

UNIVERSITÀ
POLITECNICA
DELLE MARCHE

Facoltà di Ingegneria
Master's Degree in Electronic Engineering

5G Low-Rate Polar Codes for Massive Machine-Type Communications

Codici polari 5G a basso rate per comunicazioni machine-type massive

Candidate:
Luca Perlini

Advisor:
Dr. Battaglioni Massimo

Coadvisor:
Dr. Schiavone Riccardo

Academic Year 2025-2026



UNIVERSITÀ
POLITECNICA
DELLE MARCHE

Facoltà di Ingegneria
Master's Degree in Electronic Engineering

5G Low-Rate Polar Codes for Massive Machine-Type Communications

Codici polari 5G a basso rate per comunicazioni machine-type massive

Candidate:
Luca Perlini

Advisor:
Dr. Battaglioni Massimo

Coadvisor:
Dr. Schiavone Riccardo

Academic Year 2025-2026

Università Politecnica delle Marche
Facoltà di Ingegneria
Master's Degree in Electronic Engineering
Via Brecce Bianche – 60131 Ancona (AN), Italy

Acknowledgments

Part of this thesis was developed during my research visit at the German Aerospace Center (DLR), Institute of Communications and Navigation, Oberpfaffenhofen.

I would like to express my sincere gratitude to my academic supervisor, Dr. Massimo Battaglioni, for making this experience possible and for his constant support and availability. His guidance helped me grow, both in the field of channel coding and as a person.

Special thanks go to my supervisor at DLR, Dr. Riccardo Schiavone, whose patience and ability to make complex topics easy to understand were essential: without his guidance, I would hardly have completed this work.

Finally, I thank my family, and my friends, both those who were already part of my life and those I met along the way, for making this period an experience that went far beyond studying.

To all those who contributed in any way to this thesis, I extend my sincere thanks.

Ancona, October 2026

Luca Perlini

Abstract

Massive machine-type communication is one of the key service categories of 5G, enabling connectivity for a very large number of devices that often transmit short packets under stringent reliability and complexity constraints. In this context, the design of efficient channel coding schemes operating at short blocklengths and low code rates is of particular interest. This work investigates the design and performance of polar codes, the coding scheme adopted for control channels in the 5G New Radio standard, over the binary erasure channel, focusing on the comparison of different code construction methods and the comparative evaluation of decoding strategies. Specifically, the study explores standard polar designs, Reed–Muller-like constructions, and density-evolution-based methods to determine their effectiveness in the finite-blocklength regime. Performance is assessed across a range of code rates by comparing maximum-likelihood decoding, used as the theoretical benchmark, with practical algorithms such as peeling, successive cancellation, and successive cancellation list decoding. A contribution of this work is the analysis of the decoding gap between list decoding and the optimal maximum-likelihood decoding as a function of the code rate. Numerical results show that this gap is non-monotonic: while list decoding closely approaches maximum-likelihood performance at extreme rates, very high or very low, it exhibits a peak in sub-optimality at medium rates, indicating that larger list sizes are required in this regime to approach the maximum-likelihood performance.

Sommario

Massive machine-type communication è una delle principali categorie di servizio del 5G e consente la connettività di un numero molto elevato di dispositivi che spesso trasmettono pacchetti brevi con vincoli stringenti di affidabilità e complessità. In questo contesto, la progettazione di schemi di channel coding efficienti operanti a blocklength brevi e bassi code rate è di particolare interesse. Questo lavoro studia la progettazione e le prestazioni dei polar codes, lo schema di codifica adottato per i canali di controllo nello standard 5G New Radio, sul binary erasure channel, concentrandosi sul confronto tra diversi metodi di code construction e sulla valutazione comparativa delle strategie di decodifica. Nello specifico, lo studio esplora i design polari standard, le costruzioni Reed–Muller-like e i metodi basati sulla density evolution, per determinarne l'efficacia nel regime a finite blocklength. Le prestazioni sono valutate su un intervallo di code rate confrontando la maximum-likelihood decoding, usata come benchmark teorico, con algoritmi pratici come peeling, successive cancellation e successive cancellation list decoding. Un contributo di questo lavoro è l'analisi del gap di decodifica tra la list decoding e la decodifica ottima maximum-likelihood in funzione del code rate. I risultati numerici mostrano che questo gap è non monotono: mentre la list decoding si avvicina molto alle prestazioni maximum-likelihood ai rate estremi, molto alti o molto bassi, presenta un picco di sub-ottimalità ai rate medi, indicando che in questo regime è necessaria una list size maggiore per avvicinarsi alle prestazioni maximum-likelihood.

Contents

1 Introduction **1**

2 Preliminaries **5**
2.1 Low-Rate Codes 5
2.2 Binary Erasure Channel 6
2.3 Additive White Gaussian Noise Channel 7
2.4 Hamming Distance and Minimum Distance 8
2.5 Linear Block Codes 9
2.6 Peeling Decoding over BEC 9
2.7 Maximum Likelihood Decoding over BEC 11
2.8 Singleton and Berlekamp Bounds 12

3 Low-Rate Polar Codes **15**
3.1 Polar Codes 15
3.2 Channel Polarization 15
3.3 Code Design 15
3.4 Frozen Set Constructions 17
3.5 Sparse Parity-Check Matrix Representation of Polar Codes 19
3.6 CRC Construction 21
3.7 Successive Cancellation Decoder 22
3.8 Successive Cancellation List Decoder 23
3.9 Motivation for the BEC Channel and Comparison with AWGN . . . 24

4 Numerical Results and Comparison **27**
4.1 Polar Reed-Muller-like 27
4.2 Polar Density Evolution 28
4.3 5G NR Polar Codes 29
4.4 Polar Codes at Different Rates with CRC-11 31

5 Conclusions **41**

Appendix **43**

List of Figures

2.1 Binary erasure channel 6
2.2 Tanner graph of the $(4, 2)$ code defined by $\mathbf{H}$. Circles denote variable nodes and squares denote check nodes. 10

3.1 Basic butterfly structure 16
3.2 Polar code encoding diagram 17
3.3 Factor graph of the (8,4) polar code of (3.5) 19
3.4 SCL decoding tree example 24

4.1 Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a polar code constructed using the RM-like method ($n = 1024$, $k = 100$). 28
4.2 Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a polar code constructed using the density evolution method ($n = 1024$, $k = 100$). 29
4.3 Codeword error rate as a function of the erasure probability using ML decoding, for a polar code of length $n = 1024$ with payload $A = 102$, concatenated with different CRC polynomials over the BEC. 30
4.4 Codeword error rate as a function of the erasure probability using ML decoding, for a polar code of length $n = 512$ with payload $A = 51$, concatenated with different CRC polynomials over the BEC. 30
4.5 Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(1024, 111)$ polar code concatenated with CRC-11. 31
4.6 Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(800, 111)$ polar code concatenated with CRC-11. 32
4.7 Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(200, 111)$ polar code concatenated with CRC-11. 33
4.8 Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(150, 111)$ polar code concatenated with CRC-11. 34
4.9 Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(120, 111)$ polar code concatenated with CRC-11. 35

4.10 Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(114, 111)$ polar code concatenated with CRC-11. 36
4.11 Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(111, 111)$ polar code concatenated with CRC-11, corresponding to $R = A/n \approx 0.9$. 37
4.12 Gap Δ of (4.1) between the SCL decoder with $L = 32$ and the ML decoder, and between the SCL decoder and the Berlekamp bound, as a function of the code rate, evaluated at a codeword error rate of 10^{-3}. 38
4.13 Codeword error rate as a function of the mutual information under various decoding strategies over the BEC and the AWGN channel, for a $(1024, 111)$ polar code concatenated with CRC-11. 39
4.14 Codeword error rate as a function of the mutual information under various decoding strategies over the BEC and the AWGN channel, for a $(1024, 523)$ polar code concatenated with CRC-11. 40

A.1 Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(600, 111)$ polar code concatenated with CRC-11. 43
A.2 Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(400, 111)$ polar code concatenated with CRC-11. 44

List of Tables

3.1 Comparison of the pruning strategy from [13] when applied to the parity-check matrix **H** of polar codes of different blocklengths. . . . 21

Chapter 1

Introduction

The theoretical foundations of modern digital communications were established by Shannon in his pioneering work [1], where he introduced a mathematical framework for reliable information transmission over noisy channels. In this model, the impact of noise is captured by the channel, and reliable communication is shown to be possible as long as the transmission rate R does not exceed the channel capacity C. However, Shannon's results are inherently asymptotic: they rely on arbitrarily long blocklengths and do not provide explicit constructions achieving capacity with practical complexity.

For several decades, communication systems have been primarily designed for human-centric services, such as voice calls, video streaming, and web browsing. These systems typically involve a relatively limited number of users per cell, each requiring moderate to high data rates and benefiting from coordinated resource allocation and scheduling. In this regime, classical communication techniques based on orthogonal access and centralized control have proven effective.

In contrast, modern communication systems increasingly target massive machine-type communications (mMTC), where a massive number of devices transmit short and sporadic messages with minimal or no human intervention. This paradigm is a key component of current standards such as 3GPP 5G New Radio and is expected to play an even more prominent role in future systems. Typical applications include sensing, monitoring, and control, where devices transmit small payloads under strict energy constraints.

A defining characteristic of mMTC scenarios is the presence of a very large number of potential users, of which only a small subset is active at any given time. In such random access settings, many users may transmit simultaneously over shared communication resources. Traditional multiple access schemes based on orthogonalization and scheduling become inefficient in this regime, since the associated signaling overhead and coordination complexity scale poorly with the number of devices. This motivates the adoption of uncoordinated or grant-free access schemes, where users transmit without prior resource allocation.

An achievability framework for this problem has been proposed from an information-theoretic perspective in [2], where the receiver aims to jointly detect the set of active users and decode their messages from a superposition of signals transmitted over

a common channel. A distinctive feature of this analysis is its emphasis on the finite-blocklength effects that become dominant when the payloads are small, rather than on asymptotic limits alone.

A key insight from [2] is that reliable communication in this regime requires operating with very low-rate channel codes: each device carries only a small payload, while the codeword must be very long. At the same time, practical communication systems operate under finite-blocklength constraints. In several applications, such as industrial monitoring, smart metering, and low-power telemetry, short packets are required in order to meet stringent latency and energy-efficiency requirements. This has led to significant advances in finite-blocklength information theory, providing tighter characterizations of the achievable performance in realistic scenarios [2], [3]. Despite these developments, designing practical coding schemes that approach such limits with manageable complexity remains an open challenge. Similar design constraints also arise in emerging zero-energy IoT paradigms, where devices operate under extremely stringent power budgets, often relying on energy harvesting or backscatter communication. In such scenarios, the available transmit energy is extremely limited, which naturally favors low-rate channel codes capable of ensuring reliable communication at very low signal-to-noise ratios. As a result, both massive random access and zero-energy IoT systems converge toward a common operating regime, characterized by small payloads encoded at low rates. Among the coding techniques suitable for this regime, polar codes have attracted significant attention due to their strong theoretical guarantees and practical relevance.

Polar codes are the first class of codes for which capacity-achieving performance under low-complexity decoding was rigorously proven by Arıkan [4]. Their structured construction, based on a recursive Plotkin construction, enables flexible rate adaptation and efficient encoding and decoding algorithms. Moreover, polar codes have been adopted in 3GPP 5G New Radio for control channel communications, demonstrating their practical applicability. In the context of massive random access, polar codes are particularly appealing due to their good performance at short information lengths.

Motivated by these considerations, this thesis focuses on the analysis and design of low-rate polar coding schemes for the single-user case, as a first step toward their application in massive random access scenarios. In particular, we investigate their performance in settings characterized by short payloads and low code rates. The objective is to bridge the gap between information-theoretic limits and practical implementations, enabling reliable and scalable communication for next-generation IoT systems. To address these challenges, this thesis investigates low-rate polar coding schemes through a comparative study of different polar code constructions and decoding strategies in the finite-blocklength regime. Particular emphasis is placed on understanding how code design and decoder complexity affect performance across a wide range of code rates, using maximum likelihood (ML) decoding as a benchmark and practical algorithms such as successive cancellation list (SCL) decoding as scalable alternatives. The numerical results reveal non-trivial behaviors, including a

marked dependence of the decoding gap on the code rate, providing insights that are relevant for the design of practical coding schemes for mMTC and satellite-based IoT systems. The remainder of this thesis is organized as follows. Chapter 2 introduces the theoretical background required for the analysis, including low-rate coding, channel models, and the main decoding concepts used throughout this work. Chapter 3 focuses on low-rate polar codes, describing their construction, design methodologies, and decoding algorithms, with particular attention to their relevance in the context of 5G New Radio systems. Chapter 4 presents the numerical results and performance comparisons obtained for different code constructions, decoding strategies, and code rates. Finally, Chapter 5 summarizes the main findings of this work and outlines possible directions for future research.

Chapter 2

Preliminaries

2.1 Low-Rate Codes

Forward error correction (FEC) codes are fundamental in all modern communication systems to ensure the reliability of the transmission over noisy channels. In practical scenarios, transmitted signals are affected by various impairments such as thermal noise, interference, fading, and attenuation. To mitigate these effects, redundancy is introduced at the encoder, allowing the receiver to detect and correct errors.

A linear block code is characterized by:

$$n, \text{ codeword length;} \tag{2.1}$$

$$k, \text{ number of information bits;} \tag{2.2}$$

with $n > k$.

These codes introduce redundancy (m) in every transmitted codeword. The number of redundant symbols can be computed as the difference between the length of the transmitted codeword and the number of information bits:

$$m = n - k \tag{2.3}$$

It is also possible to calculate the **code rate** as:

$$R = \frac{k}{n} \tag{2.4}$$

Low-rate codes are a class of codes with $R \ll 1$. In the context of massive random access, this regime is of particular interest: as shown in [2], achieving reliable communication in this scenario requires operating with very low-rate channel codes over relatively long blocklengths, with code ensembles on the order of $k = 100$ information bits and $n = 30000$ channel uses needed to approach the theoretical limits. This motivates the design of coding schemes that perform close to the finite-blocklength limits at very low rates.

In digital communications, the signal-to-noise ratio (SNR) is typically expressed

either per transmitted symbol or per information bit. We define

$$\gamma_s = \frac{E_s}{N_0}, \qquad \gamma_b = \frac{E_b}{N_0}, \tag{2.5}$$

where E_s is the average energy per transmitted symbol, E_b the energy per information bit, and N_0 the one-sided noise power spectral density. Since, with binary modulation, each transmitted symbol carries R information bits,

$$\gamma_s = R\,\gamma_b. \tag{2.6}$$

These two quantities are used throughout the thesis.

Since $n > k$ for any code, more symbols are always transmitted per useful bit; using a low-rate code makes this redundancy larger, potentially allowing the system to operate at a lower SNR value.

The channel capacity (C) is an upper bound on the information rate that can be communicated at an arbitrarily low error rate.

2.2 Binary Erasure Channel

The binary erasure channel (BEC) is a binary-input channel in which the transmitted bits can either be received correctly or erased, with the erasure denoted by E.

For a given erasure probability ϵ, the probability of transmitting correctly is $(1-\epsilon)$.

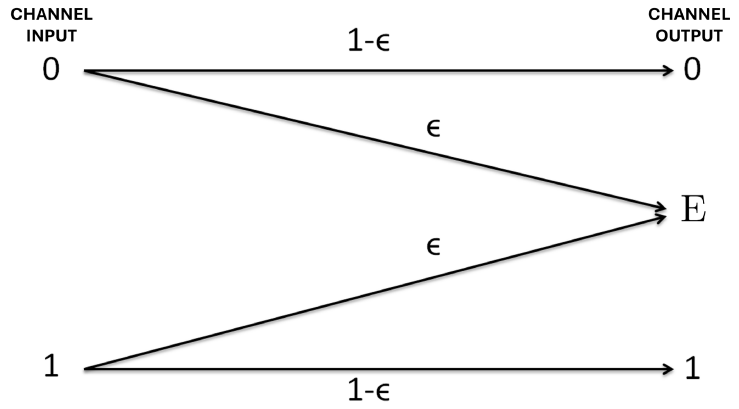


Figure 2.1: Binary erasure channel

The capacity of this channel is

$$C = 1 - \epsilon, \tag{2.7}$$

where ϵ is the erasure probability.

2.3 Additive White Gaussian Noise Channel

The additive white Gaussian noise (AWGN) channel is one of the most widely used channel models in the analysis of digital communication systems. It is an idealized communication channel in which the transmitted signal is affected only by additive noise with a Gaussian distribution and constant power spectral density over all frequencies.

The discrete system model corresponds to:

$$y_k = x_k + n_k \tag{2.8}$$

and n_k is an independent random variable with Gaussian distribution

$$n_k \sim \mathcal{N}(0, \sigma^2)$$

where σ^2 represents the noise variance and 0 its mean.

The channel capacity of the AWGN channel with binary input is given by:

$$C = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} e^{-z^2/2} \left(1 - \log_2\left(1 + e^{-2\rho - 2z\sqrt{\rho}}\right)\right) dz \tag{2.9}$$

where:

- C is the channel capacity, measured in bits per channel use, achievable with arbitrarily small error probability using suitable coding schemes;
- $\rho = E_s/\sigma^2 = 2\gamma_s = 2R\gamma_b$, with $\sigma^2 = N_0/2$.

Reliable communication with arbitrarily low error probability is possible only if the transmission rate is below the channel capacity.

In this work, the BEC is mainly adopted because it allows the use of an exact ML decoder with polynomial complexity. ML decoding selects the codeword that is most likely to have been transmitted given the received sequence; over a general channel, this requires a search whose complexity grows exponentially with the information length k. Over the binary erasure channel, instead, the decoder only needs to resolve the erased positions, which can be done with polynomial complexity. This implies that over the AWGN channel exact ML decoding becomes computationally prohibitive even for moderate blocklengths.

2.4 Hamming Distance and Minimum Distance

The Hamming distance between two vectors $\mathbf{c}$ and $\mathbf{v}$ is defined as the number of positions in which they differ:

$$d_H(\mathbf{c}, \mathbf{v}) = \sum_{i=1}^{n} \mathbb{1}\{c_i \neq v_i\} \tag{2.10}$$

where $\mathbb{1}\{\cdot\}$ denotes the indicator function, and c_i, v_i are the i-th components of $\mathbf{c}$ and $\mathbf{v}$, respectively.

For a binary linear code, the Hamming distance between two codewords equals the Hamming weight of their sum:

$$d_H(\mathbf{c}, \mathbf{v}) = w_H(\mathbf{c} + \mathbf{v}), \tag{2.11}$$

where the Hamming weight $w_H(\mathbf{c})$ of a codeword is the number of its non-zero entries. For a binary code, this reduces to

$$w_H(\mathbf{c}) = \sum_{i=1}^{n} c_i. \tag{2.12}$$

The minimum Hamming distance $d_{\min}$ of a linear block code is a parameter that determines the code's ability to detect and correct errors and erasures, and it directly influences the performance of ML decoding. Since the code is linear, the sum of two codewords is itself a codeword, and therefore the minimum distance coincides with the minimum weight over all non-zero codewords:

$$d_{\min} = \min_{\substack{\mathbf{c} \in \mathcal{C} \\ \mathbf{c} \neq \mathbf{0}}} w_H(\mathbf{c}). \tag{2.13}$$

The minimum distance determines the guaranteed correction capability:

- Error correction:

$$t = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor \tag{2.14}$$

- Erasure correction: the code is guaranteed to correct any erasure pattern with at most

$$e_{\max} = d_{\min} - 1 \tag{2.15}$$

erased positions. Beyond this number, correction may still succeed for specific erasure patterns, but it is no longer guaranteed in general.

In particular, over a binary erasure channel, ML decoding is guaranteed to recover the transmitted codeword whenever the number of erased positions is smaller than the minimum distance of the code:

$$e < d_{\min} \tag{2.16}$$

The minimum distance represents the smallest separation between any two distinct codewords in the Hamming space. A larger $d_{\min}$ implies a better separation between codewords, higher robustness against noise, and a steeper decay of the ML performance, respectively, at high SNR or low erasure probability.

The distance spectrum of a linear block code provides a characterization of the Hamming weight distribution of its codewords.

For a linear (n, k) code $\mathcal{C}$, the distance spectrum is the set $\{A_w\}_{w=0}^{n}$, where

$$A_w = |\{\mathbf{c} \in \mathcal{C} : w_H(\mathbf{c}) = w\}| \tag{2.17}$$

is the multiplicity of codewords of Hamming weight w. In particular, $A_{d_{\min}}$ denotes the multiplicity of minimum-weight codewords.

2.5 Linear Block Codes

Every linear block code can be described with a generator matrix $\mathbf{G} \in \mathbb{F}_2^{k \times n}$, such that the encoding is performed as $\mathbf{c} = \mathbf{uG}$, where $\mathbf{u} \in \mathbb{F}_2^k$ is the information vector and $\mathbf{c} \in \mathbb{F}_2^n$ is the codeword.

It can equivalently be defined by a parity-check matrix $\mathbf{H} \in \mathbb{F}_2^{(n-k) \times n}$ such that $\mathbf{H}\mathbf{x}^T = \mathbf{0}$ for every $\mathbf{x} \in \mathcal{C}$.

It is always true that

$$\mathbf{G}\mathbf{H}^T = \mathbf{0}. \tag{2.18}$$

The parity-check matrix also provides a simple mechanism for error detection: for a received vector $\mathbf{y}$, the receiver computes the syndrome $\mathbf{s} = \mathbf{H}\mathbf{y}^T$. Since $\mathbf{s} = \mathbf{0}$ whenever $\mathbf{y}$ is a valid codeword, a non-zero syndrome indicates that an error has occurred during transmission.

2.6 Peeling Decoding over BEC

The simplest example of iterative decoding is the peeling decoder introduced for the BEC by Luby et al. [5]. This decoder operates on the parity-check matrix of the code and can be applied to any linear code.

Definition 1. The Tanner graph $\mathcal{G}$ of an $m \times n$ parity-check matrix $\mathbf{H}$, with $m = n - k$, is a bipartite graph with one variable node for each code symbol and one check node for each parity check. For each non-zero entry of $\mathbf{H}$, the graph contains an edge connecting the variable node associated with the column to the check node associated with the row. Formally, $\mathcal{G} = (V \cup C, E)$, where

$$V = \{1, 2, \ldots, n\}, \quad C = \{1, 2, \ldots, m\}, \quad E = \{(i, j) \in C \times V \mid H_{i,j} \neq 0\}. \tag{2.19}$$

where V is the set of variable nodes, associated with the code symbols, and C is the set of check nodes, associated with the parity checks.

As an example, consider the parity-check matrix of a $(4, 2)$ linear code, with $m = n - k = 2$ check nodes and $n = 4$ variable nodes:

$$\mathbf{H} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix}. \tag{2.20}$$

Each column of $\mathbf{H}$ corresponds to a variable node v_j, and each row to a check node c_i. A non-zero entry $H_{i,j}$ produces an edge between v_j and c_i. The resulting Tanner graph is shown in Figure 2.2.

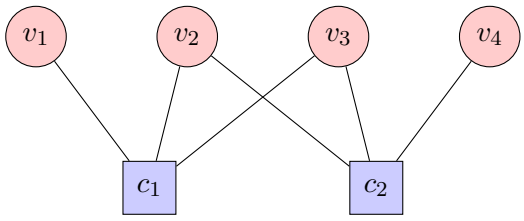


Figure 2.2: Tanner graph of the $(4, 2)$ code defined by $\mathbf{H}$. Circles denote variable nodes and squares denote check nodes.

Iteratively remove known bits from the graph as follows:

For a check node c_j, its neighborhood $N(c_j) = \{\, v_i \mid H_{j,i} \neq 0 \,\}$ is the set of variable nodes connected to it; similarly, $N(v_i) = \{\, c_j \mid H_{j,i} \neq 0 \,\}$ is the set of check nodes connected to the variable node v_i.

The peeling decoder is summarized in Algorithm 1.

Algorithm 1 Peeling decoder over the BEC

Input: received vector $\mathbf{y}$, parity-check matrix $\mathbf{H}$
Output: decoded codeword or decoding failure
for $j = 1, \ldots, m$ **do** ▷ initialization
    $e_j \leftarrow \sum_{i \in N(c_j)} \mathbb{1}(y_i = \mathrm{E})$ ▷ number of erasures at check c_j
    $s_j \leftarrow \sum_{\substack{i \in N(c_j) \\ y_i \neq \mathrm{E}}} y_i$ ▷ partial syndrome
end for
while there exists j such that $e_j = 1$ **do**
    let c_j be a degree-one check node and v_i its erased neighbor
    $y_i \leftarrow s_j$ ▷ recover the erased bit
    for $j' \in N(v_i)$ **do** ▷ update neighboring checks
        $s_{j'} \leftarrow s_{j'} + y_i$
        $e_{j'} \leftarrow e_{j'} - 1$
    end for
end while
if no erased positions remain **then**
    return decoded codeword ▷ success
else
    return decoding failure
end if

2.7 Maximum Likelihood Decoding over BEC

Let $\mathcal{C} \subseteq \{0,1\}^n$ be an (n, k) linear code. For a received vector $\mathbf{y}$, we define:

- $\mathcal{R}$: set of non-erased positions;
- $\mathcal{E}$: set of erased positions.

The ML decoder chooses the codeword $\hat{\mathbf{x}} \in \mathcal{C}$ that maximizes the likelihood of the received vector:

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x} \in \mathcal{C}} P(\mathbf{y} \mid \mathbf{x}). \tag{2.21}$$

If we are decoding over a BEC, every codeword that fits the $\mathcal{R}$ received indexes has the same probability. Thus, the ML problem reduces to finding the codewords compatible with the correctly received bits, defined as

$$\mathcal{S} = \{\mathbf{x} \in \mathcal{C} \mid x_i = y_i \ \forall i \in \mathcal{R}\}. \tag{2.22}$$

If $\mathcal{S}$ contains exactly one codeword, the decoder outputs it as the estimated codeword $\hat{x}$; otherwise, if $\mathcal{S}$ contains multiple codewords, the decoder declares a decoding failure.

Since the code is linear:

$$\mathbf{x} = \mathbf{u}\mathbf{G} \tag{2.23}$$

where:

- $\mathbf{u} \in \{0,1\}^k$ is the information vector;
- $\mathbf{G} \in \mathbb{F}_2^{(k\times n)}$.

Considering only the correctly received bits:

$$\mathbf{y}_{\mathcal{R}} = \mathbf{u}\,\mathbf{G}_{\mathcal{R}} \tag{2.24}$$

where $\mathbf{G}_{\mathcal{R}}$ is the submatrix of $\mathbf{G}$ containing only the columns corresponding to the correctly received positions. ML decoding thus reduces to solving the linear system

$$\mathbf{u}\,\mathbf{G}_{\mathcal{R}} = \mathbf{y}_{\mathcal{R}}. \tag{2.25}$$

The solution is unique if and only if $\text{rank}(\mathbf{G}_{\mathcal{R}}) = k$, in which case the decoder successfully recovers the transmitted message.

2.8 Singleton and Berlekamp Bounds

The Singleton bound, first introduced in [6], is a fundamental limit in coding theory that relates the parameters of a block code: the length n, the dimension k, and the minimum distance $d_{\min}$. For a linear (n,k) binary code, the bound states:

$$d_{\min} \leq n - k + 1. \tag{2.26}$$

The Singleton bound reflects the trade-off between redundancy and error-correcting capability: increasing k, the redundancy decreases and $d_{\min}$ usually decreases as well, limiting error correction; decreasing k, $d_{\min}$ increases, allowing more errors to be corrected.

For a BEC, the Singleton bound provides a direct limit on the number of erasures that can be recovered:

$$e_{\max} = n - k \tag{2.27}$$

where $e_{\max}$ is the maximum number of erasures that can always be corrected. Only maximum distance separable (MDS) codes achieve this bound with equality.

The block error probability associated with the Singleton bound is

$$P_B^{(S)}(n,k,\epsilon) = \sum_{e=n-k+1}^{n} \binom{n}{e} \epsilon^e (1-\epsilon)^{n-e}, \tag{2.28}$$

which represents a lower bound on the block error probability of any (n,k) binary linear code over the BEC, achieved with equality only by MDS codes.

The union bound provides an upper bound on the decoding failure probability of a linear code over the BEC under ML decoding. Unlike the Singleton bound, which constrains the minimum distance deterministically, it characterizes the block erasure probability as a function of the channel erasure probability. Consider a binary linear (n, k) code with minimum distance $d_{\min}$, over a BEC with erasure probability ϵ. By the union bound

$$P_B \leq \sum_{\mathbf{c}\in\mathcal{C},\mathbf{c}\neq\mathbf{0}} \epsilon^{w_H(\mathbf{c})}. \tag{2.29}$$

Equivalently, using the distance spectrum of the code, this can be written as

$$P_B \leq \sum_{w=d_{\min}}^{n} A_w \, \epsilon^w \tag{2.30}$$

For any non-zero codeword $\mathbf{c}$ of weight w, failure occurs whenever all w positions are erased, which happens with probability ϵ^w.

For small erasure probabilities ($\epsilon \ll 1$), the dominant contribution to this sum comes from codewords of minimum weight $d_{\min}$.

Hence:

$$P_B \approx A_{d_{\min}} \epsilon^{d_{\min}}. \tag{2.31}$$

This approximation is tight in the low-erasure regime, where failures are rare and dominated by minimum-weight codewords.

In addition to the Singleton bound, this thesis also refers to the Berlekamp random-coding bound [7], an achievability bound obtained by averaging the block error probability over the ensemble of random binary linear (n, k) codes:

$$P_B^{(\mathsf{B})}(n, k, \epsilon) = \sum_{e=n-k+1}^{n} \binom{n}{e} \epsilon^e (1-\epsilon)^{n-e} + \sum_{e=1}^{n-k} \binom{n}{e} \epsilon^e (1-\epsilon)^{n-e} 2^{-(n-k-e)}. \tag{2.32}$$

Chapter 3

Low-Rate Polar Codes

3.1 Polar Codes

Polar codes are a family of error-correcting codes that achieve the symmetric capacity of any binary-input discrete memoryless channel (B-DMC) with polynomial encoding and decoding complexity, as shown by Arıkan in [4]. This means it is possible to transmit at any rate below the symmetric capacity with an arbitrarily low error probability in the limit of large blocklengths.

3.2 Channel Polarization

These codes are named *polar* due to a channel polarization effect that transforms n i.i.d. channels W_i into a mixture of almost noise-free channels ($H(X \mid Y) \approx 0$) and noise-limited channels ($H(X \mid Y) \approx 1$), where $H(X \mid Y)$ is the conditional entropy of the input X given the channel output Y. The transformation is lossless in terms of mutual information, required to achieve the capacity, and it has low complexity, since it can be realized with $\mathcal{O}(n \log_2 n)$ complexity.

For any fixed $\delta > 0$, the fraction of the intermediate-reliability synthetic channels vanishes as $n \longrightarrow \infty$, i.e., we have

$$\lim_{n\to\infty} \frac{1}{n} \left|\left\{ i \in \{1, \ldots, n\} : \delta < H\left(W_n^{(i)}\right) < 1 - \delta \right\}\right| = 0, \tag{3.1}$$

where $W_n^{(i)}$ denotes the i-th synthetic (polarized) channel obtained from n uses of the underlying B-DMC W.

Since the transform is information-lossless, we can write:

$$\begin{aligned} \lim_{n\to\infty} \frac{1}{n} |\{i \in \{1, \ldots, n\} : H(W_n^{(i)}) \leq \delta\}| &= C(W) \\ \lim_{n\to\infty} \frac{1}{n} |\{i \in \{1, \ldots, n\} : H(W_n^{(i)}) \geq 1 - \delta\}| &= 1 - C(W) \end{aligned} \tag{3.2}$$

3.3 Code Design

Arıkan's polar codes of blocklength n are constructed with the Kronecker product of the kernel:

$$F_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \tag{3.3}$$

repeated $\log_2(n)$ times.

The kernel represents the operation in Figure 3.1,

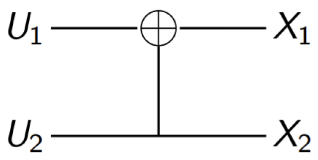


Figure 3.1: Basic butterfly structure

where

- u_1, u_2 are the input bits;
- x_1, x_2 are the output bits.

For a polar code of blocklength n, with n a power of two, the base matrix is constructed via the Kronecker product as $F_n = F_2^{\otimes p}$, where $p = \log_2 n$ is the number of polarization stages. Because of the channel polarization, the number of information bits k is smaller than n: the positions corresponding to the worst channels are frozen (set to zero). The reliability of the polarized channels induces a partition of the n index positions into two sets: the reliable positions, which carry information bits, and the unreliable ones, called the *frozen set*, whose bits (the *frozen bits*) are fixed to a known value, typically zero. From the base matrix F_n, for a given frozen set, the generator matrix $\mathbf{G}$ is obtained by keeping only the rows of F_n that do not correspond to the frozen bits.

The parity-check matrix is then created by taking the columns of F_n that correspond to the frozen positions and transposing them. Thus, for a polar code of blocklength $n = 8$, F_8 is:

$$\mathbf{F}_8 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \tag{3.4}$$

and its encoding corresponds to:

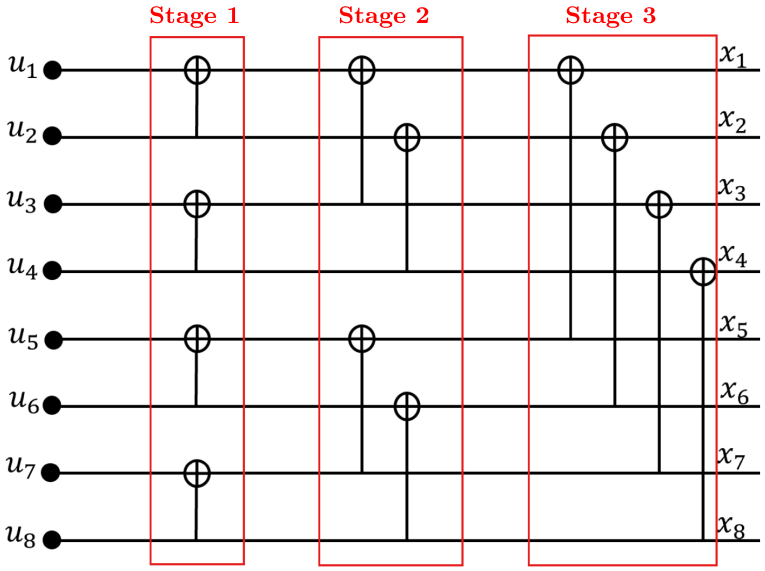


Figure 3.2: Polar code encoding diagram

For example, given the frozen set $\mathcal{F} = \{1, 2, 3, 5\}$, the parity-check matrix is:

$$\mathbf{H^T} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & & 5 & & & \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix} \tag{3.5}$$

3.4 Frozen Set Constructions

Selecting which positions belong to the frozen set is the central design choice of a polar code, since it determines which synthetic channels carry information bits and which are frozen. In this thesis we focus on four construction methods to select the frozen bit positions for an (n, k) polar code:

1. Reed-Muller-like [8], where the k rows of F_n with the largest Hamming weight are selected as information positions;

2. Density evolution [9], which tracks the evolution of the message densities to estimate the reliability of each synthetic channel;

3. β-expansion [10], where, for a given value of β, the k positions with the largest reliability metric are selected;

4. 5G-NR standard [11] construction.

For the β-expansion, the reliability metric of the row index i, whose binary representation is $(b_{p-1}, \ldots, b_1, b_0)$, is the polarization weight

$$\mathrm{PW}(i) = \sum_{j=0}^{p-1} b_j \beta^j, \tag{3.6}$$

where $b_j \in \{0, 1\}$ are the bits of the binary representation of i, and we set $\beta = 2^{1/4}$. The construction above is defined only for blocklengths that are powers of two. To transmit n coded bits with n not a power of two, the 5G NR standard [12] applies rate matching to a mother code of length $n_{\mathrm{m}} = 2^p \geq n$, constructed as described above with n_{m} in place of n. The n_{m} coded bits are reordered by a sub-block interleaver, and $n_{\mathrm{m}} - n$ of them are then removed. If $k/n \leq 7/16$, the first $n_{\mathrm{m}} - n$ interleaved bits are punctured, i.e., not transmitted and treated as erasures by the decoder. Otherwise, the last $n_{\mathrm{m}} - n$ interleaved bits are shortened, i.e., forced to zero by the encoder and treated as known by the decoder. In both cases, the bit-channel indices affected by the removed positions are frozen before selecting the k most reliable remaining positions [11]. Over the BEC, a punctured bit is therefore equivalent to an erasure, while a shortened bit is equivalent to a correctly received zero.

3.5 Sparse Parity-Check Matrix Representation of Polar Codes

The Tanner graph constructed from the parity-check matrix in (3.5) of a polar code is loopy and dense. This prevents the peeling decoder from working effectively. However, a sparse factor graph can be realized from the encoding circuit, as in [13]. An example is shown in Figure 3.3 for $n = 8$.

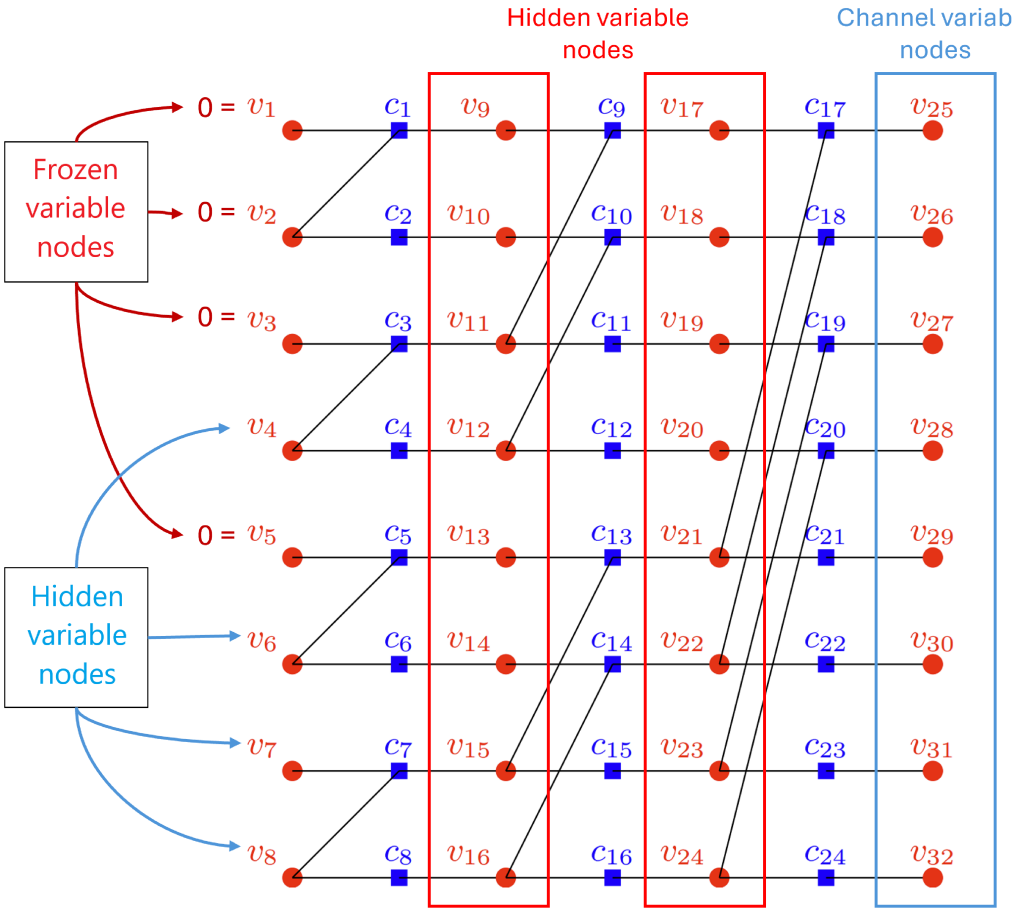


Figure 3.3: Factor graph of the (8,4) polar code of (3.5)

In the factor graph we can distinguish three different types of variable nodes: channel variable nodes (CVN), corresponding to the codeword, frozen variable nodes (FVN), and the rest are hidden variable nodes (HVN). In the example of Figure 3.3, the CVNs are $v_{25}, \ldots, v_{32}$, the FVNs are v_1, v_2, v_3, v_5, and the remaining variable nodes are HVNs.

This factor graph has $n(\log_2 n + 1)$ variable nodes and $n \log_2 n$ check nodes, and its corresponding parity-check matrix is sparse. It is much sparser than the standard one, at the cost of a larger size.

As in [13], we can prune the graph with the following procedure:

1. **FVN removal:** first we remove the columns of **H** corresponding to the FVNs.

2. **Check nodes of degree 1:** the standard polar FG contains check nodes of degree 2 and 3 only. However, after applying step 1, some check nodes can have degree 1. Their neighboring variable node must be 0. Thus, this check node and its single neighbor variable node can be removed from the FG (the column corresponding to that variable node is removed from the PCM).

3. **A CVN connected to a degree 2 check node with an HVN:** The CVN must be equal to the HVN. Thus, the connecting check node can be removed, and the HVN can be replaced with the CVN.

4. **HVN of degree 1:** This HVN does not contribute to the decoding of the other variables since it is (0, 1) with probabilities ($\frac{1}{2}$, $\frac{1}{2}$). Hence, the variable node and the check node it is connected to, can be removed from the graph.

5. **HVN of degree 2:** This HVN can be removed and the two connected check nodes can be merged.

6. **Degree 2 check node that is connected to two HVNs:** The two HVNs must be equal. Hence, the check node can be removed and the two HVNs can be merged.

Iterating over the pruning steps described above until the graph cannot be further simplified results in a smaller, valid parity-check matrix for the code.

The pruning rate of the parity-check matrix **H** was computed for block lengths $n = 2^p$, ranging from 32 to 1024, with a fixed rate of $R \approx 1/10$.

The frozen sets used in this analysis are obtained from the 5G New Radio polar code construction, which selects the information positions according to the reliability sequence standardized in 3GPP TS 38.212 [12]. Table 3.1 reports the number of rows and columns before and after pruning, together with the corresponding pruning rates. It also includes the number of ones in the matrix and their percentage with respect to the total number of elements, both before and after pruning, as well as the corresponding reduction in the number of ones.

			1s Pruning		
(n,k)	#1s	1s	#1 after Pruning	1s after Pruning	1s Pruning rate
(32,3)	400	1.30%	60	6.5%	85.0%
(64,6)	960	0.56%	152	3.7%	84.2%
(128,12)	2240	0.24%	368	2.3%	83.6%
(256,25)	5120	0.11%	804	1.1%	84.3%
(512,50)	11520	0.049%	1708	0.5%	85.2%
(1024,100)	25600	0.022%	3916	0.2%	84.7%

Table 3.1: Comparison of the pruning strategy from [13] when applied to the parity-check matrix $\mathbf{H}$ of polar codes of different blocklengths.

			Rows and Columns Pruning			
(n,k)	Rows	Cols	Final Rows	Final Cols	Pruned Rows	Pruned Cols
(32,3)	160	192	29	32	81.9%	83.3%
(64,6)	384	448	61	67	84.1%	85.0%
(128,12)	896	1024	121	133	86.5%	87.0%
(256,25)	2048	2304	261	289	87.3%	87.5%
(512,50)	4608	5120	541	591	88.3%	88.5%
(1024,100)	10240	11264	1241	1341	87.9%	88.1%

3.6 CRC Construction

A $(k+r, k)$ **cyclic redundancy check (CRC)** code is a linear code typically used for error detection [14].

A CRC is represented by its polynomial generator $g(x)$

$$g(x) = g_0 + g_1 x + g_2 x^2 + \cdots + g_r x^r \tag{3.7}$$

in $\mathbb{F}_2$ with degree r.

The CRC code can be represented also by a generator and a parity-check matrix.

The generator matrix $\mathbf{G}_{\text{CRC}}$ has size $k \times n$, with $n = k + r$, and it allows to calculate the codewords as follows:

$$\mathbf{c} = \mathbf{u}\,\mathbf{G}_{\text{CRC}} \tag{3.8}$$

where $\mathbf{u}$ is the message. The $\mathbf{G}_{\text{CRC}}$ is built directly from the generator polynomial $g(x)$ in (3.7) (with $g_0 = g_r = 1$), by placing its coefficients on each row, shifted by one position with respect to the previous row. Note that this construction yields a non-systematic encoding:

$$\mathbf{G}_{\text{CRC}} = \begin{pmatrix} g_0 & g_1 & \cdots & g_r & & \\ & g_0 & g_1 & \cdots & g_r & \\ & & \ddots & & & \ddots \\ & & & g_0 & g_1 & \cdots & g_r \end{pmatrix} \tag{3.9}$$

Equivalently, $\mathbf{G}_{\text{CRC}}$ can be constructed row by row, yielding a systematic encoding: each row corresponds to a message $\mathbf{e}_i$, where $\mathbf{e}_i = (0, \ldots, 0, 1, 0, \ldots, 0)$ has a single 1 in the i-th position (representing the polynomial $e_i(x) = x^{i-1}$), multiplied by x^r and divided by $g(x)$; the remainder $r(x)$ of this division gives the CRC part. The row of $\mathbf{G}_{\text{CRC}}$ becomes $[\mathbf{e}_i \mid \mathbf{r}]$, where the first k bits are the message and the last r bits are the CRC.

The parity-check matrix $\mathbf{H}_{\text{CRC}}$ is useful to verify the integrity of the received codeword: if the codeword is not divisible by $g(x)$, the syndrome $\mathbf{s} = \mathbf{v}\mathbf{H}_{\text{CRC}}^T$ is different from $\mathbf{0}$, meaning an error has occurred. $\mathbf{H}_{\text{CRC}}$ has size $r \times n$, with $n = k + r$, and can be obtained from $\mathbf{G}_{\text{CRC}}$ (e.g., via Gauss-Jordan elimination over $\mathbb{F}_2$), so that $\mathbf{G}_{\text{CRC}}\mathbf{H}_{\text{CRC}}^T = \mathbf{0}$.

CRC represents an efficient and structured method for error detection. Even though it is not primarily designed for correction, its use concatenated with advanced channel codes, such as convolutional or polar codes, allows for significantly improved performance in the finite-length regime and constitutes a fundamental tool in modern communication systems [15].

3.7 Successive Cancellation Decoder

Successive cancellation (SC) is a decoding algorithm for polar codes that exploits the recursive structure of polar transform.

In the case of transmission over an AWGN channel, the decoder works with log-likelihood ratios (LLRs). Given the received (or updated) observation associated to bit u_i, the LLR is defined as

$$\ell_i = \ln \frac{\Pr(u_i = 0 \mid \mathbf{y})}{\Pr(u_i = 1 \mid \mathbf{y})} \tag{3.10}$$

where $\mathbf{y}$ denotes the channel observation (or the combination of observations propagated through the decoding stages up to bit i). A positive value of ℓ_i indicates that $\mu_i = 0$ is more likely, while a negative value indicates that $\mu_i = 1$ is more likely; the magnitude $|\ell_i|$ reflects the reliability of that decision. These reliability values are progressively updated and combined as the decoder moves through the stages of the polar code structure. At each step, the decoder uses these updated LLRs to make a decision on the current bit, choosing the most likely value. This iterative refinement allows the decoder to exploit the full structure of the code, but it also means that errors made early in the process can affect subsequent decisions.

SC decoder estimates bits in increasing index order:

$$\hat{u}_0, \hat{u}_1, \ldots, \hat{u}_{n-1}.$$

At the i-th step, for $i = 0, 1, \ldots, n-1$, the decoder computes $\hat{u}_i$:

$$\hat{u}_i = \begin{cases} 0 & \text{if } i \in \mathcal{A}^C \text{ (frozen)} \\ \arg\max\limits_{b \in \{0,1\}} \Pr\left(u_i = b \mid \mathbf{y}, \hat{\mathbf{u}}_0^{(i-1)}\right) & \text{otherwise} \end{cases} \tag{3.11}$$

where $\mathcal{A}$ denotes the set of indices carrying information bits, and its complement

$$\mathcal{A}^C = \{0, \ldots, n-1\} \setminus \mathcal{A}$$

denotes the set of *frozen* bit indices, whose values are fixed and known both to the encoder and to the decoder (typically set to 0). Here $\mathbf{y}$ is the received vector and $\hat{\mathbf{u}}_0^{(i-1)} = (\hat{u}_0, \ldots, \hat{u}_{i-1})$ is the vector of previously estimated bits.

Thus, the decision depends on the received vector and all the previous estimated bits.

When considering the BEC, LLRs will take only three values, assuming the standard BPSK bit-to-LLR mapping ($0 \to +\infty$, $1 \to -\infty$):

$$\ell_i \in \{+\infty, -\infty, 0\} \tag{3.12}$$

corresponding to known 0, known 1, and erasure, respectively.

As a result, the decoder no longer needs to manipulate LLRs, but only to track whether information is available or not. The message updates reduce to simple logical rules that describe how erasures propagate through the decoding structure.

The computational complexity of SC decoding is:

$$O(n \log_2(n)).$$

3.8 Successive Cancellation List Decoder

The successive cancellation list (SCL) decoding is an extension of the standard SC decoder designed to improve finite-length performance. Decoding can be represented as a binary tree, where each level corresponds to a bit index u_i and each branch to one of its possible values $\{0, 1\}$ (see Figure 3.4). While SC follows a single decision path through this tree, SCL maintains multiple candidate paths simultaneously.

In SC decoding, once a wrong decision is made at some bit u_i, all subsequent decisions are affected. Since SC does not reconsider previous choices, early errors may propagate through the decoding tree.

The SCL decoder mitigates the issues by exploring both possible values $u_i = 0$ and $u_i = 1$ for each information bit, maintaining a list of the L most likely decoding paths and pruning the less likely paths according to a path metric.

Thus, SCL provides a trade-off between complexity and performance.

The computational complexity of SCL decoding is:

$$O(Ln \log_2(n))$$

A common enhancement consists in concatenating a CRC to the information bits: the SCL produces L candidate codewords, and the CRC check is used to select the codeword that satisfies it among the candidates; if multiple candidates pass the CRC check, the one with the best path metric is chosen; if none of them passes the check, the candidate with the best path metric is returned.

As the list size L increases, the performance of the SCL decoder approaches the ML decoder, at the cost of increased computational complexity.

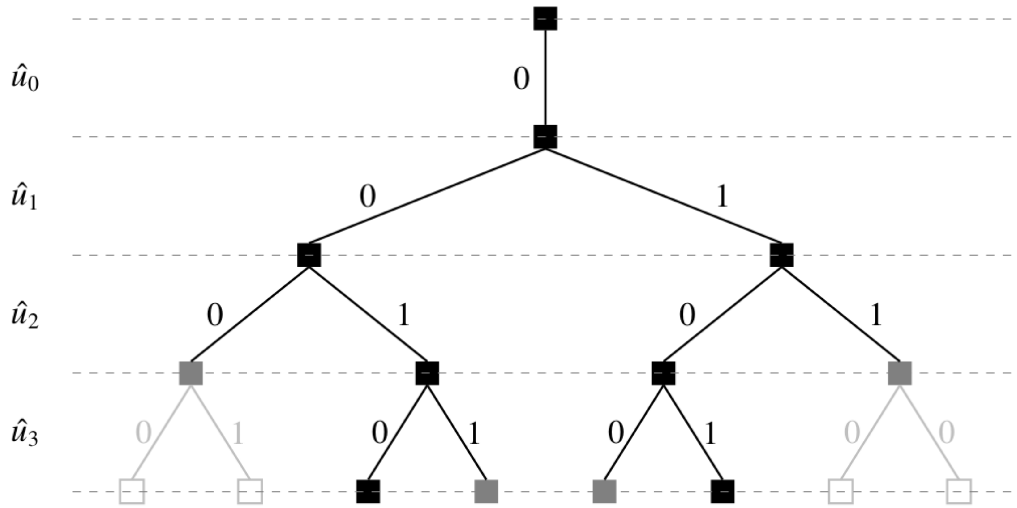


Figure 3.4: SCL decoding tree example

3.9 Motivation for the BEC Channel and Comparison with AWGN

In this work, the BEC is adopted as the primary reference model for several parts of the analysis. The main reason is that, for the considered decoding strategies, exact ML decoding over the AWGN channel requires an exhaustive search whose complexity grows exponentially with the information length, which makes direct performance benchmarking significantly more complex.

Although the analysis is carried out in the BEC setting, the goal is to understand whether the observed behavior extends to more realistic channels. In particular, the AWGN channel is of primary practical interest in wireless communications. For this reason, the results obtained in the BEC case are also interpreted as indicators of performance trends expected under AWGN conditions.

To support this comparison, both channel models are represented in terms of their mutual information, by mapping the erasure probability of the BEC and the SNR of the AWGN channel to equivalent mutual information values. Matching the mutual information provides a common scale, but it does not make the two channels equivalent at finite blocklength, so the comparison is to be understood in qualitative terms: it indicates whether the coding and decoding phenomena observed in the BEC setting persist under Gaussian noise. For the BEC, the mutual information admits a closed-form expression given by

$$I_{BEC} = 1 - \epsilon$$

where ϵ is the erasure probability.

For the AWGN channel with BPSK modulation, the mutual information is com-

puted as a function of γ_s according to

$$I_{\text{AWGN}} \approx J\left(2\sqrt{2\gamma_s}\right), \tag{3.13}$$

where

$$J(x) \approx \left(1 - 2^{-H_1 x^{H_2}}\right)^{H_3} \tag{3.14}$$

$$H_1 = 0.3073; \qquad H_2 = 1.7870; \qquad H_3 = 1.1064. \tag{3.15}$$

Chapter 4

Numerical Results and Comparison

This chapter presents the numerical evaluation of different low-rate polar code constructions over the BEC. The objective is to assess the performance of different code constructions and decoding strategies. In particular, comparisons are carried out between 5G NR polar codes, Reed-Muller-like designs, and density evolution-based constructions. The decoding methods considered include ML, SC, SCL, and peeling decoders. In some cases, the performance of the peeling decoder is not reported, as it is consistently inferior to that of the other considered decoders. The results are presented in terms of codeword error rate, defined as the probability that the decoded codeword differs from the transmitted one, as a function of the erasure probability, providing insights into the trade-offs between performance and complexity.

4.1 Polar Reed-Muller-like

The decoding methods considered include ML and peeling decoding over the sparse factor graph, constructed with the procedure presented in Sec. 3.5. The results are shown in Figure 4.1.

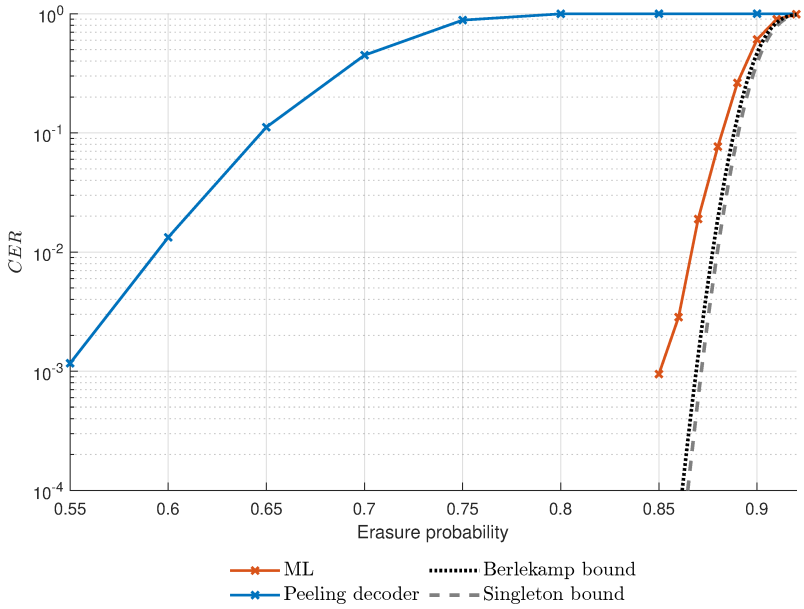


Figure 4.1: Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a polar code constructed using the RM-like method ($n = 1024$, $k = 100$).

It is clear that the ML decoder significantly outperforms the peeling decoder. This is because the effectiveness of the peeling decoder relies on the presence of degree-one check nodes throughout the decoding process, see Section 2.6, which depends on the structure of the sparse factor graph and on the stopping sets it contains, thereby limiting its performance.

4.2 Polar Density Evolution

In this section, the design erasure probability is set to 0.87, close to the channel capacity limit for a code of rate $R = 1/10$, so that the construction is optimized in the regime of interest.

Figure 4.2 shows the codeword error rate as a function of the erasure probability for the resulting code. Unlike the RM-like construction (Figure 4.1), where the peeling decoder collapses well before the ML waterfall region, here the peeling decoder performs close to the ML decoder across the entire range, with only a small gap between the two curves. This indicates that the density-evolution construction yields a parity-check matrix structure that is considerably more favorable to peeling decoding than the RM-like construction, at the cost of a worse ML performance.

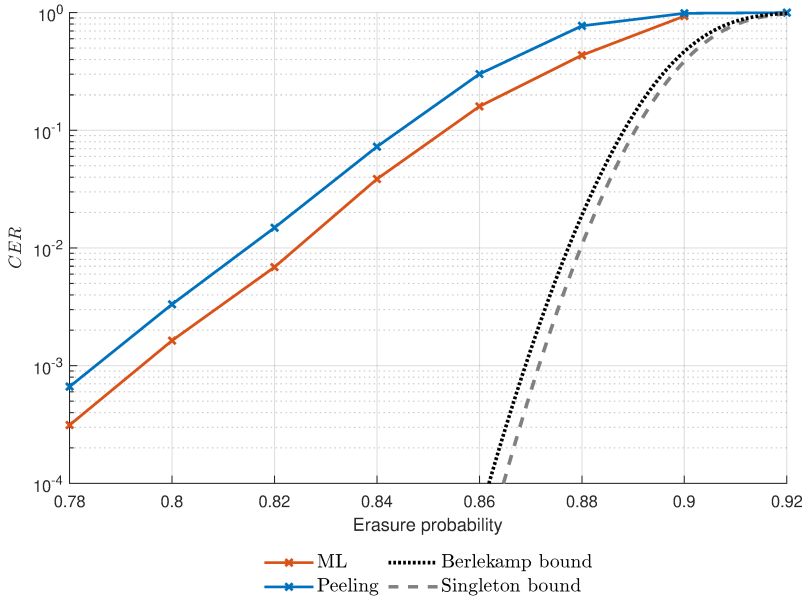


Figure 4.2: Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a polar code constructed using the density evolution method ($n = 1024$, $k = 100$).

4.3 5G NR Polar Codes

Polar codes in the 5G NR standard are concatenated with a CRC. In this section, a payload of A information bits is appended with an r-bit CRC, so that the polar encoder input has length $k = A + r$. Decoding is performed over the BEC using peeling and ML decoders for polar codes of length $n = 1024$ with $A = 102$ and of length $n = 512$ with $A = 51$.

The CRC polynomials used in this section are CRC-6, CRC-11, CRC-16, and CRC-24c, from [12].

Figures 4.3 and 4.4 show, for $n = 1024$, $A = 102$ and $n = 512$, $A = 51$ respectively, that all four CRC polynomials yield a similar improvement over the no-CRC case, shifting the waterfall region to higher erasure probabilities. Among them, CRC-6 provides a smaller improvement than the longer polynomials, namely CRC-11, CRC-16, and CRC-24c, which perform almost identically, indicating that most of the gain is achieved already with an 11-bit CRC and further redundancy brings only marginal benefits.

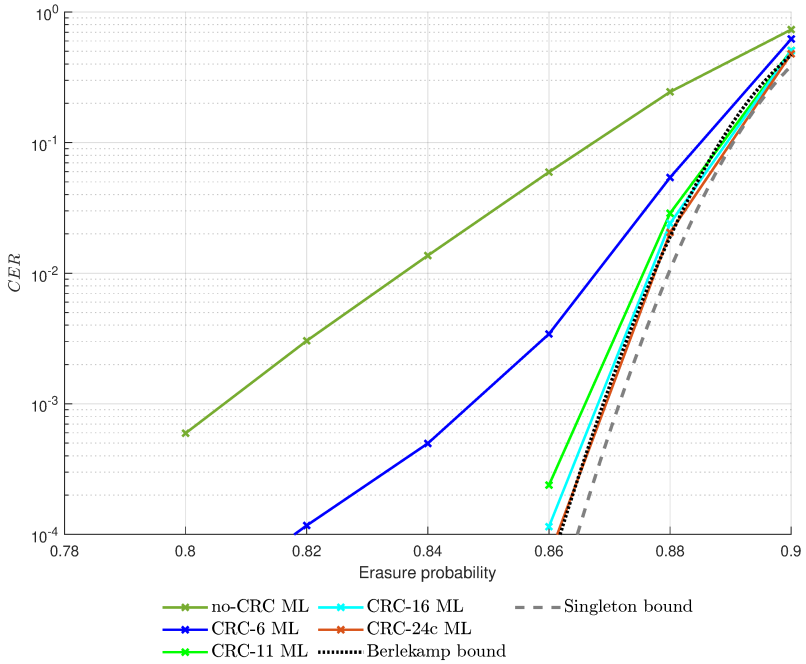


Figure 4.3: Codeword error rate as a function of the erasure probability using ML decoding, for a polar code of length $n = 1024$ with payload $A = 102$, concatenated with different CRC polynomials over the BEC.

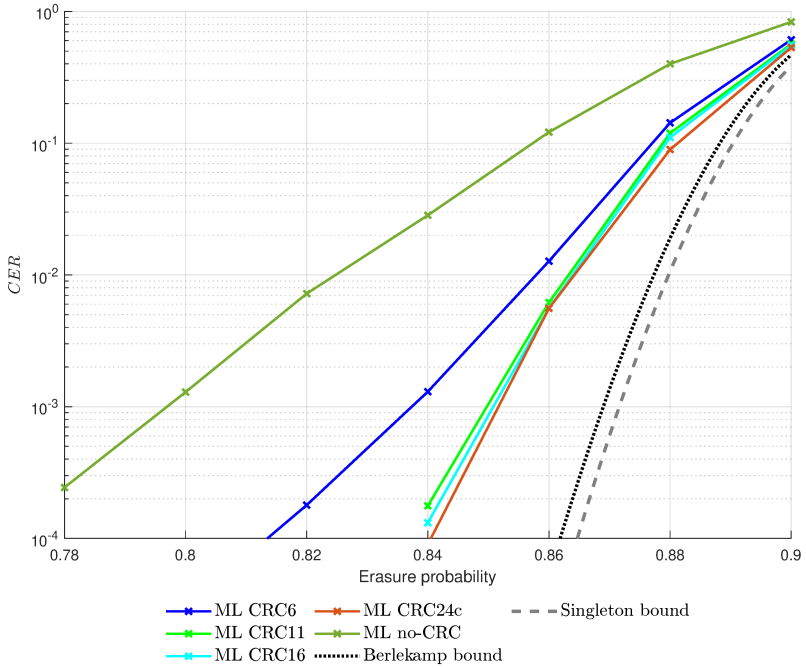


Figure 4.4: Codeword error rate as a function of the erasure probability using ML decoding, for a polar code of length $n = 512$ with payload $A = 51$, concatenated with different CRC polynomials over the BEC.

The performance of the peeling decoder is not shown, as it is significantly inferior

to that of the other decoders, for every chosen CRC polynomial.

4.4 Polar Codes at Different Rates with CRC-11

Throughout this section, $A = 100$ denotes the payload length, i.e., the number of information bits before CRC concatenation. The 11 CRC bits are appended to the payload, so the polar encoder input has length $k = A + 11 = 111$, while the rate reported in the figures is defined as $R = A/n$, with n the number of transmitted bits.

Except for $n = 1024$, the considered blocklengths are not powers of two and are obtained through the 5G NR rate matching described in Section 3.4. The codes with $n \in \{800, 600\}$ use a mother code of length $n_{\mathrm{m}} = 1024$, and the code with $n = 400$ uses $n_{\mathrm{m}} = 512$; all three are punctured. The codes with $n \in \{200, 150\}$ use $n_{\mathrm{m}} = 256$, and those with $n \in \{120, 114, 111\}$ use $n_{\mathrm{m}} = 128$; all five are shortened. The Singleton and Berlekamp bounds are computed for an (n, A) code.

The decoding performance is evaluated using ML and SCL decoding with list size $L = 32$ and $L = 1$, the latter coinciding with SC decoding.

The goal of these simulations is to analyze how the gap between SCL and ML decoding depends on the code rate, comparing two list sizes.

The considered rates are approximately $\frac{1}{10}, \frac{1}{8}, \frac{1}{6}, \frac{1}{4}, \frac{1}{2}, \frac{2}{3}, \frac{5}{6}, \frac{7}{8}, \frac{9}{10}$.

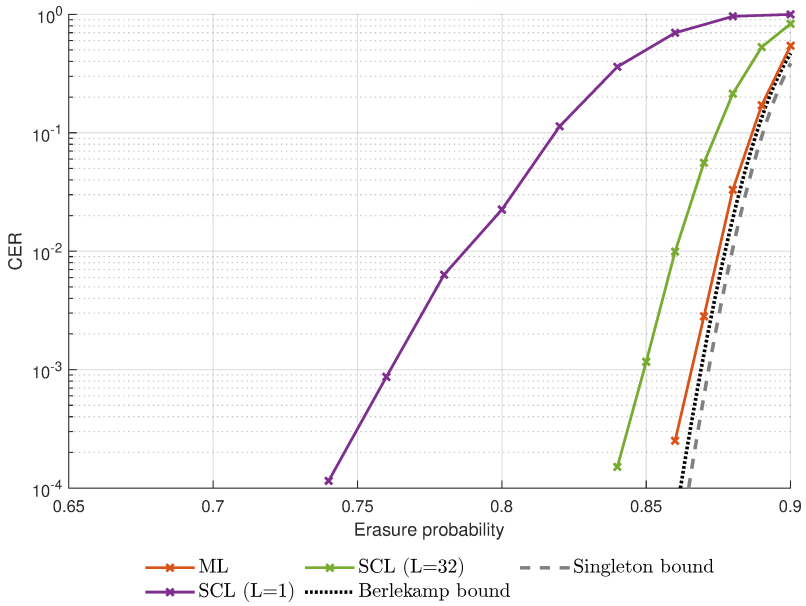


Figure 4.5: Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(1024, 111)$ polar code concatenated with CRC-11.

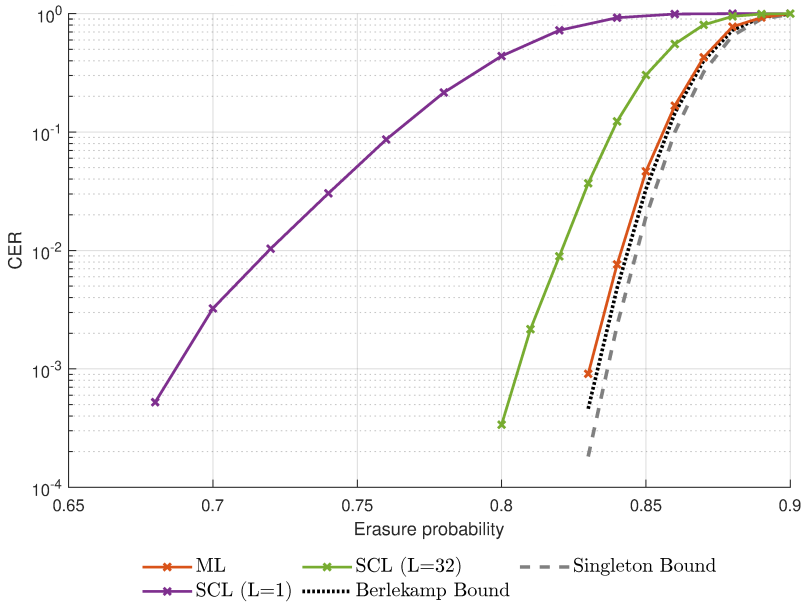


Figure 4.6: Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(800, 111)$ polar code concatenated with CRC-11.

Figures 4.5 and 4.6 show the performance for a low-rate configuration. As expected, the ML decoder provides the best performance, serving as a benchmark. The SCL decoder with list size $L = 32$ approaches the ML curve, while the SC decoder exhibits a noticeable performance gap. This behavior is due to the fact that, at low rates, the code introduces a large amount of redundancy, which allows list decoding to effectively explore multiple candidate paths and correct early decision errors. In contrast, SC decoding is more sensitive to error propagation, leading to inferior performance.

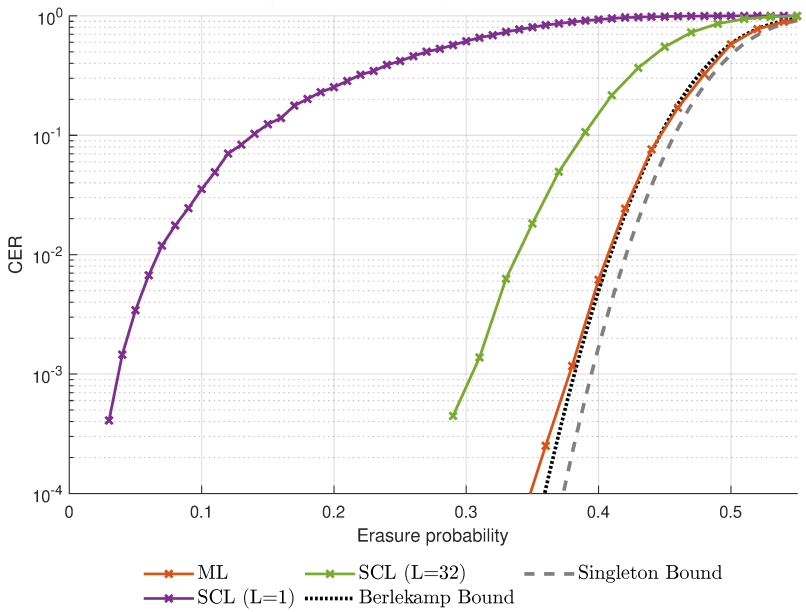


Figure 4.7: Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(200, 111)$ polar code concatenated with CRC-11.

Moving from the low-rate to the medium-rate regime, the performance gap between SCL and ML becomes more pronounced. While SCL still provides a significant improvement over SC, its ability to approximate ML decoding is reduced. This is due to the lower redundancy available at higher rates, which limits the effectiveness of path diversity in the list decoding process.

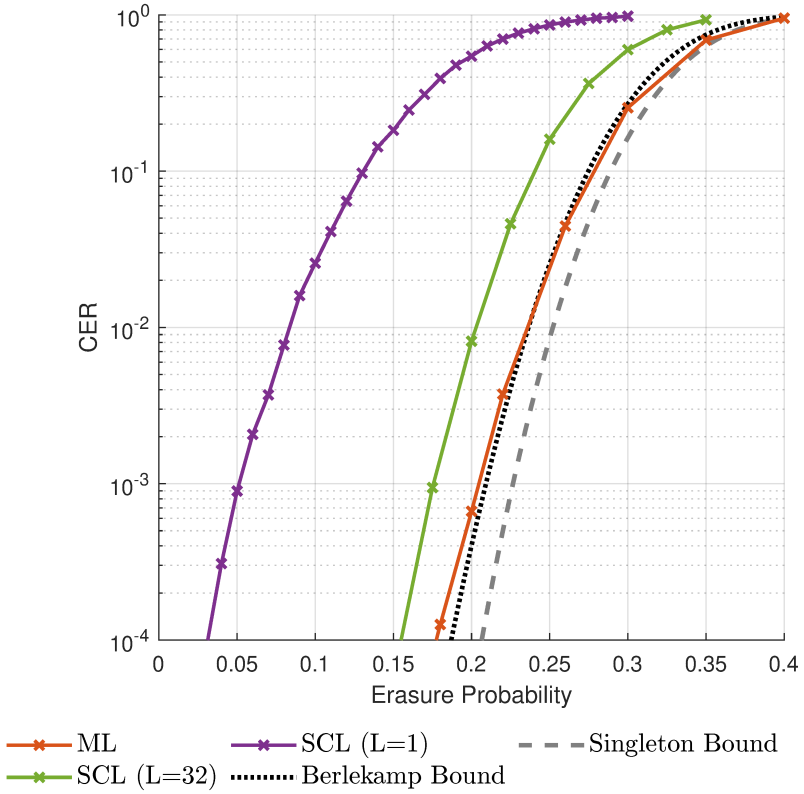


Figure 4.8: Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(150, 111)$ polar code concatenated with CRC-11.

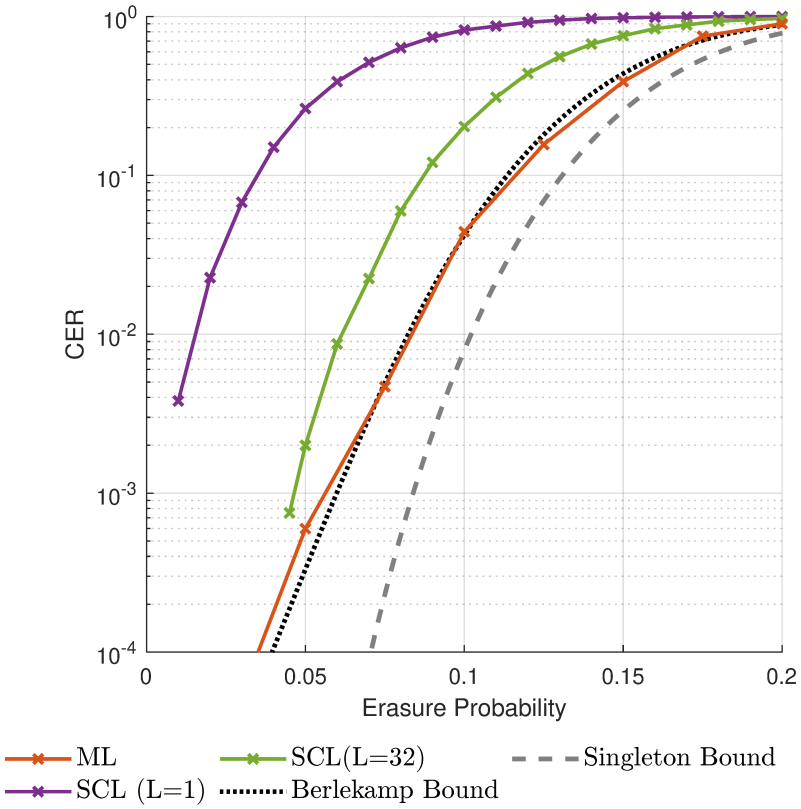


Figure 4.9: Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(120, 111)$ polar code concatenated with CRC-11.

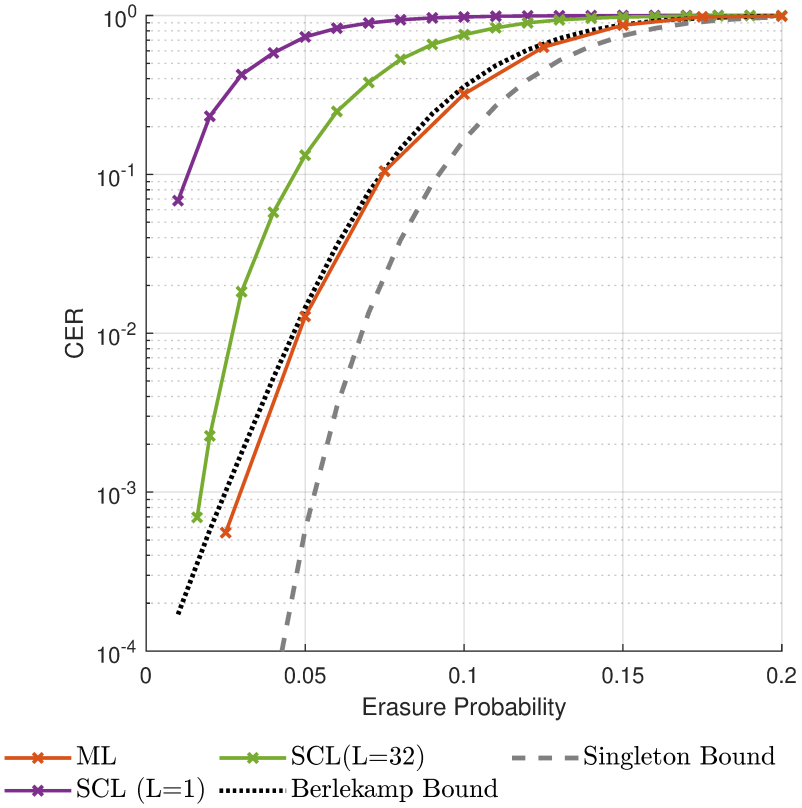


Figure 4.10: Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(114, 111)$ polar code concatenated with CRC-11.

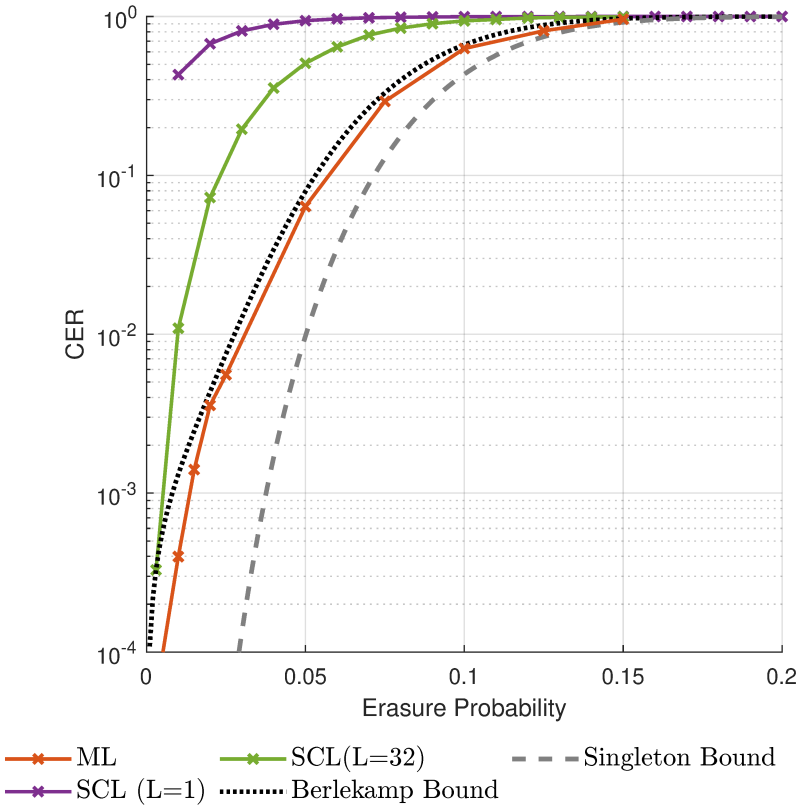


Figure 4.11: Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(111, 111)$ polar code concatenated with CRC-11, corresponding to $R = A/n \approx 0.9$.

In the high-rate regime, all decoding strategies exhibit a degradation in performance due to the reduced redundancy of the code. In this setting, even the ML decoder's performance is significantly degraded. As a consequence, the performance gap between ML and practical decoders such as SCL tends to reduce, since all schemes are ultimately limited by the inherent weakness of the code rather than the decoding strategy. In this regime, increasing the list size provides limited additional benefit, as the number of highly reliable decoding paths decreases.

To further analyze these codes, for each rate, the gap

$$\Delta = \epsilon_{\mathrm{ML}} - \epsilon_{\mathrm{SCL}} \tag{4.1}$$

was computed, where ϵ_{SCL} and ϵ_{ML} are the erasure probabilities at which each decoder attains a codeword error rate of 10^{-3}. The same definition applies to the gap with respect to the Berlekamp bound, with ϵ_{ML} replaced by ϵ_{B}.

This gap provides a measure of how close the SCL decoder with list size $L = 32$ approaches the ML performance and the Berlekamp bound. Plotting both gaps as a

function of the code rate highlights how the distance between practical decoding and the theoretical benchmarks evolves across different rates. The results are shown in Figure 4.12.

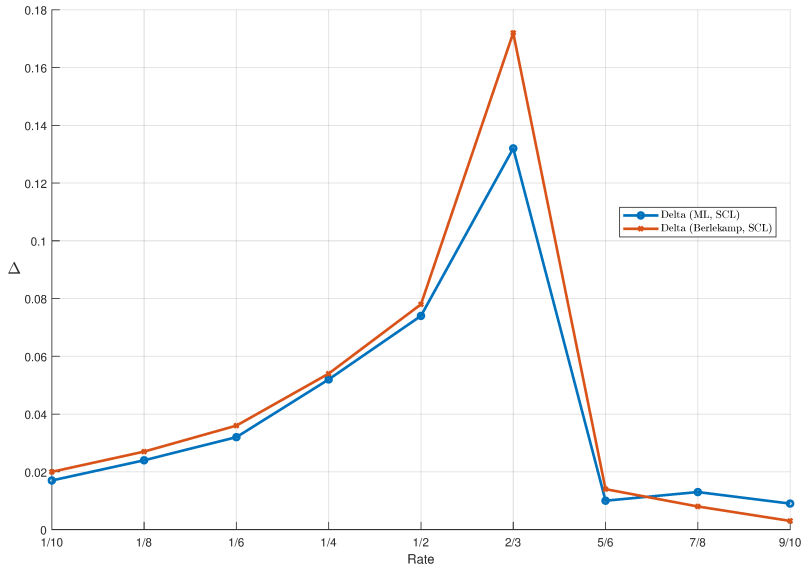


Figure 4.12: Gap Δ of (4.1) between the SCL decoder with $L = 32$ and the ML decoder, and between the SCL decoder and the Berlekamp bound, as a function of the code rate, evaluated at a codeword error rate of 10^{-3}.

As illustrated in Figure 4.12, the gap between SCL decoding and ML decoding is non-monotonic with respect to the code rate. The decoding gap peaks at medium rates, around $R = 1/2$, where a larger list size L is therefore needed to approach ML performance. Conversely, at extreme rates, high and low, the gap diminishes, indicating that SCL achieves near-ML performance with lower computational complexity. It should be noted, however, that at high rates the erasure probabilities involved are small, so that small absolute gaps may still correspond to significant relative differences. A possible explanation is that polarization provides less clear path discrimination in the medium-rate regime, although verifying this would require a dedicated analysis of the path metric distribution.

To assess whether some of the qualitative trends observed over the BEC persist over a more realistic channel, Figures 4.13 and 4.14 report the results as a function of the mutual information for a low-rate configuration, $R \approx 1/10$, and a medium-rate configuration, $R = 1/2$. SCL decoding with $L = 1$ and $L = 32$ is reported for both the BEC and the AWGN channel. ML decoding and the Singleton and Berlekamp bounds are shown for the BEC only, while for the AWGN channel the random-coding union (RCU) bound [16] is reported as a finite-blocklength achievability benchmark. Since exact ML decoding over the AWGN channel is computationally infeasible for the considered code dimensions and only two rates are considered, these results do not allow the rate dependence of the SCL-ML gap to be verified over the AWGN channel; they only indicate whether the relative behavior of the decoders is preserved.

In both configurations, the AWGN curves follow the same ordering as the BEC ones, with SCL decoding with $L = 32$ clearly outperforming SC decoding, and are shifted towards higher mutual information values. For $R \approx 1/10$, the shift is similar for both list sizes, so that the gain provided by list decoding is comparable on the two channels. For $R = 1/2$, the SCL curves with $L = 32$ nearly coincide on the two channels, while SC decoding over the AWGN channel is considerably worse than over the BEC, so that the gain provided by list decoding is larger over the AWGN channel. Moreover, on both channels the distance between SCL decoding with $L = 32$ and the corresponding achievability bound is larger for $R = 1/2$ than for $R \approx 1/10$, consistently with the trend observed in Figure 4.12. Overall, the qualitative behavior of the decoders observed over the BEC is largely preserved over the AWGN channel, which supports the use of the BEC as a tractable reference model, although a quantitative assessment would require ML benchmarks over the AWGN channel.

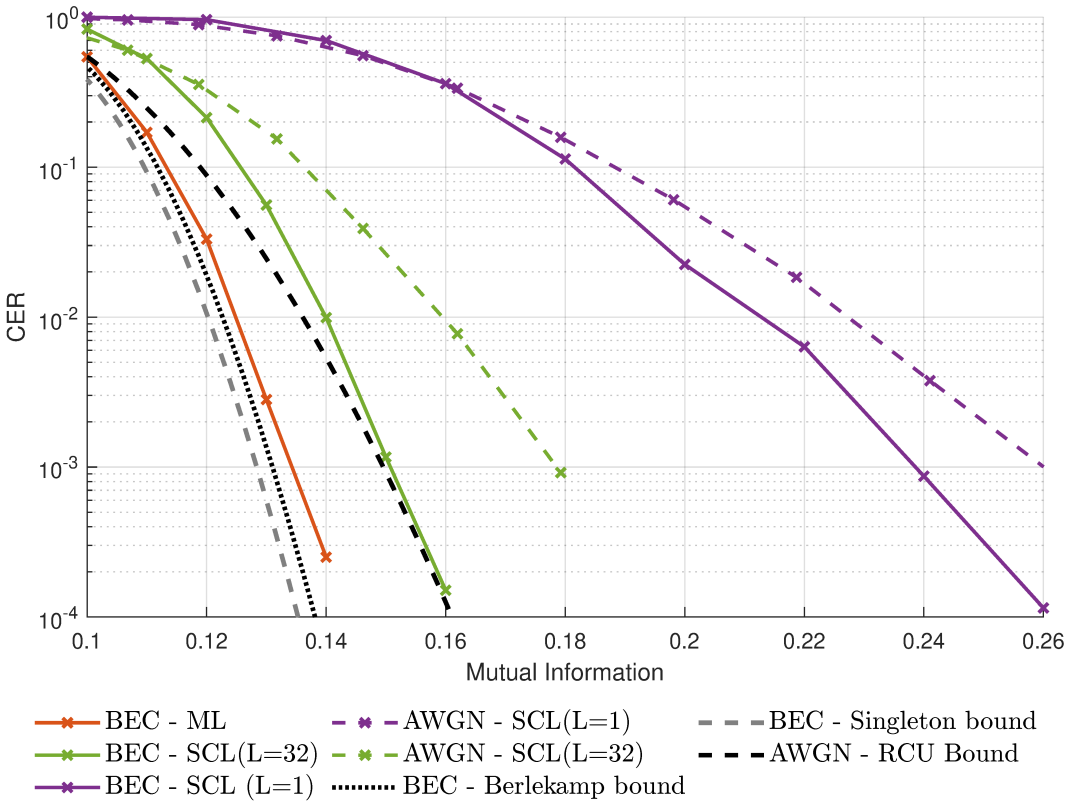


Figure 4.13: Codeword error rate as a function of the mutual information under various decoding strategies over the BEC and the AWGN channel, for a $(1024, 111)$ polar code concatenated with CRC-11.

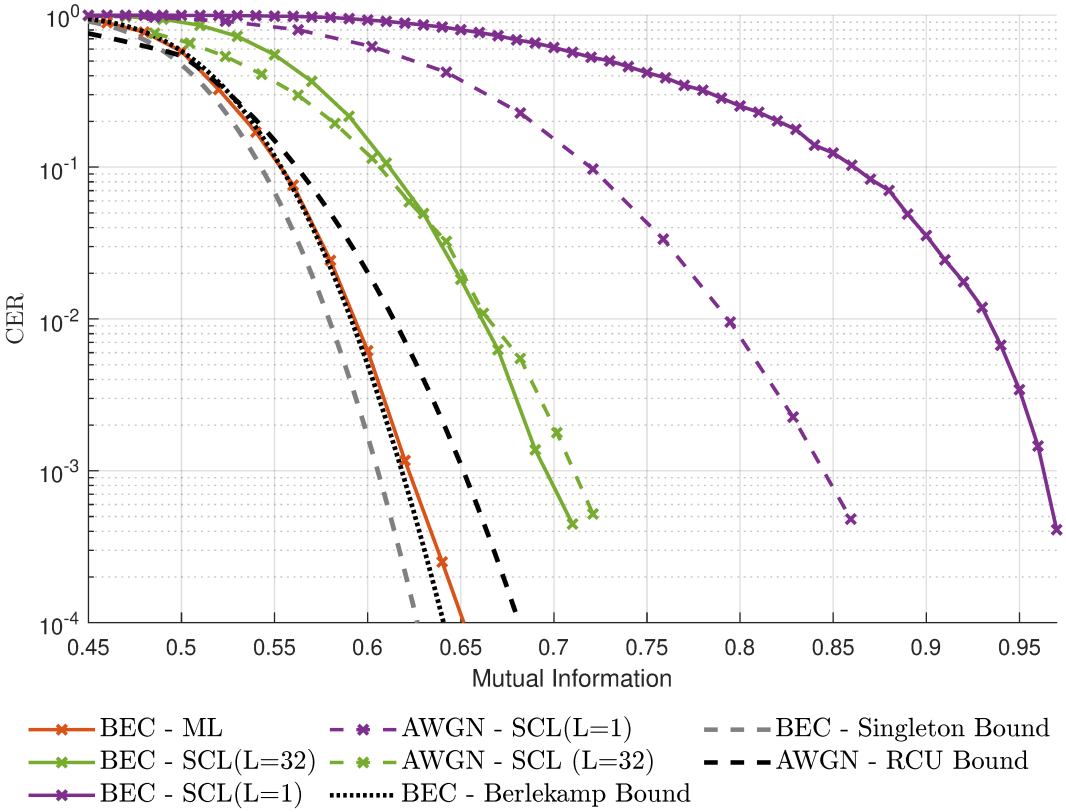


Figure 4.14: Codeword error rate as a function of the mutual information under various decoding strategies over the BEC and the AWGN channel, for a $(1024, 523)$ polar code concatenated with CRC-11.

Chapter 5

Conclusions

This work has focused on the analysis and design of low-rate polar codes for massive random access scenarios, with particular emphasis on their performance under practical decoding strategies. Polar codes, as adopted in the 5G New Radio standard, are known to achieve performance very close to the finite-length bounds under optimal ML decoding, highlighting their strong theoretical potential even in short-packet regimes.

However, in practical implementations, ML decoding is computationally infeasible, and suboptimal decoders must be employed. Among them, Successive Cancellation List (SCL) decoding represents the most widely used solution in practice. The results of this study confirm that SCL decoding is particularly effective in the low-rate regime, where it achieves performance close to optimal decoding, while the gap with respect to ML decoding is largest in the medium-rate regime. Since the focus of this work is on low-rate coding for massive random access, this behavior is especially relevant.

Despite these promising results, a significant gap remains between suboptimal decoding schemes and true ML performance. This leads to an open research question: how to design decoding algorithms that can approach ML performance while maintaining practical complexity constraints. Possible directions include the improvement of existing SCL-based architectures or the development of entirely new decoding strategies capable of better exploiting the structure of polar codes in the finite-length regime.

Addressing this gap is crucial for fully leveraging the potential of polar codes in next-generation communication systems, particularly in scenarios characterized by massive connectivity and stringent resource constraints.

Appendix

This appendix collects additional simulation results for code rates and blocklengths not discussed in Section 4.4, complementing the analysis of the decoding gap presented therein.

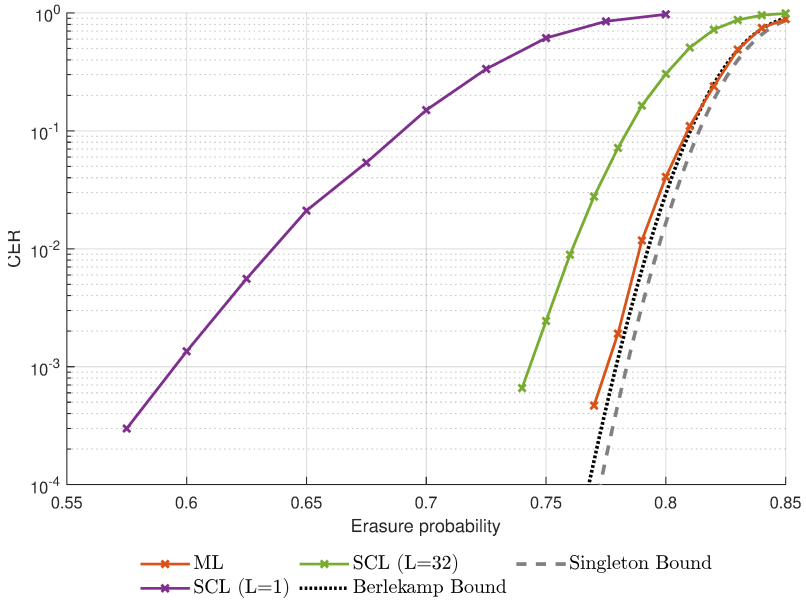


Figure A.1: Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(600, 111)$ polar code concatenated with CRC-11.

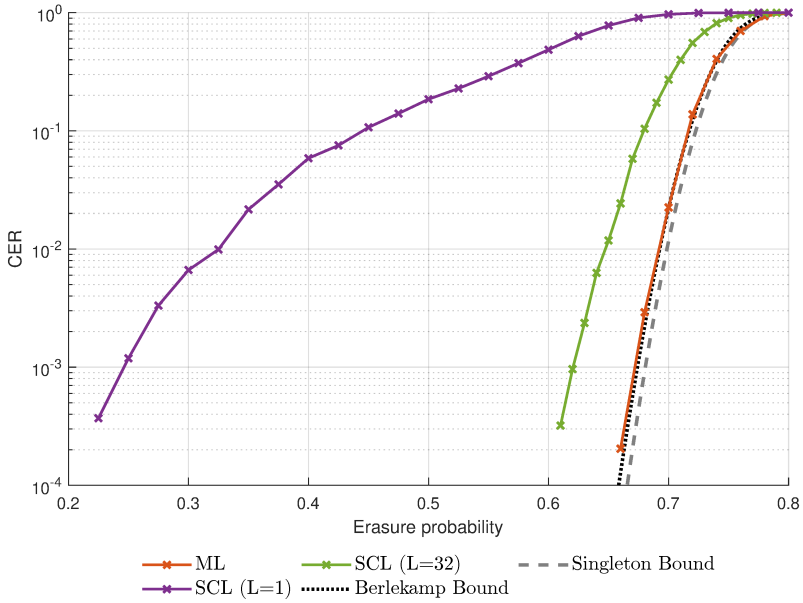


Figure A.2: Codeword error rate as a function of the erasure probability under various decoding strategies over the BEC, for a $(400, 111)$ polar code concatenated with CRC-11.

Bibliography

[1] Claude E. Shannon. "A Mathematical Theory of Communication". In: *Bell System Technical Journal* 27.3 (1948), pp. 379–423. DOI: 10.1002/j.1538-7305.1948.tb01338.x.

[2] Yury Polyanskiy. "A perspective on massive random-access". In: *2017 IEEE International Symposium on Information Theory (ISIT)*. 2017, pp. 2523–2527. DOI: 10.1109/ISIT.2017.8006984.

[3] R. Gallager. "A perspective on multiaccess channels". In: *IEEE Transactions on Information Theory* 31.2 (1985), pp. 124–142. DOI: 10.1109/TIT.1985.1057022.

[4] Erdal Arıkan. "Channel Polarization: A Method for Constructing Capacity-Achieving Codes for Symmetric Binary-Input Memoryless Channels". In: *IEEE Trans. Inf. Theory* 55.7 (2009), pp. 3051–3073.

[5] Michael G. Luby et al. "Efficient erasure correcting codes". In: *IEEE Transactions on Information Theory* 47.2 (2001), pp. 569–584. DOI: 10.1109/18.910575.

[6] Richard C. Singleton. "Maximum Distance q-Nary Codes". In: *IEEE Transactions on Information Theory* 10.2 (1964), pp. 116–118. DOI: 10.1109/TIT.1964.1053661.

[7] Elwyn R. Berlekamp. "The Technology of Error-Correcting Codes". In: *Proceedings of the IEEE* 68.5 (May 1980), pp. 564–593. DOI: 10.1109/PROC.1980.11696.

[8] Bin Li, Hui Shen, and David Tse. "A RM-Polar Codes". In: *arXiv preprint arXiv:1407.5483* (2014).

[9] Ryuhei Mori. *Properties and Construction of Polar Codes*. 2010. arXiv: 1002.3521 [cs.IT]. URL: https://arxiv.org/abs/1002.3521.

[10] Gaoning He et al. "β-expansion: A Theoretical Framework for Fast and Recursive Construction of Polar Codes". In: *GLOBECOM 2017 – 2017 IEEE Global Communications Conference*. Dec. 2017, pp. 1–6.

[11] Valerio Bioglio, Carlo Condo, and Ingmar Land. "Design of Polar Codes in 5G New Radio". In: *IEEE Communications Surveys Tutorials* 23.1 (2021), pp. 29–40. DOI: 10.1109/COMST.2020.2967127.

[12] 3GPP. *NR; Multiplexing and Channel Coding*. Technical Specification (TS) 38.212. Version 15.2.0, Release 15. 3rd Generation Partnership Project (3GPP), June 2018.

[13] Yonatan Urman and David Burshtein. "Efficient Maximum Likelihood Decoding of Polar Codes Over the Binary Erasure Channel". In: *2021 IEEE International Symposium on Information Theory (ISIT)*. 2021, pp. 1600–1605. DOI: 10.1109/ISIT45174.2021.9517748.

[14] W. W. Peterson and D. T. Brown. "Cyclic Codes for Error Detection". In: *Proceedings of the IRE* 49.1 (1961), pp. 228–235. DOI: 10.1109/JRPROC.1961.287814.

[15] Kai Niu and Kai Chen. "CRC-aided decoding of polar codes". In: *IEEE Communications Letters* 16.10 (2012), pp. 1668–1671.

[16] Yury Polyanskiy, H. Vincent Poor, and Sergio Verdú. "Channel Coding Rate in the Finite Blocklength Regime". In: *IEEE Transactions on Information Theory* 56.5 (2010), pp. 2307–2359.