Introduction to Information Theory

# Lecture 12 Part B

*Lecturer: Haim Permuter* *Scribe: Rom Hirsch and Eyal Yakir*

## I. BACKGROUND

### A. Introduction to Error-Correcting Codes

Error-correcting codes are important because they help ensure reliable communication, especially when we're dealing with channels that have a lot of noise or interference. When we send data through a communication system, it can get mixed up and corrupted due to noise and interference, which can cause errors in what we receive. Error-correcting codes are like detectives that can detect and fix these errors, making sure our data gets transmitted accurately and securely.

To comprehend error-correcting codes, it is essential to understand the notion of the channel rate. The channel rate signifies the efficiency of transmitting information over the channel and can be calculated as the ratio of the number of information bits transmitted, denoted as $k$, to the total number of bits transmitted, denoted as $n$. This includes both the information bits and any redundant bits introduced by the code. In the case of a specific code represented as $(n, k)$, where $n$ denotes the message size (in bits) and $k$ denotes the data in the message (in bits), the rate $R$ can be computed as:

$$R = \frac{k}{n} \tag{1}$$

The rate of a channel provides insights into how efficiently we can transmit the desired information, taking into account the presence of redundant bits for error correction.

**Example 1** Consider the $n$-repetition code, where a message $abc$ is encoded as $\underbrace{aa\ldots a}_{\text{n times}}\underbrace{bb\ldots b}_{\text{n times}}\underbrace{cc\ldots c}_{\text{n times}}$. We will analyze the Binary Eraser Channel, characterized by a probability of $1-p$ for correct transmission and $p$ for deletion. The channel's conditional probabilities are denoted as $p(y = i|x = i) = 1 - p$ and $p(y =?|x = i) = p$.

The Binary Eraser Channel transmits each symbol with a probability of $1 - p$ for correct transmission and $p$ for deletion. Symbol $i$ is received correctly with probability $1 - p$, while an erasure symbol is received with probability $p$.

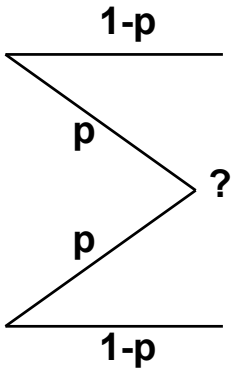


Fig. 1: channel probabilities diagram

Next, we will delve into the examination of the error probability of the code and the rate for varying values of $n$. For a given $n$, the probability of encountering an error is the probability of deletion of n bits. $p(\text{Fail}) = p^n$. As we transmit $n$ bits for every bit of data, the code corresponds to a $(n, 1)$ code, resulting in a rate of $R = \frac{1}{n}$.

## II. Polar Codes

Polar codes are a type of error-correcting code introduced by Erdal Arıkan in 2008 [1]. The concept behind polar codes is based on the idea of polarization, where a channel is transformed into subchannels with significantly different error probabilities. Polar codes leverage this polarization phenomenon to efficiently and reliably transmit information over a noisy channel. The key concept in polar codes is to identify the reliable subchannels and exploit them for transmission while effectively ignoring the unreliable subchannels. By selectively utilizing the reliable subchannels, polar codes achieve near-optimal performance in terms of error correction capability and channel capacity. Polar codes have gained significant attention as they are the first codes with an explicit proof of approaching capacity and are used in modern communication systems, such as control channels in the 5G standard.

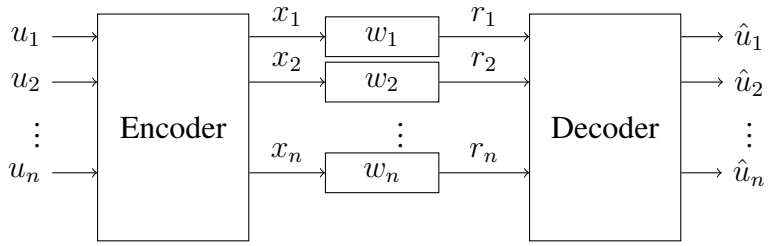


Fig. 2: A polar code diagram illustrating the encoding and decoding process.

### *A. Polar Transformation*

The channel polarization or the polarization transform is based on multiple recursive concatenations of input manipulations which transform the physical channels into virtual channels with capacities that either goes to 1 or to 0. Namely the virtual channel polarize to 0 or 1. We will present three methods to describe the polar code transformation:
1) Algebraic - Mathematically, this can be defined as:

$$\Big[x_1, \ldots, x_n\Big] = \Big[u_1, \ldots, u_n\Big] \, G_2^{\otimes \log_2(n)} \tag{2}$$

where

$$G_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \tag{3}$$

and $G_2^{\otimes n}$ is the Kronecker product of $G_2$ with itself n times define as

$$G_2^{\otimes n} = G_2 \otimes G_2 \ldots G_2 \text{ (n times)} \tag{4}$$

**Definition 1 (Kronecker Product)** The Kronecker Product of two matrices $A \in \mathbb{R}^{m \times n}, B \in \mathbb{R}^{p \times q}$ is $A \otimes B \in \mathbb{R}^{mp \times nq}$ is defined by

$$A \otimes B = \begin{bmatrix} [A]_{1,1} B & [A]_{1,2} B & \cdots & [A]_{1,n} B \\ [A]_{2,1} B & [A]_{2,2} B & \cdots & [A]_{2,n} B \\ \vdots & \vdots & \ddots & \vdots \\ [A]_{m,1} B & [A]_{m,2} B & \cdots & [A]_{m,n} B \end{bmatrix}. \tag{5}$$

**Example 2** Algebraic polar code transformation n=2,4,8

$$\begin{bmatrix} x_1 & x_2 \end{bmatrix} = \begin{bmatrix} u_1 & u_2 \end{bmatrix} \underbrace{\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}}_{G_2} = \begin{bmatrix} u_1 \oplus u_2 & u_2 \end{bmatrix} \tag{6}$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}}_{G_2^{\otimes 2}} = \begin{bmatrix} u_1 \oplus u_2 \oplus u_3 \oplus u_4 \\ u_2 \oplus u_4 \\ u_3 \oplus u_4 \\ u_4 \end{bmatrix}^T \tag{7}$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \\ u_8 \end{bmatrix} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}}_{G_2^{\otimes 3}} = \begin{bmatrix} u_1 \oplus u_2 \oplus u_3 \oplus u_4 \oplus u_5 \oplus u_6 \oplus u_7 \oplus u_8 \\ u_2 \oplus u_4 \oplus u_6 \oplus u_8 \\ u_3 \oplus u_4 \oplus u_7 \oplus u_8 \\ u_4 \oplus u_8 \\ u_5 \oplus u_6 \oplus u_7 \oplus u_8 \\ u_6 \oplus u_8 \\ u_7 \oplus u_8 \\ u_8 \end{bmatrix}^T \tag{8}$$

2) Tree - The polar code transformation can be described using a tree. For the construction using a tree: Starting with the leaf nodes $u_1, \ldots, u_n$, each parent node above is calculated using the operation:

$$parent_{node} = [child_1 \oplus child_2, child_1]$$

This operation is applied recursively until we reach the root, which consists of $n$ bits $x_1, \ldots, x_n$.

**Example 3** Tree Polar Transformation $n = 2, 4, 8$

The transformation is performed through the mapping $(u_1, u_2) \rightarrow (u_1 \oplus u_1, u_1)$, as shown in Fig. 3 as a tree and Fig.

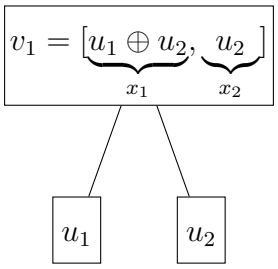


Fig. 3: Polar code tree for $n = 2$

Polar Transformation $n = 4$ - Initially, we calculate $v_1$ and $v_2$ similar to $n = 2$ transformation, and then we apply the same operation recursively with $v_1$ and $v_2$.

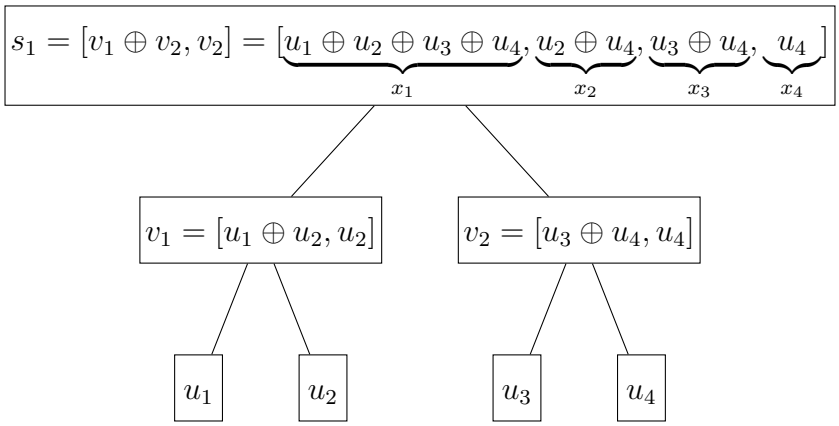


Fig. 4: Polar code tree for $n = 4$

Polar Transformation $n = 8$: The transformation is similarly structured, recursively applying the operation using the tree shown in Fig. 5.

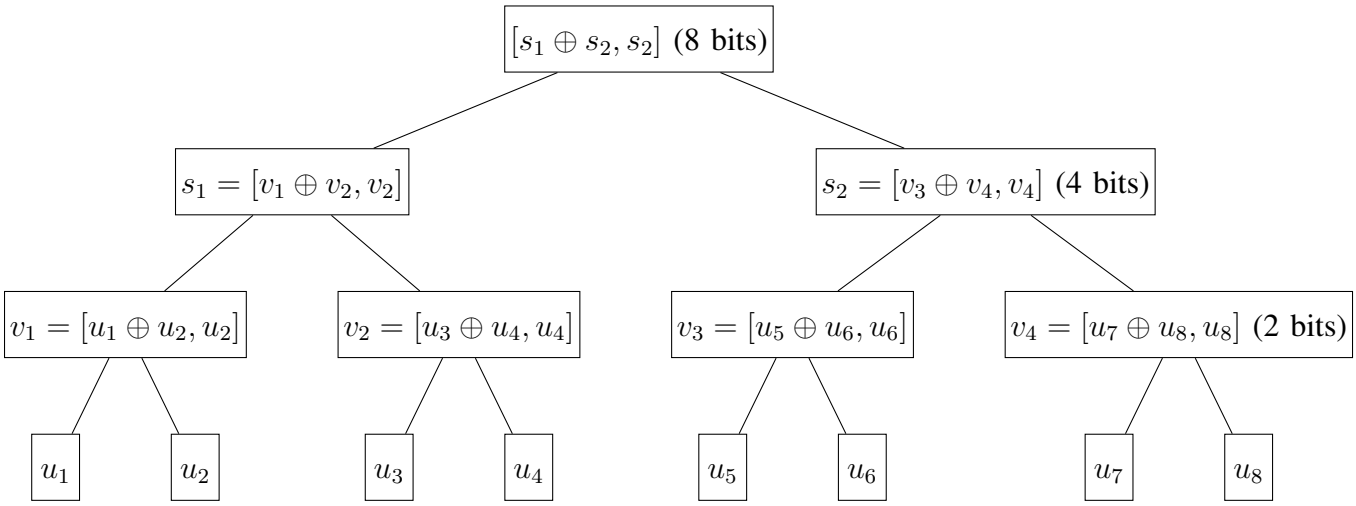


Fig. 5: Polar code tree for $n = 8$

3) Block Diagram Polar Transformation - Figure 6, Figure 7, and Figure 8 show the recursive block diagram representations for polar transformation.

**Example 4** Block Diagram Polar Transformation n=2,4,8

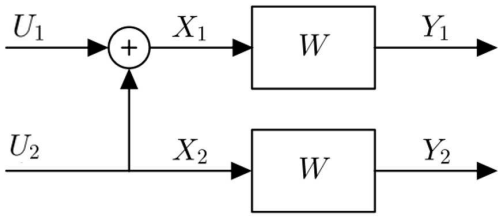


Fig. 6: Encoding scheme for $n = 2$

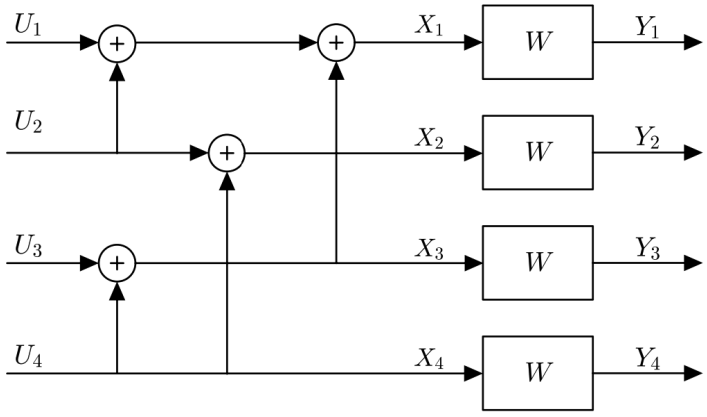


Fig. 7: Encoding scheme for $n = 4$

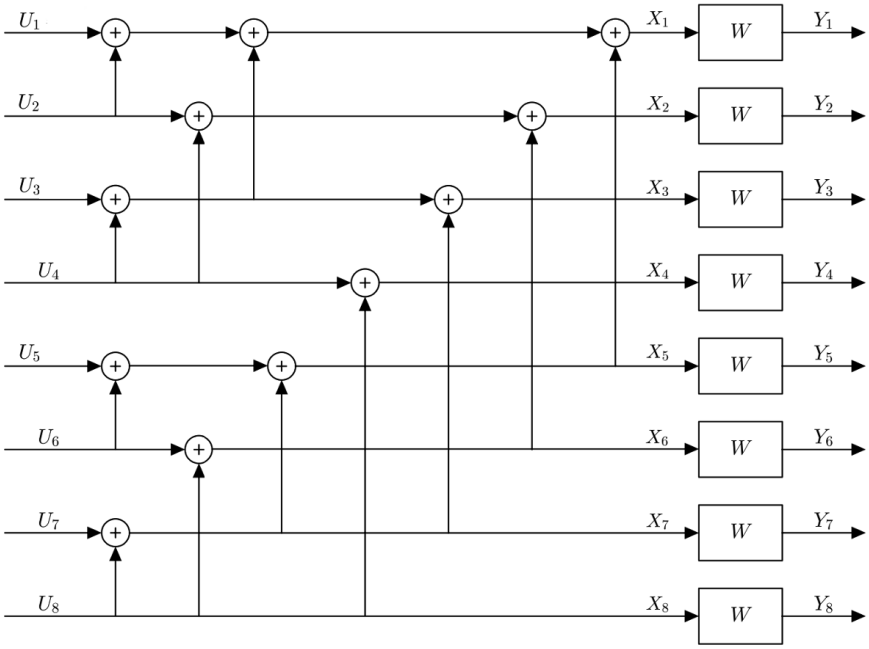


Fig. 8: Encoding scheme for $n = 8$

### *B. Encoder*

The encoder transforms a $k$-bit input into an $n$-bit output where $k < n$. This process involves assigning values based on channel reliability, where 'good' channels carry data

and 'frozen' channels remain constant, and encode it using polar transformation.

*Polar Code Encoding Algorithm:*

1) **Initialization:** Begin by constructing a $2^n \times 2^n$ polarizing matrix $G_n$ using the Kronecker power of the basic polarizing matrix, $G_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$. For instance, for $n = 3$, $G_3$ becomes an $8 \times 8$ matrix.
2) **Channel Selection:** Once $G_n$ is formed, select the top $k$ channels based on their reliability. These 'good' channels are prioritized for data transmission, while the remaining $n - k$ channels are designated as 'frozen' bits.
3) **Input Construction:** Construct the input vector $u_1^n$. Dispatch data bits onto the top $k$ channels and set the remaining $n - k$ channels to 0, representing frozen bits.
4) **Data Encoding:** Encode the input vector $u_1^n$ by multiplying it with the polarizing matrix $G_2^{\otimes log_2(n)}$. This multiplication generates the encoded output vector $x_1^n$, which contains $n$ bits.

**Example 5** Let us illustrate the polar code encoding process using an example where $n = 8$ and $k = 4$. In this case, our system employs four information bits and four frozen bits.

For a Polar code with $n = 8$, we need to construct the generator matrix $G_8$. This is achieved by performing the Kronecker product on the base matrix $G = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ three times, because $n = 2^3$.

$$G_8 = G_2^{\otimes 3} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{9}$$

Given the ranking of the channels, the indices of the frozen channels are $1, 2, 3, 5$. Suppose the input bit sequence is $u = [f, f, f, d_1, f, d_2, d_3, d_4]$, where $f$ represents a frozen bit (which is set to $0$), and $d$ denotes an information bit.

The encoding process comprises the multiplication of the information bit sequence $u$ with the generator matrix $G_8$ to yield the codeword $x = uG_8$.

$$
\begin{aligned}
x = uG_8 &= \begin{bmatrix} f & f & f & d_1 & f & d_2 & d_3 & d_4 \end{bmatrix} \times G_8 \\
&= \begin{bmatrix} d_1+d_2+d_3+d_4 & d_2+d_4 & d_1+d_3+d_4 & & & & & \\ & & & d_4 & d_2+d_3+d_4 & d_2+d_4 & d_3+d_4 & d_4 \end{bmatrix}
\end{aligned}
$$

After obtaining the codeword $x$, it is transmitted over the communication channel.

### *C. Successive Cancellation decoder*

Consider the recursive SC decoder for a polar transform of length n, our goal is to generates an estimate $\hat{u}_1^n$ of $u_1^n$ by observing the channel output $r_1^n$.We will start by showing the SC decoder for simple cases of polar codes for $n = 2$ and $n = 4$. Then, we will present the general code for (n,k).

*MAP/LLR Decoding:* When analyzing the transmission across a channel affected by noise, it becomes necessary to ascertain the original transmitted bit (0/1) from the received analog signal. The decoding procedure involves transforming the received signal into its digital representation through the utilization of the soft decision concept. Given the received bit $r$ the soft decision can be calculated as follows

$$l_u = \left( \frac{P(u=1|r)}{P(u=0|r)} \right) \tag{10}$$

where $P(u = i|r)$ is the posterior probability of $u = i$ given the channel output $r$, and therefore $l_u$ is the log of posterior ration of $u$.

Using Bayes rule, it can be written as

$$\begin{aligned} l_u &= \log\left(\frac{P(r|u=0)P(u=0)}{P(r|u=1)P(u=1)}\right) \\ &= \log\left(\frac{P(r|u=0)}{P(r|u=1)}\right) + \log\left(\frac{P(u=0)}{P(u=1)}\right) \\ &= LLR(u) + L(u)^{\text{prior}} \end{aligned} \tag{11}$$

where the first term is the log-likelihood ratio ($LLR$) of the bit $u$, and the second term is the prior ratio of $u$, In many cases in communication, the bits are uniformly distributed, i.e. $P(u=1) = P(u=0)$ and therefore

$$l_u = LLR(u) \tag{12}$$

It can be seen that the sign of the $l_u$ represents the hard decision of bits. If it is positive then the bits is 0, otherwise it is 1. The magnitude of the $LLR$ represent the reliability of that decision.

**Example 6** Let $U$ be Phase Shift Keying (BPSK) in which 0 is mapped to 1, and 1 is mapped to -1, and the channel output is Additive White Gaussian Noise (AWGN) given by $R = U + Z$ where $Z \sim \mathcal{N}(0, \sigma^2)$, the $LLR$ is equal to

$$\begin{aligned} LLR(u) &= \log\left(\frac{P(r|u=0)}{P(r|u=1)}\right) \\ &\overset{\text{(Gaussian)}}{=} \log\left(\frac{\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(r-1)^2}{2\sigma^2}\right)}{\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(r+1)^2}{2\sigma^2}\right)}\right) \\ &= \frac{2r}{\sigma^2} \\ &= r \cdot (\text{positive factor}) \end{aligned} \tag{13}$$

**Lemma 1** Repetition code $LLR$ - By extension, for n inputs $x_1^n$ (repetition code $u =$

$x1, \ldots, u = x_n$) we get:

$$\begin{aligned} LLR(u) &= \log \frac{P(u=1|r_1, r_2, ..., r_n)}{P(u=0|r_1, r_2, ..., r_n)} \\ &= \log \left( e^{(r_1+r_2+\cdots+r_n)\cdot\frac{2}{\sigma^2}} \right) \\ &= (r_1 + r_2 + \cdots + r_n) \cdot \frac{2}{\sigma^2} \\ &= (r_1 + r_2 + \cdots + r_n) \cdot \text{(positive factor)} \end{aligned} \tag{14}$$

**Lemma 2** Parity check $LLR$ - The $LLR(u)$ for the case that $u = x_1 \oplus x_2$, and the channel system (BPSK) describe in Fig. 9 is

$$LLR(u) = 2\tanh^{-1}(\tanh\left(\frac{r_1}{2}\right)\tanh\left(\frac{r_2}{2}\right)) \approx \text{sign}(r_1)\text{sign}(r_2)\min(|r_1|, |r_2|) \tag{15}$$

The proof is given in Appendix

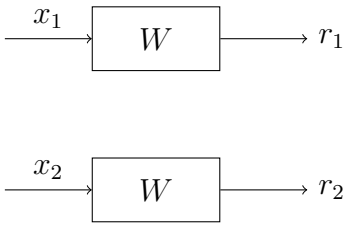


Fig. 9: diagram problem example $u = x1 \oplus x2$

**Example 7 SC polar code decoding for n=2 with BPSK + AWGN**

Let's illustrate the SC polar code decoding for $n = 2$ with BPSK + AWGN. We apply the polar encoder $G_2$ and send $x_1, x_2$ through the channel, receiving $r_1$ and $r_2$(Fig. 10). Now, we want to decode $u_1, u_2$.

The stages for SC decoding n=2 are:

1) Calculate $LLR(\hat{u}_1)$, we know that $\hat{u}_1 = x_1 \oplus x_2$ therefore from equation 20 (lemma 2 Parity check LLR)the LLR is $LLR(\hat{u}_1) = f(r_1, r_2)$.

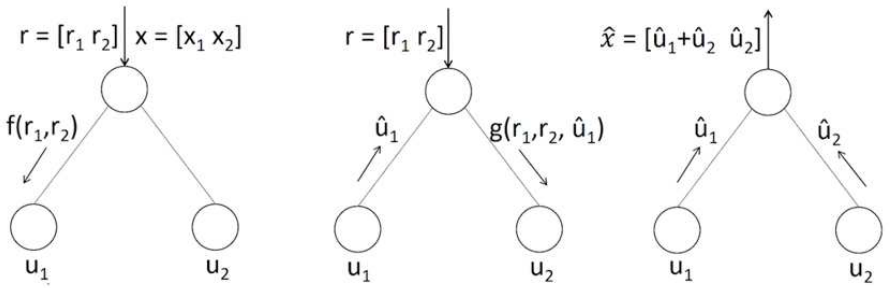


Fig. 10: Polar code decoding steps for $n = 2$ (from right to left: step 1, 2, 3)

2) Calculate $\hat{u}_1$ using the threshold method: if $f(r_1, r_2) < 0$, then $\hat{u}_1 = 1$; otherwise, $\hat{u}_1 = 0$ and send $\hat{u}_1$ to the right side. Next, calculate $LLR(\hat{u}_2)$ (repetition code equation 14 lemma 1 Repetition code):

$$LLR(\hat{u}_2) = g(r_1, r_2, \hat{u}_1) = \begin{cases} r_2 + r_1 & \text{if } \hat{u}_1 = 0 \\ r_2 - r_1 & \text{if } \hat{u}_1 = 1 \end{cases} = r_2 + (1 - 2\hat{u}_1) \cdot r_1$$

3) Calculate $\hat{u}_2$ using the threshold method: if $g(r_1, r_2, \hat{u}_1) < 0$, then $\hat{u}_2 = 1$; otherwise, $\hat{u}_2 = 0$.

**Example 8 SC decoder for n=4 BPSK + AWGN**

In this case, the steps are:

1) Step 1: Calculate the LLRs for the bits $[x_{11}, x_{12}, x_{21}, x_{22}] = [u_1 \oplus u_2 \oplus u_3 \oplus u_4, u_2 \oplus u_4, u_3 \oplus u_4, u_4]$ based on received LLRs $(r_1, r_2, r_3, r_4)$.
2) Step 2: Calculate $LLR(v_1)$ (left side) where $v_1 = [u_1 \oplus u_2, u_2]$. Use $LLR(u_1 \oplus u_2) = f(r_1, r_3)$ and $LLR(u_2) = f(r_2, r_4)$.
3) Steps 3-5: Similar to SC polar decoder for $n = 2$ starting with $[f(r_1, r_3), f(r_2, r_4)]$ instead of $[r_1, r_2]$.
4) Steps 6-7: Calculate $\hat{u}_2$ and $\hat{v}_1$ and send it back to the parent node.
5) Step 8: The right side is a repetition code, so calculate $LLR(v_2)$ where $v_2 = [u_3 \oplus u_4, u_4]$:

$$LLR(v_2) = [g(r_1, r_3, \underbrace{u_1 \oplus u_2}_{v_{11}}), g(r_2, r_4, \underbrace{u_2}_{v_{12}})]$$

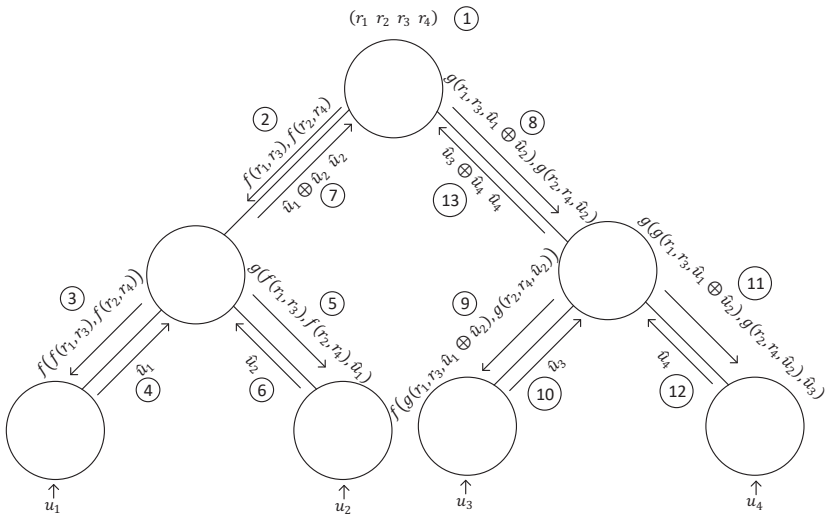


Fig. 11: SC Polar code n=4 diagram

6) Steps 9-12: Similar to SC decoder for $n = 2$ same as Step 3-5

***Successive Cancellation (SC) Decoder for a General (n,k) Polar Code:*** Having discussed the SC decoder for $n = 2, 4$, we now define it for a general $(n, k)$ code. We will start by demonstrating the operations at interior nodes as shown in Fig. 12.

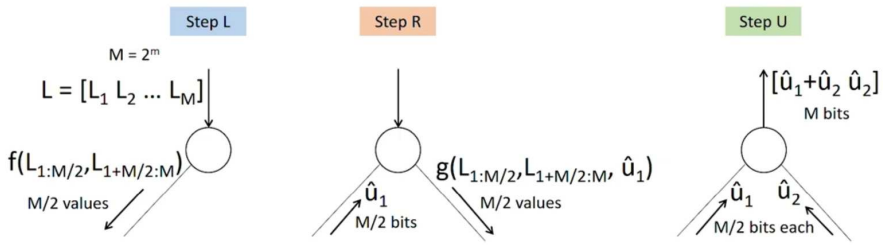


Fig. 12: SC decoder: operations in an interior node

**Step L**: This step comes into operation when the node receives a set of LLRs $L = [L_1, L_2, \ldots, L_M]$ from the parent. In this step, $L$ is sent to the left child, and the LLRs are calculated using $f(L_{1:M/2}, L_{(M/2+1):M})$. Here, the vector $L$ is divided into two halves: $L_{1:M/2}$ and $L_{(M/2+1):M}$. The computed $M/2$ LLR values are then sent to the left child

node.

$$f(a_{1:p}, b_{1:p}) = [f(a_1, b_1), f(a_2, b_2), \ldots, f(a_p, b_p)] \tag{16}$$

$$f(r_1, r_2) = \text{sgn}(r_1) \cdot \text{sgn}(r_2) \cdot \min(|r_1|, |r_2|) \tag{17}$$

**Step R**: This step is activated when it receives decisions $\hat{u}_i$ from the left child. Once the left child completes its processing, it sends the bits back to its parent. In this step, the decisions and $L$ are combined using the function $g$. It computes $M/2$ LLRs and sends them to the right child node.

$$g(a_{1:p}, b_{1:p}, c) = [g(a_1, b_1, c), g(a_2, b_2, c), \ldots, g(a_p, b_p, c)] \tag{18}$$

$$g(r_1, r_2, b) = r_2 + (1 - 2b)r_1 \tag{19}$$

**Step U**: This step comes into operation when the parent receives decisions $\hat{u}_i$ from the right child. With the decisions already received from the left child, it combines these decisions and sends them back up to the parent node.

**Decision in a leaf node**:

if $i$ is a frozen position: $\hat{u}_i = 0$ (ignoring $LLR(u_i)$)

if $i$ is a message position: $\hat{u}_i = 0$ if $LLR(u_i) \geq 0$; $\hat{u}_i = 1$ if $LLR(u_i) < 0$

**Sequence of operations**:

The decoder works in a certain sequence, starting at the root. At each node, the following steps are performed (when it is activated (receives LLRs)):

- If not a leaf, do the following in sequence:
  - Perform Step L and go to the left child.
  - When a decision is received from the left child, perform Step R and go to the right child.
  - When a decision is received from the right child, perform Step U and go to the parent.

- If a leaf, make a decision (Decision in a leaf node) and go to the parent.

**Example 9 (16,10) Polar Code using the sequence of operations for general (n,k)** Fig. 13 presents the SC polar decoding for a (16,10) code. Initially, there are 16 LLRs ($r^{(16)}$) that the root receives from the channel. It performs Step L in its first step, it goes to the left child, activating the left child node and sending only 8 LLRs using Step L. This is shown in the tree as ( $\underbrace{1}_{sequence}$ $\underbrace{L}_{step}$ : $\underbrace{L^{(8)}}_{LLR's}$) in the line between the nodes. Sequences 2-4 are again Step L (whenever a node gets activated, it always performs Step L first). We continue performing Step L until we arrive at the leaf, then we make a decision at the leaf and send it back. Then, in sequence 5, Step R is performed, and the decision is sent back (Step U is not shown in the diagram, but it is assumed to operate after receiving the two leaf decisions under the activated node). The process then continues in the same way as I described.

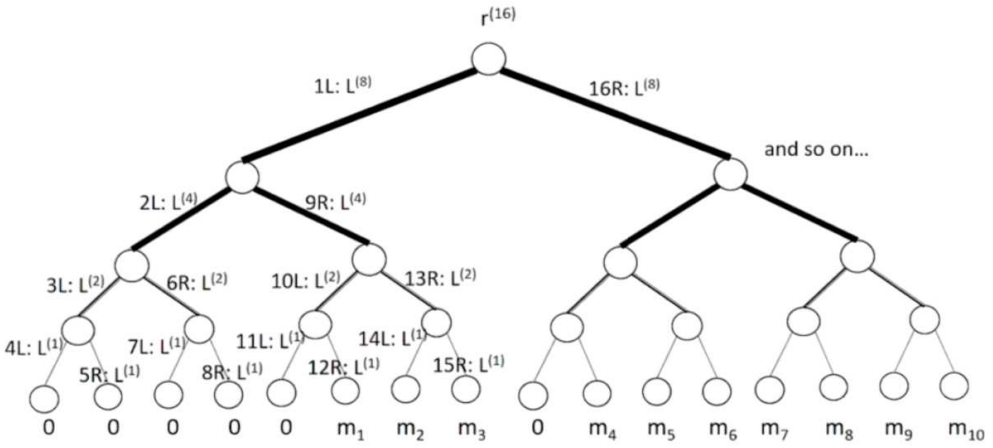


Fig. 13: SC decoder (16,10) example sequence of operations. The sequence, step and LLR's size shows is shown in the line between the nodes( $\underbrace{1}_{sequence}$ $\underbrace{L}_{step}$ : $\underbrace{L^{(8)}}_{LLR's}$) .

### *D. Successive Cancellation List Decoding (SCL)*

The motivation behind this method is to improve the performance of Successive Cancellation (SC) decoding by aiming for the maximum likelihood path. Instead of

producing a single path estimate of $\hat{u}_i, \ldots, \hat{u}_n$ which may contain errors, list decoding generates a list of possible paths. To choose the best path from the list, methods such as the sum of Log-Likelihood Ratios (LLRs) or Cyclic Redundancy Checks (CRC) can be used. Although this method adds complexity, it provides significant improvements in performance.

**List Algorithm** - The SCL decoder utilizes a parameter $L$, called the list size (always a power of two). A SCL decoder keeps track of several decoding results instead of just one, in fact for L = 1 we obtain the SC again. Instead of deciding to set the value of $u_u$ , it takes both options. i.e. when the polar SC decoder makes decisions, it considers both possible values for each bit and assigns a Decision Metric (DM) as follows:

$$\text{If } L(u_i) \geq 0 : \begin{cases} \hat{u}_i = 0 & \text{has } DM_i = 0 \\ \hat{u}_i = 1 & \text{has } DM_i = |L(u_i)| \end{cases}$$

$$\text{If } L(u_i) < 0 : \begin{cases} \hat{u}_i = 1 & \text{has } DM_i = 0 \\ \hat{u}_i = 0 & \text{has } DM_i = |L(u_i)| \end{cases}$$

Then we save the all optional paths. We repeat this process every time a decision is made, until we have more paths than the desired list size ($L$). We then select the $L$ paths according to the Path Metrics (sum of DM on a path of choices) and prune the others (discard the paths with high Path metric). This process is repeated until the decision for $u_n$ is made. Finally, we choose the path with the smallest Path Metric to be our estimated information data.

**Example 10** Let us try now to make an example for $L = 4$ and following Fig. 14. we assume $n = 4$ and all bits are unforzen. In (a) teh decoding starts and the first bit can be either 0 or 1. In the second step (b) bit assumes either 0 or 1 thus the possible words are 00, 01, 10, 11 but the nomber of path is not greater then $L = 4$ so we don't need to prune, yet. (c) shows the all possible options for the first, second and third bit but now the paths are 8 so we must keep track of only the L = 4 most likely paths (with lowest Path metrics). (d) shows that we keep only the words 010, 011, 100, 111. Decoding list

algorithm continues for the fourth bit in (e); however, the paths are 8 which is too much so it prunes for the best L = 4 paths in (f). Finally, we obtain the codewords 0100, 0110, 0111, 1111 , In the last state, we determine our estimate $\hat{u}_1^n$ using either path metrics or CRC (if CRC parity bits are added).

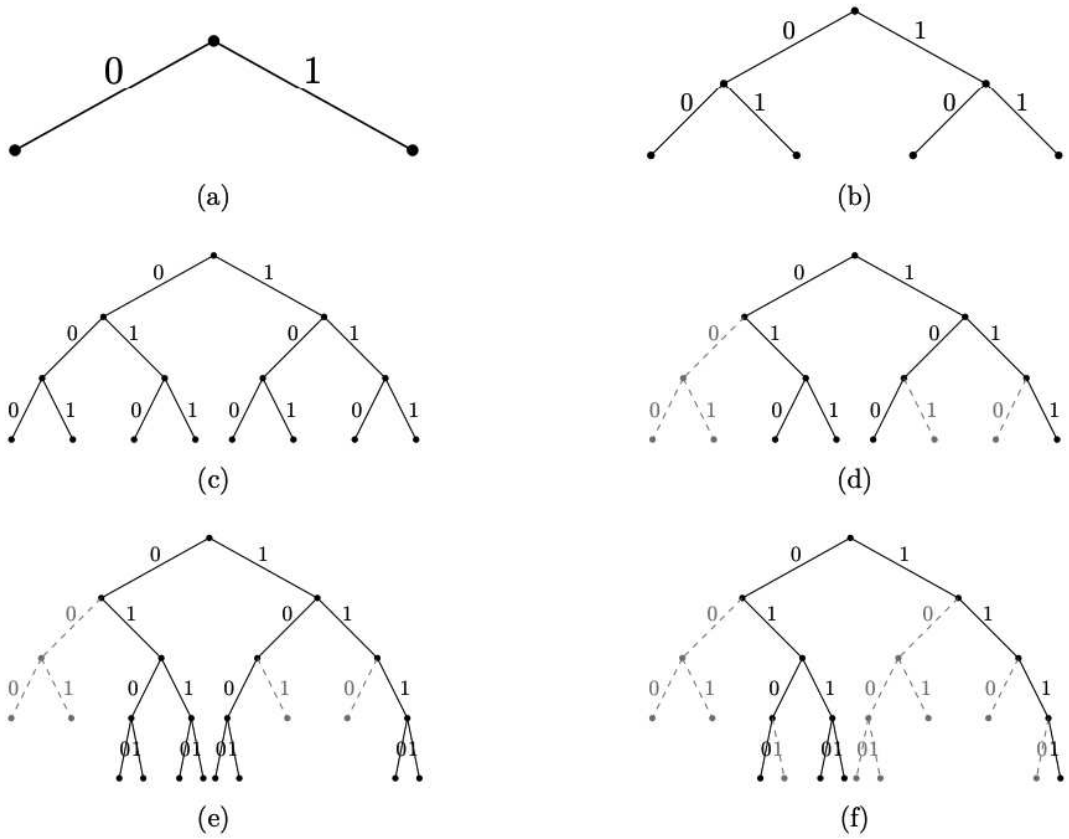


Fig. 14: SCL decoding - Evolution of decoding paths for L = 4

## REFERENCES

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

[2] NPTEL-NOC IITM, *Nptel-noc iitm youtube channel*, Accessed: 2024-07-16, 2024. [Online]. Available: https://www.youtube.com/user/nptelhrd.

## III. APPENDIX

**Proof of lemma 1**

We prove that the $LLR(c_1)$ for the case that $c_1 = c_2 \oplus c_3$ and the channel system (BPSK) described in Fig. 9 is:

$$l_1 = 2\tanh^{-1}\left(\left(\tanh(\frac{l_2}{2})\right)\tanh\left(\frac{l_3}{2}\right)\right) \tag{20}$$

Given $c_1 = c_2 \oplus c_3$, let us define the LLRs as follows:

$$l_2 = \log\left(\frac{\underbrace{P_r(c_2 = 0|r_2)}_{P_2}}{\underbrace{P_r(c_2 = 1|r_2)}_{1-P_2}}\right)$$

$$l_3 = \log\left(\frac{\underbrace{P_r(c_3 = 0|r_3)}_{P_3}}{\underbrace{P_r(c_3 = 1|r_3)}_{1-P_3}}\right)$$

Given $P_2$ and $P_3$, and knowing that $c_1 = c_2 \oplus c_3$, we have:

$$P_1 = P_2P_3 + (1-P_2)(1-P_3)$$

$$1 - P_1 = P_2(1-P_3) + (1-P_2)P_3$$

Therefore:

$$P_1-(1-P_1) = P_2(P_3-(1-P_3))+(1-P_2)((1-P_3)-P_3) = (P_2-(1-P_2))(P_3-(1-P_3))$$

$$l_1 = \log\left(\frac{P_1}{1-P_1}\right)$$

$$P_1 - (1-P_1) = (P_2 - (1-P_2))(P_3 - (1-P_3))$$

$$\frac{P_1 - (1-P_1)}{P_1 + (1-P_1)} = \frac{P_2 - (1-P_2)}{P_2 + (1-P_2)} \cdot \frac{P_3 - (1-P_3)}{P_3 + (1-P_3)}$$

$$\frac{1 - \frac{1-P_1}{P_1}}{1 + \frac{1-P_1}{P_1}} = \frac{1 - \frac{1-P_2}{P_2}}{1 + \frac{1-P_2}{P_2}} \cdot \frac{1 - \frac{1-P_3}{P_3}}{1 + \frac{1-P_3}{P_3}}$$

Using the fact that:

$$\tanh(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}}$$

We can write:

$$\frac{1 - e^{-l_1}}{1 + e^{-l_1}} = \frac{1 - e^{-l_2}}{1 + e^{-l_2}} \cdot \frac{1 - e^{-l_3}}{1 + e^{-l_3}}$$

Therefore:

$$\tanh\left(\frac{l_1}{2}\right) = \tanh\left(\frac{l_2}{2}\right) \cdot \tanh\left(\frac{l_3}{2}\right) \implies l_1 = 2\tanh^{-1}\left(\tanh\left(\frac{l_2}{2}\right) \cdot \tanh\left(\frac{l_3}{2}\right)\right)$$

Since $\tanh$ is an odd function, if $x < 0$ then $\tanh(x) < 0$, otherwise $\tanh(x) > 0$. Thus, the sign is:

$$\text{sign}(l_1) = \text{sign}(l_2) \cdot \text{sign}(l_3)$$

For the absolute value:

$$\tanh\left(\frac{|l_1|}{2}\right) = \tanh\left(\frac{|l_2|}{2}\right) \cdot \tanh\left(\frac{|l_3|}{2}\right)$$

Taking the logarithm:

$$\log(\tanh\left(\frac{|l_1|}{2}\right)) = \log(\tanh\left(\frac{|l_2|}{2}\right)) + \log(\tanh\left(\frac{|l_3|}{2}\right))$$

Let $f(x) = |\log(\tanh(|x|/2))|$, and note that $f^{-1}(x) = f(x)$. Therefore:

$$f(|l_1|) = f(|l_2|) + f(|l_3|) \implies |l_1| = f(f(|l_2|) + f(|l_3|)) \approx \min(|l_2|, |l_3|)$$

Thus, we obtain:

$$l_1 = 2\tanh^{-1}\left(\tanh\left(\frac{l_2}{2}\right)\tanh\left(\frac{l_3}{2}\right)\right) \approx \textbf{sign}(l_2)\textbf{sign}(l_3)\min(|l_2|,|l_3|)$$