POLAR CODES

Presented By :

LAB_GROUP 5

PROJECT_GROUP 1

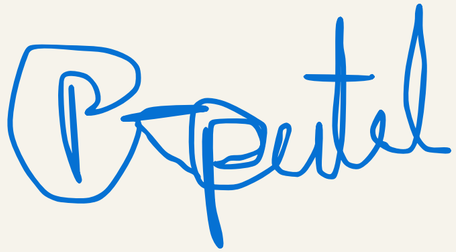
HONOR CODE

- We declare that

→ The work that we are presenting is our own work.
→ We have not copied the work (the code, the results, etc.) that someone else has done.
→ Concepts, understanding and insights we will be describing are our own.
→ We make this pledge truthfully. We know that violation of this solemn pledge can carry grave consequences.

- Signed by: all the members of the project group in the following slide.

MEMBERS OF THE GROUP

- DHWANI JOSHI 
- UTSAV PANSURIYA 
- HARSHIL KAPADIYA 
- MEET ZALAVADIYA 
- MAUNIL MODI 
- MESHV PATEL 
- JINAY VORA 
- MALAY SIDAPARA 
- SHRAVAN MAKWANA
- NEEL TANDEL 
- PARV PATEL 

OVERVIEW

- Introduction to Polar codes
- What is the idea behind polar codes?
- The concept of Encoding using generator matrix
- How to Decode?

INTRODUCTION TO POLAR CODES

- Polar codes were developed by Erdal Arikan in 2008.
- It has modest encoding and decoding complexity of **O(n logn)**
- It uses the idea of polarization which transforms the set of independent channels into a sequence of least reliable to most reliable.
- It is a linear block error-correcting code. It used for the 5G NR interface.

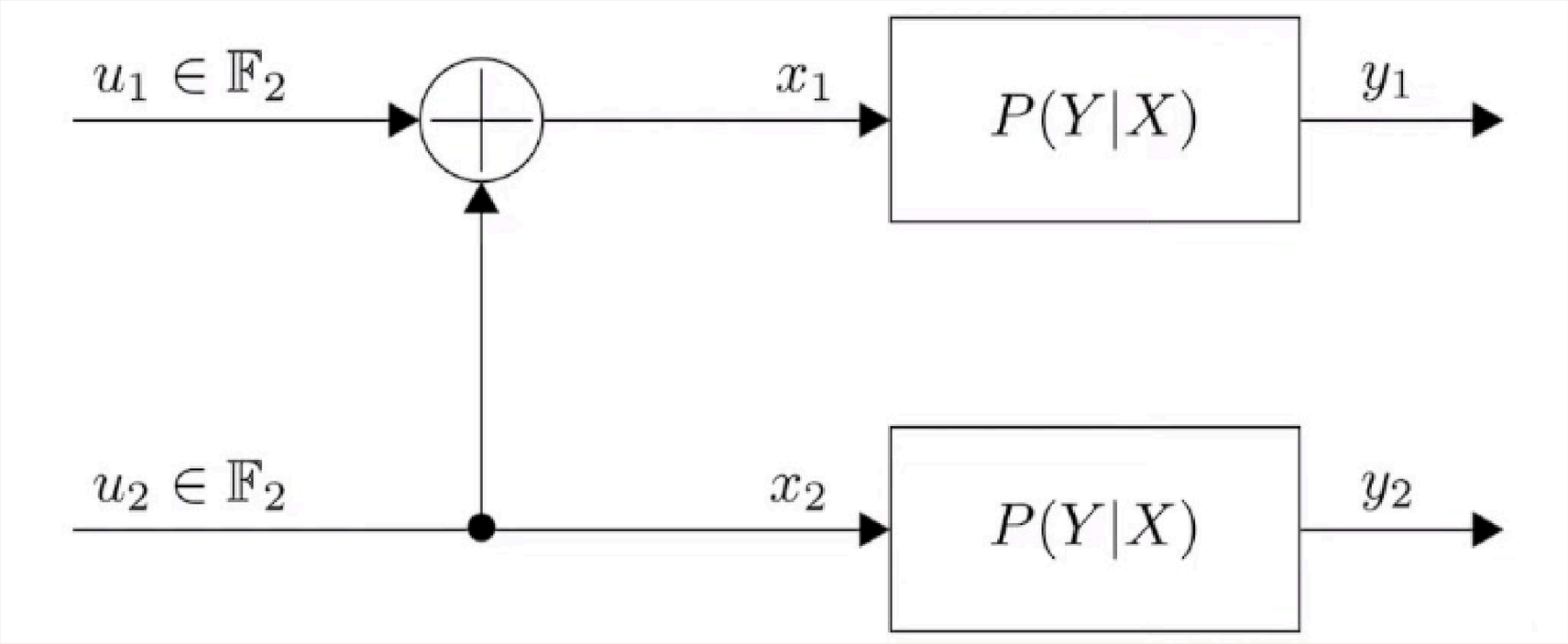
WHY TO USE POLAR CODES?

Polar codes have the property of achieving the Shannon capacity limit of a binary-input discrete memoryless channel.

Polar codes have low-complexity decoding algorithm, which is know as successive cancellation list decoding.

Polar codes exhibit excellent error correction performance. Also, it is very versatile .

The Basic Transform



- Take two input bits (u1, u2) and generate two channel bits (x1, x2) with,

$$x1 = u1 + u2$$

$$x2 = u2$$

The Basic Transform : Properties

- Communication channel P(Y|X) is input-symmetric and output-symmetric.
- Communication channel characterized by **mutual information (capacity)**

$$I_\Gamma := I(X;Y) = \sum_{x,y} P_{X,Y}(x,y) \log_2 \frac{P_{X,Y}(x,y)}{P_X(x)P_Y(y)}$$

$$= \sum_{x \in \mathbb{F}_2} \sum_{y} \frac{1}{2} P_{Y|X}(y|x) \log_2 \left(\frac{2P_{Y|X}(y|x)}{P_{Y|X}(y|X=0) + P_{Y|X}(y|X=1)} \right)$$

assuming uniformly distributed bits X1 and X2 which is the case if U1 and U2 are uniformly distributed

The Basic Tranform – Properties(2)

- We now consider the combined, 2-dimension channel

  (U1,U2)-> (Y1, Y2)

- We have

$$I(U1, U2; Y1, Y2) = I(X1, X2; Y1, Y2)$$
$$= I(X1;Y1)+ I(X2;Y2)$$
$$= 2I_{\Gamma}$$

  where (a) follows from the fact that the transformation (U1,U2)->(X1=U1+U2,X2=U2) is invertible and (b) as inputs to both memoryless channels are independent

- The basic transform does not degrade the channels, i.e., does not reduce the capacity.

The Basic Tranform – Properties(3)

- Applying the chain rule of mutual information gives

$$2I_{\Gamma} = I(U1, U2; Y1, Y2)$$
$$= I(U1; Y1, Y2) + I(U2; Y1, Y2|U1) \quad (1)$$

- Applying the chain rule of mutual information again to I(U2; Y1, Y2, U1) yields

$$I(U2; Y1, Y2, U1) = I(U2; U1) + I(U2; Y1, Y2|U1) \quad (2)$$

where **I(U1; U2) = 0** as the U1 and U2 are independent.

- Inserting (2) into (1) gives

$$2I_{\Gamma} = I(U1;Y1,Y2) + I(U2; Y1, Y2, U1).$$

The Basic Transform – Properties (4)

- This second decomposition allows us to consider the basic transformation as two equivalent channels:

1. Channel **Γ- : F2 -> y2**, a channel with input U1 and two outputs Y1 and Y2.
2. Channel **Γ+ : F2 -> y2 x F2**, a channel with input U2 and three outputs Y1, Y2 and U1.

- This decomposition has a specific operational meaning in the context of successive decoding:
  1. We first use the observations y1 and y2 to decode u1. This channel has a capacity I(U1; Y1, Y2) = I(Γ-)

The Basic Transform – Properties (4)

2. After decoding u1 we use it as additional observation together with y1 and y2 to decode u2. This second channel has a capacity of

$$I(U2; Y1, Y2, U1) := I\left(\Gamma^{+}\right)$$

Therefore , We have

$$I_{\Gamma} = I(\Gamma_{+}) + I(\Gamma_{-})$$

- We have

$$\begin{aligned} I\left(\Gamma^{+}\right) &= I(U2; Y1, Y2, U1) \\ &= I(U2; Y2) + I(U2; Y1, U1|Y2) \\ &\geq I(U2; Y2) = I(X2; Y2) = I_{\Gamma} \end{aligned}$$

where (a) is due to the chain rule of mutual information.

- This inequality leads to

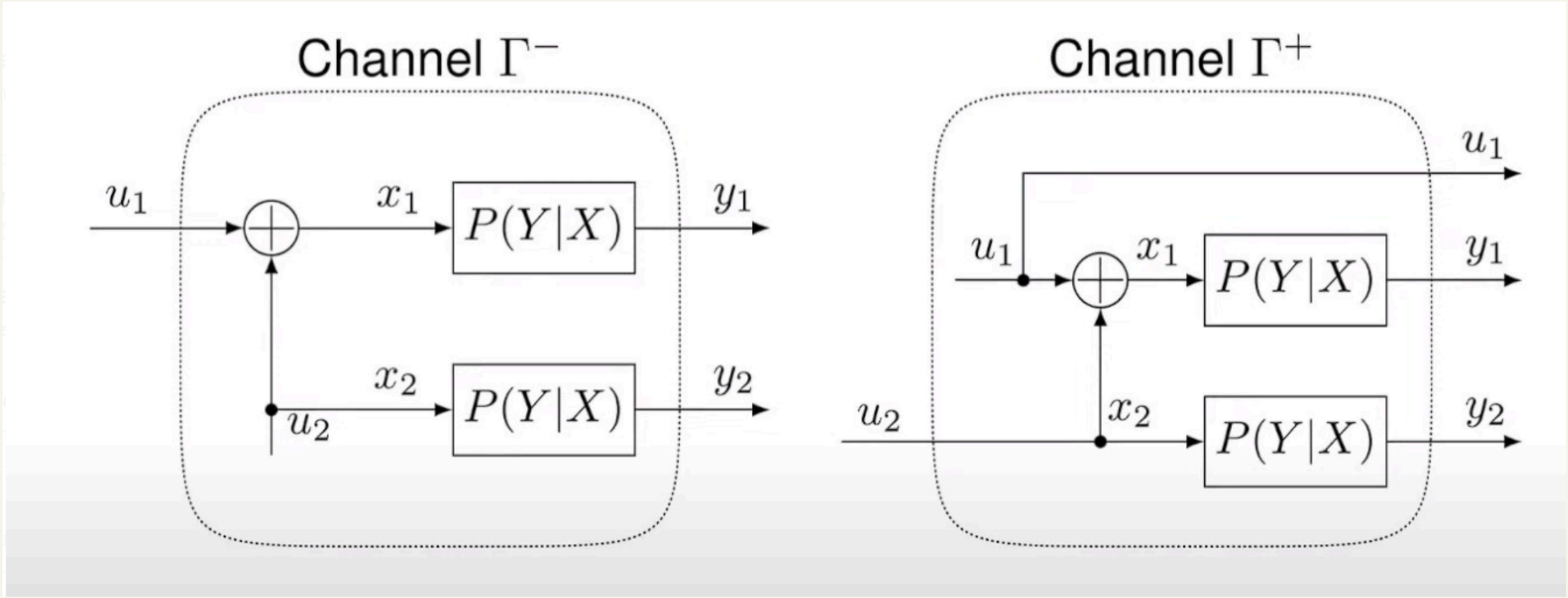
$$I_{\Gamma} + I_{\Gamma} = I\left(\Gamma^{+}\right) + I\left(\Gamma^{-}\right) \geqslant I_{\Gamma} + I\left(\Gamma^{-}\right)$$

$$I\left(\Gamma^{-}\right) \leqslant I_{\Gamma}$$

The Basic Transform – Channel Polarization

- We have generated two artificial channels Γ+ and Γ-.
- Channel Γ+ has a higher capacity than the original channel X -> Y.
- Channel Γ- has a lower capacity.
- Sum of the capacities equal the sum of the capacities of the original channel

- **And This is channel polarization!**


Channel Γ^-
u_1
x_1
$P(Y|X)$
y_1
x_2
$P(Y|X)$
y_2
u_2
Channel Γ^+
u_1
u_1
x_1
$P(Y|X)$
y_1
u_2
x_2
$P(Y|X)$
y_2

- In the case of the BEC, we generate two new BECs with

$$\Gamma\text{-} : \mathrm{BEC}(2\varepsilon - \varepsilon^2)$$

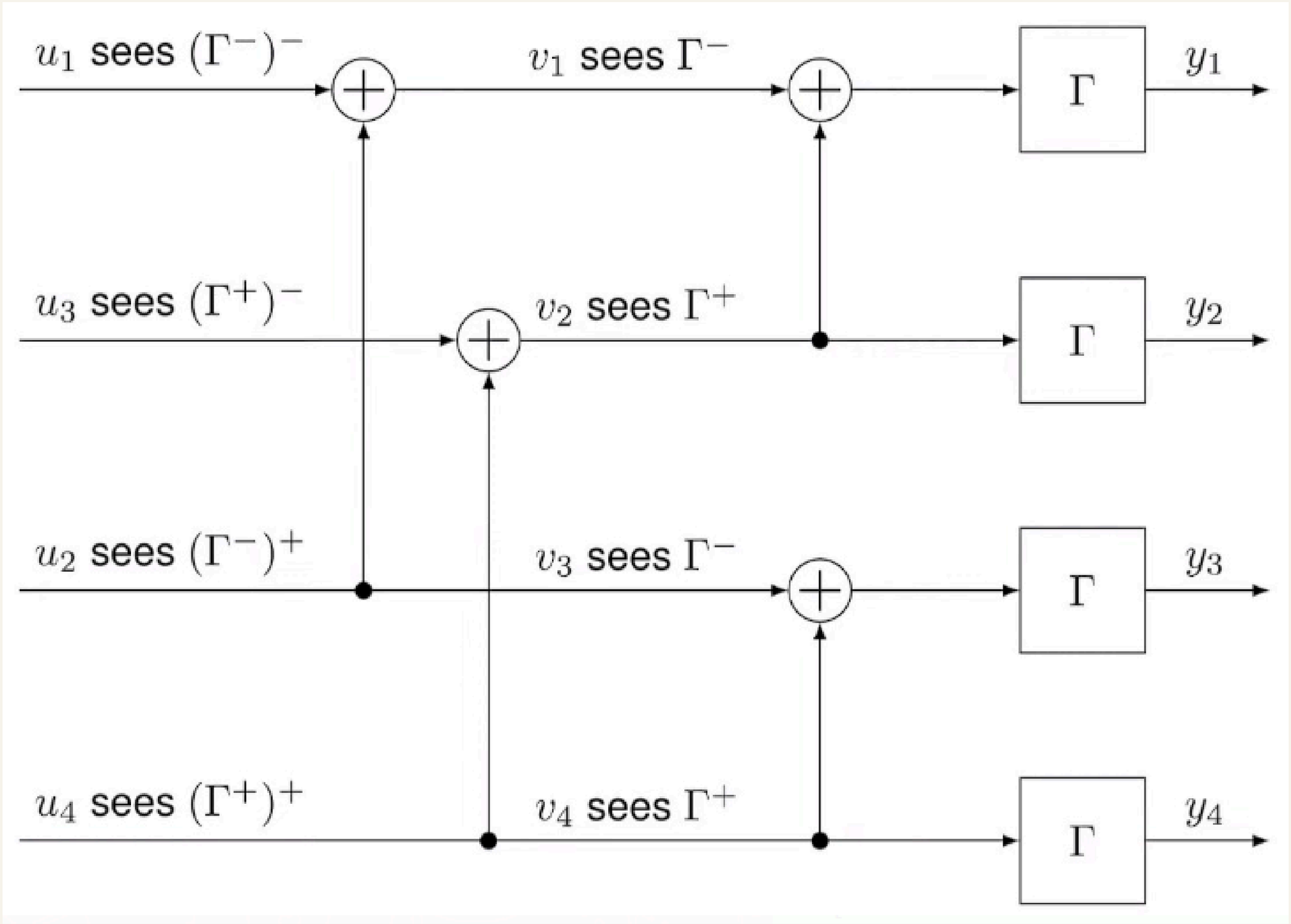
$$\Gamma\text{+} : \mathrm{BEC}(\varepsilon^2)$$

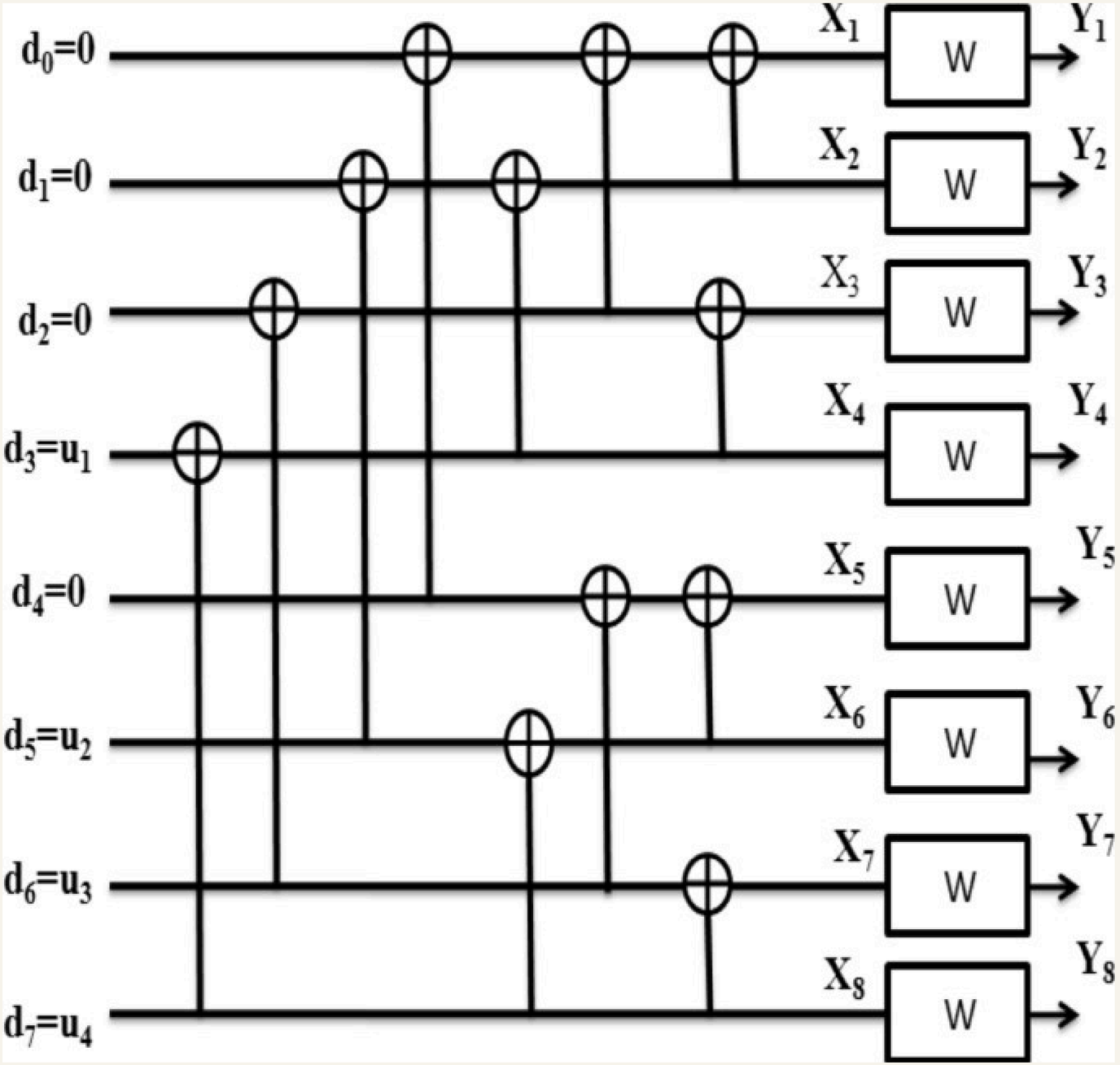
- Note that $\varepsilon^2 \leqslant \varepsilon$ for $\varepsilon \in [0,1]$
- Additionally $2\varepsilon - \varepsilon^2 \geqslant \varepsilon$ for $\varepsilon \in [0,1]$

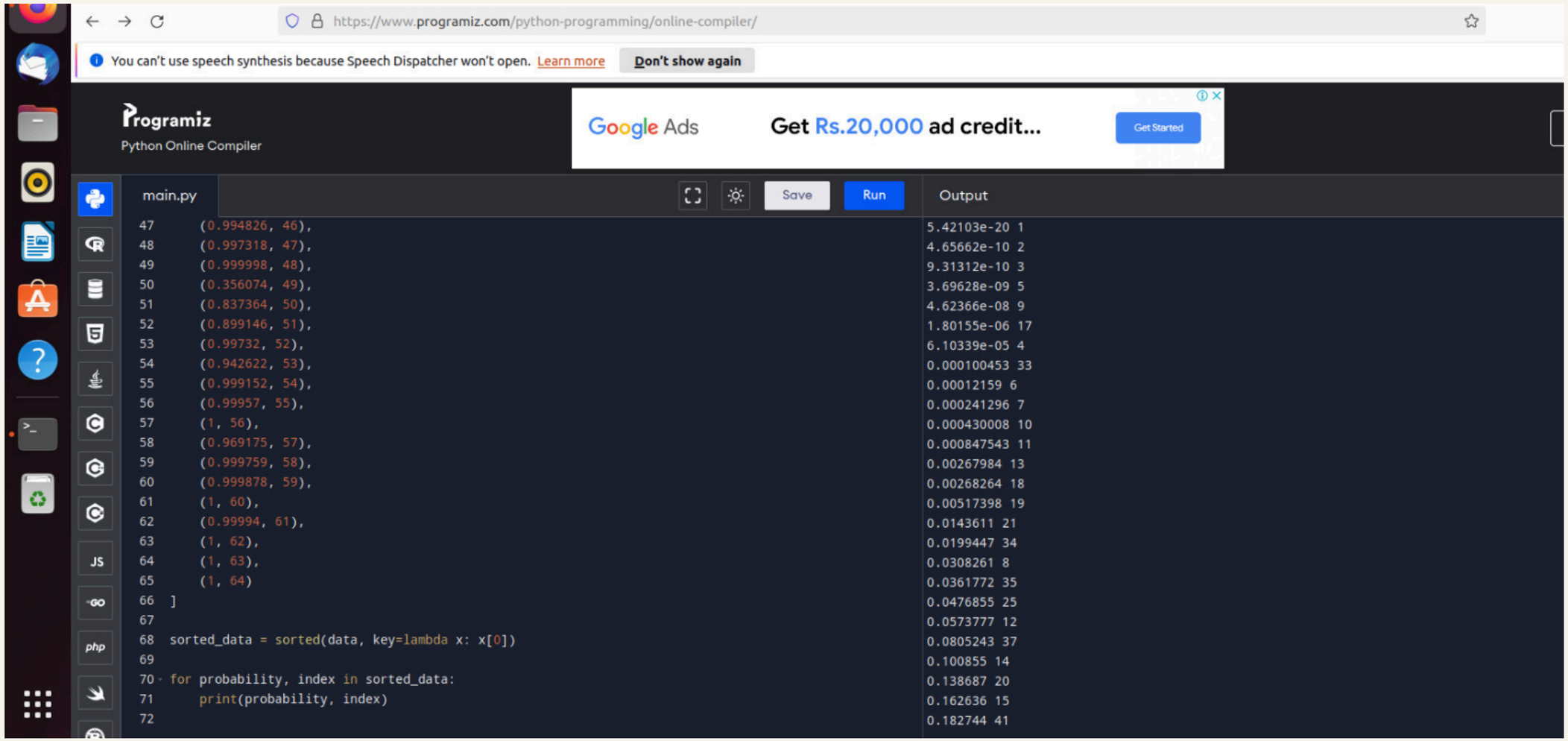
- We have shown that the basic channel transformation generates two new artificial channels which (with ideal successive decoding) "polarize" .

- One of them has a **slightly higher capacity**
- The other a **slightly lower**

- We now recursively apply the polarizing transform to construct further channels that allow for easier communication.

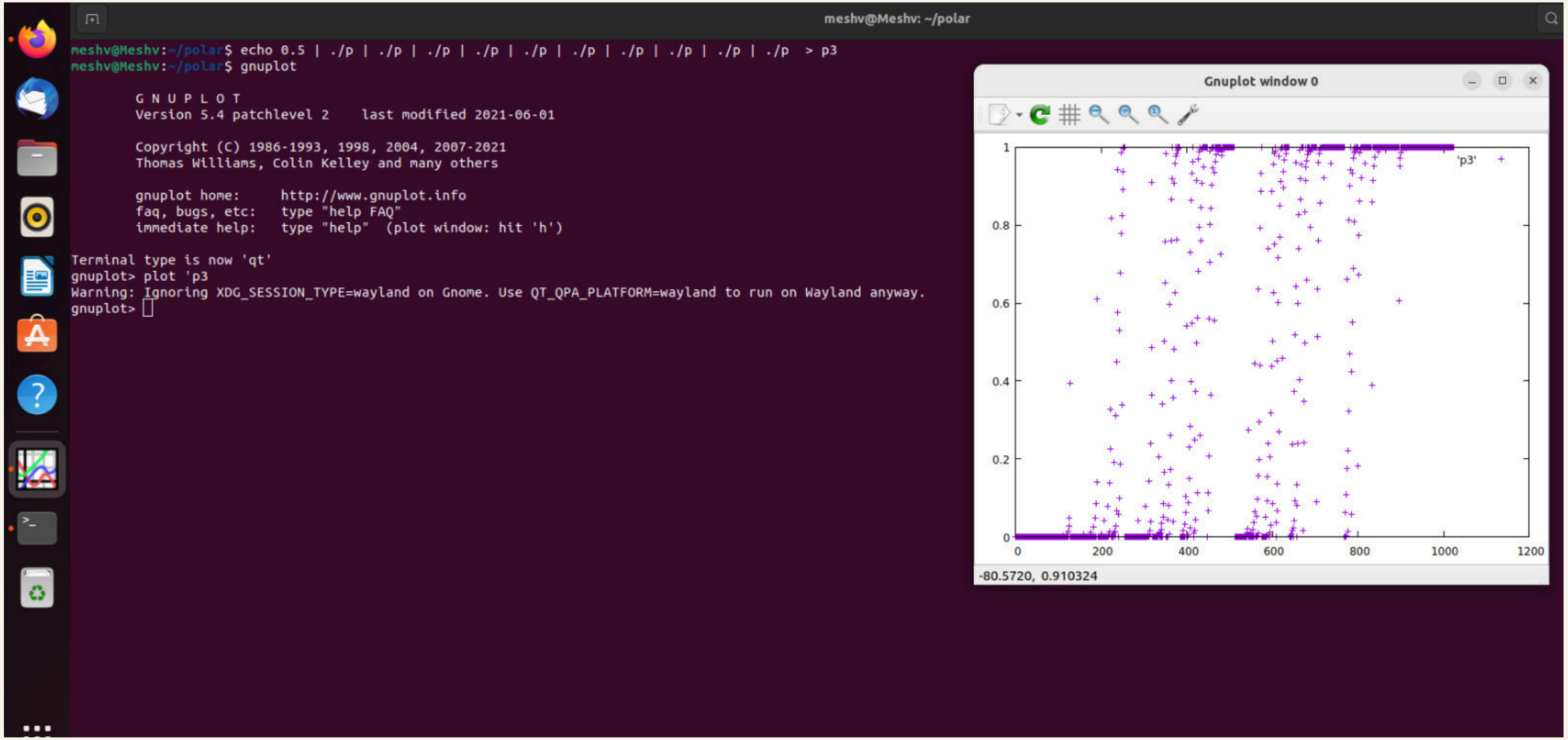

u_1 sees $(\Gamma^-)^-$
v_1 sees Γ^-
Γ
y_1
u_3 sees $(\Gamma^+)^-$
v_2 sees Γ^+
Γ
y_2
u_2 sees $(\Gamma^-)^+$
v_3 sees Γ^-
Γ
y_3
u_4 sees $(\Gamma^+)^+$
v_4 sees Γ^+
Γ
y_4


$d_0=0$
$d_1=0$
$d_2=0$
$d_3=u_1$
$d_4=0$
$d_5=u_2$
$d_6=u_3$
$d_7=u_4$
X_1
X_2
X_3
X_4
X_5
X_6
X_7
X_8
W
W
W
W
W
W
W
W
Y_1
Y_2
Y_3
Y_4
Y_5
Y_6
Y_7
Y_8

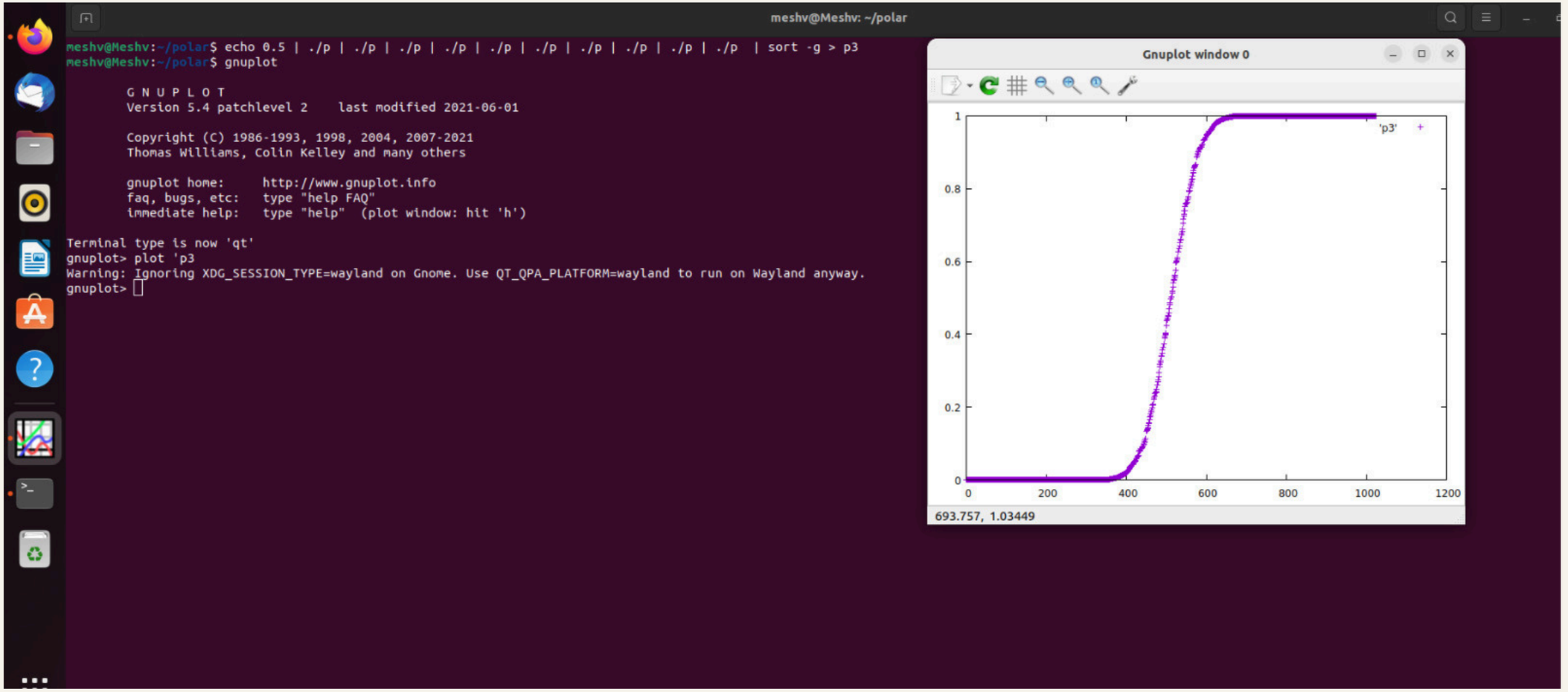


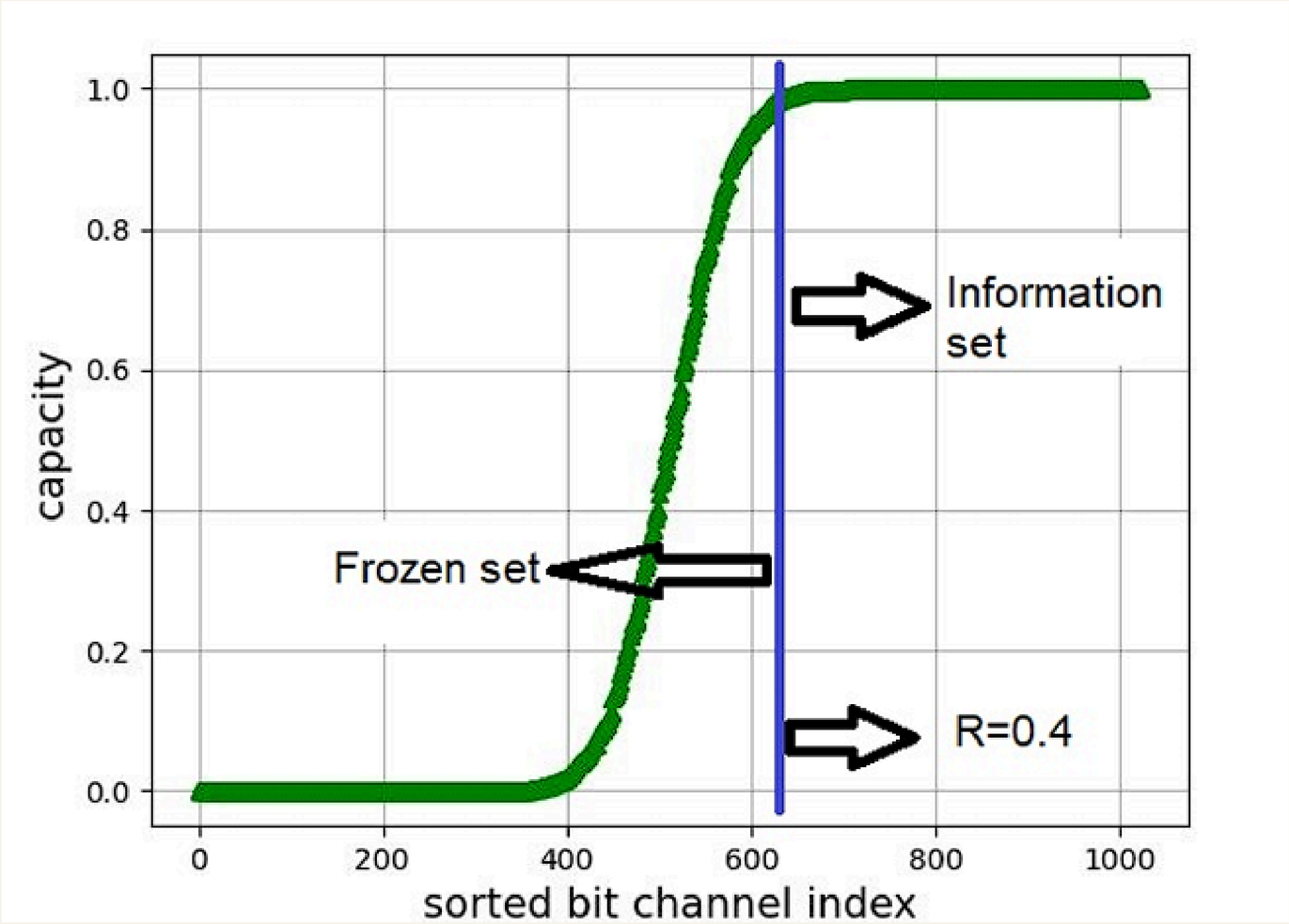
Reliability sequence : 1,2,3,5,9, ..., 64

Plot Reliability Sequence for n = 1024



Plot Reliability Sequence for n = 1024 in sorted form




1.0
0.8
0.6
0.4
0.2
0.0
capacity
0
200
400
600
800
1000
sorted bit channel index
Information set
Frozen set
R=0.4

IMPLEMENTATION

Encoding

We encode our message using channel reliability sequence and get encoded message.

AWGN

We introduce noise to our symbols through additive white gaussian noise.

$u = \sigma_n \times u_s$, where u_s N(0, 1)

σ_n = sqrt(1/SNR PER SYMBOL)

BPSK

We generate bpsk symbols from encoded message .

Decoding

We decode our final message after introduce noise from SUCCESSIVE CANCELLATION LIST DECODING ALGORITHM.

PART-I
ENCODING

Generator matrix:

Kronecker Product

BASIC IDEA OF ENCODING

1. Let the number of message bits to be transferred be k bits and N be the total number of bits.
   We will first find the reliability sequence, according to the 5G standards, for N bits.

2. Then, we will keep the N - k bits to be frozen according to the reliability sequence, as they are the least reliable. That is the U vector. Then, make N x N generator matrix through kronecker product.

3. Apply multiplication of modulo 2 to U vector and generator matrix. The vector obtained is the encoded message, which is the transferred through the bit channel.

THEORETICAL FRAMEWORK

- The generator matrix, $G_n = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}^{\otimes n}$ where n = 1,2,3,.. n and $\otimes$ is the kronecker product.

- U vector is the message bit to be encoded, including the frozen bits.

- Thus, the encoded message, $[u1\ u2]\ G_2 = [u1+u2\ \ u2]$

Encoding for N = 4 and k = 3

- Let N = 4 and k = 3. $n = \log_2 N \Rightarrow n = 2$
- The reliability sequence for N = 4 is as follows: 1 2 3 4
- So, N - k, i.e., 1 channel will be frozen bit. And the message will be transferred through 2 3 4 channels.
- Hence, the U vector will be: [0 m1 m2 m3]

- The Generator matrix for N = 4 is the kronecker product of G with itself, two times.

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

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

- Hence, the codeword or the encoded word will be:

Codeword = U G_2

$$= \quad [0\ m1\ m2\ m3] \quad X \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

- The encoded message will be:

[m1+m2+m3 m1+m3 m2+m3 m3]

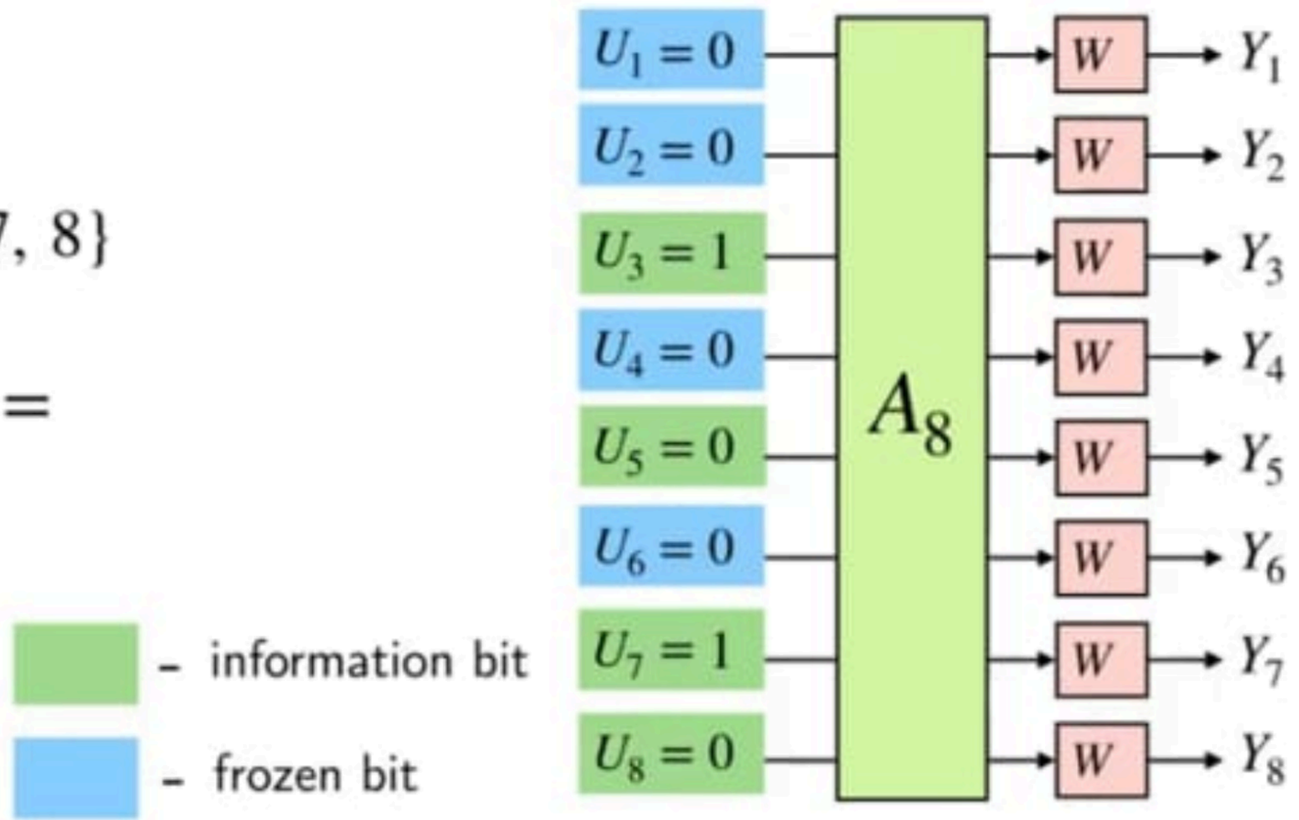
- According to the inputs, the values of m1, m2, m3 will be put.

For N = 8 and k = 4

$N = 8,\ k = 4$

Good $= \{3, 5, 7, 8\}$

message $m =$

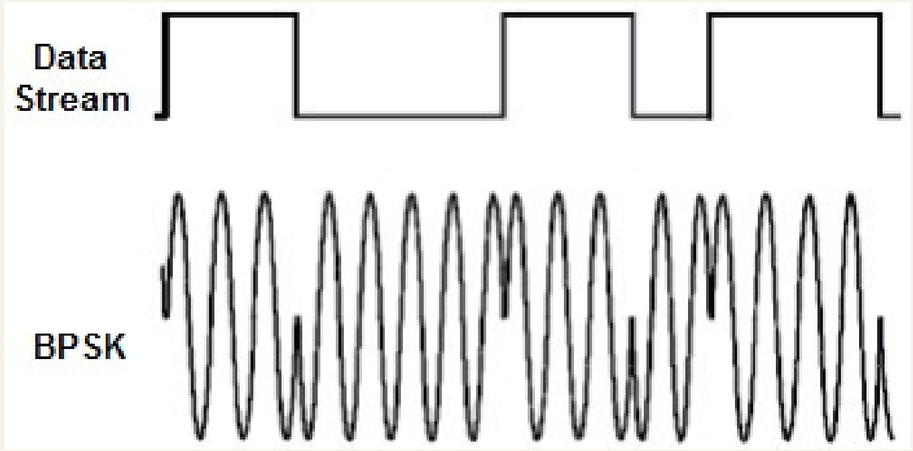


PART-II
BPSK AND AWGN

BPSK

WHAT IS BPSK?

BINARY SHIFT PHASE KEY (BPSK) IS A DIGITAL MODULATION TECHNIQUE WHERE THE PHASE OF THE CARRIER SIGNAL IS SHIFTED BY 180 DEGREES TO REPRESENT BINARY DATA.



HOW IS IT REPRESENTED?

BPSK TRANSFORMS BINARY BIT 0 TO 1 VOLT AND 1 TO -1 VOLT. IN THIS WAY, WE CAN SEND DATA. ON THE RECEIVER END, IF THE SIGNAL IS POSITIVE THEN THE RECEIVED SIGNAL IS BINARY BIT 0 AND IF IT IS NEGATIVE THE IT IS BINARY BIT 1.

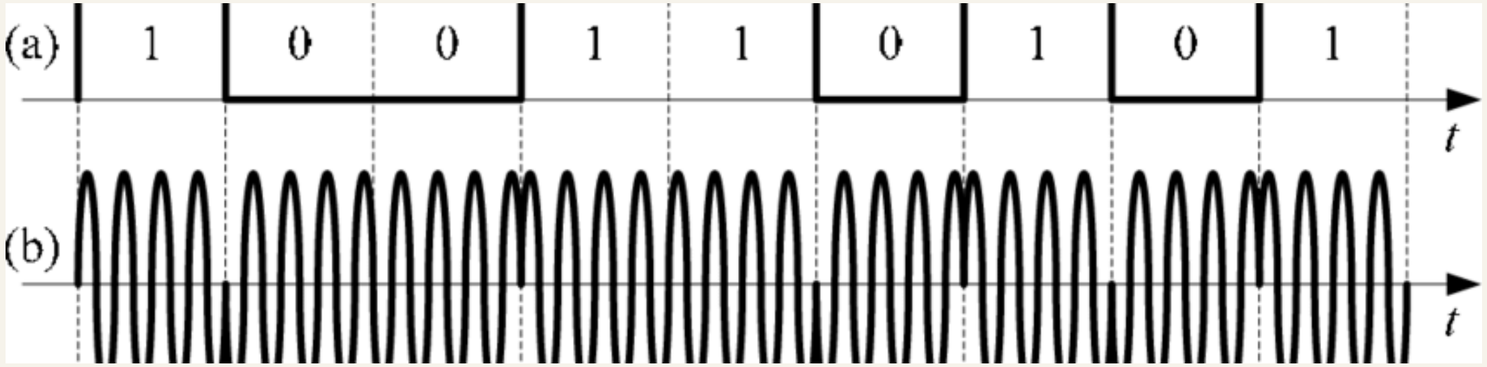
FOR INSTANCE

IF THE ENCODED MESSAGE IS AS FOLLOWS : [1 0 0 1 1 0 1 0 1].

WE CAN TRANSFROM IT USING THE FORMULA, 1 - 2B, WHERE B IS THE BIT.

THEN, THE BPSK SIGNAL WOULD BE

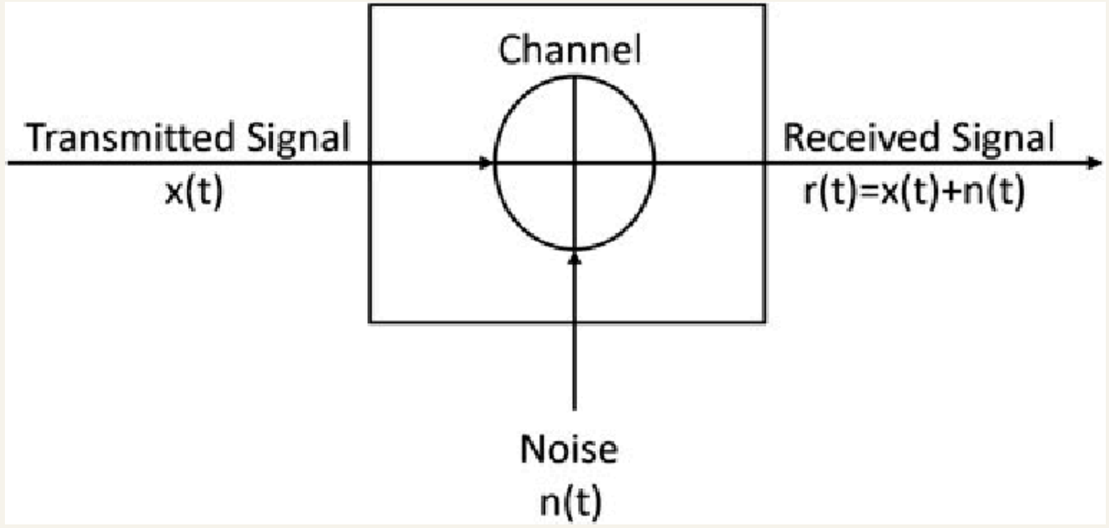
1	0	0	1	1	0	1	0	1
↓	↓	↓	↓	↓	↓	↓	↓	↓
-1	+1	+1	-1	-1	+1	-1	+1	-1



AWGN

WHAT IS AWGN?

ADDITIVE WHITE GAUSSIAN NOISE (AWGN) IS A TYPE OF NOISE THAT IS ADDED TO THE COMMUNICATION CHANNEL, WHEN THE SIGNAL IS TRANSFERRED.



HOW IS IT REPRESENTED?

WE HAVE INTRODUCED NOISE TO OUR SYMBOLS THROUGH ADDITIVE WHITE GAUSSIAN NOISE.

$u = \sigma_n \times u_s$ where $u_s \quad N(0, I)$

σ_n = sqrt(1/SNR PER SYMBOL)

PART-III
DECODING

SUCCESSIVE CANCELATION
LIST DECODING

Procedure of SC decoding :-

- **Sending Belief to the Left :** Each node starts by delivering its belief, to its left child node using a method known as the min-sum f function. This function calculates how probable it is that a bit is 0 or 1 based on the information received.
  The min-sum function $f(r1 , r2) = sgn(r1)sgn(r2)min(|r1| , |r2|)$

- **Left Node's Decision:** The left node takes this belief and utilizes it to form an initial guess about what the bit may be. This first guess is highly essential because it influences what occurs next.

- **Using Left Node's Guess:** After formulating its guess, the left node’s decision is then used by the node to communicate the next step.

- **Sending Belief to the Right :** The node takes the guess from left the node and then it transfers it into new belief sent to the right child node. It does this using another method called the g function. This function calculates the belief by taking into account the guess made by the left node.
  g function, $g(r1, r2, b) = r2 + (1 - 2b)\ r1$

- **Right Node's Decision:** With this new, updated belief, the right node then makes its own guess about the bit. This updated guess is more informed now because it uses both the received information and the estimate from the left node.

- **Final Decision to Parent:** Finally, the node combines these guesses and sends its best estimate back up to its parent node in the decoding tree.

LIMITATION OF SC DECODING

- But Successive Cancellation (SC) decoding has its own **limited error-correcting capabilities.**

- This method figures out the most likely transmitted codeword, but it might struggle to handle errors effectively at times.

- To help improve its performance, we usually use a more advanced technique known as **Successive Cancellation List (SCL) decoding.**

BEFORE WE DIVE INTO SUCCESSIVE CANCELLATION LIST DECODING, LET US FIRST SEE WHAT IS CYCLIC REDUNCY CHECK.

- The early development of CRC codes can be traced back to the work done by W. Wesley Peterson during the 1960s.

- Error detection, data integrity and reliability is the purpose of CRC bits.

How does CRC work?

n: Number of bits in data to be sent.

k : CRC length

- **First create crc polynomial of given crc bits and convert it into a binary form and that is your divisor.**
- **Append k zeros to the binary data .**
- **Divide binary data with divisor and append reminder to the binary data and generate final codeword.**

How to find CRC polynomial?

There are also algorithmic methods that can be utilized for generating CRC polynomials, such as the Berlekamp-Massey algorithm or the brute-force method.

We have used inbuilt CRC function in matlab, which is

[q , r] = gfdeconv(b , a), where q is the quotient and r is the remainder.

Example

Data word - [1 0 0 1 0 0]
No of crc bits = 3

STEP 1 : generator polynomial [x3 + x2 + 1]
Divisor = [1 1 0 1]

STEP 2 : [1 0 0 1 0 0 0 0 0]

STEP 3 : reminder = 0 0 1

Codeword -> 100100001

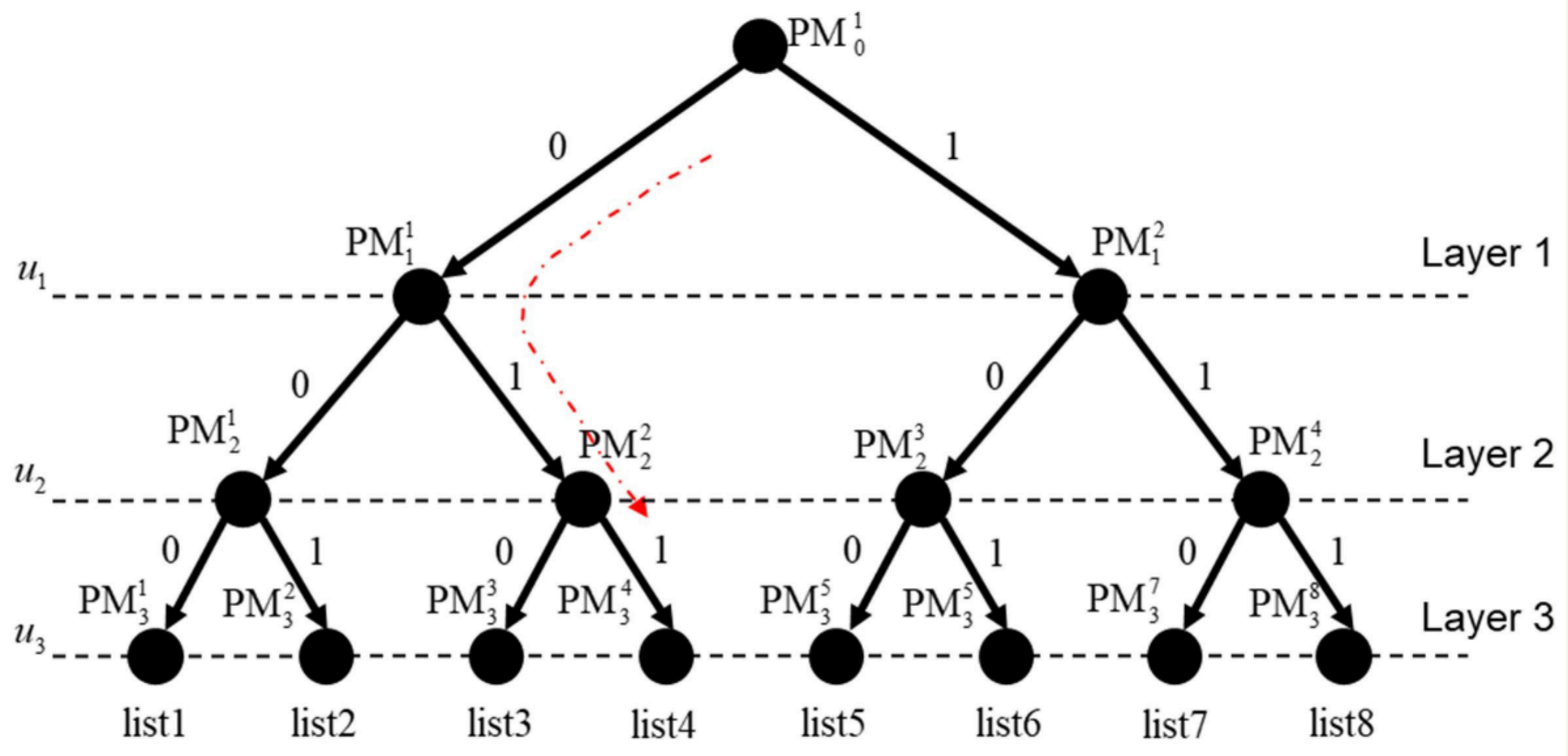
When you use CRC for error detection then you have to check that reminder of Received codeword and divisor is 0 or not if it is 0 then no error.

SC LIST DECODING

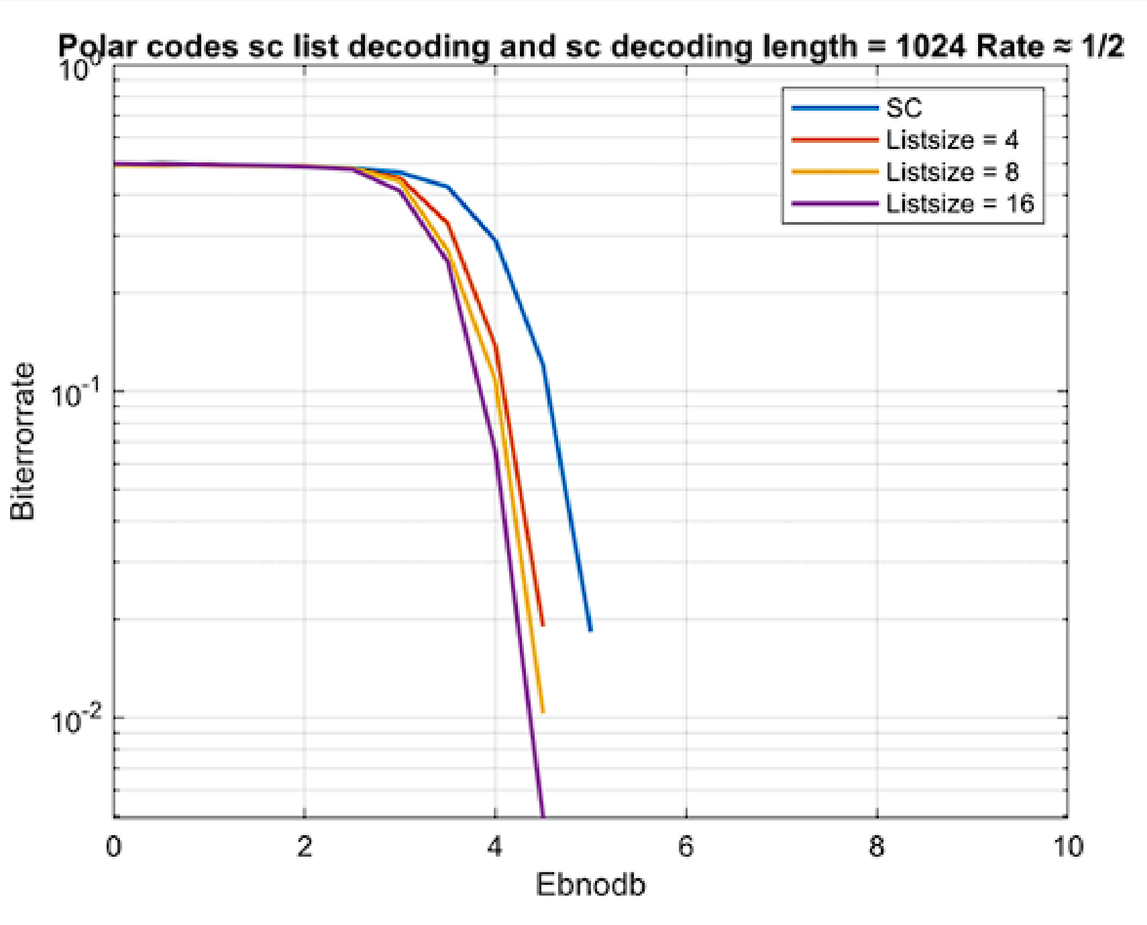
- **SC List Decoding:** The SC List decoding generates multiple possible codewords and stores them in a list. This approach allows the decoder several potential paths simultaneously, rather than a single path. The inclusion of multiple paths increases the likelihood that the correct codeword is among the considered options, thus improving error correction.

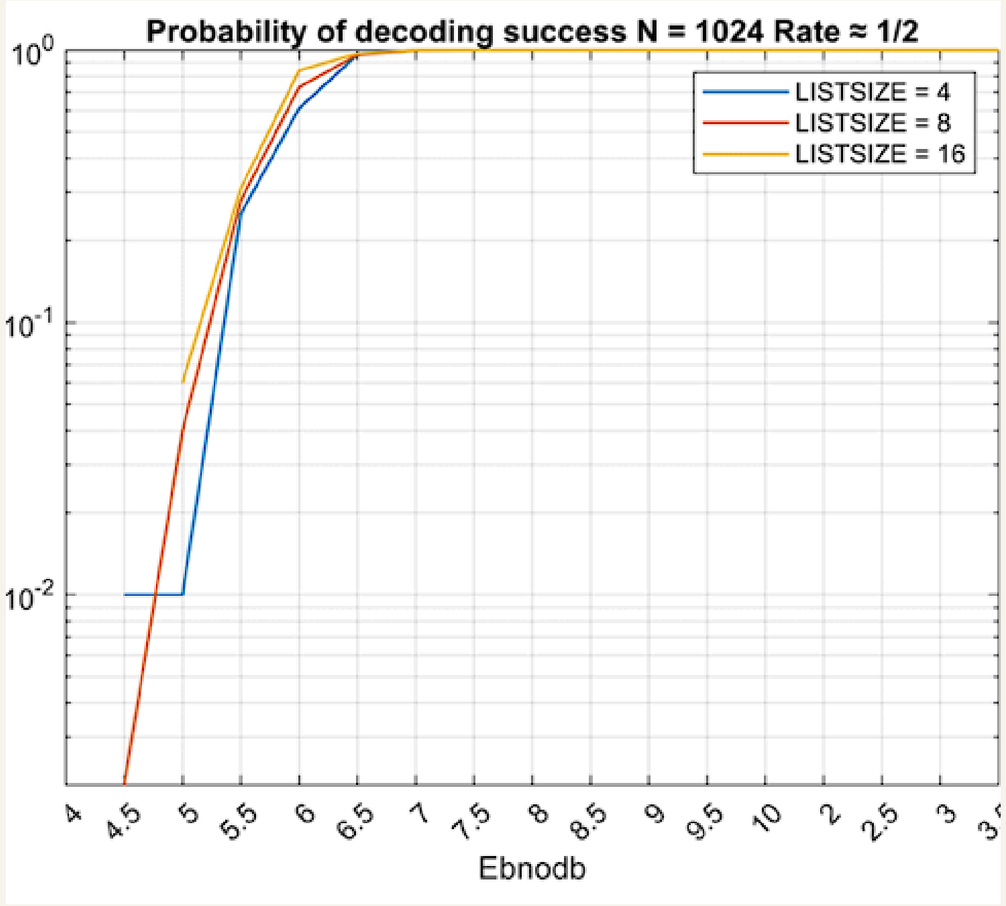
- **Using CRC for Selection:** To determine which codeword from the list is the correct one, SCL decoding utilizes Cyclic Redundancy Check (CRC), bits that are added to the message before transmission. By checking each codeword in the list against the CRC, the decoder can identify and select the most likely error-free codeword. This makes SCL decoding more reliable than standard SC decoding.

SC LIST DECODING EXAMPLE



MONTE CARLO SIMULATIONS





FOR MORE DETAILS

CLICK HERE

BIBLIOGRAPHY

- SC LIST DECODING
- Basic Transform Properties
- Arikan meets Shannon: Polar codes with near-optimal convergence to channel capacity V Guruswami.
- E. Arıkan, “Channel polarization: A method for constructing capacity achieving codes for symmetric binary-input memoryless channels
- Polar code encode
- Encoding photo
- BPSK 2
- Reliability Sequence
- BPSK 1
- AWGN

REFERENCES

- Andrew Thangaraj. Ldpc and polar codes in the 5g standard, 2019. Accessed on March 24, 2024.

- Telatar Emre. The flesh of polar codes, 2017. Accessed on March 24, 2024.

GROUP_5 | PROJECT GROUP 1

THANK YOU