Análisis y Evaluación del rendimiento de Códigos Polares en Sistemas Celulares de Quinta Generación

Naranjo Albán, Marcelo Israel

Departamento de Eléctrica, Electrónica y Telecomunicaciones

Carrera de Ingeniería en Electrónica y Telecomunicaciones

Trabajo de titulación, previo a la obtención del título de Ingeniero en Electrónica y Telecomunicaciones

Ing. Olmedo Cifuentes, Gonzalo Fernando PhD.

25 de agosto del 2021

Curiginal

Document Information

Analyzed document	Proyecto_de_grado_Marcelo_Naranjo.pdf (D111758548)
Submitted	8/27/2021 5:39:00 PM
Submitted by	
Submitter email	biblioteca@espe.edu.ec
Similarity	1%
Analysis address	ilbbioteca.GDC@analysis.urkund.com

Firmado electrónicamente por:
GONZALO FERNANDO
OLMEDO CIFUENTES

Sources included in the report

SA	**a__Codificaci_n_de_canal__5G_SDR.pdf** Document a__Codificaci_n_de_canal__5G_SDR.pdf (D109600405)	2
SA	**Proyecto_Andres_Vila.pdf** Document Proyecto_Andres_Vila.pdf (D98928069)	1
W	URL: https://ww2.mathworks.cn/help/5g/gs/polar-coding.html Fetched: 8/27/2021 5:40:00 PM	2

DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

CERTIFICACIÓN

Certifico que el trabajo de titulación, "**Análisis y Evaluación del rendimiento de Códigos Polares en Sistemas Celulares de Quinta Generación.**" fue realizado por el señor **Naranjo Albán, Marcelo Israel** el cual ha sido revisado y analizado en su totalidad por la herramienta de verificación de similitud de contenido; por lo tanto cumple con los requisitos legales, teóricos, científicos, técnicos y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, razón por la cual me permito acreditar y autorizar para que lo sustente públicamente.

Sangolquí, 25 de agosto del 2021

Firmado electrónicamente por:
GONZALO FERNANDO
OLMEDO CIFUENTES

.....

Ing. Olmedo Cifuentes, Gonzalo Fernando, Ph.D.

C.C.: 1711696342

DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

RESPONSABILIDAD DE AUTORÍA

Yo, **Naranjo Albán, Marcelo Israel**, con cédula de ciudadanía n° 0502480429, declaro que el contenido, ideas y criterios del trabajo de titulación: **" Análisis y Evaluación del rendimiento de Códigos Polares en Sistemas Celulares de Quinta Generación"** es de mi autoría y responsabilidad, cumpliendo con los requisitos legales, teóricos, científicos, técnicos, y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Sangolquí, 25 de agosto del 2021

Naranjo Albán, Marcelo Israel

C.C.: 0502480429

DEPARTAMENTO DE ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTORIZACIÓN DE PUBLICACIÓN

Yo, **Naranjo Albán, Marcelo Israel**, con cédula de ciudadanía n° 0502480429, autorizo a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación: **"Análisis y Evaluación del rendimiento de Códigos Polares en Sistemas Celulares de Quinta Generación"** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de mi responsabilidad.

Sangolquí, 25 de agosto del 2021

Naranjo Albán, Marcelo Israel

C.C.: 0502480429

Dedicatorias

Este trabajo de investigación está dedicado a una persona que estoy muy seguro, estaría muy orgullosa de este logro, mi abuelita Susana.

A toda mi familia que me apoya constantemente brindándome cariño y paciencia enseñándome a ser mejor persona, mejor hijo, mejor ser humano.

Agradecimientos

Quiero empezar agradeciendo a Dios por su misericordia, bondad y abundancia que me ha proveído todo este tiempo de carrera, estoy seguro de que sin ello no lo hubiera logrado.

Quiero agradecer a mi madre que me ha alentado cada vez que me he sentido derrotado, gracias por brindarme ese amor de madre que es eterno.

A todos mis amigos y colegas quienes fueron compañeros casi hermanos, a todos ellos un sincero, gracias.

También me gustaría expresar mi más sincera gratitud al Ing. Gonzalo Olmedo por ser mi tutor y mentor de tesis. También quiero agradecer al Ing. Daniel Altamirano quien supo encaminarme en el desarrollo de la investigación.

Son demasiadas personas para mencionarlas individualmente me han ayudado de tantas maneras durante mi trabajo de investigación, especialmente, a Fabián quien ha estado siempre pendiente de mi progreso.

Índice de Contenidos

Contenido

Índice Tablas

Índice Figuras

Índice de Acrónimos

NR – New Radio

Gbps – Gigabytes por Segundo

3GPP – 3rd Generation Partnership Project

LDPC – Low Density Parity Check

EMBB – Enhanced Mobile Broadband

URLCC - Ultra Reliable and Low Latency Communications

SC – Successive Cancellation

SCS – Simplified Successive Cancellation

SCL – Successive Cancellation List

B-DMC – Binary Discrete Memoryless Channel

ML – Maximum Likelihood

LLR – Log-likelihood Ratio

CRC – Cyclic Redundancy Check

BER – Bit Error Rate

FER – Frame Error Rate

BLER – Block Error Rate

AWGN – Additive White Gaussian Noise

Resumen

La presente tesis contiene el análisis y la evaluación del rendimiento de Códigos Polares en Sistemas Celulares de Quinta Generación. Es importante mencionar que entre los escenarios principales de uso de 5G se encuentran: banda ancha móvil mejorada (eMBB), comunicaciones ultra fiables y de baja latencia (URLLC) y comunicaciones masivas de tipo máquina (mMTC). De estos tres escenarios, la evaluación se realizará bajo las condiciones de eMBB. Para el análisis se utiliza un codificador que tiene como estructura la codificación polar sistemática y los dos tipos de decodificadores polares; cancelaciones sucesivas y lista de cancelaciones sucesivas que están asociados entre sí. El software que se utilizó para ejecutar el escenario de prueba es Matlab donde se ejecuta una simulación de un sistema de comunicaciones que utiliza una modulación BPSK y la codificación polar sobre un canal AWGN. Este sistema permite obtener métricas como el BER y FER, medidas de desempeño que permiten obtener el porcentaje de bits errados que se obtiene en el receptor comparando el flujo de bits a transmitir con el flujo de bits decodificados. Para la evaluación del desempeño se configura diferentes longitudes de mensaje y tasas de código de acuerdo con lo descrito por el 3GPP dentro de la fase de estandarización para 5G NR

PALABRAS CLAVE:

- **CODIFICACIÓN DE CANAL**
- **CÓDIGOS POLARES**
- **CANCELACIÓN SUCESIVA**
- **LISTA DE CANCELACIÓN SUCESIVA**

Abstract

This thesis contains the analysis and evaluation of the performance of Polar Codes in Fifth Generation Cellular Systems. It is important to mention that among the main 5G usage scenarios are enhanced mobile broadband (eMBB), ultra-reliable and low-latency communications (URLLC), and massive machine-type communications (mMTC). Of these three scenarios, the evaluation will be carried out under the eMBB conditions. For the analysis, an encoder is used whose structure is systematic polar coding and the two types of polar decoders; Successive cancellations and list of successive cancellations that are associated with each other. The software that was used to execute the test scenario is Matlab where a simulation of a communications system that uses BPSK modulation and polar coding is executed on an AWGN channel. This system allows metrics such as BER and FER to be obtained, performance measures that allow obtaining the percentage of erroneous bits obtained in the receiver by comparing the stream of bits to be transmitted with the stream of decoded bits. For performance evaluation, different message lengths and code rates are configured as described by 3GPP within the standardization phase for 5G NR.

KEYWORDS:

- **CHANNEL CODING**
- **POLAR CODES**
- **SUCCESSIVE CANCELLATION**
- **SUCCESSIVE CANCELLATION LIST**

Capítulo I

Planteamiento del Problema de Investigación

Antecedentes

El 5G NR hace referencia a la quinta generación de tecnología de banda ancha en redes móviles que promete brindar velocidades de hasta 10 Gbps, es decir, proporcionará velocidades y cobertura superior a las que ofrece el 4G actualmente.

En la versión 15 de los estándares de tecnología de acceso de 5G New Radio (NR) realizados por el 3GPP, se han introducido dos nuevos esquemas de codificación de canal de capa física de códigos polares y de verificación de paridad de baja densidad (*LDPC*, por sus siglas en inglés) para reemplazar a los Códigos Convolucionales y a los Códigos Turbo utilizados para 4G de evolución a largo plazo (LTE). (Bae, Abotabl, Lin, Song, & Lee, 2019)

Según el informe técnico 38.913 (3GPP, 5G; Study on Scenarios and Requirements for Next Generation Access Technologies, 2017) del 3GPP, existe tres escenarios principales de uso de 5G: banda ancha móvil mejorada (*EMBB*, por sus siglas en inglés), comunicaciones ultra fiables y de baja latencia (*URLLC*, por sus siglas en inglés) y comunicaciones masivas de tipo máquina (*mMTC*, por sus siglas en inglés). Estas aplicaciones demandan rendimiento, baja latencia y confiabilidad superior a la que proporciona el 4G con la finalidad de ofrecer altas tasas de velocidad a miles de conexiones simultáneas.

Durante la fase de estandarización 5G se determinó que los esquemas de codificación de canal deben admitir una tasa y longitud de código variable tanto para la información de control como para los datos del usuario, en función de dichos requisitos se adoptó la comprobación de paridad de baja densidad (LDPC) para los datos de los usuarios y la codificación polar para la información de control.

Justificación e Importancia

La relevancia de realizar el proyecto de investigación se basa en proporcionar una breve descripción sobre el proceso de codificación y decodificación de los códigos polares utilizados en 5G, además, comprobar el funcionamiento que propone esta nueva técnica de codificación de canal que logra cumplir los requisitos para la información de control que especifica el (3GPP, 5G; Study on new radio access technology Physical layer aspects, 2017).

El proyecto se enfoca en el desarrollo de algoritmos que permitan demostrar el rendimiento de un codificador y dos tipos de decodificadores en sus 2 formas más conocidas: cancelaciones sucesivas (SC, por sus siglas en inglés) y lista de cancelaciones sucesivas (*SCL*, por sus siglas en inglés) sobre un canal (AWGN) con las necesidades del canal de control 5G (eMBB) especificados en (3GPP, 5G; Study on Scenarios and Requirements for Next Generation Access Technologies, 2017).

Recientemente se ha realizado muchas investigaciones en el campo de la codificación de canales para mejorar las velocidades de datos en muchas de las tecnologías de acceso por radio, incluida la próxima tecnología 5G. Con el desarrollo de estos algoritmos de codificación y decodificación, se pretende que, en trabajos futuros, este trabajo sirva de guía o como línea base para mejorar los procedimientos que permita mejorar el rendimiento de los decodificadores (SC) o (SCL).

Alcance

El trabajo de titulación tiene como objetivo la evaluación del rendimiento de los códigos polares, es decir, se comprobará el funcionamiento de los algoritmos de codificación y decodificación bajo condiciones reales como son: la determinación de la información y las posiciones de los bits congelados, condiciones de la relación señal-ruido (*SNR*, por sus siglas en inglés), la longitud de código y la tasa a la que se enviarán estos datos.

Objetivos

Objetivo General

Evaluar el Rendimiento de Códigos Polares utilizados en los sistemas celulares de quinta generación.

Objetivos Específicos

- Presentar los conceptos básicos y definiciones de códigos polares usados en 5G.
- Proponer algoritmos de codificación mediante la polarización del método polarización del canal.
- Desarrollar algoritmos de decodificación: cancelación sucesiva (SC) y lista de cancelaciones sucesivas (SCL).
- Evaluar el desempeño de los códigos polares desarrollados y comparar con las funciones propuestas por Matlab.

Organización del Trabajo de Titulación

El Escrito se encuentra organizado en los siguientes capítulos: el segundo contiene el marco teórico donde se explica el modelo, la estructura, construcción y un ejemplo de cómo funciona la codificación polar. El tercer capítulo explica dos tipos de decodificadores polares: cancelación sucesiva y lista de cancelaciones sucesivas, y un ejemplo de funcionamiento de estos decodificadores. En el Capítulo 4 se presenta los resultados que se obtuvo en simulación de un sistema de comunicaciones que utiliza una modulación BPSK y la codificación polar sobre un canal AWGN, para finalmente en el Capítulo 5 presentar las conclusiones, recomendaciones del trabajo de titulación.

Capítulo II

Marco Teórico

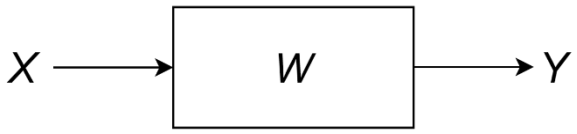
Códigos Polares para eMBB (Enhanced Mobile Broadband)

Modelo de Canal Código Polar

Para entender el modelo de canal utilizado por los códigos polares, en la Figura 1 se define W como un canal un canal binario discreto sin memoria (B-DMC, por sus siglas en inglés) donde X es la entrada binaria discreta $X = [0,1]$ y Y es la salida continua, por facilidad de notación se asume que Y es discreta.

Figura 1

Canal Binario Discreto sin memoria (B-DMC).



La probabilidad de transición: recibir y dado x de entrada, está dada por la ecuación 1

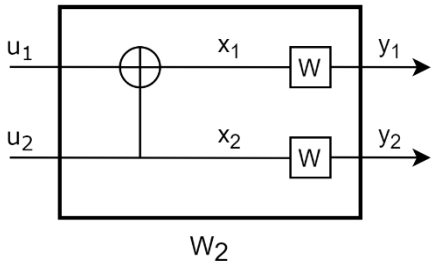
$$W(y|x)\ x \in X; y \in Y \tag{1}$$

Para lograr la capacidad simétrica $I(W)$ de un canal W, en la Figura 2, se construye un canal B-DMC W_2, siendo $I(W)$ la tasa más alta que se puede lograr. Es posible crear un segundo conjunto de N canales binarios de entrada a partir de N copias independientes de un B-DMC dado. Los canales tienen la propiedad $\{W_N^{(i)}: 1 \leq i \leq N\}$, se debe considerar que a medida que N aumenta, suceden dos cosas, una fracción de los canales I para $I\ W_N^{(i)}$ que está cerca de 0, se acerca a $1 - I(W)$, el resto de la fracción de canales para índices i para $I\ W_N^{(i)}$ cercano a 1, se acerca a $I(W)$.

Figura 2

Representación de un Canal Básico W_2

.



Este segundo grupo de canales polarizados $W_N^{(i)}$ están en "buenas" condiciones para ser codificados. Se debe asegurar que los canales que su capacidad normalizada es igual a 1 o estén cerca de 1, deben enviar datos a una tasa de velocidad igual a 1 y algunos de los canales para los que la capacidad es 0, necesitan enviar datos a una tasa de velocidad igual a 0, es decir, los canales que tienen la capacidad 0 son canales basura. Los códigos implementados bajo esta idea se denominan códigos polares.

Un canal B-DMC W contiene dos parámetros de interés. El primero es la capacidad simétrica $I(W)$

$$I(W) = \sum_{y \in Y} \sum_{x \in X} \frac{1}{2} w(y|x) \log \frac{w(y|x)}{1/2w(y|0) + 1/2w(y|1)} \tag{2}$$

Y el segundo parámetro de Battacharyya, dado por:

$$Z(w) = \sum_{y \in Y} \sqrt{w(y|0)w(y|1)} \tag{3}$$

Se considera que $Z(w)$ toma valores binarios $[0,1]$. En todo momento, se utiliza logaritmos base 2; por lo tanto, también tomará valores $[0,1]$. La capacidad simétrica es un parámetro que se utiliza como una medida de tasa y Battacharyya como parámetro de confiabilidad. La capacidad simétrica es la tasa más alta a la que se puede producir una comunicación confiable

a través de un canal W. Esto se logra utilizando las entradas del canal W con la misma frecuencia. El parámetro $Z(W)$ es un límite superior en la probabilidad de error de decisión de máxima verosimilitud (*ML*, por sus siglas en inglés) el canal W se usa una sola vez para transmitir 0 o 1.

Se define una operación conocida como producto Kronecker y se lo denota con el símbolo $\otimes$. Si A es una matriz de dimensión $m \times n$ y B una matriz $p \times q$ entonces el producto Kronecker de $A \otimes B$ es la matriz bloque de dimensión $mp \times nq$.

$$A \otimes B = \begin{bmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{bmatrix} \tag{4}$$

La n-ésima potencia Kronecker para todo $n \geq 1$ se define como $A^{\otimes n}$.

Se utilizará la notación estándar de Landau $O(N)$, para indicar el comportamiento asintótico de una función

Polarización de Canal

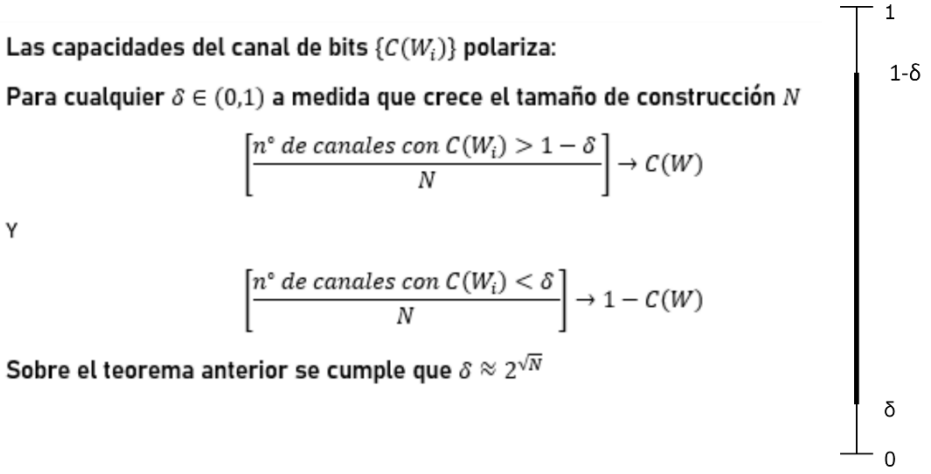
La idea principal de la polarización de canal es lograr transformar los canales ordinarios W en canales extremos, esto se logra con una transformación que produce N canales de bits independientes a partir de N copias independientes de un canal W (B-DMC). Luego, los nuevos canales sintéticos se dividen o se “polarizan” en el sentido que cada uno de ellos puede transmitir un solo bit con una fiabilidad diferente, es decir, con una probabilidad diferente de ser decodificado correctamente. Si N es lo suficientemente grande, todos los términos de capacidad simétrica $I\left(W_N^{(i)}\right)$ tienden hacia 0 (canales completamente ruidosos) o hacia 1 (canales perfectamente silenciosos).

Arikan, E. propone un teorema similar al de la polarización de fuente donde las capacidades de los canales de bits polarizan el número de canales, es decir, tienden a

moverse hacia 1 o 0. Se busca una fracción de términos para los que la entropía condicional se encuentra entre δ y $1-\delta$. Por ejemplo, en la Figura 3 donde se representa la capacidad del canal de 0 a 1, supongamos que δ es cercano al 1%, en este caso, se esperaría que el intervalo entre δ y $1-\delta$ sea grande porque ocupa casi todo el intervalo *N*-infinito, los términos entre δ y $1-\delta$ van a 0, lo que significa que no hay casi nada de canales en el medio y todos se han desplazado hacia el extremo superior 1 o hacia el extremo inferior 0.

Figura 3

Tablas PSI/SI que conforman u Representación de Polarización del Canal δ y $1-\delta$.



Nota. Tomado de (Arikan, Polar Codes tutorial, 2015).

El proceso de polarización del canal se lo puede describir en 2 fases

1. Combinación de canales
2. División de canales

Fase de Combinación de Canales

En esta fase se combina N copias de un canal (B-DMC) W de forma recursiva para generar un canal vectorial dado por la siguiente ecuación $W_N: X^N \to Y^N$ $N = 2^n$ lo que significa

que N siempre es potencia de 2 y $n \geq 0$. La recursividad comenzará en el nivel 0^{th} y se configura $W_1 = W$ con un solo conjunto de W. El valor $n = 1$ representa el primer nivel de recursividad donde 2 copias independientes de W_1 se combinan como se mostró en la Figura 2. Después de combinar las dos copias obtenemos el canal $W_2: X_2 \rightarrow Y_2$. Las probabilidades de transición obtenidas estarían representadas por la ecuación 5.

$$W_2(y_1, y_2|u_1, u_2) = W(y_1|u_1 \oplus u_2)W(y_2|u_2) \tag{5}$$

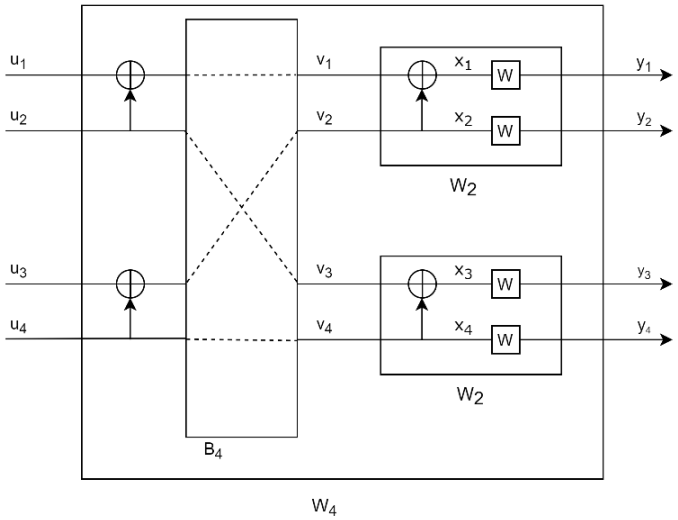
Una vez que se obtiene el canal se puede volver a combinar dos copias de W_2 para obtener una sola copia del canal, ahora el canal se llamará W_4. Esta recursividad a su vez será representada por $W_4: X_4 \rightarrow Y_4$ con probabilidad de transición:

$$W_4(y_1^4|u_1^4) = W_2(y_1^2|u_1 \oplus u_2, u_3 \oplus u_4)W_2(y_3^4|u_2, u_4) \tag{6}$$

La Figura 4 presenta una operación de *mapeo* entre la entrada u_1^4 y la salida v_1^4 $(s_1, s_2, s_3, s_4) \rightarrow v_1^4 = (s_1, s_2, s_3, s_4)$.

Figura 4

Canal de código polar W_4 y su relación con W y W_2.



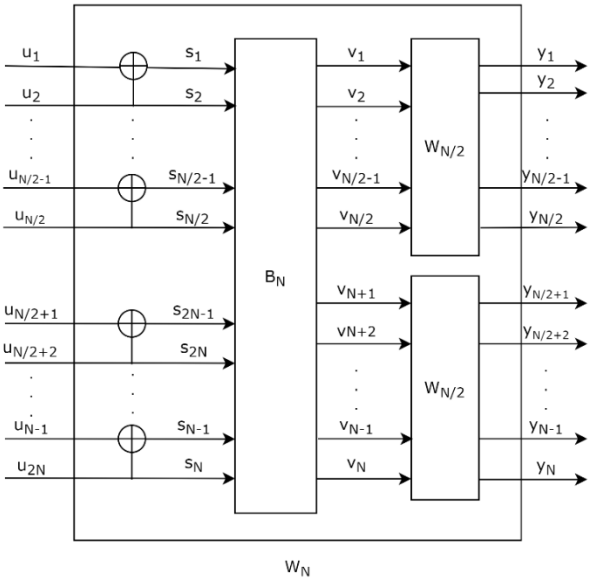
es decir, el mapeo se puede representar directamente como $u_{41} \rightarrow x_{41}$ de la entrada W_4 a la salida W^4 y se la puede escribir como $x_1^4 = u_1^4 G_4$, con la ecuación 7:

$$G_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \tag{7}$$

que se deriva una relación particular dada por la ecuación $W_4(y_1^4|u_1^4) = W_4(y_1^4|u_1^4\ G_4)$ esta relación va a ser entre la entrada y la salida W_4 y W^4. Estas dos no son más que probabilidades de transición. La forma general de recursividad se muestra en la Figura 5 se combinan dos copias independientes de $W_{N/2}$ para producir W_N, el vector de entrada u_1^N para W_N primero se transforma en el vector s_1^N tal que $s_{2i-1} = u_{2i-1} \oplus u_{2i}$ y $s_{2i} = u_{2i}$ para $1 \leq i \leq N$.

Figura 5

Construcción recursiva de W_N a partir de dos copias de $W_{N/2}$.



Existe un nombre especial para la función que desempeña B_N y se lo conoce como matriz de inversión aleatoria de bits ya que también realiza la inversión de bits para el espacio de canal

que representa la salida $v_1^N = (s_1,\ s_3, \ldots, s_{N-1}, s_2, s_4, \ldots, s_N)$ trabajando en las entradas s_1^N. Una vez realizado, comienza a actuar como un *input* a las 2 copias del canal $W_{N/2}$.

En la Figura 5, también se observa el mapeo de $u_1^N \rightarrow v_1^N$ que es un mapeo general de un extremo a otro. Se puede decir que es lineal porque todas las demás asignaciones de canales que se encuentran bajo esta asignación también son lineales, por lo tanto, esta forma se puede denotar mediante una matriz G_N que nuevamente es lineal bajo la siguiente afirmación:

$$x_1^N = u_1^N\, G_N \tag{8}$$

donde G_N es la matriz generadora de tamaño N. Los canales W_N y W^N poseen la probabilidad de transición que se representa por la ecuación:

$$W_N(y_1^N | u_1^N) = W^N(y_1^N | u_1^N\, G_N) \tag{9}$$

donde $y_1^N \in Y^N$, $u_1^N \in X^N$ son las condiciones para satisfacer por todas las variables de la ecuación anterior. Otra relación que vale la pena mencionar respecto a G_N está dada por la ecuación 9 donde B_N es la matriz de permutación de inversión de bits y $F^{\otimes n}$ es el n-ésimo producto Kronecker

$$G_N = B_N\, F^{\otimes n}\, \{N = 2^n, n \geq 0\} \tag{10}$$

La matriz $F = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ es conocida como matriz *kernel* que satisface completamente la operación de combinación de canales.

Fase de División de Canales

En la sección anterior se ha construido W^N a partir del canal vectorial W_N, ahora se realiza la operación inversa de combinar, es decir, dividir el canal. W_N en N canales de entrada binaria $W_N^{(i)}: X \rightarrow Y^N \times X^{i-1}$, $1 \leq i \leq N$ y las ecuaciones que implican la fase de división se definen

por las probabilidades de transición (Arikan, Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels, 2009):

$$W_N^{(i)}\left(y_1^N, u_1^{i-1} \middle| u_i\right) \triangleq \sum_{u_{i-1}^N} \frac{1}{2^{N-1}} W_N(y_1^N | u_1^N) \tag{11}$$

donde $\left(y_1^N, u_1^{i-1}\right)$ significa el *output* de $W_N^{(i)}$ y u_i es el *input*. Se conocerá más información sobre esto cuando tratemos de considerar un decodificador asistido por "genio" para la decodificación de cancelación sucesiva donde el i_{th} DE después de observar el y_1^N y las entradas previas u_1^{i-1} y se dice que es suministrado por el "genio asistido" independientemente de cualquier error de decisión en las etapas anteriores. Aquí se puede intentar estimar $W_N^{(i)}$ como el canal efectivo visto por el i_{th} elemento de decisión si u_1^N es uniforme a priori en X_N.

Se aprovecha el efecto de polarización para construir códigos que alcanzan tasas cercanas a $I(W)$ mediante un método que llamamos codificación polar. La idea básica de la codificación polar es crear un sistema de codificación donde se pueda acceder a cada canal sintetizado $W_N^{(i)}$ individualmente y enviar datos solo a través del subconjunto de ellos para el cual $I(W_N^{(i)})$ está cerca de 1.

Diseño y Construcción del código

Dado que los canales silenciosos tienen mayores capacidades o menores probabilidades de error que los canales ruidosos, el fenómeno de polarización del canal sugiere una nueva filosofía para la codificación de canales, a saber, seleccionar los canales silenciosos para la transmisión de bits de información. Debido a que la tasa de código se puede ajustar con precisión agregando o eliminando un canal polarizado, podemos variar casi continuamente la tasa de códigos polares. En comparación con otros esquemas de codificación, esta propiedad de velocidad compatible es una ventaja significativa. Además, a diferencia de la construcción de código tradicional para maximizar la distancia mínima de

Hamming, el objetivo de la codificación polar es minimizar directamente la probabilidad de error de los canales polarizados portadores de información. (Bioglio, Condo, & Land, 2020)

Codificación

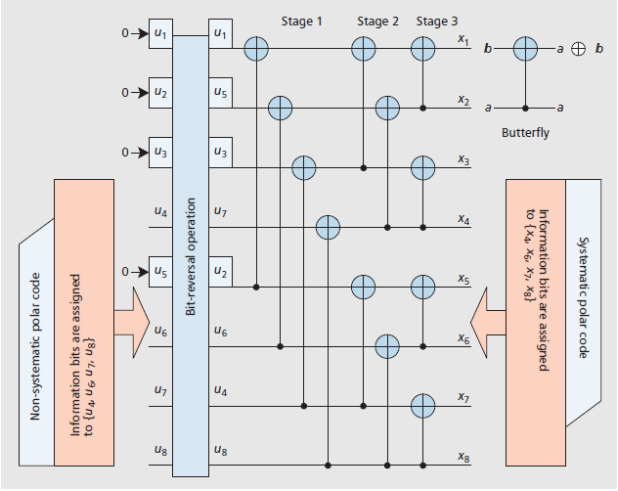
Consideremos un código polar (N, K) de longitud N bits con una tasa R, se tiene que $K = NR$ son bits de información. La codificación polar en 5G se realiza con una secuencia binaria de entrada $\boldsymbol{u} = [u_0, u_1, \dots, u_{N-1}]$ de longitud K donde las posiciones de los bits $N - K$ que pertenecen al *frozen set* se los asignará a cero $u_i = 0$ y los bits de información se los asigna a las entradas restantes. La palabra de código $x = [x_0, x_1, \dots, x_{N-1}]$ resultante se obtiene como

$$x = \boldsymbol{u} \cdot G_N = \boldsymbol{u} \cdot (G_2)^{\otimes n} \tag{12}$$

En la Figura 6, con una operación denominada "*butterfly operation",* se transforma dos bits de entrada independientes (a, b) en dos bits de salida correlacionados por la operación $(\oplus)$, XOR en binario, $(a \oplus b, a)$, dominado por la matriz *kernel* $\boldsymbol{F}$ que corresponde a la polarización de dos canales.

Figura 6

Ejemplo de un código Polar sistemático y no-sistemático



Nota. Tomado de (Ido & Vardy, 2011)

Esta operación también se puede aplicar de forma recursiva a toda la palabra de código, es decir, una palabra de código x se divide en dos partes en la etapa 3, cada una de las cuales a su vez se divide nuevamente en dos partes en la etapa 2, y así sucesivamente, hasta que se llega a una fuente única bit u_i en la etapa 1. Entonces, el proceso de codificación polar para $N = 8$ incluye una inversión de bit y tres etapas de *butterfly operations.*

Generalmente, dada una longitud de código $N = 2^n$, la transformada de polarización se puede descomponer en n etapas con cada una de las $N / 2$ *butterfly operations*, lo que da como resultado una complejidad de codificación de orden $O(N \log N)$ (Niu, Kai, Lin, & Zhang., 2014).

Códigos Polares Sistemáticos

Como dice la teoría de la codificación, casi todos los códigos de bloques lineales pueden convertirse o transformarse en un código sistemático equivalente. La codificación sistemática no es nueva. Recientemente se ha demostrado que casi todos los códigos como los Reed-Müller o códigos Turbo tienen un rendimiento superior con respecto al rendimiento de tasa de error de bit (*BER*, por sus siglas en inglés) y tasa de error de trama (*FER*, por sus siglas en inglés) si se los transforma a códigos sistemáticos. La codificación polar sistemática fue introducida en (Arikan, Systematic polar coding, 2011) y no es más que un esquema en el que los bits de entrada están integrados en la salida codificada. La codificación de los bits de información aparece como parte de la palabra de código de forma transparente, es decir, no se aplica ninguna codificación para estos bits y pueden ser observados directamente por el decodificador. En la Figura 5 se mostró un diagrama de bloques de la codificación sistemática.

En la sección anterior se definió la codificación no sistemática como $x = uG$ donde u es la palabra fuente, x la palabra de código. La palabra fuente u se divide en dos partes $u =$

(u_A, u_{A^c}), u_A contiene datos de usuario y u_{A^c} contiene los *frozen bits* donde el mapeo se puede escribir como

$$x = u_A G_A + u_{A^c} G_{A^c} \tag{13}$$

donde G_A y G_{A^c} son las submatrices de G. Ahora se fija un código, tal como lo especifica un codificador no sistemático como en la ecuación (12), y se considera varios codificadores sistemáticos posibles para este código:

$$x_B = u_A G_{AB} + u_{A^c} G_{A^c B} \tag{14}$$

$$x_{B^c} = u_A G_{AB} c + u_{A^c} G_{A^c B^c} \tag{15}$$

En (Arikan, Systematic polar coding, 2011) se busca un codificador sistemático en el que x_B desempeñe el papel que desempeñó u_A en la codificación no sistemática como portador de datos, mientras que u_{A^c} contiene los *frozen bits*. Para cualquier parámetro de codificador no sistemático (A, u_{A^c}) se dice que existe un codificador sistemático (B, u_{A^c}) si (si y solo sí) A y B tienen el mismo número de elementos, G_{AB} es una matriz invertible y las ecuaciones para x_B y x_{B^c} son verdaderas

$$u_A = (x_B - u_{A^c} G_{A^c B})(G_{AB})^{-1} \tag{16}$$

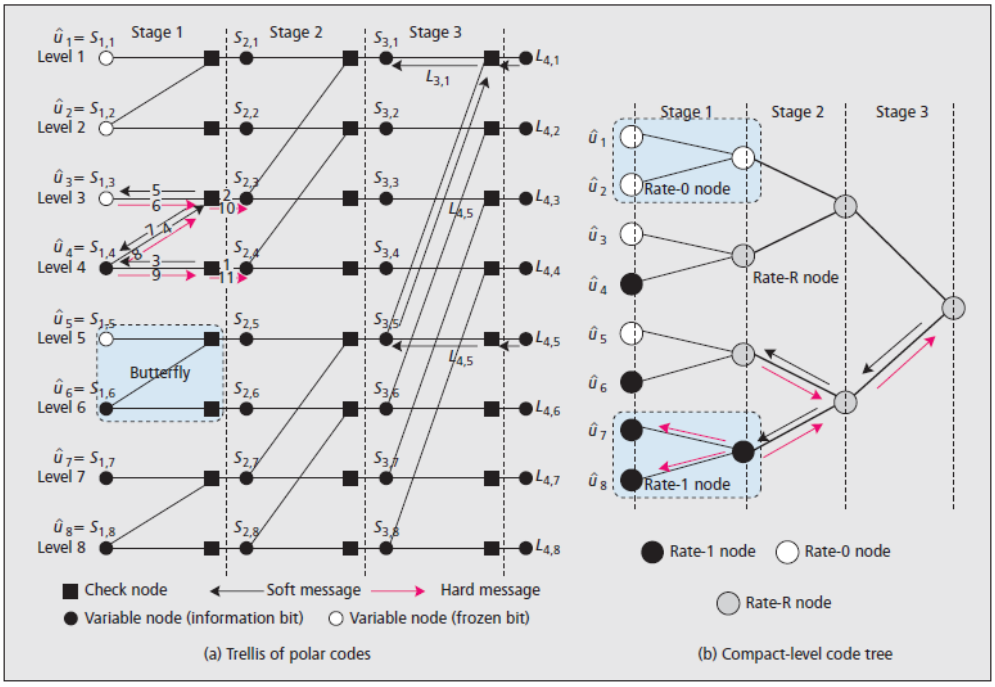
Finalmente se reemplaza u_A en la ecuación 14 para hallar x_{B^c}.

La codificación polar sistemática muestra una mejora en el BER con un método de decodificación que calcula x indirectamente, calculando primero u como si el código no fuera sistemático, se esperaría que cualquier error de decodificación en u (que sufre la propagación de errores) se amplificaría en el paso de recodificación $x = uG$. Paradójicamente, los resultados de la simulación muestran que este no es el caso.

Para construir un código polar sobre un B-DMC simétrico arbitrario, en (Mori & Tanaka, 2009) se propuso el uso de herramientas *density evolution* (DE) para rastrear la función *probability density function* (PDF) de las relaciones log-likelihood ratios (LLR) en la variable y comprobar los nodos en el gráfico de decodificación de códigos polares, como se muestra en la Figura 7. Las convoluciones de los PDF LLR se realizan en los nodos de verificación y variable. Esta técnica DE se usa ampliamente en el diseño de códigos LDPC y es igualmente aplicable al diseño de códigos polares. Con base en las PDF LLR en la primera etapa de los nodos variables, se pueden obtener las probabilidades de error de todos los canales polarizados.

Figura 7

Representaciones de Code Tree y 'trellis' de códigos polares. (a) Trellis de códigos polares; (b) Code Tree de nivel compacto.



Nota. Tomado de (Ido & Vardy, 2011).

En (Ido & Vardy, 2011) se propuso un método eficaz para resolver este problema controlando los errores de cuantificación y mediante la aproximación adecuada. Allí se introdujo dos métodos de aproximación, llamados actualización y degradación de cuantificación, para obtener un límite superior e inferior en la probabilidad de error de cada canal. Ambos métodos transforman el canal relevante en uno nuevo con un alfabeto de salida más pequeño en términos de m. Aunque la precisión de las funciones de densidad de este algoritmo se puede mejorar aumentando el tamaño del alfabeto de salida, la complejidad del algoritmo también aumenta. Para los canales AWGN, principalmente de los teóricos de la codificación, se puede aplicar un método alternativo presentado en (Trifonov, 2012) llamado *Gaussian Approximation* (GA) en la construcción de códigos polares. El GA tiene una complejidad menor que el método de Tal y Vardy cuando se aplica a canales AWGN de entrada binaria, pero produce casi la misma precisión. Desde el punto de vista práctico, GA es una opción más atractiva que otros métodos de códigos polares.

Funcionamiento

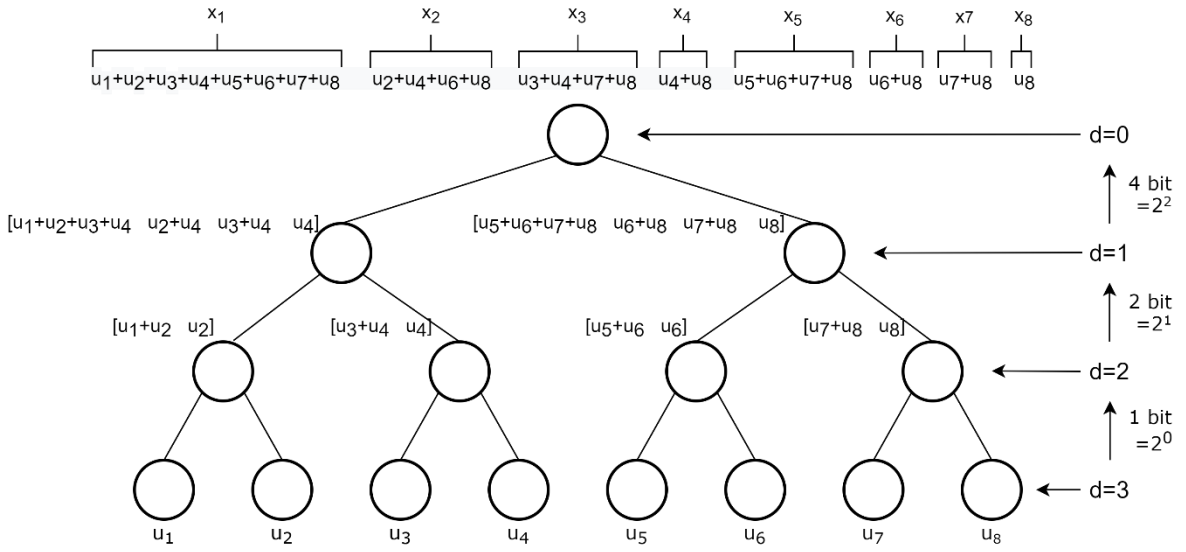
Para comprobar el funcionamiento del codificador se parte de un ejemplo básico utilizando la representación *code tree*. Se define un $N = 8$ bits, entonces, como entrada se tiene el vector $u = \{u_1, u_2, \dots, u_8\}$ que se los asigna al final del *code tree* y como salida se obtendrá el vector $x = \{x_1, x_2, \dots, x_8\}$. En la Figura 8 se observa la representación del ejemplo mencionado.

Se asigna los bits de entrada en los nodos correspondientes y se procesa un nivel a la vez. En la parte derecha se asigna un valor de nivel d, cuyo valor está dado por $d = \log_2 N$ y corresponde al número de niveles que tendrá el *code tree.* Nótese que el número de bits en cada nivel siempre será igual a N, pero a medida que el nivel avanza, se combina más bits entre sí.

El proceso comienza en el nivel más alto, en este caso $d = \log_2(8) = 3$, hasta llegar al nivel cero $d = 0$, otro proceso es el que combina los bits en cada nivel. Primero se combina 1 bit a la vez, y este valor aumenta a medida que se cambia que los niveles avanzan, generalizando, se combina $m * 2$ bits en cada nivel.

Figura 8

Representación de Code Tree para $N = 8$.



La combinación inicia con u_1, el siguiente salto es $2 * m$ a la derecha hasta N, entonces, si se combina m bits a la vez, los siguientes m bits a combinar son $i + 2 * m - 1$ y el siguiente salto a $i + 2 * m - 1$. Finalmente se combina las dos partes en pasos de m para obtener el valor de cada carga. Una vez que el proceso de combinación finaliza, se cambia al siguiente nivel y el número de bits combinados se duplica con $m * 2$

Una vez definido el proceso de codificación se verifica con un ejemplo. Se define $N = 8$ bits de información codificada y $K = 4$ bits de información. La secuencia de confiabilidad se calcula para cada caso dependiendo del valor de N. Para el ejemplo que se presenta en la Figura 9, de la secuencia original, se toma todos los números que son menores o iguales a

$N = 8$, de los cuales, se asigna los $N - K$ primeros bits como *frozen bits* o bits menos confiables y el resto de los bits más confiables como bits de mensaje.

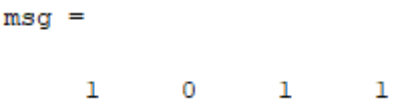
Figura 9

Secuencia de confiabilidad para $N = 8$ con frozen bits y bits más confiables.

En la Figura 10, para este ejemplo, se muestra la generación de un mensaje aleatorio de longitud $K = 4$ bits.

Figura 10

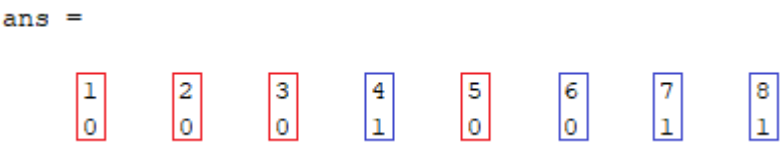
Mensaje aleatorio de 4 bits.



De los 8 bits que se espera del mensaje codificado, se asigna “0” las posiciones de los *frozen bits* y los bits del mensaje en las posiciones de los bits más confiables como se observa en la Figura 11.

Figura 11

Asignación de los frozen bits y los bits de mensaje.

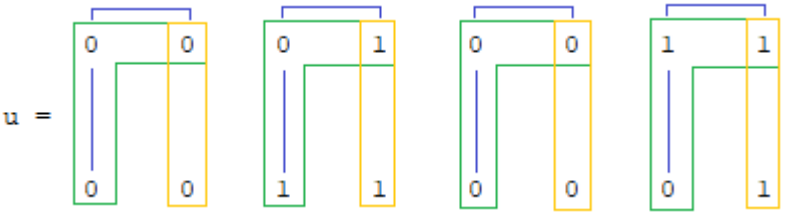


En la figura 11 se puede observar las posiciones 1, 2, 3 y 5, señaladas con color rojo, corresponde a los frozen bits y las posiciones 4, 6, 7, 8, señaladas con azul, corresponde a las posiciones donde se ubica los bits de mensaje.

Luego, se ejecuta el proceso de combinación. En la Figura 12 se presenta el tercer nivel de profundidad donde se combina 2 bits a la vez con la operación *XOR* y el segundo bloque de bits se repite. Los 2 bits, señalados con verde en la parte superior, son los bits que se opera con la operación XOR y el producto se ubica en la segunda fila. Los bits señalados con amarillo, corresponde a los bits que se repite.

Figura 12

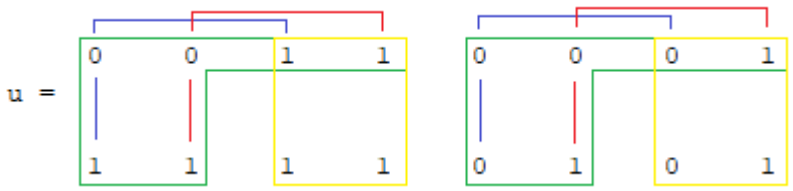
Combinación de $2^1 = 2$ *bits.*



En la Figura 13 se presenta el segundo nivel de profundidad donde se combina 4 bits y se repite los 2 últimos bits. Los 4 bits, señalados con verde en la parte superior, son los bits para operar, los bits señalados con azul y rojo se combina con la operación XOR y el producto se ubica en la segunda fila. Los bits señalados con amarillo, corresponde a los bits que se repite.

Figura 13

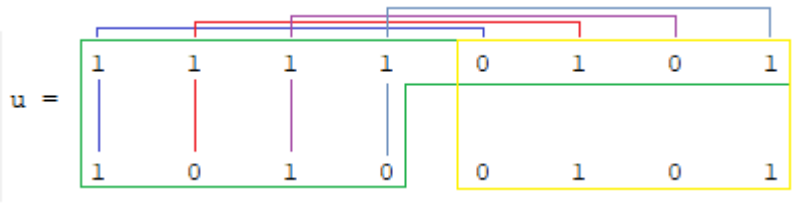
Combinación de $2^2 = 4$ *bits.*



Finalmente, en la Figura 14 se obtiene la palabra codificada en nivel 0 de profundidad. Los bits se combinan en pares con la operación XOR, señalados con azul, rojo, púrpura y gris. Los bits señalados con amarillo, corresponde a los bits que se repite.

Figura 14

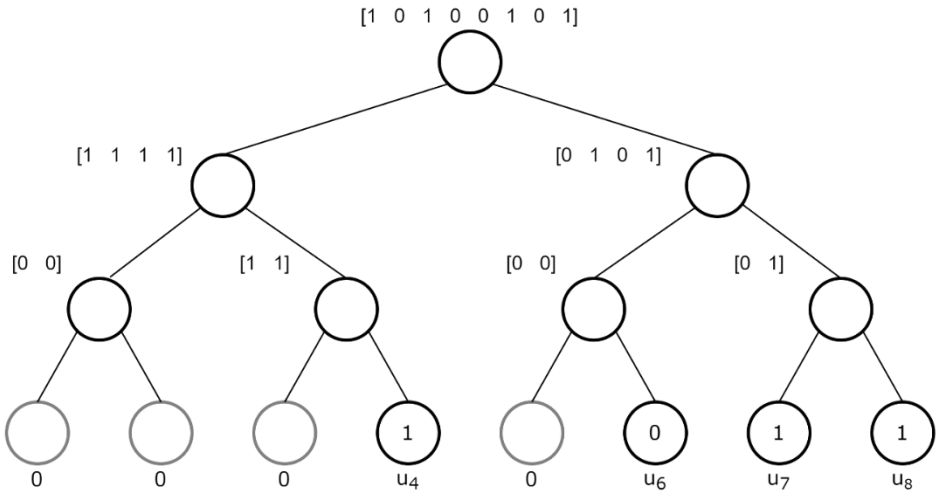
Combinación de $2^3 = 8$ *bits.*



Todo el proceso resumido se lo puede observar desde la representación de *code tree* que se muestra en la Figura 15. El proceso comienza de desde la parte inferior donde se combina los bits a medida que sube de nivel hasta llegar al mensaje codificado en la parte superior.

Figura 15

Codificación a través de un Code Tree $N = 8$ con un mensaje binario de 4 bits aleatorios.



Capítulo III

Decodificadores polares

Decodificador Successive Cancellation (SC)

Desde que se introdujo por primera vez en 2009, la codificación polar generalmente está ligada al decodificador SC como un método de decodificación subóptimo. Hoy en día, aunque existen algunos métodos de decodificación que funcionan mejor, la decodificación SC sigue desempeñando un papel integral en la codificación polar.

Los *butterfly units* del codificador polar introduce la correlación entre los bits de origen, de modo que cada bit codificado con un índice dado se basa en todos sus bits precedentes con índices más pequeños. Este tipo de correlación puede tratarse conceptualmente como una interferencia en el dominio de bits de origen que, cuando es aprovechado, conduce a un rendimiento de decodificación mucho mejor, por lo tanto, constituye la idea central de un algoritmo de decodificación básico, llamado decodificación SC. La cancelación sucesiva de la "interferencia" provocada por los bits anteriores mejora la fiabilidad en la recuperación de los bits fuente. Debido a la estructura regular de los códigos polares, el algoritmo SC se puede describir en términos de un *trellis* o una estructura de *code tree*.

La decodificación SC puede verse como un algoritmo de paso de mensajes soft/hard sobre la estructura *trellis* del código polar. El *trellis* consta de n etapas y N niveles. Cada etapa incluye $N/2$ *butterfly units* y cada unidad contiene un par de verificación y nodo variable. El decodificador SC actualiza los mensajes paso a paso y decide secuencialmente la estimación bit a bit. La Figura 7a mostró la estructura *trellis* de un código polar de longitud $N = 8$. Los mensajes *hard* propagados sobre estructura *trellis* son los bits de estimación correspondientes a los nodos variables designados como $s_{i,j}$, donde i y j indican la etapa y el índice de nivel en el *trellis*, respectivamente, donde $1 \leq i \leq n$ y $1 \leq j \leq N$. Los mensajes *soft* correspondientes

a estos bits son los valores LLR $L_{i,j} = L_{(s_i,j)}$. En este ejemplo, los valores de bit (en el lado izquierdo) del *trellis* son las estimaciones del bloque fuente, es decir, $s_{1,j} = \hat{u}_j$ y los mensajes *soft* correspondientes son $L_{1,j} = L(\hat{u}_j)$. Sea y_j la señal recibida después de la permutación de inversión de bits, el LLR correspondiente está dado por

$$L_{n+1,j} = \log \frac{P(y_j|u_i = 1)}{P(y_j|u_i = 0)} \tag{17}$$

La regla de actualización y decisión se puede expresar de la siguiente manera.

Regla de actualización de mensajes soft

Observe la ecuación 18, donde $i = 1, 2, \dots, n$, $j = 1, 2, \dots, N$ y $\lfloor \cdot \rfloor$ es la operación *floor*.

$$L_i^j = \begin{cases} 2\tanh^{-1}\left[\tanh\left(\frac{L_{i+1,j}}{2}\right) \cdot \tanh\left(\frac{L_{i+1,j-2^{i-1}}}{2}\right)\right], & \left\lfloor \frac{j-1}{2^{i-1}} \right\rfloor mod\ 2 = 0 \\ \left(1 - 2s_{i,j-2^{i-1}}\right)\left(L_{i+1,j-2^{i-1}}\right) + L_{i+1,j}, & caso\ contrario \end{cases} \tag{18}$$

Se define el mensaje *soft* en el nodo de verificación con índice de nivel que satisface $\left\lfloor \frac{j-1}{2^{i-1}} \right\rfloor mod\ 2 = 0$ se actualiza utilizando la primera fórmula de la ecuación 18, que es similar al cálculo en el nodo de verificación de los códigos LDPC, mientras que la actualización del mensaje en el nodo variable se realiza mediante la segunda fórmula de la ecuación, lo que requiere el conocimiento de las restricciones del nodo variable y del mensaje *hard*.

Regla de actualización de mensajes hard

Partiendo de la definición de la ecuación 19 y las otras notaciones son las mismas con la ecuación 18

$$s_{i+1,j} = \begin{cases} s_{i,} \oplus s_{i,j-2^{i-1}}, & \left\lfloor \frac{j-1}{2^{i-1}} \right\rfloor mod\ 2 = 0 \\ s_{i,j}, & caso\ contrario \end{cases} \quad (19)$$

Se define el mensaje *hard* en el nodo de verificación con índice de nivel específico se actualiza utilizando la primera fórmula de la ecuación 19; de lo contrario, el mensaje *hard* se actualiza mediante la segunda fórmula.

Regla de decisión

En la etapa 1, la regla de decisión de bit es la siguiente: para un bit de información, si los mensajes *soft* $L_{1,j} \geq 0$, entonces $\hat{u}_i = 0$, en caso contrario $\hat{u}_i = 1$; para un bit congelado, simplemente se configura en un valor predeterminado, $\hat{u}_i = 0$.

En el ejemplo de *Trellis* mostrado en la Figura 7a, el decodificador SC realiza los cálculos de mensajes *soft* y pasa a través del *Trellis* de derecha a izquierda; luego, los mensajes *hard* se calculan y se propagan en dirección inversa.

Como ejemplo, en la primera *butterfly unit* (desde la parte superior del *trellis*) de la etapa 3, el mensaje *soft* $L_{3,1}$ se calcula utilizando $L_{4,1}$ y $L_{4,5}$. Además, como otra instancia, en la segunda *butterfly unit* (desde la parte superior del *trellis*) de la etapa 1, los mensajes suaves $L_{2,4}$ y $L_{2,3}$ se envían a los nodos de verificación relacionados después de los pasos 1 y 2 (ilustrados por el flechas y números), respectivamente. Después de los pasos 3 al 6, la ecuación obtiene el mensaje *soft* $L_{1,3}$ es obtenido mediante la ecuación 18. Sin embargo, no importa qué valor tome $L_{1,3}$, el mensaje *hard* correspondiente $s_{1,3} = 0$ se envía de vuelta al nodo de verificación porque el bit u_3 es un bit congelado.

Luego, el mensaje *soft* $L_{1,4}$ se calcula en el paso 7 usando la ecuación 18 con el mensaje *hard* $s_{1,3}$ y los mensajes suaves ($L_{2,4}$ y $L_{2,3}$). El mensaje *hard* correspondiente $s_{1,4}$ se envía de vuelta a los nodos de verificación en los pasos 8 y 9, respectivamente. En los pasos 10 y 11, el mensaje *hard* $s_{2,3}$ y $s_{2,4}$ se calculan usando la ecuación 19.

Una implementación directa del decodificador SC es activar las *butterfly unit* una por una. Este método de procesamiento secuencial tiene un rendimiento deficiente. Si se activa simultáneamente múltiples *butterfly unit* se puede lograr un procesamiento con mayor rendimiento. Los cálculos de mensajes *soft* en una *butterfly unit* se puede considerar como una operación básica en la decodificación SC. Dado que el *trellis* consta de $(N/2)log_2(N)$ *butterfly unit*, la complejidad general del decodificador SC es $O(NlogN)$ y requiere $O(NlogN)$ unidades de memoria para almacenar los valores LLR.

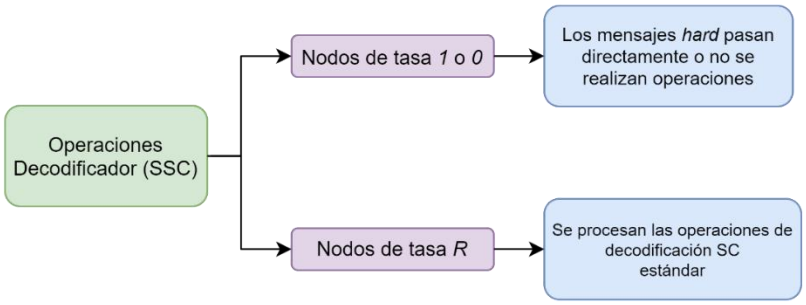
Decodificador Simplified Successive Cancellation (SSC)

Desde un punto de vista práctico, reducir la complejidad de la decodificación es un tema importante. (Alamdar-Yazdi & Kschischang, 2011) propuso un decodificador *Simplified Successive Cancellation* para disminuir los cálculos redundantes en la decodificación SC sin afectar la característica de error. Según la característica de la polarización progresiva, los códigos polares se pueden presentar mediante un c*ode tree* de nivel compacto, como se muestra en la Figura 7b. En la tercera etapa solo se mantiene dos niveles porque todos los canales están polarizados en dos tipos. Del mismo modo, hay cuatro y ocho niveles en las otras dos etapas, respectivamente. Todos los nodos del c*ode tree* se pueden dividir en tres tipos: nodos de tasa cero cuyos descendientes son todos los bits congelados; nodos de tasa uno que son todos los bits de información y nodos de tasa R que son los bits congelados parciales y de información.

Las operaciones que intervienen en la decodificación SSC se pueden resumir como se muestra en la Figura 16

Figura 16

Operaciones del Decodificador SSC



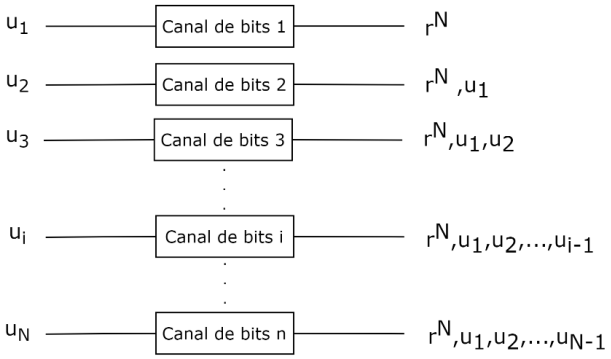
En comparación con la decodificación SC estándar, la complejidad se reduce aproximadamente de 2 a 20 veces por la simplificación en los nodos de tasa cero y tasa uno.

En la práctica, los canales de bits que se crea en el codificador deben recrearse en el decodificador. La idea básica de polarización es combinar todos los bits de información utilizando la matriz G que se presentó en el capítulo 2, luego se divide en un conjunto de N canales de bits donde cada bit u_i ingresa por ese nuevo canal y se obtiene una salida distinta.

La primera salida para u_1 es el vector completo r^N. Para el segundo canal de bits se tiene el vector completo r^N y el bit anterior u_1, entonces, para le tercer canal de bits nuevamente se tiene el vector completo r^N y los dos bits anteriores. La representación de esta idea básica se la puede visualizar en la Figura 17.

Figura 17

Construcción de los canales de bits.

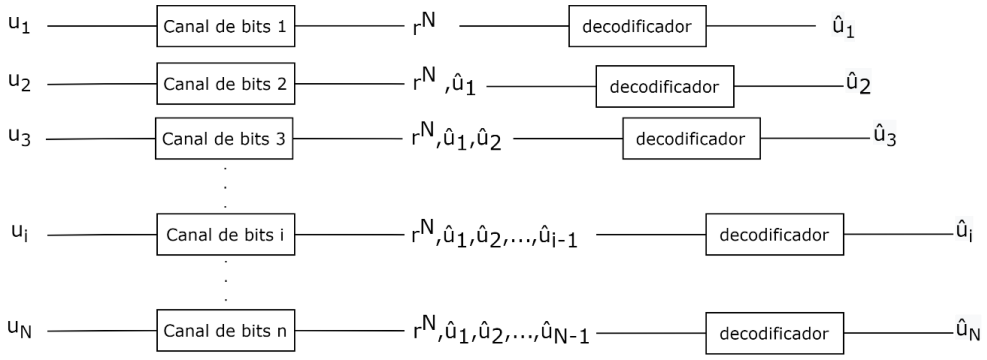


¿Ahora, se puede obtener el bit anterior a partir del segundo canal de bits? Tal vez no existe una forma exacta de obtener ese bit anterior, pero si es posible estimarlo. Nuevamente surge otra interrogante, es posible decodificar correctamente u_1 a partir de r^N?. En algunos casos u_1 puede ser un *frozen bit* por lo que no sería necesario decodificarlo y directamente se puede determinar que es cero.

Observe la Figura 18 secuencialmente desde la parte superior. Si se recibe r^N se realiza una decodificación y se obtiene $\hat{u}_1$. Se debe considerar que si el bit u_i es un *frozen bit*, no decodifica y $\hat{u}_i = 0$; si el bit u_i es un bit de información, se lo decodifica con las reglas de decisión de mensajes *soft* y *hard*.

Figura 18

Decodificación secuencial de los canales de bits.



En resumen, los canales de bits pueden recrearse en el decodificador. Una vez que se obtiene $\hat{u}_1$ junto con r^N se lo utiliza como entrada para decodificar $\hat{u}_2$. Después de obtener $\hat{u}_2$ se lo utiliza para obtener $\hat{u}_3$ junto con r^N y $\hat{u}_1$ y así sucesivamente. Los *frozen bits* demuestran su importancia ya que, si los canales de bit no son realmente confiables, eventualmente el decodificador fallará y aquí es donde cumple su función, por lo que al congelar en posiciones frecuentes se sabe con certeza que el $\hat{u}_i$ es igual a 0 y no propagará errores en las decodificaciones posteriores.

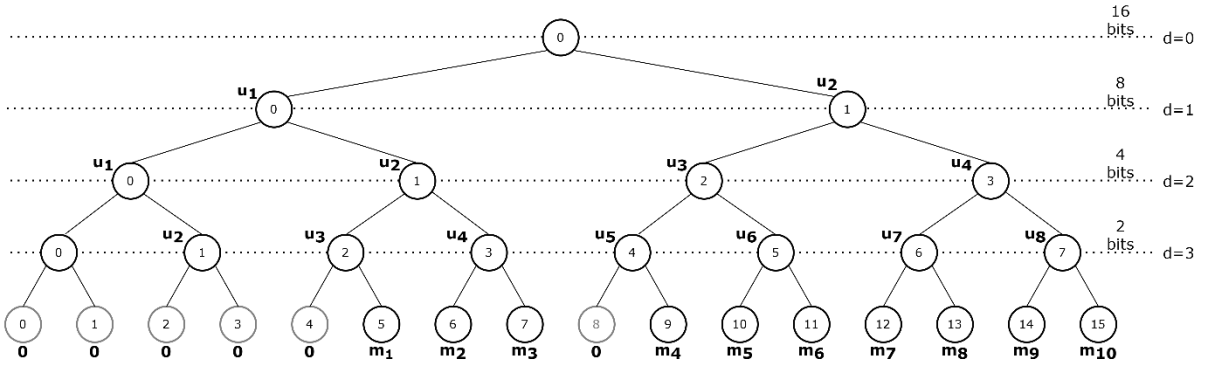
Notación

Un pequeño ejemplo de un decodificador (16,10), con la representación de *code tree,* se puede observar en la Figura 19 donde $N = 16$ y $K = 10$, nuevamente d determina el nivel y en cada nivel se puede observar que se opera 2^n bits. Se numera los nodos en cada nivel, entonces en cada nivel d se tiene de 0 a $2^d - 1$ nodos de izquierda a derecha. El nodo superior $d = 0$ es el nodo raíz. Para cada nodo n en la profundidad d es necesario identificar el *'hijo izquierdo'*, el *'hijo derecho'* y el *'padre'*. Se observa que hay un patrón para el 'hijo izquierdo', es múltiplo de 2 por lo que se lo puede generalizar como $2n$ en cada $d + 1$. De forma análoga,

el *'hijo derecho'* se lo puede generalizar como $2n + 1$ en cada nivel $d + 1$. El *'padre'* se lo obtiene con la operación *floor,* descrito para las ecuaciones 18 y 19, de $\left\lfloor \frac{n}{2} \right\rfloor$ en el nivel $d - 1$.

Figura 19

Representación Code Tree para un decodificador (16,10).



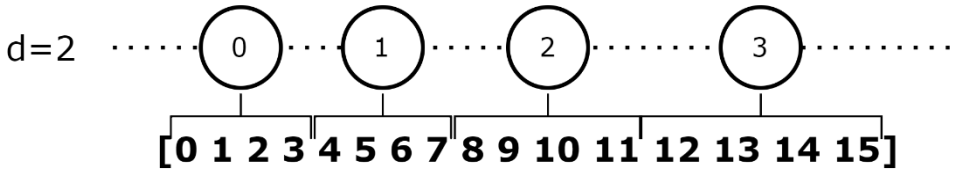
Una vez que se tiene esta notación de profundidad, nodo derecho y nodo izquierdo se puede encontrar fácilmente los *hijos* y el *padre.*

Almacenamiento

El proceso empieza desde la parte superior en el nivel 0 donde se recibe las 16 'creencias' que son bits recibidos o 'creencias', en el nodo 0 del nivel 1 tiene 8 'creencias' y el nodo 1 también tiene 8 'creencias', para el nivel 2 se tiene 4 'creencias' en cada nodo. En general, se observa que en cada nivel se tiene 16 'creencias' por lo que se almacenará en una matriz I de 5 filas y 16 columnas, para este caso. Una vez almacenadas estas 'creencias' se las puede asociar con los nodos como se observa en la Figura 20, entonces, cada nodo i en el nivel d tiene 2^{n-d} 'creencias' entrantes, es decir, se agrupa todos los nodos 'hijo' en la matriz I pero se asocia en diferentes bloques asignados los nodos 'padres' para facilitar el almacenamiento.

Figura 20

Agrupación de bits a los nodos en el nivel $d = 2$



De forma general I es una matriz bidimensional, las filas tendrán n elementos y las columnas tendrán $2^n - 1$ elementos, así $I[0{:}\,n\,, 0{:}\,2^n - 1]$.

Un detalle que es importante mencionar en este proceso, a diferencia del codificador en el que se parte de la parte inferior del *code tree*, y los siguientes pasos consecutivos progresan hasta llegar al nivel 0 y no existe la necesidad de almacenar ningún elemento intermedio, es decir, solo avanza en una dirección. Por su lado, el decodificador no tiene una dirección fija, a medida que avanza en los niveles hacia abajo, vuelve a subir y en ese paso intermedio necesita recordar el mensaje o 'creencias' que recibió por lo que es necesario almacenar todos los mensajes que fluyen en cada nivel.

Estados

Para establecer los estados que se utilizará en el proceso de decodificación se definirá 3 pasos básicos: L, R, U que resumirá el proceso descrito anteriormente. Estas operaciones suceden en cada nodo a partir del nivel 1 por lo que son operaciones que ocurren al interior del nodo. Los nodos del último nivel del *code tree* se lo conoce como 'nodo hoja'

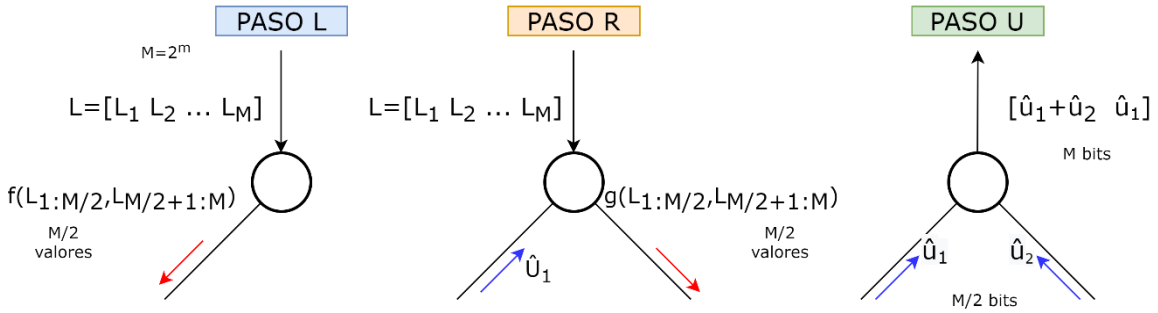
El primer paso L , está representado por el vector L que recibirá un conjunto de M 'creencias' entrantes y lo divide en dos partes; la primera parte tiene una longitud de $M/2$ elementos cuyos índices van desde 1 hasta $M/2$ y los índices de la segunda parte van de

$M/2+1$ hasta M luego de ejecutar la operación da como resultado $\hat{u}_1$ de $M/2$ bits que son los resultados del 'hijo izquierdo'

El siguiente paso es la operación R, realiza una operación que está dada por la segunda parte ecuación 18. Una vez más L se divide en dos partes, opera los dos vectores y aplica la regla de decisión junto con los valores que se obtiene del paso L obteniendo así $\hat{u}_2$ de $M/2$ bits. La Figura 21 presenta los estados L, R y U y las operaciones que intervienen en cada estado.

Figura 21

Estados L, R y U utilizados en la decodificación SSC.



Una vez que hayan finalizado los pasos L y R, en el paso *U*, los valores obtenidos $\hat{u}_1$ y $\hat{u}_2$ regresan al nodo y se combinan $[\hat{u}_1 \oplus \hat{u}_2 \;\; \hat{u}_2]$ dando como resultado un vector de M bits, esto a su vez será los valores que se enviarán al siguiente nivel para continuar operando.

Para almacenar los estados se utilizará una notación ligeramente distinta, esta notación está dada por la tabla 1.

Tabla 1

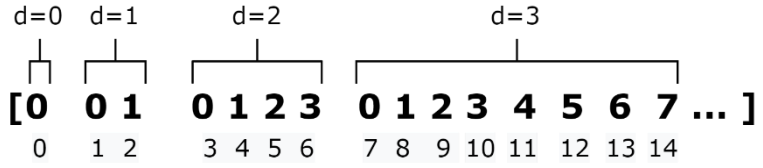
Notación utilizada para describir el estado de las operaciones.

Notación	Estado
0	Aún por activar
1	Paso L terminado
2	Paso R terminado
3	Paso U terminado

En la Figura 22 se observa que en cada nivel d existe 2^d nodos, estos valores se almacenará en un vector lineal.

Figura 22

Número de nodos en cada nivel d.



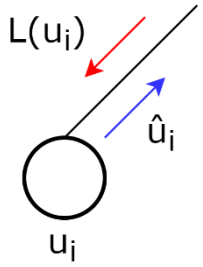
Entonces, para cada nodo i en el nivel d está la posición $(2^d - 1) + i$, por ejemplo, si se ubica en el nivel 2, posición 2 será $(3) + 2$ que concuerda con la quinta ubicación. Entonces, el vector estado empieza en 0 y termina en $2^{n+1} - 2$ por lo que la longitud total será $2^{n+1} - 1$. Inicialmente el vector estado almacenará todos los 0 y cada nodo en particular pasará por todos los estados que indica la tabla 1.

Se debe diferenciar entre nodos y 'nodos hoja', a diferencia de los nodos, los 'nodos hoja' no tienen hijos y son los nodos finales del *code tree*. El proceso que ocurre en los 'nodos hoja' es simple, se recibe una 'creencia' $L(u_i)$ a la vez y esta puede estar en dos situaciones

diferentes, la posición i puede corresponder a un *frozen bit* o a un bit de mensaje como se observa en la Figura 23.

Figura 23

Representación de un *'nodo hoja'.*



Si i corresponde a una posición congelada $\hat{u}_i = 0$, Si i corresponde a una posición de mensaje $\begin{cases} \hat{u}_i = 0 & si \quad L(u_i) \geq 0 \\ \hat{u}_i = 1 & si \quad L(u_i) < 0 \end{cases}$

Secuencia de operación

El decodificador funciona en una secuencia determinada que comienza en el nodo raíz o el nodo del nivel 0. En los nodos de los siguientes niveles, si no es un 'nodo hoja', se activa cuando recibe las 'creencias' y el nodo anterior se desactiva. Luego se ejecuta el paso L y pasa al 'hijo izquierdo' de ese nodo. Cuando se ha recibido la decisión del 'hijo izquierdo', realiza el paso R y el control pasa al 'hijo derecho'. Cuando recibe la decisión del 'hijo derecho' se ejecuta el paso U y le envía una decisión de vuelta al padre, una vez que se haya recibido ambas decisiones este recupera el control y continúa el proceso hasta llegar a los 'nodos hoja'.

Si es un 'nodo hoja', cuando recibe una 'creencia' verifica si está congelada o no. Si está congelada envía un 0 de vuelta o toma una decisión como se describió en la sección anterior y transfiere el control a su nodo 'padre'.

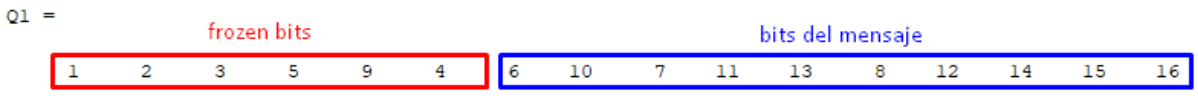
La razón para que este proceso funcione en una secuencia determinada es que los canales de bits están ordenados de acuerdo con una secuencia de confiabilidad desde canales malos a canales muy buenos congelando ciertos canales para evitar la propagación de errores.

Funcionamiento

Partiendo del ejemplo mencionado en el que *N=16* y *K=10*, en la Figura 24 se calcula las posiciones de los *frozen bits* con la secuencia de confiabilidad, el resto de las posiciones de bits ocuparán los bits del mensaje.

Figura 24

Posiciones de los $N - K = 6$ *frozen bits y bits de mensaje.*



Luego que el mensaje ha sido codificado se lo transmite a través de un canal AWGN con modulación BPSK. Los valores recibidos del canal se almacenan en un vector Ln que se muestra en la Figura 25 y corresponde a la raíz en el nivel $d = 0$.

Figura 25

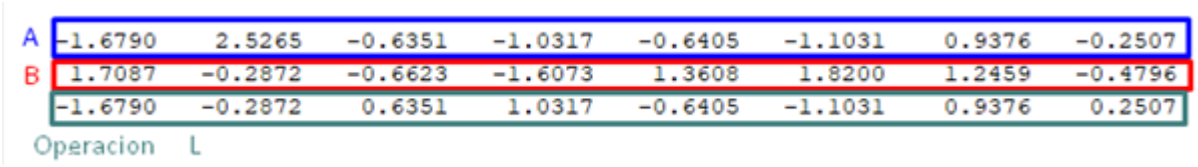
Valores recibidos por el decodificador luego de una transmisión a través de un canal AWGN con una modulación BPSK.

```
Ln =

  Columns 1 through 12

   -1.6790    2.5265   -0.6351   -1.0317   -0.6405   -1.1031    0.9376   -0.2507    1.7087   -0.2872   -0.6623   -1.6073

  Columns 13 through 16

    1.3608    1.8200    1.2459   -0.4796
```

Para continuar con el nodo 0 del nivel 1, el vector Ln se divide en 2 partes a los que se denominará A, para la primera parte y B para la segunda parte, también se ejecuta el paso *L* donde ejecuta la regla de actualización de mensajes *soft* que está dada por la primera parte de la ecuación 18. La tercera fila que indica la Figura 26 indica el resultado tal operación.

Figura 26

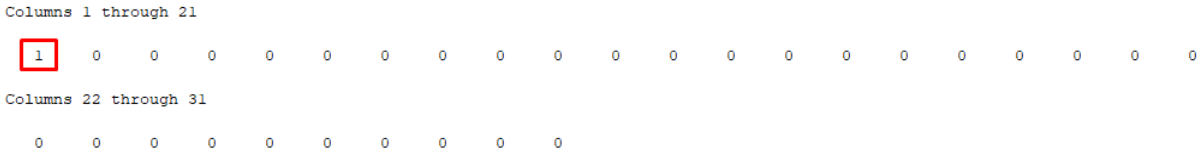
Resultado de la operación L entre los vectores A y B.



En la Figura 27 se puede comprobar que la matriz de estados en el que indica que se encuentra realizando el paso L de acuerdo con la Tabla 1.

Figura 27

Matriz de estados en la primera interacción.

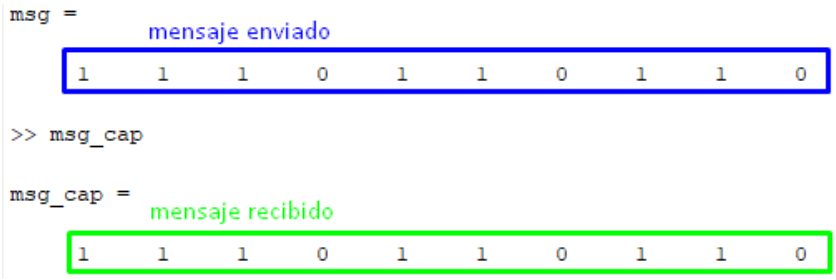


El proceso avanza al nodo 0 del siguiente nivel hasta llegar a un 'nodo hoja' para continuar con los pasos R y U. Una vez que se encuentre en un 'nodo hoja' procederá con el paso R, envía una decisión al padre y pasará el control al siguiente nodo. Se puede verificar, el nodo 0 en el nivel 1 a la izquierda, continúa al nodo 0 en el nivel 1 hasta llegar al nodo 0 del nivel 3, es decir al 'nodo hoja'. Luego pasa a la derecha del nivel 3 y sube hasta nodo 2 del nivel 0 donde nuevamente pasa a la derecha y continuar con el proceso.

Una vez que el proceso de decodificación ha concluido, se obtiene el mensaje recibido ya decodificado. En la Figura 28, en la parte superior, se observa los bits de mensaje enviados, y los bits de mensaje recibido, en la parte inferior, ya decodificado donde además se verifica que el decodificador no presentó errores en el proceso.

Figura 28

Palabra de código enviada por el codificador (azul) y mensaje que se ha recibido luego de ser decodificado (verde).



Decodificador Successive Cancellation List (SCL)

La decodificación Successive Cancellation List (SCL) propuesta en (Ido & Vardy, 2011) es una extensión del decodificador básico Successive Cancellation (SC). En este decodificador, en cada etapa de decodificación, se considera L caminos de decodificación simultáneamente, siendo L un número entero. Una vez que el proceso de decodificación llega a su fin, la ruta que se seleccionará es la ruta más probable y se designa como la única palabra de código en la salida del decodificador.

La cantidad de rutas de decodificación para bits de información individual se puede duplicar mediante el uso de un algoritmo de codificación de lista preciso que logra esta presentación; más tarde, las rutas L más probables se guardan y todas las demás rutas se descartan usando un proceso denominado *proceso de poda*. Se define un parámetro conocido como tamaño de lista L. Por lo general, un valor de L más alto logra tasas de error menores, pero tiempos de ejecución más aumenta y con ello, mayor uso de memoria.

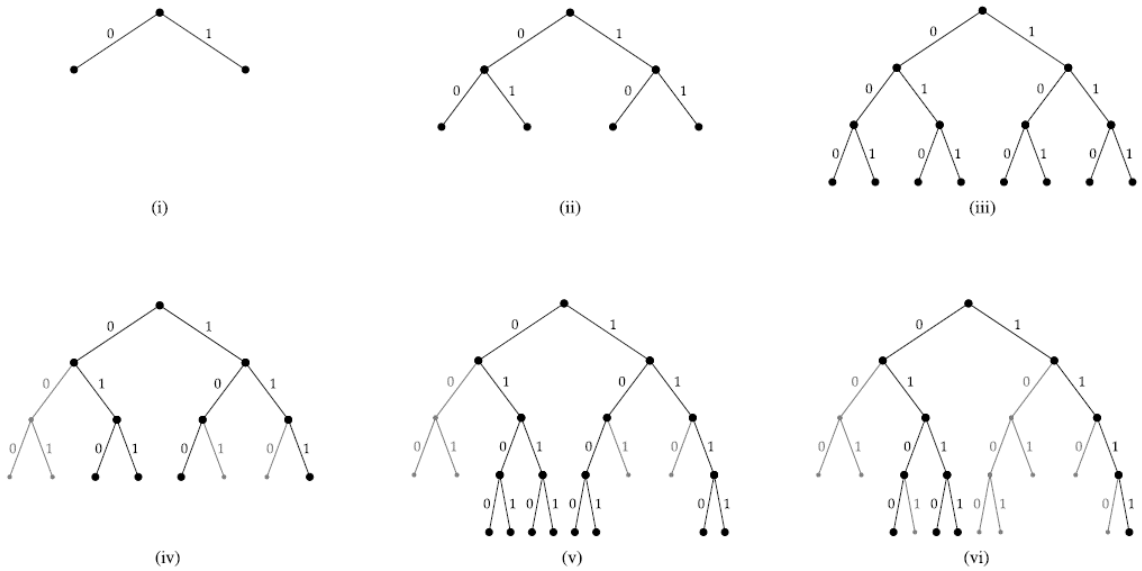
A diferencia del decodificador SC donde en cada fase se debe decidir el valor de u_i, en un decodificador SCL, en lugar de decidir el valor de u_i no congelado en “0” o “1”, se inspecciona ambas opciones. Es decir, se deja que un "camino" sea una determinada decisión sobre los valores de u_0^i, para $0 \leq i \leq n$

En la Figura 29, por simplicidad, se supone que $n = 4$ y todos los bits están descongelados. El tamaño de la lista es $L = 4$, cada nivel tiene como máximo 4 nodos con rutas que continúan hacia abajo. Los caminos descontinuados son de color gris.

Al decodificar un bit no congelado u_{i+1}, se divide la ruta de decodificación en dos nuevas rutas. Ambas rutas tendrán u_0^i como prefijo. Una de las nuevas rutas terminará en "0" mientras que la otra terminará en "1". Dado que cada división duplica el número de rutas a examinar, se debe podar, y la cantidad máxima de rutas permitidas es el tamaño de lista especificado, L. Naturalmente, se buscaría mantener las "mejores" rutas en cada etapa, por lo tanto, requieren un criterio de poda. El criterio de poda será mantener los caminos más probables.

Figura 29

Evolución de las rutas de decodificación.



En la figura 29 se indica que la figura (i) Inicia el algoritmo. El primer bit no congelado puede ser 0 o 1. (ii) Los segundos bits no congelado pueden ser 0 o 1. El número de rutas no es superior a L = 4, por lo que aún no es necesario podar. (iii) Teniendo en cuenta todas las opciones para el primer, segundo y tercer bit, se obtienen 8 rutas de decodificación; demasiado, ya que L = 4. (iv) Se poda los 8 caminos en L = 4 caminos más prometedores. (v) Continúa las 4 rutas activas considerando ambas opciones del cuarto bit no congelado. El número de caminos se duplica a 8, que es demasiado (L = 4). (vi) Nuevamente, se poda a L = 4 mejores caminos.

En cuanto a la implementación en (Ido & Vardy, 2011) sugiere una estructura eficiente en el espacio para implementar el decodificador SC y las complejidades $O(N \log N)$ de tiempo y $O(N)$ de espacio. Una implementación directa del decodificador SCL requerirá $O(LN^2)$ cálculos. Se introduce una técnica denominada "lazy copy" basada en la estructura de intercambio de memoria entre las rutas candidatas para reducir las operaciones de copia redundantes. Por lo tanto, el decodificador SCL se puede implementar con complejidad de tiempo $O(LN \log N)$. Al igual que para los decodificadores SC y SCL, estas características también se pueden aplicar en la implementación de los decodificadores SC y SSC. Los cálculos reales de SSC, suponiendo que la profundidad del *stack* D sea lo suficientemente grande, se vuelven mucho menores que los de SCL cuando se trabaja en el régimen de relación señal/ruido (SNR) moderada o alta.

Decodificación Successive Cancellation List asistidas por CRC (SCL+CRC)

Cyclic redundancy check (CRC) es la técnica más utilizada en la detección y corrección de errores en el espacio de la teoría y la codificación de la información. También se usa ampliamente en estándares celulares. Por ejemplo, (3GPP) incorpora CRC ampliamente en la mayoría de sus tecnologías de acceso por radio como UMTS, LTE, etc.

Si se considera un bloque de entrada de longitud K del codificador de corrección de errores, k bits de información de longitud y la longitud de la secuencia de CRC m bits, entonces, $K = k + m$. Los bits CRC se pueden considerar como parte de los bits de origen para el código de corrección de errores, la tasa de código R se definiría como $R = K/N$.

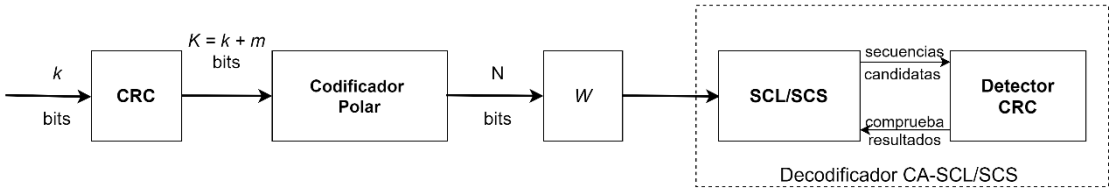
Para mejorar aún más el rendimiento de los códigos polares, recientemente se han propuesto esquemas de decodificación SCL/SCS asistidos por CRC, como CA-CL/CA-SCS, propuestos por (Niu, Kai, Lin, & Zhang., 2014) y (Niu & Chen, CRC-Aided Decoding of Polar Codes, 2012). En estos esquemas, el decodificador SCL/SCS envía los caminos candidatos a un detector CRC, y los resultados de la verificación se utilizan para detectar la palabra de código correcta. Bajo estos esquemas de decodificación asistida por CRC, el rendimiento de los códigos polares mejora sustancialmente y supera al de la decodificación *maximal likelihood* (ML).

Como se observa en el receptor de la Figura 30, el decodificador SCL/SCS envía las secuencias candidatas al detector CRC y este último retroalimenta los resultados de la verificación para ayudar a la determinación de la palabra de código.

El criterio de parada es uno de los aspectos más importantes a considerar cuando hablamos de decodificación asistida por CRC. Cuando se emplea un proceso de decodificación iterativo, los resultados de CRC proporcionan un criterio de parada o inician las solicitudes de retransmisión. Tradicionalmente, los resultados de CRC proporcionan un criterio de parada para el proceso de decodificación iterativo o inician las solicitudes de retransmisión.

Figura 30

Decodificador polar asistido por CRC.

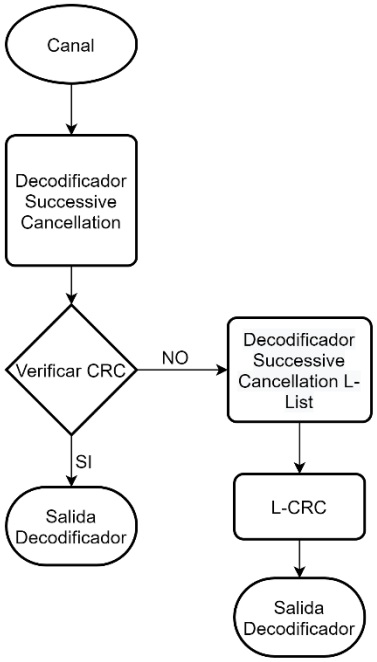


El rendimiento del algoritmo SC se puede mejorar utilizando el algoritmo de decodificación SCL que reconoce las mejores rutas de decodificación de L juntas. El rendimiento se puede mejorar aún más introduciendo CRC en la decodificación SCL seleccionando una ruta válida de CRC entre las L mejores rutas de decodificación al final de la decodificación. Sin embargo, el algoritmo de decodificación SCL adolece de una latencia prolongada y un rendimiento bajo debido a los cálculos $O(L\,N\log N)$ de alta complejidad a medida que aumentan L y N.

(Huilgol, 2017) propone una mejora en el rendimiento de SCL mediante el uso de un decodificador adaptativo que proporciona el rendimiento del SC con la productividad de SCL. La Figura 31 presenta un diagrama de flujo del decodificador, el cual abarca tres componentes principales, decodificadores SC, SCL y CRC. Inicialmente, se activa el decodificador SC y se calcula un vector de estimación de *hard decision*. Después de eso, el decodificador CRC controla si el vector de *hard decision* es correcto. Si el CRC es válido, es muy probable que el vector de *hard decision* sea el vector de información correcto. En este caso, el decodificador adaptativo se termina inmediatamente sin la activación del decodificador SCL.

Figura 31

Diagrama de Flujo de un decodificador asistido por CRC.



En otro caso, cuando el CRC no es válido, se activa el decodificador SCL y se genera L vectores de información candidatos. Entre estos candidatos, el decodificador CRC selecciona un candidato, que tiene un vector CRC válido. Si más de un candidato de vector CRC es válido, se selecciona el más probable entre estos candidatos. Por último, cuando ninguno de los candidatos tiene un vector CRC válido, el decodificador CRC selecciona el vector de estimación de decisión más probable para reducir el BER.

Capítulo IV

Análisis del Desempeño

Con la ayuda del software MatLab (*MATrix Laboratory*, por sus siglas en inglés), se ejecuta una simulación de un sistema de comunicaciones que utiliza una modulación BPSK y la codificación polar sobre un canal AWGN. Este sistema permitirá obtener métricas como el BER y FER, medidas de desempeño que permiten obtener el porcentaje de bits errados que se obtiene en el receptor comparando el flujo de bits a transmitir con el flujo de bits decodificados. Para la evaluación del desempeño se configura diferentes longitudes de mensaje y tasas de código.

Se configura parámetros de codificación utilizados para la simulación y se define que: N es la longitud de la palabra de código y K es la longitud del mensaje. Es importante considerar los valores que proporciona el informe técnico (3GPP, 5G; Study on new radio access technology Physical layer aspects, 2017) en la Sección 8.2.1.5.2 y en la Sección 8.3.1.5.2 donde especifica la longitud máxima del mensaje codificado de acuerdo con la Tabla 2.

Tabla 2

Longitud máxima del mensaje codificado para el enlace descendente y ascendente.

Tamaño máximo del mensaje codificado con código Polar	
$\boldsymbol{N_{max,DCI} = 512}$	Información de control del enlace descendente
$\boldsymbol{N_{max,UCI} = 1024}$	Información de control del enlace ascendente

Determinar la información y las posiciones de los bits congelados se puede realizar en una forma recursiva de ecuaciones simples en forma cerrada como se discutió en el capítulo 2. En un sistema práctico esto implica tiempo de procesamiento. Una forma posible de evitar

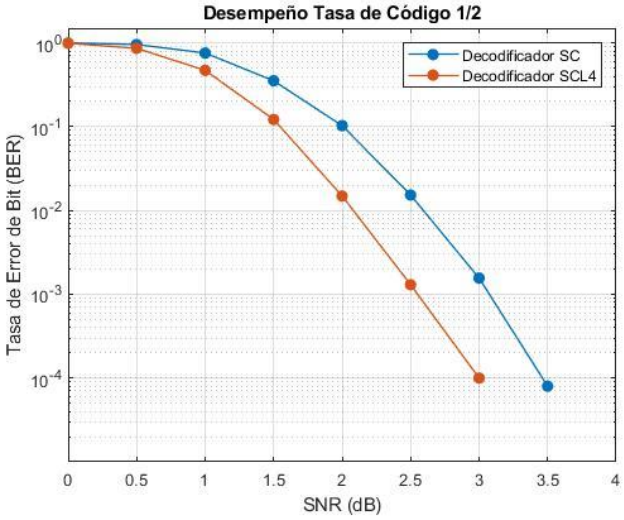
el desafío numérico antes mencionado, para la determinación de la secuencia de información, sería almacenar la secuencia de información pre calculada, esta secuencia de confiabilidad proporciona el (3GPP, 5G; NR; Multiplexing and channel coding, 2018) en la sección 5.3.1.2 y se utilizará de aquí en adelante para la evaluación del desempeño.

Los resultados presentados a continuación corresponden al decodificador SC y SCL con $L = 4$ y $CRC = 11$. Para cada punto de SNR se generó simulando, hasta 1000 errores de trama o un máximo de 10000 tramas, lo que ocurra primero.

En la Figura 32 se observa los resultados a una tasa de codificación $R = 1/2$ donde a partir de $SNR = 3$, la tasa de error de bit que se obtiene en el decodificador SCL es igual a cero, es decir, ya no presenta errores por lo que se puede decir que es ligeramente más eficiente respecto al decodificador SC que ya no presenta errores a partir de $SNR = 3.5$. La diferencia que se tiene entre los dos decodificadores es de $0.5\ dB$ para esta codificación.

Figura 32

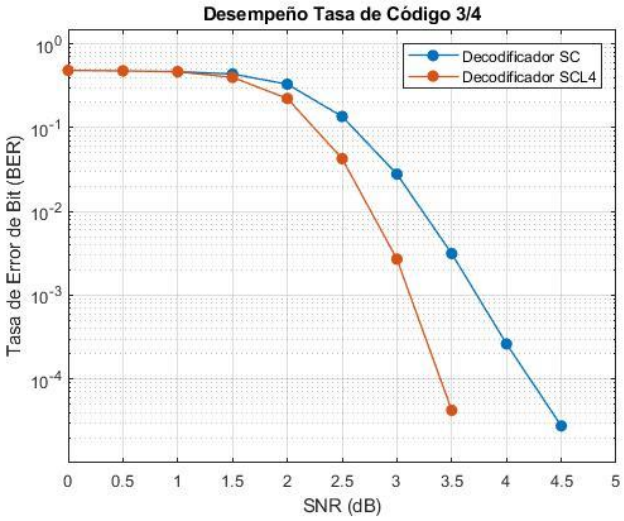
Desempeño a una tasa de código $1/2$*.*



La Figura 33 muestra el desempeño del sistema con una tasa de código igual a $3/4$. En el intervalo $0 \leq SNR \leq 2$, se observa que los decodificadores obtienen un resultado similar. A partir de un $SNR = 2.5$ ya se puede notar una diferencia, sin embargo, a partir de $SNR = 3.5\, dB$ la tasa de error de bit del decodificador SCL es prácticamente cero mientras que el decodificador SC necesita de un $SNR = 4.5$ para no presentar errores.

Figura 33

Desempeño a una tasa de código $3/4$.



En la Figura 34 se puede visualizar el desempeño que se obtuvo para una tasa de código igual a $2/3$. En el rango de $0 \leq SNR \leq 1.5$, los decodificadores presentan un desempeño similar y a medida que este aumenta, el decodificador SCL muestra una mejora significativa respecto al decodificador SC, de hecho, el decodificador SCL ya no presenta errores desde $SNR = 3$ mientras que el decodificador SC no presenta errores a partir de $4\, dB$ de SNR.

En la Figura 35 se observa claramente una diferencia en el desempeño de los decodificadores. El decodificador SCL es mucho más eficiente, ya no presenta errores a partir de un $SNR = 3.5\, db,$ respecto al decodificador SC que empieza a presentar una tasa de error

de bit igual a cero en $5dB$ de SNR. En esta tasa de código evidencia la eficiencia que logra el decodificador SCL, logra una diferencia de $1.5dB$, que, en términos de potencia, es una gran diferencia.

Figura 34

Desempeño a una tasa de código $2/3$.

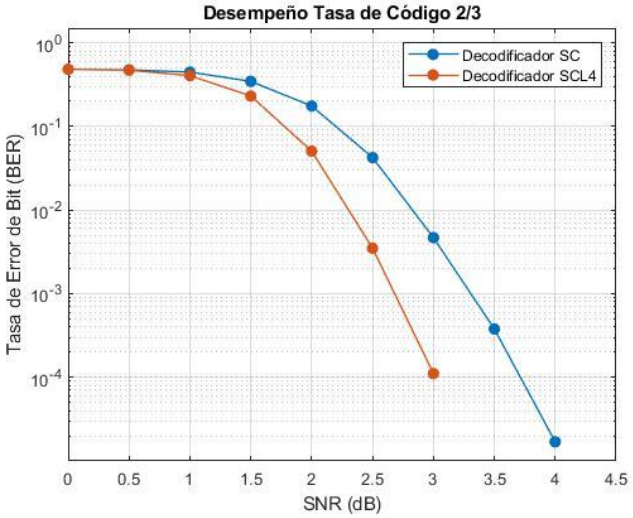
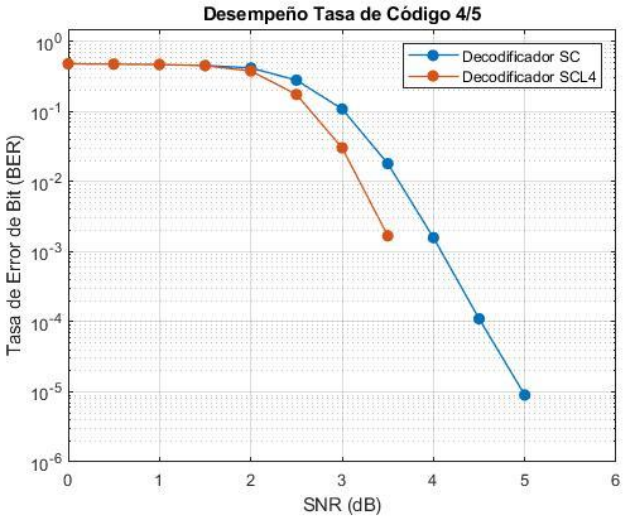


Figura 35

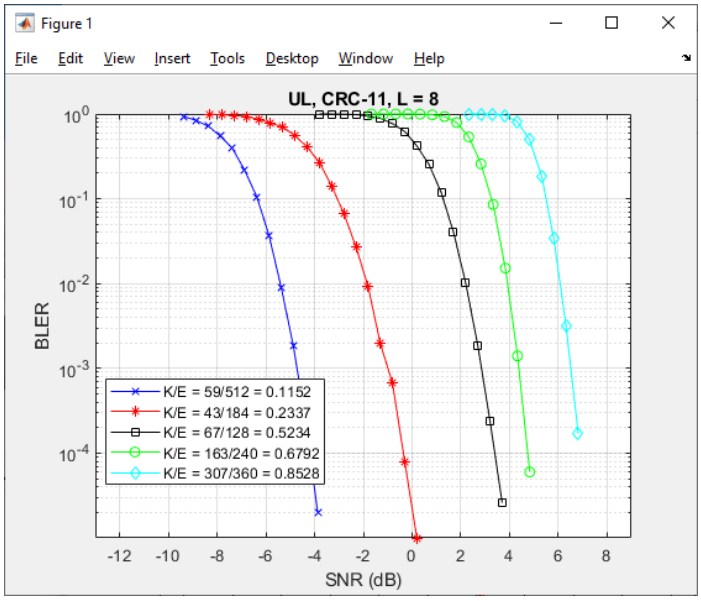
Desempeño a una tasa de código $4/5$.



Para obtener resultados significativos, las simulaciones deben realizarse durante más tiempo, para su efecto MatLab utiliza scripts que encapsulan el procesamiento en una función que admite la generación de código C. En la Figura 35 se presentan los resultados para diferentes velocidades de código y longitudes de mensaje para ambas direcciones de enlace con modulación QPSK. Este ejemplo destaca uno de los esquemas de codificación polar (polar asistido por CRC) especificado por 3GPP para la información del canal de control de 5GNR.

Figura 36

Resultados de simulación para diferentes velocidades de código y longitudes de mensaje.



Nota. Tomado de (MATLAB™, 2021).

La diferencia significativa de rendimiento frente al código evaluado es que las funciones que implementa Matlab muestran el uso de componentes para todas las etapas del procesamiento (codificación, ajuste de velocidad, recuperación de velocidad y decodificación) y los usa en un enlace con QPSK a través de un canal AWGN.

Los resultados de rendimiento entre el código evaluado y las funciones que implementa MatLab para diferentes tasas de código y longitudes de mensajes muestran concordancia con

las tendencias publicadas, dentro de las variaciones de los supuestos paramétricos y de simulación.

Capítulo V

Conclusiones y Recomendaciones

Conclusiones

Este trabajo de investigación presentó los conceptos que se utiliza en el proceso de construcción y codificación, propuesto por Erdal Arikan con varios métodos de decodificación polar propuesto por varios autores. Se analizó un método conocido como polarización de canal que proporciona un esquema de baja complejidad para construir canales polarizados para el escenario eMBB de 5G.

Se simuló un sistema donde permitió observar el desempeño de la codificación polar a diferentes tasas de codificación donde se observó el rendimiento que tiene el decodificador SCL frente al decodificador SC. Los resultados ayudaron a determinar que el decodificador SCL tiene un mayor rendimiento respecto al SC, sin embargo, el decodificador SC es la base de todos los decodificadores polares desarrollados hasta el momento.

El decodificador SC funciona bien para una gran longitud de bloques de código, logra alcanzar un Eb/No de 3.5 dB para una tasa de codificación de $1/2$ y 4.5 dB para una tasa de codificación de 3/4 que es aproximadamente una diferencia de 1dB para un BER de hasta 10^{-4}. Cuando la tasa de codificación se fija en 2/3 el rendimiento mejora logrando una diferencia de 1.5 dB de ganancia con un BER de 10^{-4}.

El decodificador SCL asistido por CRC tiene una mejora de rendimiento de 0,5 dB sobre el decodificador SCL ya que se sabe que los esquemas CRC ampliamente utilizados en esquemas 3GPP mejoran el desempeño resultante que los esquemas básicos.

Recomendaciones

Con base a la bibliografía existente, el estudio de la codificación polar de forma nativa puede tornarse compleja ya que implementa funciones de probabilidad que pueden volverse difícil de entender por lo que se recomienda que empiece revisando la codificación polar simplificada con la representación code tree de nivel compacto.

Durante la evaluación del desempeño se utilizó, en el decodificador SCL únicamente se evaluaron los resultados para un tamaño de lista $L = 4$, sin embargo, se puede extender esta evaluación para una mayor longitud, así mismo, se puede evaluar los resultados para los diferentes polinomios de CRC que propone 3GPP en (3GPP, 5G; NR; Multiplexing and channel coding, 2018).

Bibliografía

3GPP. (2017). *5G; Study on new radio access technology Physical layer aspects.* Sophia Antipolis - France: Telecommunications Standards Institute .

3GPP. (2017). *5G; Study on Scenarios and Requirements for Next Generation Access Technologies.* Sophia Antipolis - France: European Telecommunications Standards Institute.

3GPP. (2018). *5G; NR; Multiplexing and channel coding.* Sophia Antipolis - France: European Telecommunications Standards Institute.

Alamdar-Yazdi, A., & Kschischang, F. R. (2011). A Simplified Successive-Cancellation Decoder for Polar Codes. *IEEE Communications Letters, Vol. 15, No. 12*, 1378-1380.

Arikan, E. (2009). Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Transactions on Information Theory 55, no. 7*, 3051-3073.

Arikan, E. (2011). Systematic polar coding. *IEEE communications letters 15, no. 8*, 860-862.

Arikan, E. (15 de Enero de 2015). *Polar Codes tutorial.* Obtenido de UC berkely: https://simons.berkeley.edu/sites/default/files/docs/2691/slidesarikan.pdf

Bae, J. H., Abotabl, A., Lin, H.-P., Song, K.-B., & Lee, J. (2019). An overview of channel coding for 5G NR cellular communications. *APSIPA Transactions on Signal and Information Processing Vol.7*, 1-14.

Bioglio, V., Condo, C., & Land, I. (2020). Design of Polar Codes in 5G New Radio. *IEEE Communications Surveys and Tutorials*, 1-12.

Dizdar, O., & Erdal, A. (2016). A High-Throughput Energy-Efficient Implementation of Successive Cancellation Decoder for Polar Codes Using Combinational Logic. *IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 63, no. 3*, 436-447.

Huilgol, S. (2017). *Channel Coding Techniques For 5G Using Polar Codes.* Arlington, Texas: The University of Texas at Arlington.

Ido, T., & Vardy, A. (2011). List decoding of polar codes. *Information Theory Proceedings (ISIT) 2011 IEEE International Symposium*, 1-5.

MATLAB™. (2021). *5G New Radio Polar Coding*. Obtenido de Mathworks helper Center: https://ww2.mathworks.cn/help/5g/gs/polar-coding.html

Mori, R., & Tanaka, T. (2009). Performance of Polar Codes with the Construction using Density Evolution. *IEEE Communications Letters, Vol. 13, No. 7*, 519-521.

Niu, K., & Chen, K. (2012). CRC-Aided Decoding of Polar Codes. *IEEE COMMUNICATIONS LETTERS, VOL. 16, NO. 10*, 1668-1671.

Niu, K., Kai, C., Lin, J., & Zhang., Q. T. (2014). Polar codes: Primary concepts and practical decoding algorithms. *IEEE Communications magazine 52, no. 7*, 192-203.

Trifonov, P. (2012). Efficient Design and Decoding of Polar Codes. *IEEE Transactions on Communications, vol. 60, no. 11*, 3221-3227.