Departamento de Teoría de la Señal y Comunicaciones

PROYECTO FIN DE CARRERA

APLICACIÓN DE CÓDIGOS IRA EN UN CANAL RELÉ DEGRADADO GAUSSIANO

Autor: Tamara Benito Matías
Tutor: Ángel Bravo Santos

Leganés, octubre de 2012

Agradecimientos

Quiero dedicar la posible finalización de este proyecto a mis padres, familia y tutor. Si no hubiera sido por su incansable tenacidad y apoyo, no hubiera podido poner punto y final a una de las etapas más importantes de mi vida.

Por ello, espero y deseo que algún día pueda compensar todas esas horas que habéis tenido que sufrir a mi lado.

Gracias de todo corazón.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción y objetivos

1.1 Prefacio

A lo largo de este capítulo y sucesivos se va a proporcionar una visión general de la importancia que ha adquirido el estudio de codificación de red (“Network coding”) aplicada al canal relé, de tal forma que se pueda dar brevemente una pequeña idea de lo que se pretende obtener tras la realización del mismo.

Hoy en día, las redes de comunicación comparten el mismo principio fundamental de operación, ya se trate de paquetes a través de Internet, o señales en una red telefónica. La información se transporta de la misma manera, como vehículos que comparten la misma carretera o fluidos que fluyen por la misma tubería. Es decir, los flujos independientes de datos pueden compartir recursos de red, pero la información en sí misma es independiente. Enrutamiento, almacenamiento de datos, control de datos, control de errores, y en general, todas las funciones de red se basan en este supuesto.

La codificación de red [1] es un campo reciente en la teoría de la información que rompe con la suposición anterior. En lugar de simplemente reenviar los datos, los nodos pueden recombinar algunos paquetes de entrada en uno o varios paquetes de salida. Un ejemplo sencillo en un contexto inalámbrico sería una topología de tres nodos, como puede observarse en la figura Fig. 1. En dicha figura, los nodos A y B intercambian paquetes a través de un nodo intermedio S (por ejemplo, una estación base inalámbrica). El envío de paquetes llevado a cabo en este tipo de topología mediante la codificación de red será: A [respectivamente B] envía un paquete a [respectivamente b] a S, difundiendo

después una secuencia $a \oplus b$ en vez de a y b. Tanto A como B pueden recuperar el paquete de interés, mientras el número de transmisiones se ve reducido.

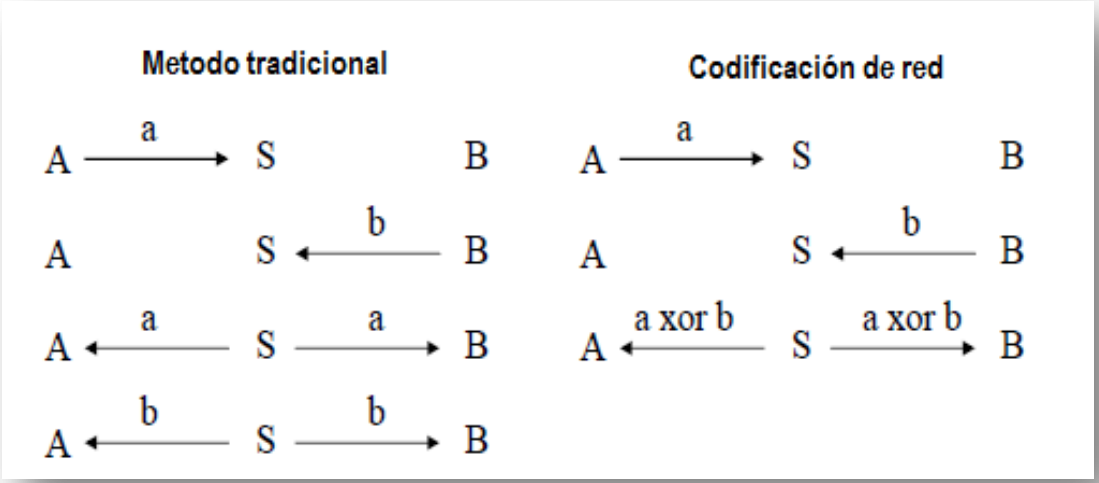


Fig. 1. ***Ejemplo sencillo de codificación de red.***

La codificación lineal de red, en general, es similar a este ejemplo, a diferencia de que la operación xor se sustituye por una combinación lineal de datos, interpretados como un conjunto finito de elementos. Este hecho permite un grado mayor de flexibilidad, de tal manera que los paquetes se pueden combinar. Además de los beneficios de rendimiento mostrados en el ejemplo anterior, la codificación de red se considera altamente apropiada para entornos en los que parte de la información se encuentra disponible al tomar decisiones. La recepción exitosa de la información no depende de recibir el contenido específico de un paquete, sino más bien en la recepción de un número suficiente de paquetes independientes. Las combinaciones lineales requieren un mayor cómputo en los nodos de red. Sin embargo, de acuerdo con la ley de Moore, el procesamiento es cada vez menor y menos costoso. Por tanto, el cuello de botella se ha desplazado al ancho de banda de la red para soportar la creciente demanda de aplicaciones y las garantías de calidad de servicio sobre amplias redes no seguras.

Para reforzar el concepto de codificación lineal de red, considérese un sistema que actúa como un relé de información, como puede ser un router, un nodo en una red ad-hoc o bien un nodo en una red de distribución punto a punto. Tradicionalmente, cuando se reenvía un paquete de información destinado a algún otro nodo, simplemente se repite. Con la codificación de red, se permite al nodo combinar un número de paquetes que ha recibido o creado en uno o varios paquetes de salida.

Supóngase que cada paquete se compone de L bits. Cuando los paquetes al ser combinados no tienen el mismo tamaño, los más cortos se rellenan con ceros finales. Se puede interpretar s bits consecutivos de un paquete como símbolo sobre un espacio $\mathbb{F}_{2^s}$, en el que cada paquete consta de un vector de L/s símbolos. Con una codificación lineal de red, los paquetes de salida son combinaciones lineales de los paquetes originales, donde las operaciones de suma y multiplicación se desarrollan sobre el espacio $\mathbb{F}_{2^s}$. La razón para elegir un marco lineal es que los algoritmos de codificación y decodificación no presentan una gran complejidad computacional.

No se entiende por combinación lineal el método de concatenación ya que si se combinan linealmente paquetes de longitud L, el paquete codificado resultante también tendrá longitud L. En contraste con la concatenación, cada paquete codificado contiene sólo una fracción de la información contenida en los paquetes originales. Se puede llegar a pensar que la codificación lineal de red es una forma de difusión de información. Por lo tanto, un primer resultado de la codificación de red es que permite aumentar la capacidad en una red multidifusión. Aunque los beneficios más significativos de dicha codificación puede apreciarse en términos de robustez y adaptabilidad.

Si se aplica la teoría de codificación de red en redes inalámbricas, el concepto de relé se propone como un medio para aumentar la capacidad del sistema y/o área de cobertura para la comunicación sobre largas distancias o alrededor de obstáculos. En un sistema relé, un nodo relé intermedio ayuda a la transmisión desde un nodo fuente a un nodo destino mediante el reenvío del mensaje de la fuente al destino [2]. En un caso típico, el relé se encuentra en la trayectoria entre el origen y el destino, y ayuda a reducir pérdidas por trayecto. El relé reenvía mensajes usando comúnmente una estrategia de "Amplify-And-Fordward" (AAF), ver figura Fig. 2, donde se amplifica el mensaje recibido y se reenvía al destino, o bien por un esquema de "Decode-And-Forward" (DAF), ver figura Fig. 3, en el que el mensaje de la fuente se decodifica, se reenvía al destino con redundancia incremental o repetición de código [3].

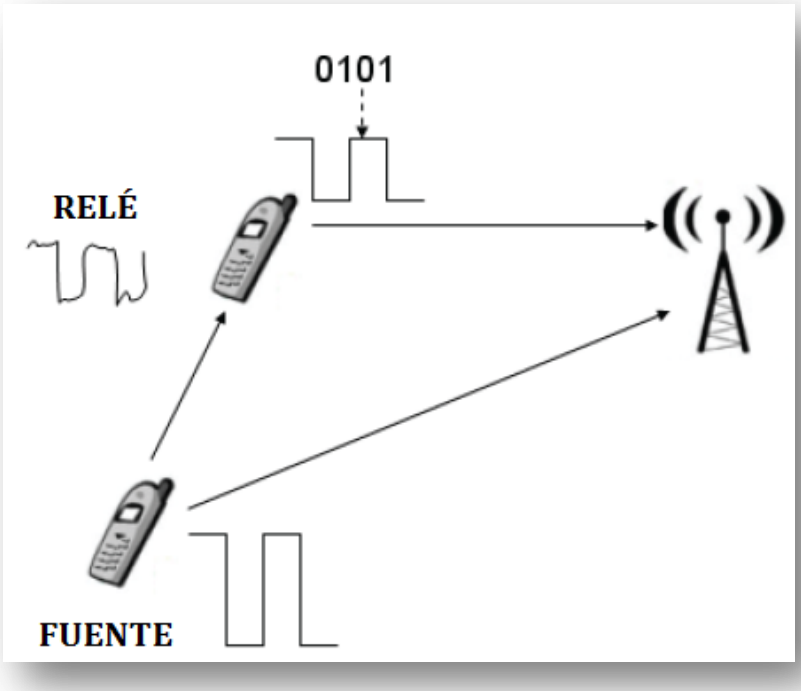


Fig. 2. ***Técnica DAF*.**

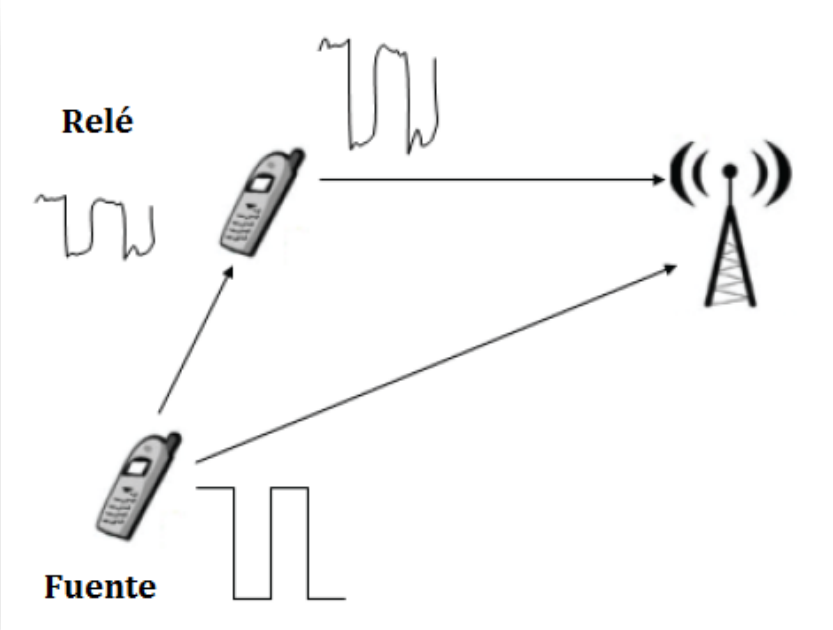


Fig. 3. ***Técnica AAF.***

1.2 Objetivos

Debido al despliegue de redes de comunicaciones con gran diversidad de dispositivos autónomos, a lo largo de los años se ha avanzado en el estudio de nuevas posibilidades de códigos eficientes basados en codificación de red. En particular, la cooperación entre los diferentes nodos ha sido identificada como una tecnología que permite satisfacer la demanda cada vez mayor de los recursos. Por lo que el objetivo principal del proyecto se centra en el estudio de diferentes códigos y la estrategia de codificación para la cooperación entre canales relé mediante la estrategia de decodificación y reenvío.

Como primer objetivo, se plantea desarrollar la problemática concebida por Razagui [4], en la que se desarrolla una técnica de "binning" para la estrategia DAF en una canal relé degrado. Primeramente, como indica Razagui en su estudio se construyen códigos LDPC para su posterior retransmisión a través de la estrategia DAF basada en la investigación de Cover y El Gamal para un escenario en el que la capacidad del enlace relé-destino es lo suficientemente grande. Después, se considerará un caso más específico de análisis, donde se realiza un estudio [4] en profundidad de las capacidades que atañen a ese caso.

En segundo lugar, debido a que en [4] no se propone una simulación completa para códigos de dos capas, se propone una nueva estrategia de codificación a través de códigos IRA para el canal relé "full-dúplex". La construcción del código IRA, al igual que los códigos propuestos en [4], muestra que la operación de binning para el canal relé es fundamentalmente más sencilla de implementar en la práctica que otras técnicas de binning [5].

Finalmente, tras encuadrar todo el marco teórico del estudio, se proponen varias implementaciones prácticas de estructuras de código en el relé y la aplicación de diferentes algoritmos de decodificación iterativos con el fin de adaptar su construcción a varios escenarios.

2.1 Estructura de la memoria

Este documento comprende una serie de capítulos claramente diferenciados cuyos contenidos se describen a continuación:

Capítulo 1: Introducción y objetivos	• En este primer capítulo, se introduce un acercamiento a la teoría de codificación de red. Además, se asociará su aplicación al caso de estudio: el canal rele. Por último, se presentarán los objetivos que se han perseguido para dicho estudio y las dificultades obtenidas.
Capítulo 2: Trabajo previo	• En el segundo capítulo, se lleva a cabo una descripción detallada de todos aquellos conceptos teóricos relacionados con el proyecto y qué finalidad última se quiere aportar con el mismo.
Capítulo 3: Canal relé degrado	• En el tercer capítulo, se presenta el canal relé como un caso especial de canal físicamente degradado.
Capítulo 4: Códigos sobre Grafo de Factores	• En el cuarto capítulo, se desarrolla una explicación en profundidad de las diferentes posibilidades de códigos para la etapa de codificación en el canal relé.
Capítulo 5: Aplicación de códigos IRA sobre un canal relé degrado	• En el quinto capítulo, se tiene como objetivo mostrar la aplicación técnica de los conceptos teóricos estudiados a lo largo del resto de capitulos. Además, se fijará una nomenclatura para las distintas etapas de la modelización de la simulación.
Capítulo 7: Resultados de simulación	• En el séptimo capítulo, se plamarán los resultados obtenidos a los largo de las diferentes simulaciones. Además, se irán añadiendo los comentarios permitentes para cada una de las gráficas obtenidas.
Capítulo 8: Conclusiones	• En el octavo capítulo, se detallan las conclusiones obtenidas, dificultades encontradas, decisiones tomadas,... Además, se comentan las experiencias que se han ido adquiriendo en la consecución de las diferentes etapas.
Capítulo 9: Lineas futuras	• En el noveno capítulo, se proponen posibles mejoras y ampliaciones que permitirán dar continuidad al trabajo desarrollado en este proyecto fin de carrera.
Capítulo 10: Presupuesto	• En el último capítulo, se planteará también una previsión de los recursos que se van a emplear para su realización, haciéndose una valoración económica del coste total generado.
Anexo 1: Glosario	• En este anexo, se proporciona una relación con todos los acrónimos empleados a lo largo de este proyecto.
Anexo 2: Referencias	• En este último anexo, se recoge todas las referencias bibliográficas empleadas en la realización del proyecto.

Capítulo 2

Trabajo previo

2.1 Secuencias típicas

2.1.1 Introducción

Shannon introdujo el término Secuencias típicas en 1948 [6]. Para comenzar con el desarrollo teórico, consideró una fuente discreta sin memoria (DMS) representada por un dispositivo que transmitía símbolos procedentes de un alfabeto finito X, independientemente e idénticamente distribuidos, figura Fig. 4.

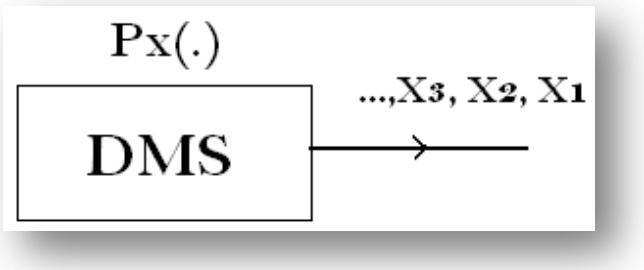


Fig. 4. ***Fuente discreta sin memoria con distribución*** $P_X(.)$ **[7]**

Supóngase una fuente con una distribución de probabilidad que es la siguiente:

$$P_X(0) = \frac{2}{3}\ y\ P_X(1) = \frac{1}{3}$$

Considérese el siguiente experimento: se genera una secuencia de longitud 18 a través de un generador de números aleatorios con una distribución de probabilidad mostrada anteriormente. A continuación, se muestra la secuencia junto con otras tres secuencias generadas artificialmente:

a) 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
b) 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0
c) 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0
d) 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1.

Si se calculan las probabilidades con las que fueron emitidas las secuencias por la fuente, se obtendría:

a) $\left(\frac{2}{3}\right)^{18} * \left(\frac{1}{3}\right)^{0} \approx 6.77 \cdot 10^{-4}$
b) $\left(\frac{2}{3}\right)^{9} * \left(\frac{1}{3}\right)^{9} \approx 1.32 \cdot 10^{-6}$
c) $\left(\frac{2}{3}\right)^{11} * \left(\frac{1}{3}\right)^{7} \approx 5.29 \cdot 10^{-6}$
d) $\left(\frac{2}{3}\right)^{0} * \left(\frac{1}{3}\right)^{18} \approx 2.58 \cdot 10^{-9}$

Por lo tanto, la primera secuencia es la más probable. Sin embargo, no será sorprendente averiguar que fue la secuencia c) la que fue obtenida por el generador aleatorio. Entonces ¿por qué la intuición nos engaña?, para explicar esto, se puede definir con más detalle lo que significa secuencias típicas.

2.1.2 Concepto teórico

Se define un canal discreto que consta de un alfabeto de entrada $\mathcal{X}$ y un alfabeto de salida $\mathcal{Y}$ y una colección de funciones de masa de probabilidad $p(y|x)$ que expresa la probabilidad de observar el símbolo de salida y habiendo enviado x. El canal se dice que es sin memoria si la distribución de probabilidad de la salida depende únicamente de la entrada en ese momento y es condicionalmente independiente de anteriores entradas y salidas en el canal.

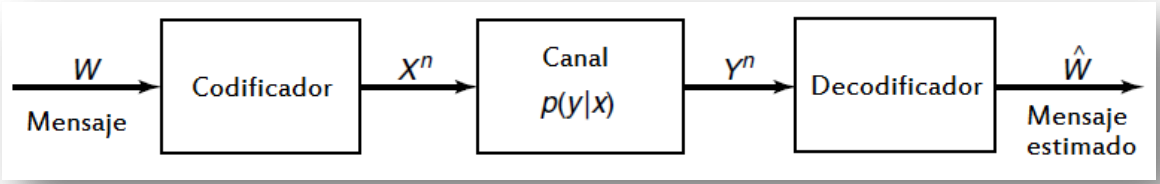


Fig. 5. ***Sistema de comunicaciones*** **[7]**

El mensaje W trazado desde un conjunto de índices $\{1,2,\dots,M\}$ se codifica obteniendo $X^n(W)$, recibiendose por el destino como una secuencia aleatoria $Y^n \sim p(y^n|x^n)$.

En términos generales, se podrá decodificar una salida de un canal Y^n con índice i, si la palabra código $X^n(i)$ es "conjuntamente típica" con la señal recibida Y^n. A continuación, se definirán algunos conceptos básicos sobre secuencias típicas y cómo encontrar la probabilidad conjuntamente típica, cuando $X^n(i)$ produce como salida Y^n y cuando no [8].

El conjunto A_ϵ^n de secuencias típicas $\{(x^n, y^n)\}$ con respecto a la distribución $p(x,y)$ es un conjunto de n-secuencias con una entropía empírica cercana a ϵ para llegar a la entropía teórica:

$$A_\epsilon^n = \left\{(x^n, y^n) \epsilon\, \mathcal{X}^n x\, \mathcal{Y}^n : \left|-\frac{1}{n} log\, p(x^n) - H(X)\right| < \epsilon, \right.$$
$$\left. \left|-\frac{1}{n} log\, p(y^n) - H(Y)\right| < \epsilon, \left|-\frac{1}{n} log\, p(x^n, y^n) - H(X,Y)\right| < \epsilon \right\}$$

donde:

$$p(x^n, y^n) = \prod_{i=1}^{n} p(x_i, y_i)$$

La propiedad de equipartición asintótica (AEP) es una propiedad desarrollada en teoría de la información y es análoga a la Ley de los Grandes Números. La AEP establece que la propiedad de observar una secuencia $s = x_1, x_2, \dots, x_n$, procedente de una fuente F, es próxima a 2^{-nH}. Además, va a permitir separar las secuencias generadas por la fuente en dos tipos: típicas y atípicas. Distinción importante a la hora de codificar la fuente ya que permitirá aplicar o no una técnica de compresión.

Atendiendo al teorema de la AEP conjunta, se mantiene que dado $\{(X^n, Y^n)\}$, secuencias de longitud n, siendo éstas i.i.d.[1] de acuerdo con la siguiente expresión: $p(x^n, y^n) = \prod_{i=1}^{n} p(x_i, y_i)$. Entonces:

1. $Pr\left((X^n, Y^n) \epsilon\, A_\epsilon^{(n)}\right) \xrightarrow[n\to\infty]{} 1$

[1] i.i.d, abreviatura que proviene del término "Independent and identically distributed random variables". En teoría de probabilidad y estadística, una secuencia o un conjunto de variables aleatorias son independientes e identicamente distribuidas si cada variable aleatoria tiene la misma distribución de probabilidad que las otras y todas son mútuamente independientes.

2. $\left|A_{\epsilon}^{(n)}\right| \leq 2^{n(H(X+Y)+\epsilon)}$
3. Si $\left(\tilde{X}^n, \tilde{Y}^n\right) \sim p(x^n)p(y^n)$, es decir, $\tilde{X}^n$ y $\tilde{Y}^n$ son independientes con los mismos marginales que $p(x^n, y^n)$, entonces:

$$Pr\left((X^n, Y^n) \epsilon A_{\varepsilon}^{(n)}\right) \geq 2^{-n(I(X;Y)-3\varepsilon)}$$

Si n es lo suficientemente grande, se puede expresar la ecuación anterior como:

$$Pr\left((X^n, Y^n) \epsilon A_{\epsilon}^{(n)}\right) \geq (1-\varepsilon)2^{-n(I(X;Y)+3\epsilon)}$$

El conjunto de secuencias conjuntamente típico, se podrá visualizar gráficamente en la figura Fig. 6. En ella, habrá alrededor de $2^{nH(X)}$ secuencias típicas de X en $A_{\epsilon}^{(n)}(P(X))$, y alrededor de $2^{nH(Y)}$ secuencias típicas de Y en $A_{\epsilon}^{(n)}(P(Y))$. Sin embargo, dado que sólo hay $2^{nH(X,Y)}$ secuencias típicas, no todos los pares típicos X^n e Y^n se consideran también conjuntamente típicos.

La probabilidad de que cualquier par elegido arbitrariamente sea conjuntamente típico es aproximadamente $2^{-nI(X;Y)}$ cuando n es suficientemente grande. Este hecho sugiere que encontraremos alrededor de $2^{nI(X;Y)}$ señales distinguibles en X^n.

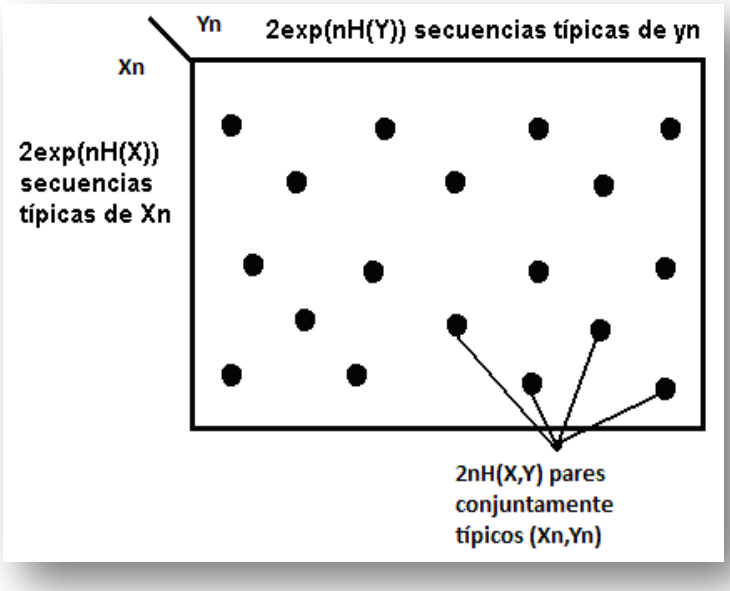


Fig. 6. Ejemplificación de secuencias conjuntamente típicas

Otra forma de verlo es en términos de un conjunto de secuencias conjuntamente típicas para una secuencia fija de salida Y^n, presumiblemente la secuencia de salida proveniente de la entrada correcta X^n. Para esta secuencia Y^n, hay alrededor de $2^{nH(X|Y)}$ entradas condicionalmente típicas. La probabilidad de que algunas entradas X^n elegidas

al azar sean conjuntamente típicas con Y^n es aproximadamente $2^{nH(X|Y)}/_{2^{nH(X)}} = 2^{-nI(X;Y)}$, si se tiene en cuenta que $H(X,Y) = H(X) + H(Y|X)$. Eso sugiere que se pueden elegir alrededor de $2^{nI(X;Y)}$ palabras código $X^n(W)$ antes de que una de estas palabras código se confunda con la palabra código que causó la salida Y^n.

2.2 El canal relé

El canal relé fue estudiado por primera vez por van der Meulen [9], [10], [11, p.7 y pp. 32-34] en 1971, y también por Sato [12]. En esos primeros estudios, se plantearon canales de comunicaciones que disponían de tres terminales. El problema que se investigaría, ver figura Fig. 7, se centra en cómo enviar información desde un terminal específico (llamado terminal 1) a otro (llamado terminal 3) sobre el canal, de la manera más efectiva posible, asumiendo que todos los terminales cooperan en el proceso de transmisión. Definió formalmente van der Meulen un canal discreto sin memoria con tres terminales mediante un conjunto de probabilidades de transición $P(y_1, y_2, y_3|x_1, x_2, x_3)$ en $\mathcal{B}_1 \times \mathcal{B}_2 \times \mathcal{B}_3$ (siendo $\mathcal{B}_t$ el alfabeto de salida del terminal t), habiendo tantas $P(y_1, y_2, y_3|x_1, x_2, x_3)$ como entradas $(x_1, x_2, x_3)\, \epsilon\, \mathcal{A}_1 \times \mathcal{A}_2 \times \mathcal{A}_3$ (siendo $\mathcal{A}_t$ el alfabeto de entrada para el terminal t). Para dicha definición se debe cumplir que:

$$P(Y_1, Y_2, Y_3|X_1, X_2, X_3) = \prod_{k=1}^{n} P(y_{1k}, y_{2k}, y_{3k}|x_{1k}, x_{2k}, x_3)$$

para todo $X_t = (x_{t1}, \dots, x_{tn})\, \epsilon\, A_t^{(n)}$; $Y_t = (y_{t1}, \dots, y_{tn})\, \epsilon\, B_t^{(n)}$; $t = 1,2,3$ y $n \geq 1$.

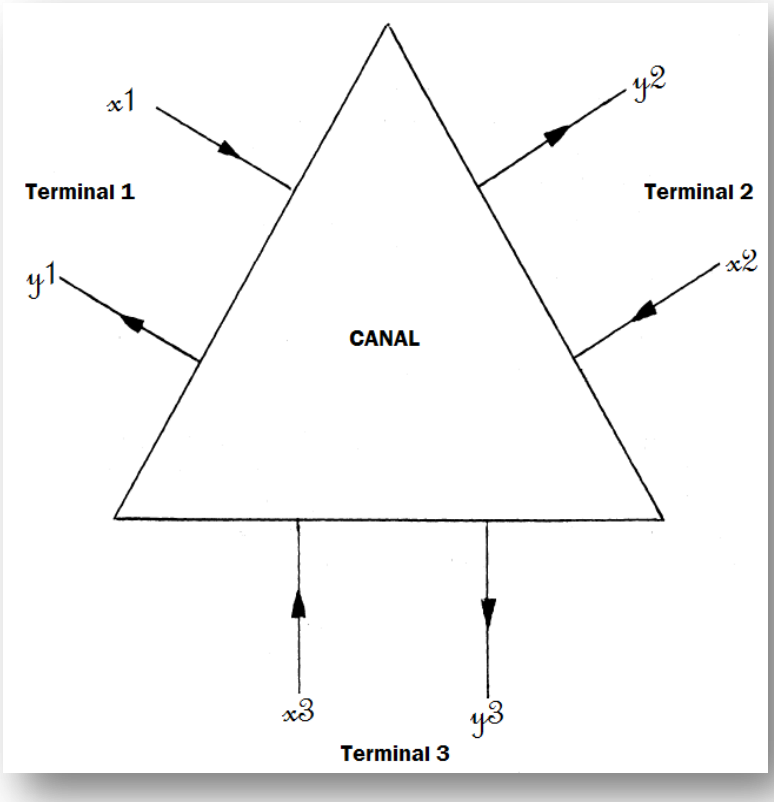


Fig. 7. ***Modelo de canal relé propuesto por van der Meulen***

Más tarde, Cover y El Gamal en su trabajo de 1979 [13], proporcionan un análisis en profundidad de un canal relé individual, centrando el estudio en obtener la región de

capacidad para una versión del canal físicamente degradado. Hasta hace poco, no se habían llevado a cabo grandes avances para extender estos resultados a los canales relés múltiples. Sin embargo, un interés renovado, en redes ad-hoc y la teoría de la información de red, ha desatado una nueva investigación sobre los canales relé. Una serie de resultados recientes sobre esta área, se recoge en el trabajo de Gupta y Kumar [14] donde han demostrado resultados sobre la región de tasa alcanzable para un tipo de red de comunicaciones generalizada, en la que se destaca el caso del canal relé degradado. Los resultados en [14] son tanto para un canal discreto sin memoria como para un canal AWGN. Un artículo posterior, escrito por Xie y Kumar [15], establece una expresión explícita para la tasa alcanzable en un canal Gaussiano degrado con múltiples relés, que en general, excede la tasa mostrada en [14].

El exhaustivo trabajo de Cover y El Gamal [13] describe dos estrategias básicas para el canal relé. Estos dos esquemas son ampliamente conocidos hoy en día como "Decode And Forward" (DAF) y "Compress-And-Forward" (CAF). Estas dos estrategias de codificación representan diferentes tipos de cooperación. En DAF, la cooperación es relativamente obvia, el relé decodifica el mensaje procedente de la fuente, y la fuente y relé cooperativamente transmiten la información comúnmente construida al destino en el bloque siguiente. En CAF, el método de cooperación es más complicado de reconocer. En este técnica se cuantifica la señal recibida, más delante se comprime y se retransmite hacia el destino; donde se decodifica el mensaje original a través de la señal recibida por el relé y la fuente. Cover y El Gamal demostraron que la estrategia DAF es útil para una clase de canales relé degradados [16].

En [16] se define el canal relé como un canal en el que hay una fuente y un destino con un número de nodos intermedios que actúan como relé para ayudar en las comunicaciones desde que se transmite hasta que se recibe, ver figura Fig. 8. El canal relé más sencillo tiene un solo nodo intermedio o nodo relé, por lo que se prestará especial interés en este estudio.

En este caso, el canal va a consistir en cuatro conjuntos finitos $\mathcal{X}$, $\mathcal{X}_1$, $\mathcal{Y}$ e $\mathcal{Y}_1$ y una colección de funciones de probabilidad $p(y, y_1|x, x_1)$ en $\mathcal{Y}\ x\ \mathcal{Y}_1$, uno por cada $(x, x_1) \in \mathcal{X}\ x\ \mathcal{X}_1$. Se considera x la entrada al canal e y la salida del canal, y_1 corresponde con la observación del relé y x_1 es la entrada elegida por el relé que depende únicamente de las observaciones del pasado $(y_{11}, y_{12}, \dots, y_{1i-1})$ como se muestra en la figura Fig. 8.

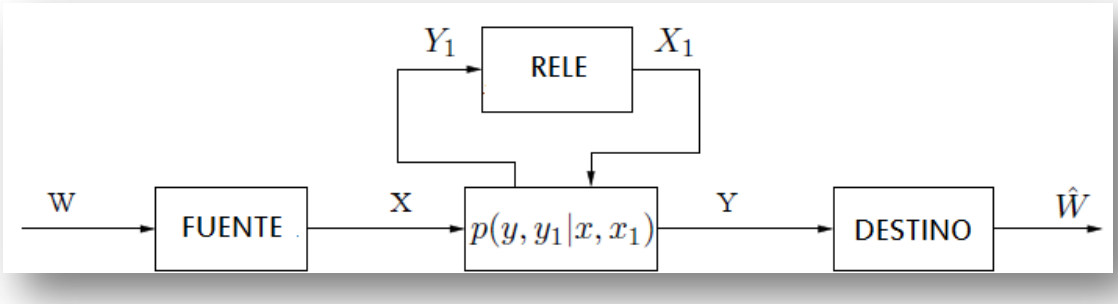


Fig. 8. ***Canal relé básico***

Una vez mostrada la nomenclatura del canal, el problema que se plantea será encontrar la capacidad del canal entre la fuente X y el destino Y.

El canal relé combina un canal de difusión entre fuente-relé y fuente-destino y un canal de acceso múltiple entre el relé y el destino. La capacidad será conocida para el caso especial de un canal relé físicamente degradado, el cual se explicará con detalle más adelante.

Un código $(2^{nR}, n)$ para un canal relé consiste en un conjunto de enteros $\mathcal{W} = (1, 2, 3, \dots, 2^{nR})$, una función de codificación $X: (1, 2, 3, \dots, 2^{nR}) \rightarrow \mathcal{X}^n$, un conjunto de funciones relé $\{f_i\}_{i=1}^{n}$ que pueden expresarse como $x_{1i} = f_i(Y_{11}, Y_{12}, \dots, Y_{1i-1}),\ 1 \leq i \leq n$, y una función de decodificación $g: \mathcal{Y}^n \rightarrow\ (1, 2, 3, \dots, 2^{nR})$.

Hay que darse cuenta que la definición anterior de las funciones de codificación incluyen la condición de no anticipación en el canal relé. Por tanto, se permite que la entrada del canal relé dependa solamente de las observaciones del pasado $(y_{11}, y_{12}, \dots, y_{1i-1})$. El canal además se considerará sin memoria, de tal forma que (Y_i, Y_{1i}) depende solamente del pasado a través de los símbolos actuales transmitidos (X, X_{1i}). Así, para cualquier elección $p(w)$ con $w \epsilon \mathcal{W}$, el código elegido $X: (1,2,3, \dots, 2^{nR}) \rightarrow \mathcal{X}_i^n$, y las funciones relé $\{f_i\}_{i=1}^{n}$, la función de masa de probabilidad conjunta en $\mathcal{W}\ x\ \mathcal{X}^n\ x\ \mathcal{X}_1^n\ x\ \mathcal{Y}_n\ x\ \mathcal{Y}_1^n$ vendrá dada por la siguiente expresión:

$$p(w, x, x_1, y, y_1) = p(w) \prod_{i=1}^{n} p(x_i|w) p\ (x_{1i}|y_{11}, y_{12}, \dots, y_{1i-1}) * p(y_i, y_{1i}|x_i, x_{i1})$$

Si el mensaje enviado es $w\ \epsilon\ [1, 2^{nR}]$, entonces: $\lambda(w) = Pr\{g(Y) \neq w|w\ enviada\}$ denota la probabilidad condicional de error.

Para concluir, se puede definir la probabilidad de error media del código como:

$$P_e^{(n)} = \frac{1}{2^{nR}} \sum_{w} \lambda(w)$$

La probabilidad de error se puede calcular a través de una distribución uniforme sobre la palabra código $w\ \epsilon\ [1, 2^{nR}]$. La tasa R se dice que es alcanzable por el canal relé si existe una secuencia de $(2^{nR}, n)$ códigos con $P_e^n \rightarrow 0$.

2.3 Tipos de codificación para el canal relé

Los últimos avances sobre dispositivos ad-hoc inalámbricos y redes de sensores han ido promoviendo una nueva oleada de investigaciones sobre una codificación aplicable a canales relé, abarcando tanto aspectos teóricos [17] [18] [19] como aspectos de aplicación práctica sobre la cooperación de sistemas de comunicación [20] [21]-[27], como puede ser el caso estudiado en [20] en el que varios dispositivos móviles comparten sus antenas para alcanzar una mejor tasa de transmisión de subida, ver figura Fig. 9.

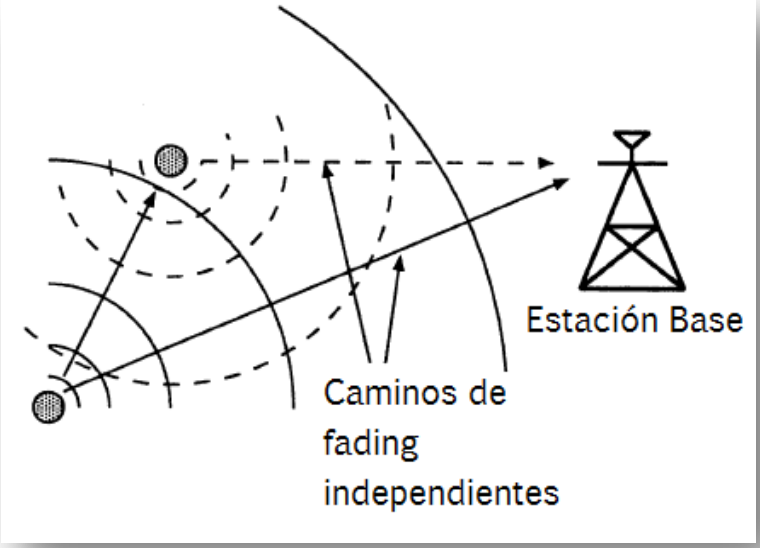


Fig. 9. ***Ejemplo de cooperación entre móviles*** **[20]**

Trabajos relacionados con el uso de códigos en canales relé se remontan a teorías de codificación ya propuestas en [20] [28] [29] [30]. Más adelante, para proporcionar una mayor variedad de tipos de codificación aplicables a un terminal relé, se comenzó con el empleo de los conocidos turbo códigos [20] [28] [30] donde el relé es el encargado de generar los bits paridad.

Mientras que, los documentos mencionados anteriormente se centran en mejorar la cooperación en un contexto de turbo códigos, el caso de estudio en [5] se concentra en conseguir acercase a la tasa teórica alcanzable en DAF utilizando códigos LDPC. Se decidió el uso de estos códigos debido a que su capacidad puede acercase al rendimiento en un canal con un solo usuario y además por su flexibilidad de construcción. Además, en [5] se dispone de un conjunto completo de técnicas de diseño para canales de un solo usuario, permitiendo de esta forma desarrollar procedimientos similares para canales relé.

En el contexto de la codificación LDPC, la construcción del código presente en [5] se relaciona con varios estudios recientes sobre métodos de codificación tales como: decodificación y envío para canales relé half-dúplex y full-dúplex [24], [31]-[34].

En [31], se estudia una estructura de códigos LDPC similar a los códigos LDPC expurgados de dos capas propuestos en [5] como solución para el diseño del canal relé, y utilizándose para el diseño del código de chequeo un enfoque de evolución de densidad.

El análisis del código, junto con la metodología de diseño de [31], está basado en la evolución de densidad para el canal de un solo usuario, donde el rendimiento de los códigos de dos capas expurgados puede aproximarse por códigos LDPC convencionales.

El trabajo llevado a cabo en [32] propone una codificación a través de la técnica DAF con códigos LDPC y un esquema de decodificación aplicados a un canal relé BIAWGN. En [32] se considera el uso de fuentes independientes y un diccionario de palabras código en el relé, asumiendo que el relé es capaz simultáneamente de recibir y transmitir. En este caso, el éxito de la decodificación del código expurgado en el destino se encuentra garantizado, y el diseño del código óptimo implica el diseño de un código LDPC convencional para el canal entre la fuente y el relé. Cabe destacar que aunque el uso de diccionarios de palabras código independientes es óptimo para ciertos canales con fading en los que una transmisión coherente puede no ser posible [35, Teorema 6], en general, los códigos convencionales LDPC no son adecuados para una estrategia DAF en escenarios con canales muy estrictos, donde el ancho de banda y/o energía al enlace fuente-relé, relé-destino y fuente-destino, son mínimos para lograr una tasa de DAF determinada.

En otra serie de estudios, [33] y [34] aplican códigos LDPC convencionales optimizados para canales de un solo usuario a un canal relé con fading, y emplearán particiones aleatorias para conseguir que el código LDPC trabaje a dos tasas diferentes. Usando este enfoque, los autores informan sobre un rango de umbrales de entre 0,4 a 0,7 dB para varias configuraciones de canal con parámetros de transmisión fijos. (Los códigos LDPC tienen un umbral de SNR teórico, para valores inferiores la $P_e \neq 0$, que es superior al límite de la capacidad, en [34, Fig. 6] [35, Fig.8] muestran que es posible acercarse al umbral teórico).

Finalmente, [24] utiliza un esquema basado en particiones de los códigos LDPC convencionales, donde múltiples esquemas de codificación entrada-salida son adaptados para la estrategia DAF. El trabajo desarrollado por [5] se diferencia de [24], [31]-[33], en que en vez de usar particiones aleatorias para conseguir que el código LDPC convencional trabaje a dos tasas diferentes, en [5] utilizará técnicas de códigos expurgados o bien técnicas de prolongación de código, junto con un diseño explícito de códigos LDPC de dos capas, que son capaces de acercarse simultáneamente a las capacidades de los enlaces fuente-relé y fuente-destino. Un diseño adecuado de códigos LDPC es imprescindible para aproximarse a la tasa teórica prometida con la estrategia DAF; ya que los códigos LDPC convencionales están ajustados para funcionar eficientemente con una determinada parametría de canal. Por lo tanto, un código fuente LDPC convencional no puede acercarse a los valores de capacidad de los enlaces fuente-relé y fuente-destino, debido a la diferencia en los respectivos parámetros de cada enlace. Por último, en [5] se mostrarán dos diferentes estructuras de grafo LDPC, llamados códigos expurgados de dos capas y códigos de prolongación de dos capas, que serán necesarias para llegar a aproximar la tasa teórica DAF para todo el rango de parámetros del canal.

El problema de diseño de códigos para un canal relé, también se asocia con el concepto general de códigos sin tasa llamados LT, popularizado gracias al descubrimiento de los códigos fuente [36] y, más recientemente, los códigos Raptor [37]. En la estructura del código Raptor, mostrada en la figura Fig. 10, los símbolos de entrada son añadidos por símbolos redundantes (cuadrados gris oscuro) en el caso de un pre-

código sistemático. Un apropiado código LT se utiliza para generar símbolos de salida desde los símbolos de entrada pre-codificados.

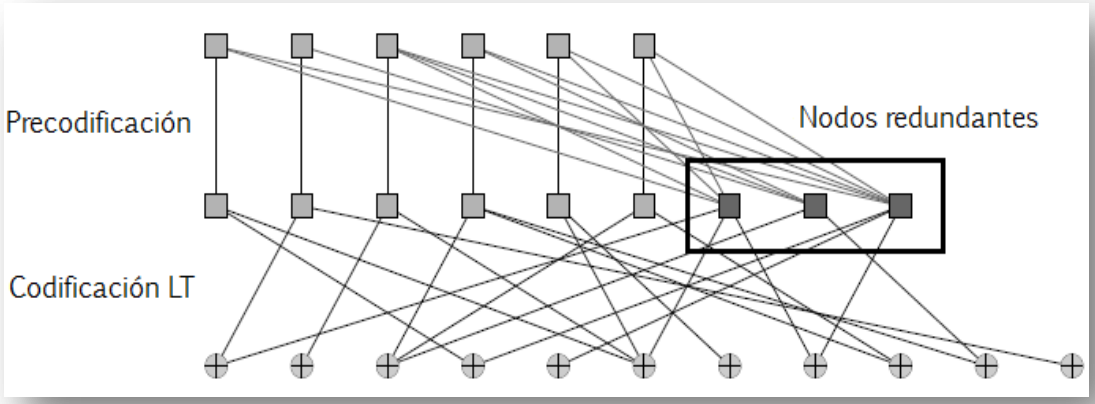


Fig. 10. ***Códigos Raptor*** **[37]**

Los códigos LDPC de dos capas que se proponen en [5] son similares a los códigos sin tasa debido a que ambos son capaces de trabajar a múltiples tasas. Sin embargo, el requisito de diseño del código para el canal relé es fundamentalmente diferente en varios aspectos que se explican más adelante. En un canal relé propuesto en [5], la información adicional codificada para el segundo código se transmite de forma aislada a través del enlace relé-destino. En contraposición, el modelo de canal para códigos sin tasa, por lo general, asume que los bits adicionales se envían a través de un mismo enlace (por consiguiente, pueden estar corrompidos por ruido del canal). Además, los códigos fuente y los códigos Raptor están diseñados específicamente para un canal de borrado binario (BEC); en la práctica los códigos sin tasa existen para diversos modelos de canal, aunque hoy en día todavía se mantienen abiertas varias investigaciones [38]. En este sentido, la metodología de diseño para los códigos fuentes y los códigos Raptor no puede aplicarse directamente a un canal relé general, excepto en el caso de un canal de borrado.

Para concluir, la construcción de códigos propuesta en [5] también está relacionada con el uso de métodos de partición de código LDPC para los protocolos IR-HARQ, utilizados para canales con transmisión inalámbrica [32], [40]-[45]. En la configuración HARQ, los bits adiciones de los datos codificados son enviados cuando el decodificador falla en el proceso de la decodificación. Una vez más, los bits adicionales pueden ser potencialmente corrompidos por el canal, resultando un problema añadido a la decodificación comparado al canal relé. Por lo tanto, los esquemas de codificación existente en HARQ no son directamente aplicables a la estrategia DAF.

Capítulo 3

Canal relé degradado

3.1 Introducción

El canal relé $(\mathcal{X}\ x\ \mathcal{X}_1, p(y, y_1|x, x_1), \mathcal{Y}, \mathcal{Y}_1)$ se dice que está degradado si $p(y, y_1|x, x_1)$ puede representarse a través de la siguiente ecuación:

$$p(y, y_1|x, x_1) = p(y_1|x, x_1)p(y|y_1, x_1).$$

Equivalentemente, un canal relé está degradado si $p(y, y_1|x, x_1) = p(y|y_1, x_1)$, es decir, $X \to (X_1, Y_1) \to Y$ configura una cadena de Markov. Así Y es la degradación aleatoria de la señal Y_1 del relé.

Para el canal relé degradado, la capacidad puede expresarse a través de la siguiente ecuación, donde el supremo se halla sobre todas las distribuciones conjuntas $p(x, x_1)$ en $\mathcal{X}\ x\ \mathcal{X}_1$:

$$C = \sup_{p(x,x_1)} \min\{I(X, X_1; Y), I(X; Y_1|X_1)\}$$

Por tanto, debido a la condición de degradación:

$$I(X; Y, Y_1|X_1) = I(X; Y_1|X_1)$$

Esta capacidad es alcanzable por un conjunto de combinaciones de las siguientes técnicas:

1. Codificación aleatoria
2. Lista de códigos
3. Partición de Slepian-Wolf
4. Codificación para un canal de múltiple acceso en modo cooperación (MAC).
5. Superposición de código
6. Codificación de bloques de Markov en el relé y en el destino.

Hay que tener en cuenta que todas las construcciones anteriores, incluyendo la codificación de superposición, la partición de Slepian-Wolf, y la codificación para la cooperación MAC, pueden aplicarse sin cambios a un canal relé arbitrario. Sin embargo, la suposición de degradación es necesaria para establecer que la tasa C, que será de hecho la capacidad.

Este proyecto se centrará principalmente en el caso de una codificación aleatoria, por lo que los pasos que se deberían seguir son:

- **Generación aleatoria del código:** Para este paso fijaremos $p(x_1)p(x|x_1)$. Primeramente, se generaran $M_0 = 2^{nR_0}$ secuencias i.i.d aleatorias en el espacio $\mathcal{X}_1^n$, cada una generada de acuerdo a $p(x_1) = \prod_{i=1}^{n} p(x_{1i})$, clasificándolas como $x_1(s), s \in [M_0 = 1,2, \dots, 2^{nR_0}]$. Para cada $x_1(s)$, se van a generar 2^{nR} secuencias condicionalmente independientes $x(w|s)$, $w \in [1,2, \dots, 2^{nR}]$, cada una generada de acuerdo a $p\big(x|x_1(s)\big) = \prod_{i=1}^{n} p(x_i|x_{1i}(s))$. Estas relaciones definen el diccionario aleatorio de palabras código $\mathcal{C} = \{x(w|s)|x_1(s)\}$. La partición aleatoria $\mathcal{S} = \{S_1, S_2, \dots, S_{2^{nR_0}}\}$ de $\{1,2,3, \dots, 2^{nR}\}$ se define como: Cada entero $w \in \{1,2,3, \dots, 2^{nR}\}$ será asignado independientemente, atendiendo a una dsitribución aleatoria sobre los índices $s = 1,2, \dots, 2^{nR_0}$, a las celdas S_s.

- **Codificación:** Sea $w_i \in \{1,2,3, \dots, 2^{nR}\}$ el nuevo indice que se envía en el bloque i, y que s_i sea la partición correspondiente a w_{i-1}. El codificador de la fuente envía $x(w_i|s_i)$. El relé estima el anterior w_{i-1} a través de $\widehat{\widehat{w}}_{i-1}$, además asume que $\widehat{\widehat{w}}_{i-1} \in S_{\widehat{\widehat{s}}_i}$. Más tarde, el relé envía la palabra código $x\big(\widehat{\widehat{s}}_i\big)$ en el bloque i.
- **Decodificación:** Se asume que al final del bloque i-1, el destino conoce $(w_1, w_2, \dots, w_{i-2})$ y $(s_1, s_2, \dots, s_{i-1})$, mientras que el relé conoce $(w_1, w_2, \dots, w_{i-1})$ y en consecuencia $(s_1, s_2, \dots, s_i)$. El proceso de decodificación al final del bloque i es el siguiente:
    1. Al conocer s_i y esperando hasta recibir $y_1(i)$, el relé estima el mensaje de la fuente $\widehat{\widehat{w}}_i = w$ si y sólo si existe un único w tal que $\big(x\big(w|\widehat{\widehat{s}}_i\big), x_1(\widehat{\widehat{s}}_i), y_{1(i)}\big)$ son secuencias ϵ conjuntamente típicas. Atendiendo a las propiedades de las secuencias conjuntamente típicas se demuestra que $\widehat{\widehat{w}}_i = w$ con una probabilidad de error pequeña si $R < I(X; Y_1|X_1)$ y n suficientemente grande.
    2. El destino determinará que $\hat{s}_i = s$ si existe un y sólo un s tal que $\big(x_1(s), y_{(i)}\big)$ son secuencias conjuntamente típicas. Atendiendo a la

teoría de secuencias típicas se sabe que s_i puede ser decodificada con una probabilidad de error arbitrariamente pequeña si $R_0 < I(X_1;Y)$ y n es lo suficientemente grande.

3. Asumiendo que s_i es decodificada correctamente por el destino, éste construye un diccionario $\mathcal{L}\big(y(i-1)\big)$ de índices que el destino considera que son secuencias conjuntamente típicas con $y(i-1)$ en el bloque $(i-1)$. El destino, posteriormente, declara $\widehat{w}_{i-1} = w$ como el indice enviado en el bloque $(i-1)$ si hay un único w en $S_{S_i} \cap \mathcal{L}\big(y(i-1)\big)$. Si n es lo sufientemente grande y si $R < I(X;Y|X_1) + R_0$, entonces $\widehat{w}_{i-1} = w_i$, con una probabilidad de error arbitrariamente pequeña. Atendiendo a las tasas binarias alcanzables en el punto 1 y 2, se puede decir que la tasa binaria final será:

$$R < I(X;Y|X_1) + I(X_1;Y) = I(X,X_1;Y)$$

Si se quisiera un mayor detalle en análisis de la probabilidad de error, se puede acudir a [13].

3.2 Canal Relé degradado Gaussiano

El estudio llevado a cabo por Cover y El Gamal [13] sobre la capacidad en un canal relé degradado, permite determinar que el mejor modelo que ilustra dicho canal es el caso Gaussiano.

Considérese un canal relé degradado Gaussiano , mostrado en la figura Fig. 11, donde Z_1 y Z_2 son secuencias de variables aleatorias i.i.d con media 0 y varianza $N_1 = \sigma_1^2$ y $N_2 = \sigma_2^2$ respectivamente. El destino Y, es una versión modificada del relé Y_1, condicionada por X_1.

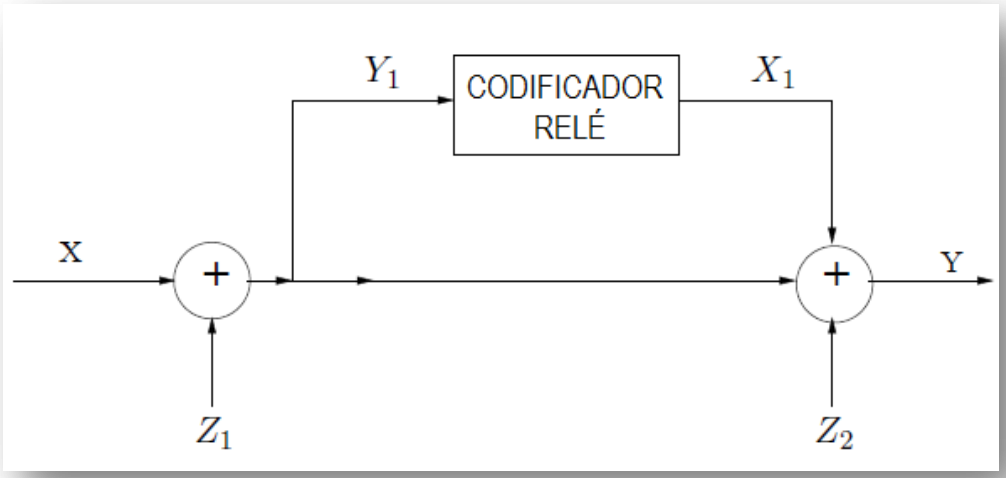


Fig. 11. ***Canal relé degradado Gaussiano.***

La definición del canal relé degradado Gaussiano vendrá dada por las siguientes expresiones:

$$Y_1 = X + N_1$$
$$Y = X + N_1 + X_1 + N_2$$

La codificación obtenida por el canal relé es una secuencia causal representada por:

$$X_{1i} = f_i(Y_{11}, Y_{12}, \dots, Y_{1i-1})$$

Además, la potencia transmitida está limitada por:

$$\frac{1}{n}\sum_{i=1}^{n} {x_i}^2(w) \leq P, w\epsilon\{1,2,\dots,M\}$$

$$\frac{1}{n}\sum_{i=1}^{n} {x_1}^2{}_i(y_{11}, y_{12}, \dots, y_{1i-1}) \leq P_1\,, (y_{11}, y_{12}, \dots, y_{1i-1})\,\epsilon\,\mathbb{R}^n$$

La fuente X transmite con potencia P y el relé X_1 con potencia P_1. La capacidad en consecuencia será:

$$C = \max_{0 \le \alpha \le 1} \min \left\{ C\left(\frac{P + P_1 + 2\sqrt{\alpha P P_1}}{N_1 + N_2}\right), C\left(\frac{\alpha P}{N_1}\right) \right\}$$

Siendo:

- $\alpha = \alpha - 1$
- $C(x) = \frac{1}{2}\log(1 + x)$

Si se considerara la siguiente desigualdad: $\frac{\mathrm{P_1}}{\mathrm{N_2}} \geq \frac{\mathrm{P}}{\mathrm{N_1}}$, se puede concluir que $C = \mathrm{C}\left(\frac{\mathrm{P}}{\mathrm{N_1}}\right)$, lográndose a su vez si $\alpha = 1$, ya que el segundo término es el más pequeño. El canal se considera libre de ruido después del relé, y de esta manera, se puede obtener un valor de la capacidad entre X y el relé de $C\left(\frac{P}{N_1}\right)$. Por consiguiente, la tasa $C\left(\frac{P}{N_1+N_2}\right)$ sin la presencia del relé, puede llegar a aumentar hasta $C\left(\frac{P}{N_1}\right)$ en el momento que haya presencia del mismo. Para un valor alto de $\mathrm{N_2}$, y para $\frac{\mathrm{P_1}}{\mathrm{N_2}} \geq \frac{\mathrm{P}}{\mathrm{N_1}}$, se puede observar que el incremento en la tasa desde $C\left(\frac{P}{N_1+N_2}\right) \approx 0$ a $C\left(\frac{P}{N_1}\right)$. Si $\frac{\mathrm{P_1}}{\mathrm{N_2}} < \frac{\mathrm{P}}{\mathrm{N_1}}$, entonces $I(X, X_1; Y) < I(X; Y_1|X_1)$. En este caso, el relé no puede garantizar una perfecta transmisión de s, entonces la fuente tiene que cooperar para enviar s. Claramente la maximización de $\alpha = \alpha^*$ es estrictamente menor que 1, y se obtiene resolviendo α en:

$$\frac{1}{2}\left(1 + \frac{P + P_1 + 2\sqrt{\bar{\alpha} P P_1}}{N_1 + N_2}\right) = \frac{1}{2}\ln\left(1 + \frac{\alpha \mathrm{P}}{N_1}\right)$$

Suponiendo que $R < C\left(\frac{\alpha P}{N_1}\right)$, será necesario el uso de dos conjuntos de palabras código. El primero de ellos, compuesto por 2^{nR} palabras código con una potencia de αP. El segundo de ellos, tiene 2^{nR_1} palabras código con una potencia de $\tilde{\alpha} P$. Se usarán palabras código pertenecientes a los dos conjuntos alternativamente para crear la posibilidad de cooperación del relé. Se comenzará enviando una palabra código desde el primer conjunto ($i \in 2^{nR}$). El relé ahora tiene conocimiento del índice de esta palabra código siempre y cuando $R < C\left(\frac{\alpha P}{N_1}\right)$, aunque el nodo correspondiente dispondrá de un diccionario de palabras código de tamaño $2^{n\left(R - C\left(\frac{\alpha P}{N_1 + N_2}\right)\right)}$.

En el siguiente bloque, la fuente y el relé tienen que cooperar para resolver la incertidumbre del destino sobre la palabra código enviada previamente. Desafortunadamente, la fuente y el relé no pueden estar seguros de que la palabra código enviada esté en el diccionario del destino, porque no poseen información de la señal recibida Y. Por consiguiente, se llevará a cabo el binning del primer conjunto de palabras código en 2^{nR_1} bloques con el mismo número de palabras en cada bloque. El relé, la fuente y el destino son conocedores de dicho bin. El relé y el destino localizan el bloque del bin donde se encuentra la palabra código del primer conjunto y se envía dicha palabra al segundo conjunto con ese índice, es decir, tanto X como X_1 enviarán el mismo código designado. El relé tiene que escalar la palabra código para cumplir las especificaciones de potencia P_1. Desde este momento, tanto el relé como la fuente podrán transmitir

simultáneamente sus palabras código. Un punto a tener en cuenta es que la información enviada por el relé y la fuente es coherente, así que la potencia vista por el destino Y será: $\left(\sqrt{\tilde{\alpha}P} + \sqrt{\mathrm{P1}}\right)^2$.

Sin embargo, el proceso explicado anteriormente no es exactamente lo que llevará a cabo la fuente en el segundo bloque. Sino que la fuente también elige una palabra código reciente del primer diccionario de palabras código, añade ésta a la palabra código compartida de la segunda lista de códigos y envía la suma al canal.

La primera acción que debe realizar en el segundo bloque el destino *Y* tras el proceso de recepción, será encontrar el índice compartido por el segundo diccionario de palabras código a través de la búsqueda de la palabra código más cercana en ese diccionario. Dicho nodo resta la palabra código de la secuencia recibida y luego calcula un diccionario de índices de tamaño 2^{nR_1} correspondiente a todas las palabras código del primer conjunto que quizás hayan sido enviadas en el segundo bloque.

Ahora, sería el momento en el que el destino completara la palabra código del primer conjunto enviada por el primer bloque. El destino cogería el diccionario de posibles palabras código que han podido ser enviadas por el primer bloque y miraría si coincide con la celda de la partición que aprendió de la transmisión del relé del segundo bloque. Las tasas y potencias ya han sido elegidas, por lo que es probable que haya una única palabra que coincida.

En cada nuevo bloque, la fuente y el relé van a cooperar para resolver la incertidumbre surgida en el diccionario de bloques enviados con anterioridad. Además, la fuente superpone nueva información del primer conjunto para la transmisión del segundo conjunto y poder así transmitir la suma.

El destino siempre se encuentra un bloque por detrás, pero tendrá suficientes bloques para no tener ningún efecto en la tasa global de recepción.

3.3 Teorema de capacidad para el canal relé

Atendiendo a la definición en el apartado 2.2 del canal relé discreto y sin memoria a través de $(\mathcal{X}x\mathcal{X}_1, p(y, y_1|x, x_1), \mathcal{Y}x\mathcal{Y}_1)$ se puede plantear el problema de encontrar la capacidad del canal entre la fuente y el destino.

En [9], se utilizó una aproximación en el reparto de tasas para encontrar los límites inferiores de la capacidad C. Los límites superiores fueron descubiertos en [9] y [12]. Sin embargo, dicha capacidad C fue establecida únicamente para canales degradados.

Por ello, surge otro motivo más para centrar el estudio en canales relés degradados cuyo ejemplo hemos visto anteriormente en el apartado 3.2 en la figura Fig. 11. Además, se partirá de la ecuación capacidad en ese mismo apartado para estudiarla en más detalle.

$$C^* = \max_{0<\alpha<1} \min\left\{C\left(\frac{P + P_1 + 2\sqrt{\bar{\alpha}PP_1}}{N_1 + N_2}\right), C\left(\frac{\alpha P}{N_1}\right)\right\}$$

Una interpretación para lograr C^* consiste, por ejemplo, en que relé Y_1 decodifique perfectamente la palabra código de la fuente X, luego el relé y la fuente cooperan coherentemente en el siguiente bloque para conseguir resolver la incertidumbre de x de los restantes y.

Generalizando, las funciones de codificación $x_1(\cdot)$, $f_i(\cdot)$ y la función de decodificación $g(\cdot)$ vistas en el apartado 3.2 se consideran funciones estocásticas.

La entrada al relé x_{1i} depende únicamente del pasado $y_1^i = (y_{11}, y_{12}, \dots, y_{1i-1})$ siendo ésta definitivamente la definición utilizada por van der Meulen. El canal es sin memoria en el sentido de que (y_i, y_{1i}) depende del pasado (x^i, x_1^i) a través de los actuales símbolos trasmitidos (x_i, x_{1i}). Por lo tanto, para cualquier elección de $p(w)$, $w\epsilon M$, y elección de código $x\colon \mathcal{M} \to X^n$ y funciones relé $\{f_i\}_{i=1}^n$, el máximo de la función de masa de probabilidad conjunta en $\mathcal{M}x\mathcal{X}^n x\mathcal{X}_1^n x\mathcal{Y}x\mathcal{Y}_1^n$ viene dada por la siguiente expresión como se vio en el apartado 2.2:

$$p(w, x, x_1, y, y_1) = p(w)\prod_{i=1}^{n} p(x_i|w)p(x_{1i}|y_{11}, y_{12}, \dots, y_{1i-1})p(y_i, y_{1i}|x_i, x_{1i})$$

Para recodar conceptos teóricos, si el mensaje $w\epsilon\mathcal{M}$ es enviado, se denotará la probabilidad condicional de error como $\lambda(w) = Pr\{g(Y) \neq w\}$. Se definirá la probabilidad de error media del código como $\overline{P_n}(e) = \frac{1}{M}\sum_w \lambda(w)$. La probabilidad de error se calcula bajo una distribución uniforme aplicada a $w\epsilon[1, M]$. Finalmente, $\lambda_n(w) = \max_{w\epsilon\mathcal{M}} \lambda(w)$ será la máxima probabilidad de error para el código (M, n). La tasa R de un código (M, n) se define por $R = \frac{1}{n}\log(M)$ bits/transmisión. Se dice que la tasa R es alcanzable por el canal relé si para cualquier $\epsilon > 0$ y cualquier n lo suficientemente grande, existe un código (M, n) con $M \geq 2^{nR}$ tal que $\lambda_n < \epsilon$. La capacidad C de un canal relé es el supremo del conjunto de tasas alcanzables.

Ahora se considera una familia de canales relé en los que el relé Y_1 es mejor que el destino Y, en el sentido de que el canal relés es degradado.

Por tanto, atendiendo al apartado 3.1 la capacidad C para un canal relé degradado vendrá dada por:

$$C' = \sup_{p(x_1,x_2)} min\big(I(X,X_1;Y), I(X;Y_1|X_1)\big)$$

donde el supremo se aplica a todas las distribuciones conjuntas $p(x,x_1)$ en $\mathcal{X}x\mathcal{X}_1$.

El primer término entre paréntesis de la ecuación anterior, sugiere que una tasa $I(X,X_1;Y)$ puede ser alcanzable siendo $p(x,x_1)$ arbitraria. Sin embargo, la tasa sólo será alcanzable si existe una completa cooperación entre el relé y la fuente. Para fijar esta cooperación el relé debe conocer x. Por lo tanto, la tasa de transmisión de x debería ser menor que $I(X;Y_1|X_1)$.

Para un mejor entendimiento, se ha elegido aplicar los resultados de la capacidad a un simple ejemplo introducido por Sato [12]. El canal es mostrado en la figura Fig. 12 tiene $\mathcal{X} = \mathcal{Y} = \mathcal{Y}_1 = \{0,1,2\}$, $\mathcal{X}_1 = \{0,1\}$ y la probabilidad condicional $p(y,y_1|x,x_1)$ satisface la condición de canal relé degradado $y_1 \equiv x$.

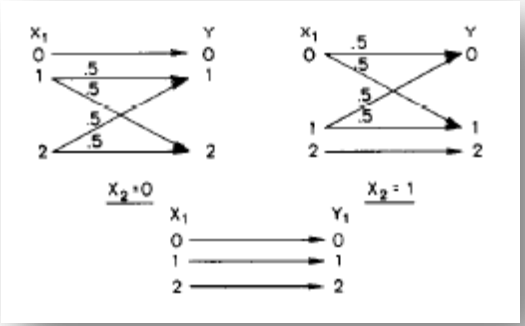


Fig. 12. ***Canal relé ternario*** **[13]**

		y = 0	y = 1	y = 2
$p(y\|y_1, x_1 = 0) =$	$y_1 = 0$	1	0	0
	$y_1 = 1$	0	0.5	0.5
	$y_1 = 2$	0	0.5	0.5

		y = 0	y = 1	y = 2
$p(y\|y_1, x_1 = 1) =$	$y_1 = 0$	0.5	0.5	0
	$y_1 = 1$	0.5	0.5	0
	$y_1 = 2$	0	0	1

Sato calculó un límite superior con cooperación de información para la capacidad del canal, $R_{UG} = \max_{p(x,x_1)} I(X,X_1;Y) = 1.170$. En el caso de que se restringiesen las funciones de codificación relé a:

1. $X_{1i} = f_i(y_{11}, \dots, y_{1i-1}) = f(y_{1i-1})\ \ 1 < i \leq n,$
2. $X_{1i} = f_i(y_{11}, \dots, y_{1i-1}) = f(y_{1i-2}, y_{1i-1}) \quad 1 < i \leq n.$

Sato calculó dos límites superiores para la capacidad de canal:

1. $R_1 = 1.0437$
2. $R_2 = 1.0549$

Para la capacidad C' se obtiene que el valor de la capacidad es de $C' = 1.161878$. La distribución conjunta óptima viene dada por los valores de la tabla Tabla 1.

	$x_1 = 0$	$x_1 = 1$	$x_1 = 2$
$x_2 = 0$	0.35431	0.072845	0.72845
$x_2 = 1$	0.072845	0.072845	0.35431

Tabla 1. ***Tabla $p(x, x_1)$ para el canal relé ternario***

Se observa que a pesar de permitir que las funciones de decodificación en el relé dependan solamente de un número finito de y_i transmisiones, se puede lograr una capacidad dependiente de x_1 e $y_1$de fomar similar que en el artículo [46].

3.4 Análisis del estudio de Razaghi y Yu sobre el canal relé aplicando la estrategia DAF con códigos LDPC de dos capa

3.4.1 Introducción

En el artículo [5] se vio que los códigos LDPC, como se ha venido explicando, han demostrado ser códigos potentes capaces de acercarse a la capacidad de canales de un único usuario. La idea clave de los códigos LDPC es implementar el esquema de codificación aleatoria de Shannon forzando un conjunto de restricciones aleatorias de chequeo de paridad en los bits de información.

Mientras que un código aleatorio es un elemento fundamental de la teoría de información para un único usuario, el "binning" es imprescindible en un entorno multiusuario. Tras intentar dar una visión general de la utilidad de los códigos LDPC, el análisis llevado a cabo en [5], propondrá una implementación del binning aplicado a códigos LDPC para el canal relé.

El trabajo en [5] se centra en la aplicación práctica de la estrategia DAF para canales relé. A pesar de ello, se restringirá el estudio a canales relé Gaussianos con baja relación señal a ruido (SNR). Se mostrará que, dentro de un marco lineal de decodificación, la estrategia de binning, en la que el índice de un grupo de la palabra código se transmite del relé al destino, puede ser interpretada como un esquema de reenvío de paridad. Por otra parte, el código de diseño óptimo para la estrategia DAF implica el diseño de un código LDPC que trabaje con diferentes SNR dependientes del camino a seguir: uno de ellos con una mayor relación señal a ruido correspondiente al relé y el otro con una menor SNR que afecta al destino. Este procedimiento representa una nueva metodología en la construcción de códigos LDPC llamado códigos LDPC de dos capas [5].

Los principales resultados que se obtendrán tras el estudio de esta nueva metodología serán dos conjuntos nuevos de códigos LDPC [4], los cuales abordan de forma simultánea las capacidades de dos canales Gaussianos con dos diferentes SNR. El análisis de rendimiento y las metodologías de diseño para estos dos nuevos conjuntos de códigos LDPC de dos capas se desarrollarán a través de la evolución de la densidad [47] para estandarizar los códigos LDPC a códigos de dos capas. Una técnica de diseño basada en la programación lineal se ha desarrollado para optimizar los grados de los códigos de dos capas. Las dos estructuras para códigos de dos capas se conciben para dar cabida a la optimización de los grados de los nodos de chequeo. Conjuntamente, se demostrará que los códigos LDPC irregulares con dos capas con una selección cuidadosa de secuencias de grados de nodos variable y de chequeo puede asintóticamente aproximarse a la tasa DAF teórica (para entradas binarias) de un canal relé dentro de una fracción de decibelio para diferentes condiciones de canal. Por último, la propuesta de diseño del código se puede generalizar para redes relé con múltiples relés y quedará demostrado que el enfoque de codificación que se propone es aplicable a redes más generales.

3.4.2 Codificación en el canal relé degradado AWGN

3.4.2.1 Estrategia DAF

Para el desarrollo teórico de la estrategia DAF se parte de un canal relé Gaussiano, como se muestra en la figura Fig. 11 del apartado 3.2.

Se recuerda que la fuente tendrá una restricción de potencia P, al igual que la restricción que hay en el relé es P_1.

Antes de continuar con el detallado desarrollo de dicha estrategia se planteará una breve revisión de la misma [13, Sección IV]. En la estrategia DAF, la fuente selecciona un nuevo mensaje $w_i \in \{1,2,\dots,2^{nR}\}$ para cada subconjunto i. El conjunto de mensajes de la fuente $\{1,2,\dots,2^{nR}\}$ se particiona aleatoriamente en subconjuntos de 2^{nR_1} bins $(R_1 \leq R)$ cuyo tamaño es $2^{n(R-R_1)}$. En el bloque i, el mensaje del relé se representa por s_i que es el índice del bin de w_{i-1} siendo éste el mensaje de la fuente en el bloque $i-1$. En el bloque i, el relé transmitirá $X_1(s_i)$, mientras que la fuente transmitirá:

$$X(w_i, s_i) = \tilde{X}(w_i) + \sqrt{\frac{(1-\alpha)P}{P_1}} X_1(s_i) \quad (3)$$

Siendo:

- $\tilde{X}(w_i)$ y $X_1(s_i)$ vectores aleatorios Gaussianos de tamaño n.

$\tilde{X}(w_i)$ será la palabra código tras codificar w_i y $X_1(s_i)$ será la palabra código tras codificar s_i, el proceso de codificación se lleva a cabo por medio de un diccionario de palabras código de tamaño 2^{nR} y 2^{nR_1}, generado mediante la distribución de probabilidad $p_{\tilde{X}}(\tilde{x}) \sim \mathcal{N}(0, \alpha P)$ y $p_{X_1}(x_1) \sim \mathcal{N}(0, \alpha P_1)$. Se tiene que tener encuenta que la ecuación (3) implica que la fuente divide de forma óptima su potencia total P en una fracción de αP para el nuevo mensaje transmido w_i y una fracción de $(1-\alpha)P$ para la transmisión conjunta del índice del bin s_i del anterior mensaje de la fuente w_{i-1}.

El proceso de decodificación en el bloque i será explicado en las líneas siguientes. El relé primero decodifica w_i en base a la siguiente expresión:

$$Y_1 = X + Z_1 = \tilde{X}(w_i) + \sqrt{\frac{(1-\alpha)P}{P_1}} X_1(s_i) + Z_1 \quad (4)$$

A continuación, el relé calcula s_{i+1}, el índice del bin de w_i, que será transmitido en el siguiente bloque. Dado que $X_1(s_i)$ es conocido por el relé, éste puede ser restado. Por tanto, una decodificación exitosa de $\tilde{X}(w_i)$ sería posible si:

$$R \leq I(X; Y_1 | X_1) = \frac{1}{2} \log\left(1 + \frac{\alpha P}{N_1}\right) \quad (5)$$

Como resultado, en el destino se puede observar:

$$Y = X + X_1 + Z_2 = \tilde{X}(w_i) + \left(1 + \sqrt{\frac{(1-\alpha)P}{P_1}}\right) X_1(s_i) + Z_2 \quad (6)$$

La decodificación de w_i tiene lugar en dos etapas. Primero, el módulo decodificador trata de realizar el proceso de decodificación sobre s_i y el índice del bin de w_{i-1}, mientras que el término $\tilde{X}(w_i)$ se considera ruido. La decodificación se habrá realizado con éxito si:

$$R_1 \leq \frac{1}{2}\log\left(1+\frac{\left(\sqrt{P_1}+\sqrt{(1-\alpha)P}\right)^2}{\alpha P+N_1+N_2}\right) \quad (7)$$

Una vez que s_i es conocido, el destino ahora puede restar $X_1(s_i)$ a Y y proceder con la decodificación de w_i en la segunda etapa. En esta etapa se llevará a cabo el siguiente bloque de codificación, después de que el índice del subconjunto de s_{i+1} sea decodificado. El índice del subconjunto de s_{i+1} restringe el número de candidatos w_i en un conjunto de tamaño $2^{n(R-R_1)}$. Por lo tanto, la decodificación de w_i (en el bloque $i+1$) tendrá éxito si:

$$R-R_1 \leq I\left(\tilde{X};Y|X_1\right)=\frac{1}{2}\log\left(1+\sqrt{\frac{(1-\alpha)P}{P_1}}\right) \quad (8)$$

Combinando las ecuaciones (5), (7) y (8), se puede ver que la tasa de DAF para un canal relé Gaussiano es:

$$R=\max_\alpha \min\left\{\frac{1}{2}\log\left(1+\sqrt{\frac{(1-\alpha)P}{P_1}}\right),\frac{1}{2}\log\left(1+\frac{P+P_1+2\sqrt{(1-\alpha)PP_1}}{N_1+N_2}\right)\right\} \quad (9)$$

El factor de cooperación óptimo en las expresiones anteriores es $\alpha=1$ si $\frac{P}{N_1}\geq\frac{P_1}{N_2}$, caso en el que la estrategia óptima no asigna una porción de la potencia de la fuente para la cooperación con el mensaje del relé [13]. Por lo tanto, no será necesaria una transmisión coherente entre el relé y la fuente [48, sección 42].

En el caso $\frac{P_1}{N_2}<\frac{P}{N_1}$, el valor óptimo de α se encuentra estrictamente entre 0 y 1. Este valor se halla si se igualan los dos términos para minimizar la expresión en (9), ver el apartado 3.2.

3.4.2.2 Codificación para la estrategia DAF

Obsérvese que el problema de diseño de código para la estrategia óptima DAF consiste en la construcción de dos códigos: $\mathcal{X}_1$ con una tasa R_1 y $\widetilde{\mathcal{X}}$ con una tasa R. Mientras el diccionario de palabras código del relé $\mathcal{X}_1$ se construye como un código convencional de corrección de errores que garantiza el éxito de la decodificación en el destino, el diccionario de palabras código de la fuente $\widetilde{\mathcal{X}}$ se construye de modo que se pueda aplicar la decodificación tanto en el relé como en el destino. El relé debe ser capaz de decodificar $\widetilde{\mathcal{X}}$ con una SNR:

$$SNR_+=\frac{\alpha P}{N_1} \quad (10)$$

Mientras el destino será capaz de decodificar $\widetilde{\mathcal{X}}$ bajo una diferente relación señal a ruido gracias a la ayuda del índice de un bin extra de información procedente del relé:

$$SNR_{-} = \frac{\alpha P}{N_1 + N_2} \quad (11)$$

La problemática de la construcción del código puede esquematizarse a través de la representación visualizada en la figura Fig. 13, donde las ecuaciones (12) y (13) se utilizan para indicar la eficiencia entre el enlace fuente-relé y la tasa entre fuente y destino.

$$R_{+} = \frac{1}{2}\log\left(1 + SNR_{+}\right) \quad (12)$$
$$R_{-} = \frac{1}{2}\log\left(1 + SNR_{-}\right) \quad (13)$$

La tasa general DAF en términos de R_{+} y R_{-} ahora llegará a ser:

$$R = \min\{R_{+}, R_1 + R_{-}\} \quad (14)$$

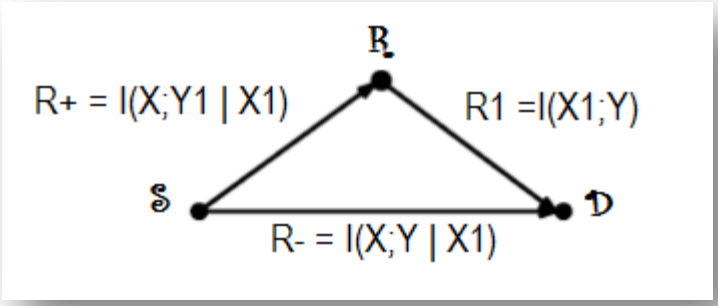


Fig. 13. ***Definición de la problemática de construcción de códigos DAF*** **[4]**

En resumen, el problema de construcción de códigos DAF se centra en dos subproblemas: la construcción de un diccionario de palabras código de la fuente para acercase al mismo tiempo a las tasas R_{+} y R_{-}, y la construcción de un diccionario convencional del relé para acercarse a la tasa $R = R_1 + R_{-}$.

Obsérvese que cada vez que $\frac{P_1}{N_2} < \frac{P}{N_1}$, la mejor opción del factor de asignación de potencia α en (9) siempre conducirá a que $R_{+} = R_1 + R_{-}$. Esto implica que el diccionario de palabras código de la fuente debe ser diseñado para abordar simultáneamente las capacidades efectivas de la fuente-relé y las capacidades efectivas de la fuente-destino en dos diferentes SNRs y en dos diferentes tasas.

La condición $R_{+} = R_1 + R_{-}$ no es necesariamente válida, si en el enlace relé-destino se cumple $\frac{P_1}{N_2} > \frac{P}{N_1}$, o si el factor de asignación de potencia α no es elegido óptimamente, llegando a encontrar una tasa DAF subóptima. En estos casos, un código convencional diseñado para canales de un solo usuario puede ser suficiente [24], [32]-[34].

Por ejemplo, cuando $R_{+} \ll R_1 + R_{-}$, la tasa general DAF está limitada por R_{+}. El relé puede transmitir el exceso de información de las palabras código de la fuente al destino sin coste alguno sobre la tasa general DAF. En este caso, el código fuente-destino puede operar a una tasa por debajo de R_{-}. Del mismo modo, cuando $R_{+} \gg R_1 + R_{-}$, la

tasa general DAF se encuentra limitada por $R_1 + R_-$. El código fuente-relé, en este caso puede operar a una tasa por debajo de R_+. En ambos casos, una de las dos restricciones entre $\min\{R_+, R_1 + R_-\}$ no se podrá llegar a obtener. En consecuencia, un código LDPC convencional para un único usuario se diseña para cumplir la limitación más estricta logrando $\min\{R_+, R_1 + R_-\}$. En general, sí y sólo si $\frac{P_1}{N_2} < \frac{P}{N_1}$, se podría optimizar el funcionamiento del relé con el fin de maximizar la tasa global. Este paso conlleva a que $R_+ = R_1 + R_-$, donde surge la necesidad de un código de la fuente que pueda lograr simultáneamente la capacidad a dos diferentes tasas y a dos diferentes SNRs.

3.4.2.3 Método de binning a través de la generación de paridad

Un factor principal en la estrategia DAF es el proceso de binning. Una de las principales preguntas que se podría plantear es cómo se puede aplicar el binning a un escenario práctico. Si se restringe la atención a los canales Gaussianos con baja relación señal a ruido (por ejemplo, $R < 1$) para que los códigos lineales sean óptimos, entonces el binning se implementa mediante la generación de bits de paridad adicional en las palabras código de $\widetilde{\mathcal{X}}$. La generación de estos bits de paridad (o síndromes) es una forma natural de partición de un diccionario de palabras códigos en bins, con lo que las palabras código en cada bin satisfacen un conjunto particular de ecuaciones de paridad. Los bits de paridad son exactamente los índices del bin. La idea de implementar la estructura de binnig a través de síndromes fue utilizada en el pasado por Slepian-Wolf [49] [50].

Para llevar a cabo el proceso de binning y la codificación en bloques de Markov usando esta idea, el relé tiene que decodificar la palabra código $\tilde{X}(w_i)$ transmitida en bloque i, generar bits de paridad extra para $\tilde{X}(w_i)$, codificar después con un diccionario de palabras código $\mathcal{X}_1$ independiente, y envíar los bits codificados al destino en el siguiente bloque.

El destino decodifica $\tilde{X}(w_i)$ utilizando los bits de paridad extra. Por lo tanto, la estrategia DAF es una estrategia de reenvío de paridad, y da lugar a una construcción de códigos de dos capas. El sistema de codificación basado en esta idea se describe en la figura Fig. 14, mostrándose de forma esquemática como:

a) El mensaje de la fuente se codifica con un código LDPC $(n, n - k_1)$.
b) El relé decodifica la palabra código proveniente de la fuente.
c) El relé genera k_2 bits adicionales de paridad.
d) Los k_2 bits de paridad son transmitidos al destino con un diccionario de palabras código diferente.
e) El destino primero decodifica los k_2 bits extra de paridad, luego decodifica el mensaje de la fuente sobre el código de dos capas mediante la búsqueda de la palabra código que satisfaga k_1 bits de paridad cero y k_2 bits de paridad diferentes de cero

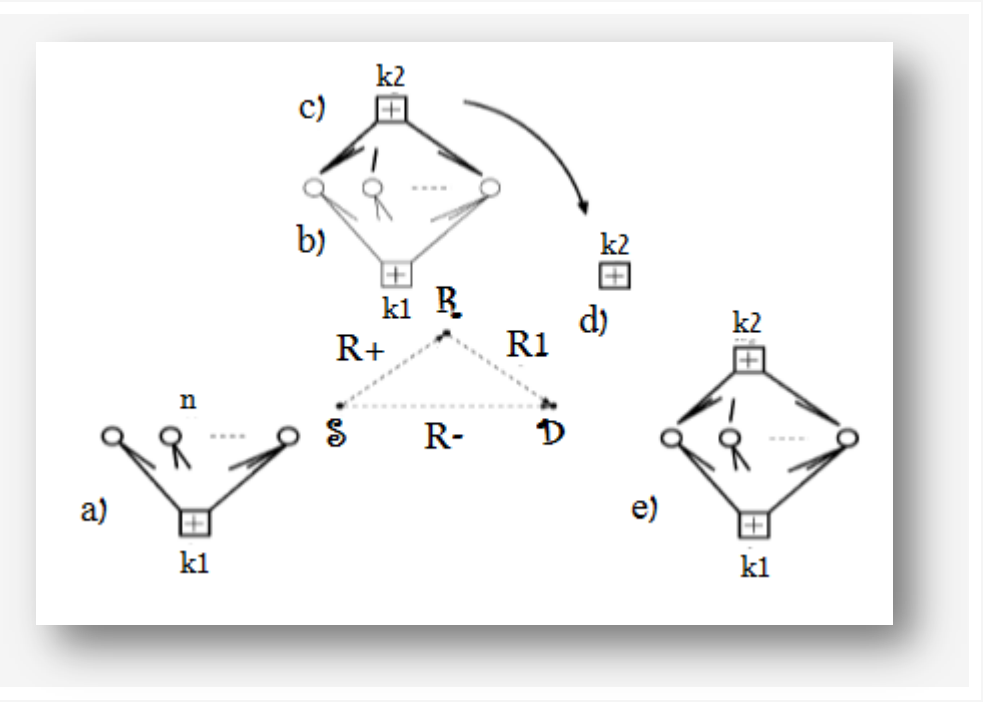


Fig. 14. ***Paridad de reenvío de la aplicación de DAF con códigos LDPC.*** **[4]**

Antes de considerar el diseño de códigos de dos capas para acercarse a la capacidad de canal de un canal relé Gaussiano, es útil preguntarse si ese tipo de códigos existe en la teoría. Es bien sabido que códigos lineales aleatorios pueden lograr la capacidad requerida debido a la simetría de un canal binario (y para el canal Gaussiano a una baja SNR) bajo la máxima probabilidad de decodificación. Un subcódigo de un código aleatorio es también un código aleatorio. Consecuentemente, bajo la máxima probabilidad de decodificación, un código de dos capas, que puede lograr la capacidad con dos diferentes SNRs, se puede encontrar.

La pregunta se vuelve más interesante, si tenemos en cuenta métodos iterativos prácticos de decodificación. En este ámbito, resultados teóricos se encuentran disponibles sólo para un modelo de canal BEC, para lo cual lograr la creación de secuencias de los grados para los códigos LDPC bajo métodos iterativos de decodificación pueden ser identificados en [114]-[116].

Capítulo 4

Códigos sobre grafos de factores

4.1 Introducción

En este capítulo se intenta presentar la conexión existente entre el grafo de factores (FG) y el algoritmo de decodificación "Sum-Product" (SP), dando lugar a una forma sencilla de entender un gran número de algoritmos aparentemente diferentes que han sido desarrollados en el área de la ingeniería. Se van a tener en cuenta algoritmos que hacen frente a complicadas funciones "globales" de muchas variables y que derivan su eficiencia computacional explotando la forma de convertir los factores de la función global en productos simples de funciones locales, en el que cada uno depende de un subconjunto de variables. Tal factorización puede ser visualizada utilizando el FG, es decir, un grafo bipartito que expresa qué variables son argumentos de funciones locales.

Por tanto, el objetivo será la descripción de un algoritmo de paso de mensajes genérico llamado algoritmo SP, que opera con el FG e intenta calcular diversas funciones marginales asociadas con la función global. Las ideas básicas son sencillas, y sin embargo, como se mostrará más adelante, cubren una sorprendente variedad de algoritmos desarrollados en inteligencia artificial, procesamiento de señales y en comunicaciones digitales que se pueden derivar como casos específicos del algoritmo SP, operando de forma apropiada con un FG.

Genealógicamente, los FGs son una generalización directa de los "grafos de Tanner" [51], entre otros [52]. Tanner [53], introdujo el concepto de grafos bipartitos, ver figura Fig. 15, para describir familias de códigos que son generalizaciones de códigos de

Gallager de verificación de paridad de baja densidad (LDPC) [54] y también el algoritmo de SP, en este contexto.

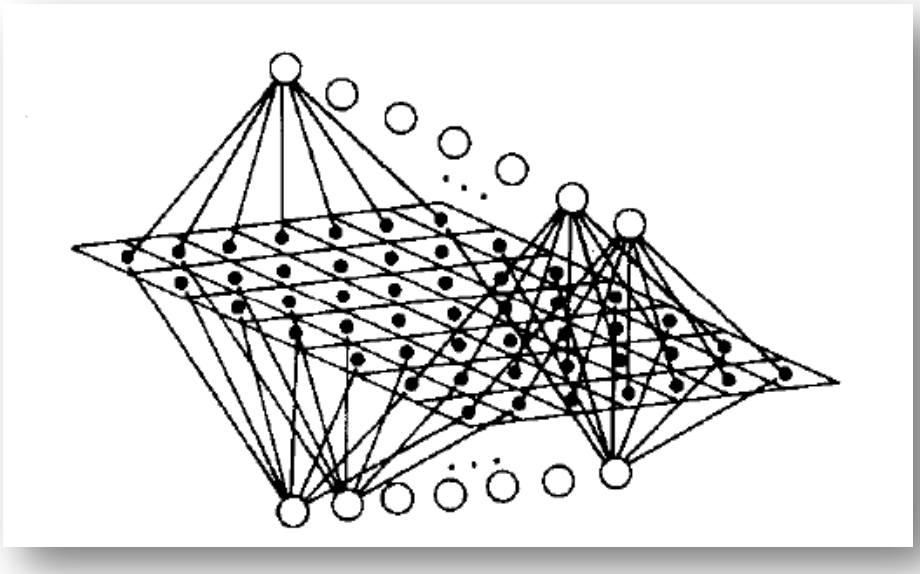


Fig. 15. ***Grafo bipartito para el producto de dos códigos Hamming [7, 4, 3].*** **[53]**

En la formulación original de Tanner, todas las variables son símbolos de la palabra código y por tanto "visibles", Wiberg y otros introdujeron las variables de estado "latentes" y sugirieron también aplicaciones más allá de la codificación. Los FGs llevan los modelos teóricos de grafo un paso más allá, aplicándolo también a funciones. Desde la perspectiva del FG, un grafo de Tanner para un código representa una factorización particular de los indicadores de la función código. Si bien puede parecer intuitivamente razonable que algunos algoritmos deben explotar la manera en que una función global de factores puede transformarse en un producto de funciones locales, la idea fundamental es que muchos algoritmos bien conocidos resuelven el problema de la marginalización del producto de funciones ("MPF"). En un artículo de referencia [55], Ají y McEliece desarrollaron una "ley distributiva generalizada" (GDL), que en algunos casos resuelve el problema MPF con una representación de "árbol de uniones" de la función global. Los FGs pueden ser vistos como un enfoque alternativo más estrecho a las gráficas de Tanner y un desarrollo para la representación gráfica de los códigos.

Esencialmente, cada resultado desarrollado en la unión árbol/GDL puede traducirse en un resultado equivalente en el FG/algoritmo SP y viceversa. Se considera que el último ajuste es el más óptimo, no sólo porque conecta mejor con los enfoques mencionados anteriormente, sino también porque los FGs son de alguna manera más fácil de describir. Además, el algoritmo SP a menudo puede aplicarse con éxito en situaciones donde las soluciones exactas al problema del MPF (conforme a lo dispuesto a las uniones del árbol) se convierten computacionalmente en un hecho intratable. El ejemplo más destacado es el proceso iterativo de decodificación de códigos turbo y códigos LDPC.

También, hay estrechos vínculos entre los FGs y los modelos gráficos para las distribuciones de probabilidad multidimensional, tales como los campos aleatorios de Markov [56], [57], [58] y las redes Bayesianas [59] [60]. Al igual que los FGs, estos modelos gráficos codifican en su estructura una factorización particular de la función de

probabilidad conjunta de varias variables aleatorias. El algoritmo de Pearl, "Belief-Propagation" (BP) [59], que opera a través de "paso de mensajes" en una red bayesiana, se traduce inmediatamente en una instancia del algoritmo SP operando en un FG que expresa la misma factorización. Las redes Bayesianas y las de BP han sido utilizadas anteriormente para explicar el proceso iterativo de decodificación para códigos turbo y LDPC [61], [62], [63], [64], [65], [66]. Nótese, sin embargo, que Wieberg [51] había descrito con anterioridad estos algoritmos de decodificación como ejemplos del algoritmo de SP [67].

En apartados posteriores dentro de este capítulo, se presentará un sencillo ejemplo que ilustra el funcionamiento del algoritmo SP a través de un FG. Se verá que cuando el FG no presenta ciclos, entonces la estructura de este no solamente permite codificar a través de funciones de factores, sino que también codifica expresiones para el cálculo de varias funciones marginales asociadas a una función dada. Dichas expresiones dan lugar directamente al algoritmo SP.

En el apartado 4.4, se muestra como el FG se puede usar como una herramienta de modelado de sistemas y señales, viéndose que el FG es compatible con los estilos de modelización probabilística y conductual.

4.1.1 Modelando sistemas con el grafo de factores

Ahora, se describirán diversas formas en que el FG puede ser utilizado para la modelización de sistemas, es decir, colecciones de interacciones entre variables.

En el modelado probabilístico de sistemas, el FG puede ser usado para representar la función de probabilidad conjunta de las variables que comprenden el sistema. La factorización de esta función puede proporcionar información trascendental acerca de las dependencias estadísticas entre estas variables.

Del mismo modo, en el modelado conductual de sistemas, como en la obra [52], el comportamiento del sistema se especifica en términos de la teoría de conjuntos especificando que configuraciones particulares de las variables son válidas. Este enfoque puede ser aceptado por el FG que representa la función característica para el comportamiento dado. La factorización de esta función característica puede proporcionar información estructural valiosa sobre el modelo.

En algunas aplicaciones, incluso se pueden combinar estos dos estilos de comportamiento. Por ejemplo, en la codificación de canal, se modela tanto el comportamiento válido (es decir, el conjunto de palabras código), como la función de probabilidad conjunta a posteriori sobre las variables que definen las palabras códigos dadas por la salida recibida de un canal dado. (Si bien incluso puede ser factible para modelar canales complicados con memoria [69], a lo largo de este apartado se tratarán canales sin memoria).

En el modelado conductual, "La convención de Iverson [70 p.24]" puede ser útil. Si P es un predicado (proposición booleana) que envuelve a un conjunto de variables, entonces $[P]$ es la función evaluada en $\{0,1\}$ que indica la verdad de P, es decir:

$$[P] \coloneqq \left\{ \frac{1, si\ P\ es\ cierto}{0,\ en\ caso\ contrario} \right\}$$

Si ^ denota la conjunción lógica o el operador "AND", entonces una importante propiedad de la convención de Iverson es que:

$$[P_1 \text{^} P_2 \text{^} \dots \text{^} P_n] = [P_1][P_2] \dots [P_n]$$

4.1.2 Modelado conductual

Sea $x_1, x_2, \dots, x_n$ una colección de variables configurado el espacio $\mathcal{S} = \mathcal{A}_1 \times \mathcal{A}_2 \times \dots \times \mathcal{A}_n$. Por un comportamiento en $\mathcal{S}$, se refiere a cualquier subconjunto B de $\mathcal{S}$. Los elementos de B serán las configuraciones válidas. Puesto que un sistema se especifica a través de su comportamiento B, dicho enfoque se conoce como modelado conductual [52].

La utilización del modelado conductual es natural para los códigos. Si el dominio de cada variable es un alfabeto finito A, de modo que la configuración espacial es n veces el producto Cartesiano $S = A^n$, por lo tanto, un comportamiento $C \subset \mathcal{S}$ es llamado código bloque de longitud n sobre A, y las configuraciones válidas son llamadas palabras código.

La función característica de un comportamiento B se define como:

$$X_B(x_1, x_2, \dots, x_n) \coloneqq [(x_1, x_2, \dots, x_n) \epsilon B]$$

Obviamente, especificar X_B es equivalente a especificar B. (También se podría proporcionar a X_B una interpretación probabilística señalando que es proporcional a una función de probabilidad que sea uniforme sobre todas las configuraciones válidas).

En muchos casos importantes, la pertenencia a una configuración particular en un comportamiento B puede ser determinada mediante la aplicación de una serie de pruebas (chequeos), cada uno implicando un cierto subconjunto de variables.

Una configuración se considerará válida si y sólo si se pasan todas las pruebas; es decir, el predicado $(x_1, x_2, \dots, x_n) \epsilon B$ puede ser escrito como un conjunto lógico de una serie de predicados más simples. Por tanto, los factores de X_B se relacionan con un producto de funciones características, cada una indicando si un determinado subconjunto de variables es un elemento de algún comportamiento local.

4.2 Definición

El FG está considerado como la aplicación sistemática y eficiente de la propiedad distributiva a la hora de marginalizar funciones de varias variables que toman valores dentro de un conjunto discreto $\mathcal{X}$ ya que utiliza la factorización para reducir el coste computacional.

Para la representación del FG se define una función y su factorización en producto de funciones: $f \rightarrow f_1, f_2, f_3, f_4$ en la que:

- A cada variable se le asocia un nodo variable representado por un círculo.
- A cada función f_1, f_2, f_3, f_4 se le asocia un nodo factor representado por un cuadrado.
- Se conecta un nodo variable con un nodo factor si la función correspondiente depende de ese nodo factor.

La figura Fig. 16 muestra la representación de la factorización $f(x_1, x_2, x_3, x_4, x_5, x_6) = f_1(x_1, x_2, x_3)\, f_2(x_1, x_4, x_6) f_3(x_4) f_4(x_4, x_5)$ donde se quiere obtener:

$$f(x_1) = \sum_{x_2, x_3, x_4, x_5, x_6} f(x_1, x_2, x_3, x_4, x_5, x_6) = \sum_{x_2, x_3, x_4, x_5, x_6} f(x_1, x_2, x_3, x_4, x_5, x_6)$$

Aplicando la factorización mostrada en la figura Fig. 16 se puede concluir que:

$$f(x_1) = \left[\sum_{x_2, x_3} f_1(x_1, x_2, x_3)\right] \times \left[\sum_{x_4} f_3(x_4) \sum_{x_6} f_3(x_1, x_4, x_6) \sum_{x_5} f_4(x_4, x_5)\right]$$

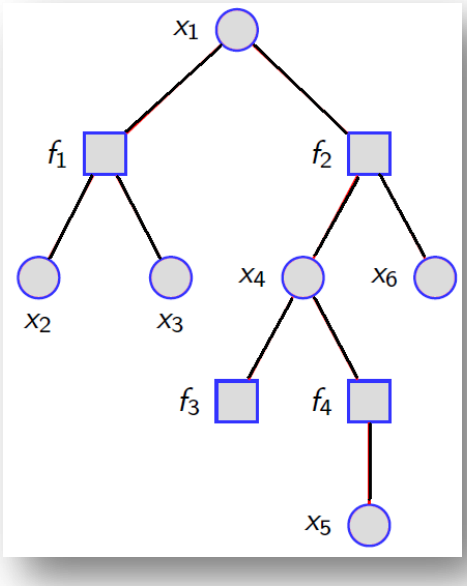


Fig. 16. ***Ejemplo de un FG.*** **[68]**

4.3 Gráfico de Tanner para códigos lineales

La función característica para cualquier código lineal definida por una matriz de chequeo de paridad H de $r \times n$ puede ser representada por un FG con n nodos variable y r nodos factor. Por ejemplo, si C es un código lineal binario con una matriz de chequeo:

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

Entonces C es el conjunto de todas las 6-tuplas binarias $x \triangleq (x_1, x_2, \ldots, x_6)$ que satisface tres ecuaciones simultáneas expresadas en la forma de la matriz como $Hx^T = 0$. La pertenencia en C está completamente determinada por comprobar si cada una de las tres ecuaciones se satisface.

Por lo tanto, aplicando a la matriz de paridad H el concepto de convención de Iverson se tiene que:

$$\begin{aligned} X \subset (x_1, x_2, \ldots, x_6) &= [(x_1, x_2, \ldots, x_6) \in \mathrm{C}] \\ &= [x_1 \oplus x_2 \oplus x_5 = 0][x_2 \oplus x_3 \oplus x_6 = 0][x_1 \oplus x_3 \oplus x_4 = 0] \end{aligned}$$

donde $\oplus$ denota la suma GF(2). El correspondiente FG se representa en la figura Fig. 17, donde se ha utilizado un símbolo especial para representar las comprobaciones de paridad (un cuadrado con el sigo "+"). Aunque estrictamente hablando, el FG representa la factorización de la función característica del código, que con frecuencia se refiere al FG como una representación del código en sí mismo. El FG obtenido de esta manera recibe el nombre del Grafo de Tanner [53].

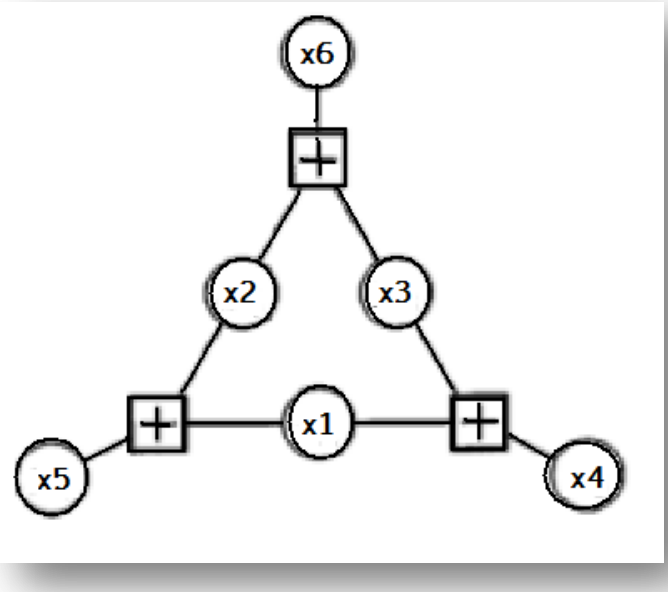


Fig. 17. ***Grafo de Tanner para el código lineal representado por H*** **[62]**

Resulta obvio que un grafo de Tanner, para cualquier código lineal de bloque $[n, k]$, puede ser obtenido a través de una matriz de chequeo de paridad $H = [h_{ij}]$. Esta matriz de chequeo de paridad tiene n columnas y al menos $n - k$ filas. Los nodos variable corresponden a las columnas de H y los nodo factor (de chequeo) a las filas de H, con una arista conectando al nodo factor i con el nodo variable j si y sólo si $h_{ij} \neq 0$. Puesto que hay muchas matrices de chequeo de paridad para la representación de un código dado, hay en general, muchos grafos de Tanner para la representación de dicho código.

4.4 Códigos definidos en grafos

Desde 1948, cuando Claude Shannon introdujo la notación de capacidad de canal [6], siendo el objetivo final de la decodificación de canal encontrar una aproximación práctica del canal. De acuerdo con el teorema de capacidad de canal de Shannon, la comunicación será fiable con una tasa R ($\frac{bits}{uso\ de\ canal}$) sobre un canal Gaussiano de ruido blanco (AWGN), siempre y cuando se cumplan las condiciones mínimas de señal a ruido del canal, cumpliendo el límite de Shannon. El canal AWGN es un modelo de comunicación en el que el deterioro de la comunicación sólo se debe a la suma de un ancho de banda lineal o ruido blanco con una densidad espectral constante y una distribución Gaussiana en amplitud. Dicho modelo no tiene en cuenta el fading, la selectividad en frecuencia, la interferencia, la no linealidad o la dispersión. Sin embargo, con este modelo se consiguen generar modelos matemáticos simples y manejables que son útiles para obtener información sobre el comportamiento subyacente de un sistema antes de que estos fenómenos se tengan en cuenta.

Si hablamos sobre ratios normalizados de densidad de energía a ruido de bit $\frac{E_b}{N_0}$, la comunicación tendrá lugar de forma fiable si se cumple la siguiente condición de tasa R: $\frac{E_b}{N_0} \geq \frac{2^{2R}-1}{2^R}$, donde E_b es la energía media por bit transmitido y $\frac{N_0}{2}$ es la varianza media en canales con ruido Gaussiano. El límite de Shannon está referido al mínimo alcanzable en $\frac{E_b}{N_0}$.

Aproximarse al límite de Shannon con niveles bajos de decibelios fue posible gracias al uso de códigos con una gran complejidad de decodificación como los códigos convolucionales, pero una mayor reducción de dichos niveles requería una complejidad inalcanzable hasta el descubrimiento de los turbo códigos [71]. Una de las más importantes innovaciones introducidas por los turbo códigos fue la creación de una clase de reglas de baja complejidad para una decodificación subóptima, como por ejemplo los algoritmos de paso de mensajes iterativos. Usando el paso de mensaje en el decodificador, los turbo códigos proveen un excelente desarrollo y un pequeño salto para alcanzar el límite de Shannon con una complejidad de decodificación baja. Atendiendo a la figura Fig. 18, podemos observar la comparación de rendimientos entre los turbo códigos y los códigos convolucionales dentro de un canal AWGN. El alto rendimiento observado en el gráfico lleva a prestar una gran atención a los turbo códigos, gracias a los cuales se ha promovido una nueva clase de códigos llamados códigos definidos en grafos.

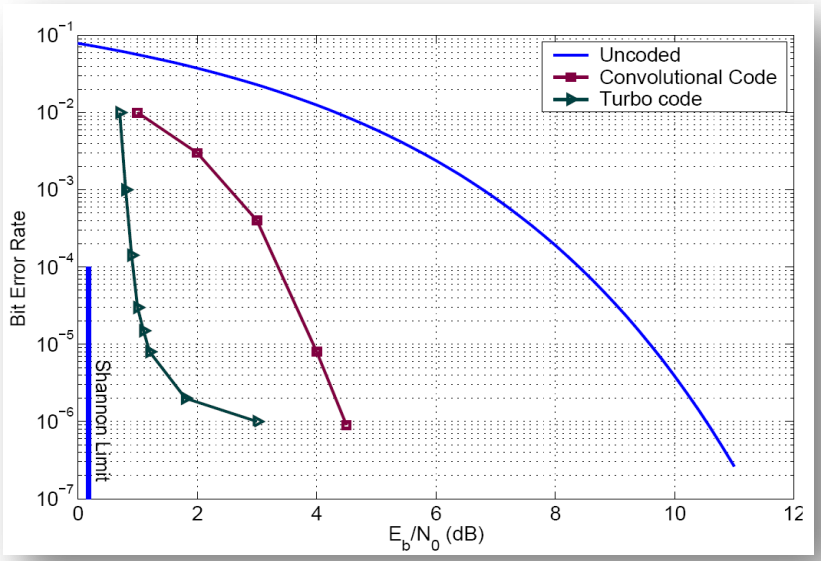


Fig. 18. ***Gráfico rendimientos entre turbo códigos y códigos convolucionales*** **[72]**

Los códigos definidos a través de grafos pueden ser decodificados con el algoritmo de paso de mensaje. La utilización de este algoritmo para esta familia de códigos hace que sea una combinación bastante atractiva ya que la decodificación está muy cerca de alcanzar los rendimientos óptimos y en la práctica, la complejidad aumenta linealmente con la longitud del código. Por lo tanto, 50 años después del trabajo de Shannon, especialistas en la decodificación han sido capaces de descubrir códigos que permitan alcanzar el límite de Shannon con una complejidad de decodificación razonable.

4.5 Códigos LDPC

4.5.1 Introducción

En este apartado se quiere introducir los términos de análisis, diseño y decodificación de una serie de códigos para el control de errores que poseen una gran potencia y flexibilidad, dichos códigos se conocen con el nombre de "*Códigos de chequeo de paridad con baja densidad*", esta familia de códigos de chequeo de paridad son conocidos por el nombre de LDPC que provienen del término anglosajón "Low-density parity-check codes". Los códigos LDPC han sido diseñados para manejar de forma eficiente el término capacidad, mencionado en apartados anteriores, para una gran variedad de canales con una cierta complejidad de decodificación. Se ha estudiado que con estos códigos se puede alcanzar la capacidad de muchos canales, e incluso se ha demostrado que se puede lograr la capacidad de un canal binario de borrado (BEC) a través la decodificación iterativa. El canal de borrado binario [73], constituye un simple ejemplo de un modelo de canal no trivial.

4.5.2 Historia sobre códigos LDPC

La comprensión de esta familia de códigos comenzó con el estudio de Tanner sobre grafos en códigos lineales [53]. Más tarde, Wiberg descubrió que los turbo códigos también podían representarse gráficamente [69]. Después de este descubrimiento, tanto en [63] como [66], se demostró que el algoritmo de decodificación turbo con una representación gráfica de un turbo código era un caso especial de propagación en una red Bayesiana [59].

Paralelamente a la investigación de los turbo códigos, en 1996 Mackay y Neal en [26], como Sipser y Spielman en [74] redescubrieron una familia de códigos olvidados llamados códigos LDPC. Esta clase de códigos fueron originalmente propuestos en 1962 por Gallager [54], pero fueron descartados por ser demasiado complejos para la época.

Los códigos LDPC llamaron mucho la atención debido a que con ellos era posible alcanzar el límite de Shannon con una diferencia mínima de dBs. Otra gran característica de estos códigos es su sencilla representación gráfica que se basa en la representación de Tanner para códigos lineales [53].

Esta simple estructura permite un análisis preciso de los códigos LDPC [75] [47], como también permite un buen diseño para códigos LDPC irregulares, optimizados mediante limitaciones específicas.

Desde el descubrimiento de los códigos LDPC, se han llevado a cabo una gran variedad de investigaciones y mejoras en el área de la utilización de grafos para códigos. Sin lugar a dudas, la investigación sobre códigos LDPC ha desempeñado y seguirá desempeñando un papel importante en este campo, ya que muchas de las nuevas clases de códigos definidos a través de grafos se verán influidos por los códigos LDPC. Algunos ejemplos que cabe destacar podrían ser: Códigos de repetición acumulada (RA) [76], Códigos de transformación Luby [36] y Códigos concatenados en forma de árbol [77].

Algunas claves sobre las mejoras en el ámbito de códigos gráficos y su importancia se describen a continuación.

- **Códigos LDPC irregulares**. Como se muestra en [61], los códigos LDPC irregulares provienen del desarrollo de códigos LDPC regulares. Todos los códigos LDPC que son capaces de aproximarse al límite de Shannon para los diferentes canales son los códigos LDPC irregulares. El descubrimiento de los mismos, ocasionó la transformación de estructuras irregulares para otros códigos definidos por grafos tales como turbo códigos irregulares [78] y códigos RA irregulares [79].
- **Códigos de repetición acumulada**: Uno de los problemas de los códigos LDPC es su complejidad en la codificación. Una de las mejores soluciones para restar parte de su complejidad es introducir nuevas estructuras a estos códigos, siendo una de las mejores soluciones los códigos RA [76] que sufren una pérdida de rendimiento en comparación con los códigos LDPC. La complejidad de los códigos RA aumenta linealmente con la longitud del bloque.
- **Obtención de capacidad con códigos LDPC en un modelado de canal BEC**: Shokrollahi encontró una familia de códigos LDPC irregulares que podrían lograr la capacidad en un canal BEC [80] [81].
- **Análisis de la evolución de los códigos LDPC**: Un análisis en profundidad de los códigos LDPC bajo diferentes núcleos de decodificación fue realizado en [47]. La principal idea es seguir la evolución de la densidad de los mensajes en el decodificador. Gracias a este análisis, diseñar buenos códigos LDPC irregulares que ya habían sido estudiados para canales BEC fue posible trasladarlo a otro tipo de canales.
- **Análisis Gaussiano sobre códigos LDPC y turbo códigos [82]-[84]:** Debido a la evolución de la complejidad computacional de la densidad, se han llevado a cabo numerosos estudios para conseguir una aproximación válida para esa densidad. En particular, la aproximación de densidad de mensajes con la densidad Gaussiana parece ser la más efectiva.
- **Códigos de transformación de Luby [36]**: La idea sobre códigos de baja tasa de envío es una de las últimas invenciones en el campo de la teoría de la codificación. Estos códigos son de gran utilidad con transmisores con modelos de radiodifusión, cuyos canales hacia el destino son diferentes unos con otros. En tales casos, no está muy claro qué tasa de código debería ser utilizada para la protección de datos. Los códigos LT pertenecen a esta familia de códigos de baja tasa de envío que solventan este problema. Para ser más específicos, cada destino obtiene una tasa de envío de datos dependiendo de las condiciones del canal.

En conclusión, los códigos LDPC son unos de los más importantes códigos definidos a través de grafos. Esto se debe a su excelente rendimiento, así como a su estructura simple y flexible. Los códigos LDPC se utilizan hoy en día para algunos estándares, tales como ETSI EN 302 307 para transmisión de video digital [85] y IEEE 802.16 (Grupo de trabajo en acceso de banda ancha inalámbrico) para codificación en accesos múltiple por división en frecuencia ortogonal (OFDMA) [86].

4.5.3 Códigos LDPC y su análisis

4.3.3.1 Modelos gráficos y decodificación por paso de mensajes

Los modelos gráficos son ampliamente utilizados en muchos sistemas probabilísticos multivariable clásicos, abarcando campos como la estadística, teoría de la información, reconocimiento de patrones, aprendizaje automático y teoría de la codificación.

Uno de los pasos importantes en la representación gráfica de códigos fue la introducción del concepto del factor de un grafo [87]. El FG representa la factorización de una función multivariable en funciones simples. Por ejemplo, en la siguiente ecuación se muestra el FG junto con la factorización pertinente:

$$f(x_1, x_2, x_3, x_4) = f_1(x_1, x_2) \cdot f_2(x_2, x_3, x_4) \cdot f_3(x_4)$$

En la figura Fig. 19, se representarán las variables a través de nodos circulares y las funciones por medio de nodos cuadrados. Una función nodo es adyacente al resto de nodos por medio de sus argumentos.

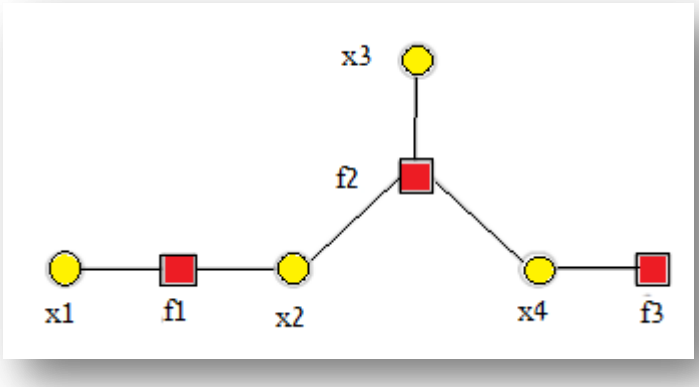


Fig. 19. ***FG que representa la ecuación de factorización anterior*** **[72]**

La función de densidad de probabilidad (FDP) se factoriza a menudo por medio de funciones de densidad locales, cada una de ellas es una función de un subconjunto de variables. Por lo tanto, el FG es un modelo gráfico que es usado convenientemente dentro de una problemática probabilística.

Cuando el FG no tiene bucles, como es el caso de existir un único camino entre un par de nodos, usando el algoritmo SP [87], todas las funciones de densidad de probabilidad marginales pueden calcularse. El algoritmo SP es equivalente al algoritmo de propagación de Pearl para redes Bayesianas [87] y reduce el proceso de marginalización de una función global en un conjunto de operaciones locales de paso de mensajes.

Estudios recientes [80], [88]-[91] han demostrado el efecto de los bucles en el funcionamiento de estos algoritmos. En la mayoría de los análisis en códigos definidos a través de grafos, no se tiene en cuenta el efecto causado por los bucles.

4.3.3.2 Códigos LDPC: Estructura

Un código LDPC se clasifica dentro de los códigos lineales de bloque y por lo tanto, se configuran a través de una matriz de chequeo de paridad. La diferencia entre un código LDPC de un código lineal convencional es la matriz de chequeo de paridad que será aleatoria, por ejemplo, el número de entradas distintas de cero es mucho más pequeño que el número total de entradas.

La representación gráfica de estos códigos se ha extendido tan rápidamente que la mayoría de la gente piensa que hablar de códigos LDPC en términos de estructura es lo mismo que su FG.

Un FG es siempre un grafo bipartito cuyos nodos se dividen en elementos de nodos variables y nodos funciones o como es lo mismo nodos de chequeo [87] [92]. Para proporcionar una visión en profundidad del concepto que se quiere desarrollar, se proporcionará a continuación un ejemplo sobre un grafo bipartito y como un código lineal puede ser transformado en él.

Considérese un grafo bipartito G con n nodos variables, r nodos de chequeo y E aristas. En la figura Fig. 20, se puede observar un ejemplo gráfico de la definición de grafo bipartito. En la figura, se verán representados los nodos variables mediante círculos y los nodos de chequeo mediante cuadrados.

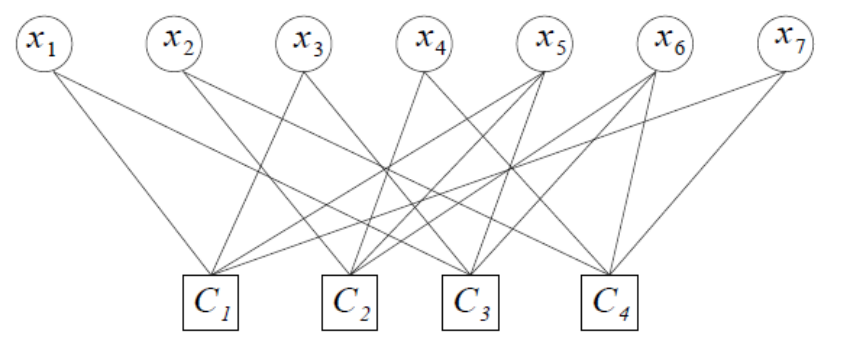


Fig. 20. ***Código de chequeo de paridad a través de un grafo bipartito*** **[72]**

Un nodo variable v_j es una variable binaria perteneciente al alfabeto {0,1} y un nodo de chequeo c_i corresponde a una restricción de paridad que deben cumplir sus nodos variable vecinos, es decir,

$$\bigotimes_{j:v_j \,\epsilon\, n(c_i)} v_j = 0.$$

Siendo:

- $n(c_i)$: conjunto de todos los vecinos de c_i.
- $\otimes$: operador matemático que representa la suma en módulo 2.

El resultado será un código binario lineal de longitud n y de dimensión $k \geq n - r$, si y sólo si, todas las restricciones de paridad son linealmente independientes. La matriz de verificación de paridad H de este código es la matriz de adyacencia del grafo G, es decir, la entrada (i, j) de H, h_{ij}, es 1 si y sólo si c_i está conectado a v_j.

Los códigos LDPC se pueden extender al espacio $GF(q)$ si se considera un conjunto de pesos distintos de cero $w_{i,j} \epsilon\, GF(q)$ para las aristas de G. En este caso, la matriz de chequeo de paridad estará formada por el conjunto de pesos. En otras palabras, $h_{ij} = w_{ij}$.

Una vez demostrado gráficamente el concepto de grafo bipartito, puede entenderse que cualquier grafo bipartito G puede dar lugar a un código lineal. En el caso de los códigos LDPC, el número de asistas E del FG es comparable con el número de aristas de un grafo bipartito construido al azar. Como se explicará más adelante para un código LDPC con una tasa fija R, el número de aristas es de orden n, mientras que para un grafo bipartito construido al azar tiene $\frac{1}{2}n^2(1 - R)$ aristas. Por lo tanto, si n aumenta, el ratio $\frac{2E}{n^2(1-R)}$ decrece, resultando así un código más reducido.

Los códigos LDPC, dependiendo de su estructura, pueden ser clasificados como códigos regulares o irregulares. Los códigos regulares tienen un grado fijo tanto para los nodos variable como para los nodos de chequeo. El grado de un nodo variable de un código regular se denota como d_v y el grado de un nodo de chequeo como d_c, cumpliéndose siempre las siguientes igualdades:

$$E = r \cdot d_c = n \cdot d_v$$

Por lo tanto, la tasa de código R se puede calcular de la siguiente forma:

$$R \coloneqq \frac{k}{n} \geq \frac{n - r}{n} = 1 - \frac{d_v}{d_c}.$$

Si las filas de H son linealmente independientes, entonces $R = 1 - \frac{d_v}{d_c}$. Generalmente, el ratio $\frac{n-r}{n}$ se conoce con el nombre tasa de diseño [75], pero puede existir una dependencia lineal entre las filas de H que será ignorada, y la tasa de diseño y la tasa real se asumirán iguales.

Se considera ahora el conjunto de códigos LDPC regulares con grado variable d_v, grado de chequeo d_c y longitud n. Si n es lo suficientemente grande, el comportamiento promedio de casi todos los casos de este conjunto se centran en torno a un comportamiento esperado [75]. Por lo tanto, los códigos LDPC regulares se definen a través de su grado variable, de chequeo y su longitud. Cuando el rendimiento y las propiedades de los códigos LDPC regulares con una longitud suficientemente larga adquieren interés, se representarán únicamente mediante el grado de sus nodos variable y de chequeo. Por ejemplo, un código LDPC (3,6) hace referencia a un código con nodos variable de grado 3 y los nodos de chequeo de grado 6. La tasa de diseño siguiendo la ecuación anterior será de $\frac{1}{2}$.

Aunque los códigos LDPC regulares operan cerca de su capacidad, la capacidad de los turbo códigos es mucho más eficiente existiendo un margen entre ellos. La principal ventaja de los códigos LDPC regulares frente a los turbo códigos es el error de piso. Como puede verse en la figura Fig. 18, comparando valores bajos de SNR, para valores de SNR altos y moderados, la curva de BER de los turbo códigos tiene una pendiente menor. Este es el fenómeno conocido como error de piso. Este fenómeno es una propiedad fundamental debido a la baja distancia de los turbo códigos [93]. Por lo tanto, los turbo códigos experimentarán un error de piso incluso por debajo de una decodificación óptima. Los códigos LDPC, sin embargo, como puede comprobarse en [94] y verse en la figura Fig. 18 tienen un error de piso mejor. Además, en [65] se demuestra que los códigos LDPC pueden alcanzar el límite de Shannon bajo una decodificación óptima.

Los códigos LDPC comienzan a ser vistos de forma más atractiva cuando Luby demuestra que ese margen de capacidad puede ser reducido utilizando códigos LDPC irregulares [61]. Un código LDPC se denomina irregular si su FG e nodos variable (y/o de chequeo) no tienen el mismo grado.

Un conjunto de códigos LDPC irregulares se define por su distribución de grados de las aristas de los nodos variable $\Lambda = \{\lambda_2, \lambda_3, \dots\}$ y la distribución de grados de las aristas de los nodos de chequeo $\mathrm{P} = \{\rho_2, \rho_3, \dots\}$, donde λ_i representa la fracción de aristas incidentes en los nodos variable con el grado i y ρ_i denota la fracción de aristas incidentes en los nodos de chequeo con el grado j. Otra forma de describir el mismo conjunto de códigos es mediante la representación de las secuencias Λ y P a través de su polinomio generador $\lambda(x) = \sum_i \lambda_i x^{i-1}$ y $\rho(x) = \sum_i \rho_i x^{i-1}$. Ambas notaciones fueron introducidas en [61] y se han utilizado en documentación posterior. Se destaca que el grafo en este caso se caracteriza en términos de fracciones del grado de cada arista y no como anteriormente a través del grado de los nodos.

Al igual que los códigos regulares, como puede verse en [75], el comportamiento en promedio de todos los casos de un conjunto de códigos irregulares se centra entorno de un comportamiento esperado siempre y cuando la longitud de los códigos sea suficientemente larga.

Dada una distribución de grados de un código LDPC y su número de aristas E, es sencillo comprobar que el número de nodos variable es $n = E \sum_i \frac{\lambda_i}{i} = E \int_0^1 \lambda(x) d_x$ y el número de nodos de chequeo es $r = E \sum_i \frac{\rho_i}{i} = E \int_0^1 \rho(x) d_x$. Por lo tanto, la tasa de diseño del código será $R = 1 - \frac{\sum_i \frac{\rho_i}{i}}{\sum_i \frac{\lambda_i}{i}}$.

La búsqueda de una buena familia de códigos irregulares de longitud grande es equivalente a encontrar una buena distribución de grados. La tarea de encontrar una distribución de grados cuyos resultados cumpla las propiedades de una familia de códigos no es trivial.

4.3.3.3 Códigos LDPC: Métodos de decodificación

Atendiendo a la siguiente expresión $n = E\sum_i \frac{\lambda_i}{i}$, se puede comprobar que una vez fijado en la distribución de grados de los nodos variable, el número de aristas del factor de grafo de un código es proporcional a n. Ésta es la principal condición de los códigos LDPC para que la complejidad de decodificación sea lineal con respecto a la longitud del código a través de un número constante de iteraciones. La linealidad de la complejidad se debe a que la decodificación se realiza mediante paso de mensajes a través de las aristas del grafo, por lo tanto la complejidad de una iteración es del orden de E.

Existen gran variedad de algoritmos para el paso de mensajes en códigos LDPC. En este apartado se intentará dar una visión general de algunos de ellos. Se comenzará con el algoritmo suma-producto, incluyendo una interpretación de su decodificación que permitirá dar sentido a otros algoritmos de decodificación.

- **ALGORITMO SP**: Se opta por un caso simplificado en el que los nodos variable pertenecen al alfabeto $\{0,1\}$ y los nodos función son limitaciones para el control de paridad.

- **ALGORITMO MIN-SUM:** En el algoritmo min-sum, la regla de actualización en un nodo variable es la misma que en el algoritmo de SP especificado a continuación, excepto por la salvedad que la actualización de un nodo de chequeo c se simplifica a:

$$m_{c\to v} = \min_{y\epsilon n(c)-\{v\}}\left(|m_{y\to v}|\right)\cdot \prod_{y\epsilon n(c)-\{v\}} sign\left(m_{y\to v}\right)$$

  Comparando esta regla de actualización con la del algoritmo SP cabe destacar que $\tanh^{-1}$ del productorio de $\tanh$ toma aproximadamente el mismo valor que el mínimo del valor del absoluto del producto de los signos. Esta aproximación se hace más precisa a medida de que la magnitud de los mensajes aumenta. Así que en iteraciones posteriores, cuando la magnitud de los mensajes suele ser mayor, el rendimiento de este algoritmo es casi el mismo que el algoritmo SP.

- **ALGORITMO DE DECODIFICACIÓN DE GALLAGER A:** En este algoritmo introducido por Gallager [54], el mensaje sigue un alfabeto binario $\{0,1\}$. La regla de actualización de los nodos de chequeo c siguen la siguiente ecuación:

$$m_{c\to v} = \bigotimes_{y\epsilon n(c)-\{v\}} m_{y\to v}$$

  En un nodo variable v, los mensajes de salida $\mathrm{m_{v\to c}}$ coinciden con los mensajes intrínsecos, a menos que todos los mensajes extrínsecos no sean compatibles con los mensajes intrínsecos. En este caso los mensajes de salida coinciden con los mensajes extrínsecos, es decir:

$$m_{c\to v} = \begin{cases} m_0 \; si \; \exists y \epsilon n(c) - \{v\} : m_{y\to v} = m_0 \\ \overline{m_0} \; en \; otro \; caso \end{cases}$$

Siendo:

- m_0: mensaje intrínseco binario
- $\bar{x}$: complementario binario del valor de x

El algoritmo de Gallager A tiene un peor rendimiento comparado con otros mecanismos de decodificación blanda, pero computacionalmente es mucho menos complejo.

- **ALGORITMO DE DECODIFICACIÓN DE GALLAGER B:** Este algoritmo también se debe a Gallager, y al igual que el algoritmo A, los mensajes toman valores binarios. La única diferencia entre ambos es la regla de actualización del nodo variable. En un nodo variable v los mensajes de salida $m_{c\to v}$ son:

$$m_{c\to v} = \begin{cases} \overline{m_0} \; si \; \exists y_1, y_2, \dots, y_b \epsilon n(c) - \{v\} : m_{y_1\to v} = \dots = m_{y_b\to v} = \overline{m}_0 \\ m_0 \; en \; otro \; caso \end{cases}$$

Siendo:

- b: un entero cuyo rango se encuentra $\left\lfloor \frac{d_v - 1}{2} \right\rfloor < b < d_v$

En este caso, el mensaje de salida de un nodo variable es el mismo que el del mensaje intrínseco, a no ser que los mensajes extrínsecos de b no sean compatibles. El valor de b puede cambiar de una iteración a otra. El valor óptimo de b para un código LDPC regular lo calculó Gallager [54] y corresponde con el menor entero de b que cumpla:

$$\frac{1 - p_0}{p_0} \leq \left[\frac{1 + (1 - 2p)^{d_c - 1}}{1 - (1 - 2p)^{d_c - 1}} \right]^{2b - d_v + 1}$$

Siendo:

- p_0: representa la probabilidad de cruce del canal (tasa de error de los mensajes intrínsecos)
- p: representa la tasa de error de los mensajes extrínsecos.

Atendiendo a diversos estudios [72], se demuestra que el algoritmo B es el mejor algoritmo de paso de mensajes binario para códigos LDPC. Para los códigos LDPC irregulares, también se demuestra que es el óptimo entre todos los algoritmos de paso de mensajes binarios cuando se desconoce el grado de los nodos vecinos.

Tanto en los algoritmo A como B, los mensajes no poseen información del decodificador blando. Por lo tanto, no es una sorpresa que el rendimiento de ambos algoritmos, en comparación con el algoritmo SP, sea muy pobre. Mientras un decodificador blando (por ejemplo, el

decodificador SP) itera hacia la convergencia, la magnitud de los mensajes va creciendo, por lo que la información cualitativa se hace menos útil. A pesar de ello, se utiliza los algoritmos A y B para acelerar el proceso de decodificación debido a que es computacionalmente menos costoso.

4.6 Códigos IRA

4.6.1 Introducción

Desde el descubrimiento de los turbo códigos [71], ha habido varios descubrimientos notables en el campo de códigos con similitudes probabilísticas. En particular, el redescubrimiento de los códigos LDPC, originalmente propuesto en [54], la introducción de códigos LDPC irregulares [95], [96], y la introducción de códigos de repetición acumulada (RA) [76].

En [95], [96], se muestra que los códigos LDPC irregulares son capaces de alcanzar asintóticamente la capacidad de un canal binario de borrado (BEC) bajo el método de decodificación iterativo de paso de mensajes. Aunque el canal BEC es el único canal en el que existen resultados en la actualidad, los códigos LDPC irregulares han sido diseñados para otros canales con entradas binarias (como por ejemplo, el canal binario simétrico (BSC) y el canal con interferencia entre símbolos (ISI) [97]-[99]) y son capaces de alcanzar óptimos rendimientos.

Los primeros intentos para optimizar los códigos LDPC irregulares ([100] para canales BEC y [75] para otros canales) con la técnica de la evolución de densidad (DE), calculan el rendimiento esperado para un conjunto de códigos con similitudes probabilísticas en el límite de la longitud de código bloque infinita. Con el fin de reducir la carga de cálculo del conjunto de optimización basado en DE, se proponen técnicas más rápidas basadas en la aproximación del DE por un sistema dinámico unidimensional recursivo. Estas técnicas son exactas solamente para el canal BEC (para el cual el DE es de una dimensión). Las técnicas más populares propuestas hasta ahora se basan en la aproximación de Gauss (GA) de los mensajes intercambiados en la decodificación de paso de mensajes. Además de la condición de simetría de la densidad de los mensajes, GA implica que la densidad Gaussiana de mensajes puede ser expresada por un único parámetro. Las diversas técnicas difieren en el seguimiento del parámetro y en las funciones de mapeo que definen la dinámica del sistema [82], [91], [101], [102]- [105].

La introducción del concepto de códigos LDPC irregulares motivó otros esquemas como los códigos irregulares de repetición acumulada (IRA) [79] obteniendo similares resultados en la capacidad de canales BEC y en los turbo códigos irregulares [106]. Los códigos IRA son, de hecho, subclases especiales tanto de los códigos LDPC irregulares como los turbo códigos irregulares. En los códigos IRA, figura Fig. 21, una fracción f_i de información de bits es repetida i veces, siendo $i = 2,3, \ldots$. La distribución $\{f_i \geq 0, i = 2,3, \ldots : \sum_{i=2}^{\infty} f_i = 1\}$ se conoce como el perfil de repetición, y se mantiene con un grado de libertad en la optimización del conjunto del código IRA. Después de la etapa de repetición, la secuencia resultante es intercalada y se convierte en la entrada de una maquina recursiva con finitos estados (llamada acumulador), dando como salida un bit para cada símbolo de entrada, haciendo referencia al factor de agrupamiento como parámetro de diseño.

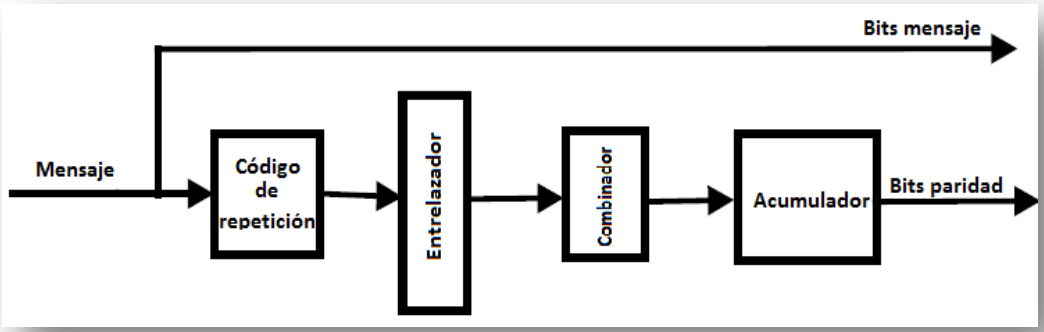


Fig. 21. ***Código IRA sistemático visto como un turbo código***

Los códigos IRA son una opción atractiva debido a que el codificador es extremadamente simple, su rendimiento es bastante competitivo con respecto a turbo códigos y LDPCs, y pueden ser decodificados con un esquema iterativo de baja complejidad.

Se puede observar trabajos sobre el método de diseño de códigos IRA y de cómo elegir de forma óptima el factor de agrupamiento y el perfil de agrupación en [79] y [107]. La maquina recursiva con finitos estados es tan sencilla que da libertad para elegir cualquier número racional entre 0 y 1 como la tasa de codificación. El desarrollo sobre códigos IRA se centrará únicamente en utilizar la misma recursión [79], aunque se podría esperar una mejora en los códigos mediante la inclusión de una máquina de estados finita con un grado de libertad en todo el conjunto de optimización. El método utilizado en [79] para seleccionar el perfil de repetición se basó en la longitud de bloque infinita de GA de la decodificación de paso de mensajes propuesta en [84].

En los siguientes apartados, se presentará el diseño sistemático de un codificador IRA y su correspondiente decodificador: algoritmo de paso de mensajes. Resultados existentes en el análisis del decodificador son resumidos y aplicados en el conjunto de códigos IRA. Esto conduce a un sistema dinámico bidimensional cuyo estado se define en el espacio de distribuciones simétricas, por lo que se deriva en una condición de estabilidad local.

Un código IRA [79] puede ser representado por un grafo de Tanner, figura Fig. 22, siendo éste un grafo bipartito con dos tipos de nodos: N nodos variables (círculos) y M nodos de chequeo (cuadrados). Los nodos variable (VN) pueden ser particionados en $K = N - M$ nodos de información (IN) y M nodos de paridad (PN). El número de nodos de chequeo se calculará mediante la siguiente expresión: $M = (N \sum_i f_i)/a$. Las entradas IN son de diverso grado. Cada nodo de información está conectado a un número de nodos de chequeo: f_i que denota la fracción de entradas IN con grado i, con $\sum_i f_i = 1$. Cada nodo de chequeo está conectado a un número a de nodos de información. Las correspondientes $a * M$ conexiones entre CN e IN son "permutaciones arbitrarias" (Π_1). Los nodos de chequeo están además conectados a los nodos de paridad siguiendo un patrón fijo en zigzag (Π_2). El código puede ser descrito por la distribución de grados $f = (f_2, f_3, \dots, f_j)$ y un entero positivo a.

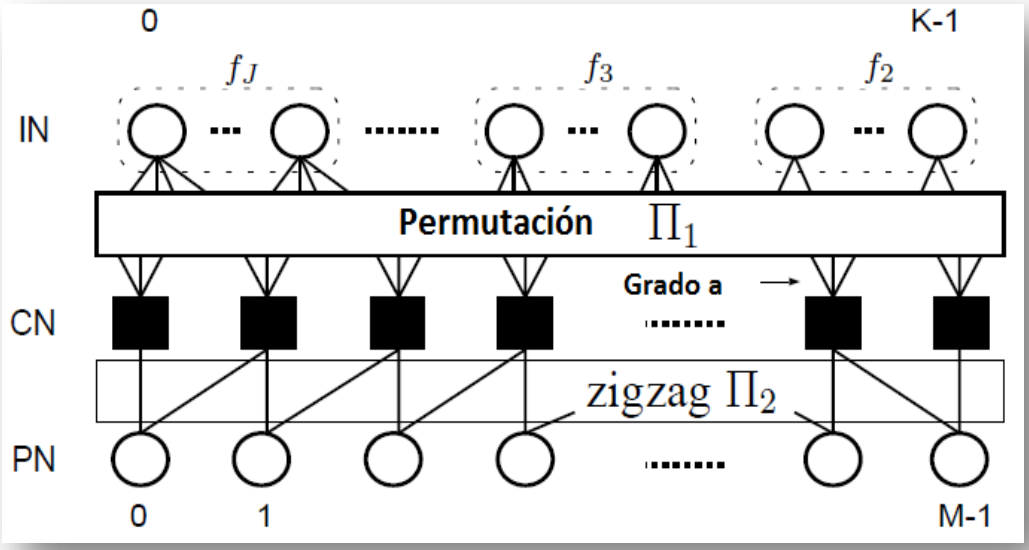


Fig. 22. ***Gráfico de Tanner para códigos IRA***

Si la "permutación arbitraria" en la figura Fig. 22 es fija, el gráfico de Tanner representa un código binario lineal con N bits de información $(u_1, u_2, \dots, u_N)$ y M bits de paridad $(x_1, x_2, \dots, x_M)$. Cada uno de los bits de información se asocia con uno de los nodos de información, y cada uno de los bits de paridad se asocia con uno de los nodos de paridad. El valor de un bit de paridad se determina únicamente por la condición de que la suma en modulo 2 de valores de los nodos variable conectados a cada uno de los nodos de chequeo sea cero. Para ver esto, convencionalmente se establece que $x_0 = 0$. Entonces, si los valores de bits en las $a * M$ aristas que salen del bloque de permutación son $(v_1, v_2, \dots, v_{a*M})$, se puede obtener la siguiente fórmula recursiva:

$$x_j = x_{j-1} + \sum_{i=1}^{a} v_{(j-1)a+i} \ \ siendo\ j = 1,2, \dots, M$$

De hecho, esta fórmula corresponde con el algoritmo de codificación, y así que si a es un entero positivo fijo y $n \to \infty$, la complejidad de codificación es $O(n)$.

Existen dos versiones del código IRA en la figura Fig. 22: la versión sistemática y no sistemática. La versión no sistemática corresponde a un código (M, N), en el que la palabra código correspondiente a los bits de información $(u_1, u_2, \dots, u_N)$ es $(x_1, x_2, \dots, x_M)$. La versión sistemática corresponde a un código $(N + M, N)$, en el que la palabra código es $(u_1, u_2, \dots, u_N; x_1, x_2, \dots, x_M)$.

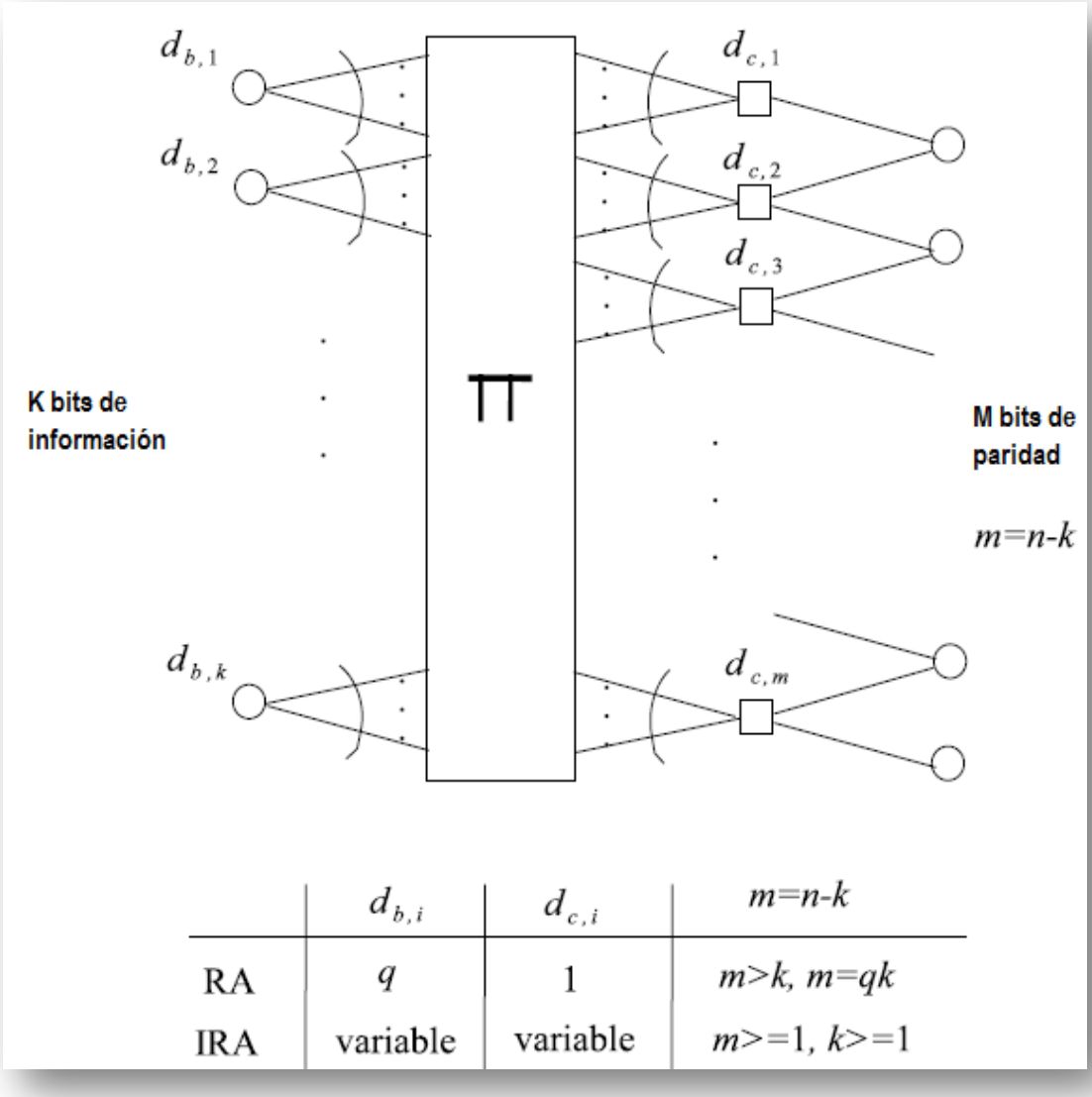
La tasa de un código no sistemático se ve fácilmente que es $R_{NSIS} = {}^{a}/_{\sum_i i f_i}$, mientras que para un código sistemático su tasa es $R_{SIS} = {}^{a}/_{a + \sum_i i f_i}$.

Por ejemplo, los códigos originales AR son códigos no sistemáticos con $a = 1$ y con exactamente un valor de f_i igual a 1, de tal forma que $f_q = 1$ y el resto 0, en cuyo caso la

tasa de este código quedaría como $R = {}^{1}/_{q}$. (Sin embargo, el desarrollo que se llevará a cabo a lo largo de este trabajo se centra sobre los códigos sistemáticos IRA).

4.6.2 Construcción matriz de paridad para códigos IRA

Los códigos sistemáticos IRA son una generalización de los códigos sistemáticos RA en el que la tasa de repetición puede diferir en K bits de información. Además, de acuerdo con la figura Fig. 23, en [79] y su asociada discusión, $M > K$ y los nodos de chequeo están todos conectados a exactamente $a \geq 1$ nodos de información. En el contexto de la figura Fig. 23, $d_{b,i}$ varía con i y $d_{c,i}$ varía con a para cualquier valor de a fijo siempre y cuando $a \geq 1$. Sin embargo, como han señalado otros investigadores y H. Jin [108], las limitaciones de $M > K$ y $d_{c,i} = a$ son menos restrictivas en [79].



	$d_{b,i}$	$d_{c,i}$	m=n-k
RA	q	1	$m>k$, $m=qk$
IRA	variable	variable	$m>=1$, $k>=1$

Fig. 23. ***Gráfico de Tanner para una estructura de códigos RA e IRA*** **[109]**

La matriz de chequeo de paridad para códigos sistemas RA e IRA tiene la siguiente forma: $H = [H_1\ H_2]$, donde H_2 es una matriz cadrada doblemente diagonal:

$$H_2 = \begin{bmatrix} 1 & & & & \\ 1 & 1 & & & \\ & & \ldots & & \\ & & 1 & 1 & \\ & & & 1 & 1 \end{bmatrix}$$

Para códigos RA, H_1 es una matriz regular, la cual tiene columnas con peso q y filas con peso 1. Para los códigos IRA, H_1 tiene columnas con peso $\{d_{b,i}\}$ y filas con peso $\{d_{c,i}\}$. El codificador mostrado en la figura Fig. 24 es un diseño útil para un análisis conjunto y a través del cual se puede obtener H_1 a través de $\Pi^T B^T$, donde Π es la matriz de permutación.

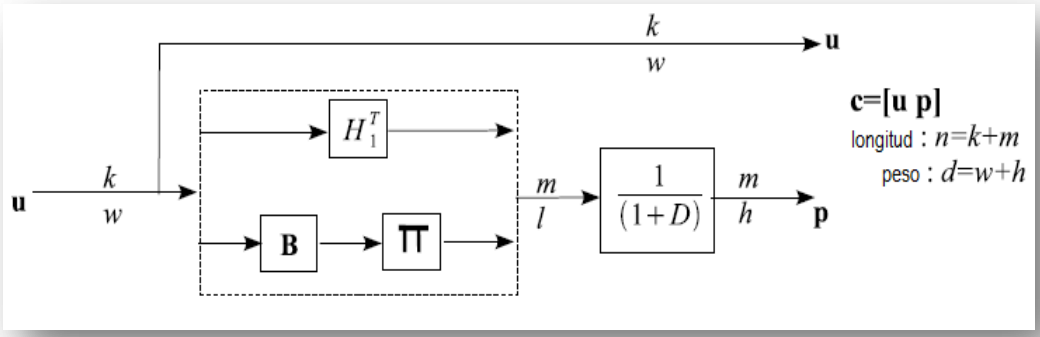


Fig. 24. ***Codificador sistemático para una estructura de códigos RA e IRA*** **[109]**

4.6.3 Evolución de densidad: distribución de grados

En los códigos IRA se hace distinción entre aristas, conexiones entre nodos de información y nodos de chequeo, y aristas que conectan los nodos de paridad con los de chequeo.

La distribución de grados de las aristas se denota como $(\lambda(x), \rho(x))$, donde λ_i corresponde con la fracción de aristas que procede de un nodo de información de grado i y ρ_i es la fracción de aristas que proviene de los nodos de chequeo de grado i (donde el grado de un nodo de chequeo es el número de aristas a las cuales están conectados).

Para la distribución de grados en los nodos, es útil definir las siguientes funciones:

$$v(x) = v_2 x + \cdots + v_i x^{i-1} + \cdots + v_{v_{máx}} x^{v_{máx}-1}$$
$$h(x) = h_2 x + \cdots + h_i x^{i-1} + \cdots + h_{v_{máx}} x^{v_{máx}-1}$$

En este caso, v_i se considera la fracción de nodos de información con grado i y h_i la fracción de nodos de chequeo con grado i.

Un método para determinar el rendimiento, en un canal AWGN, de un conjunto de códigos con una determinada distribución de grados, llamado DE, se propuso para códigos LDPC [75] y se modificó para los códigos IRA en [117]. Para un canal AWGN, la DE proporciona, como resultado, un valor alto de varianza de ruido, y por consiguiente, una relación señal a ruido baja.

El esquema de DE para los códigos IRA, se basa en representaciones del grafo de Tanner como el que se puede observar en la figura Fig. 22.

Dada una longitud N y una tasa R, el código IRA, con una distribución de grados $\lambda(x)$, tendrá:

$$v_i = N \frac{\lambda_i / i}{\sum_j \lambda_j / j}; \; i > 2$$

$$h_i = \frac{\rho_i / i}{\sum_j \rho_j / j}$$

Para el caso $i = 2$, los nodos de información de grado 2 se representan como:

$$v_2 = N \frac{\lambda_2 / 2}{\sum_j \lambda_j / j} - (1 - R) * N$$

Una distribución de grados óptima para códigos IRA, para una tasa dada, se puede conseguir por medio de DE [75]. A pesar de ello, en el caso de estudio que se nos plantea y para reducir la complejidad de diseño, se partirá únicamente de la restricción de que la matriz de permutación, al menos, contenga M nodos de grado 2.

4.7 Paso de mensajes

4.7.1 Introducción

Sean $x_1, x_2, \dots, x_n$ una colección de variables, en las que para cada i, cada x_i toma valores en un dominio finito, el alfabeto $\mathcal{A}_i$. Sea $g(x_1, x_2, \dots, x_n)$ una función cuyo dominio es $S = \mathcal{A}_1 \times \mathcal{A}_2 \times \dots \times \mathcal{A}_n$ y codominio $\mathcal{R}$. El dominio S de g se denomina conjunto de configuración para una colección de variables dadas, y cada elemento de S es una configuración particular de las variables, por ejemplo, una asignación de un valor a cada variable. El codominio $\mathcal{R}$ se asumirá inicialmente que es un conjunto de números reales [55].

Si se asume que el sumatorio en $\mathcal{R}$ está bien definido, entonces las asociaciones con cada función $g(x_1, x_2, \dots, x_n)$, serán n funciones marginales $g(x_i)$. Para cada $a \epsilon \mathcal{A}_i$, el valor de $g_i(a)$ se obtiene a través del sumatorio del valor de $g(x_1, x_2, \dots, x_n)$ sobre todas las configuraciones de las variables que cumplan $x_i = a$. Debido a que este tipo de suma es muy importante a lo largo de este trabajo, se planteará a continuación un ejemplo sencillo de nomenclatura. Si h es una función de tres variables x_1, x_2, x_3, entonces el sumatorio para x_2 se denotaria de la siguiente manera:

$$\sum_{\sim x_2} h(x_1, x_2, x_3) = \sum_{x_1 \epsilon \mathcal{A}_1} \sum_{x_3 \epsilon \mathcal{A}_3} h(x_1, x_2, x_3)$$

En esta notación se entiende que:

$$g_{i(x_i):=} \sum_{\sim x_i} g(x_1, \dots, x_n)$$

En general, ha despertado gran interés el desarrollo de procedimientos eficaces para operar funciones marginales que:

a. Explotan la utilidad de los factores de las funciones marginales, utilizando la ley distributiva para simplificar las sumas.
b. Reutiliza sumas parciales.

Supóngase que $g(x_1, x_2, \dots, x_n)$ se factoriza en un producto de varias funciones locales, teniendo cada una un subconjunto de $\{x_1, x_2, \dots, x_n\}$ como argumentos, por ejemplo, supóngase que:

$$g(x_1, x_2, \dots, x_n) = \prod_{j \epsilon J} f_j(X_j)$$

donde J es un conjunto de indices discretos, X_j es un subconjunto de $\{x_1, x_2, \dots, x_n\}$, y $f_j(X_j)$ es una función que tiene elementos de X_j como argumento.

Como ya hemos visto, un FG es una representación gráfica estándar bipartita de una relación matemática, que en este caso es un argumento de relaciones entre variables y funciones locales.

Un ejemplo clásico de la representación del FG puede ser el siguiente, sea $g(x_1, x_2, x_3, x_4, x_5)$ una función de 5 variables, y supóngase que g puede expresarse como un producto de 5 factores, por lo que $J = \{A, B, C, D, E\}$, $X_A = \{x_1\}, X_B\{x_2\}, X_C = \{x_1, x_2, x_3\}, X_D = \{x_3, x_4\}$ y $X_E = \{x_5\}$.

$$g(x_1, x_2, x_3, x_4, x_5) = f_A(x_1) f_B(x_2) f_C(x_1, x_2, x_3) f_D(x_3, x_4) f_E(x_5)$$

La representación gráfica de la expresión anterior puede observarse en la siguiente figura Fig. 25.

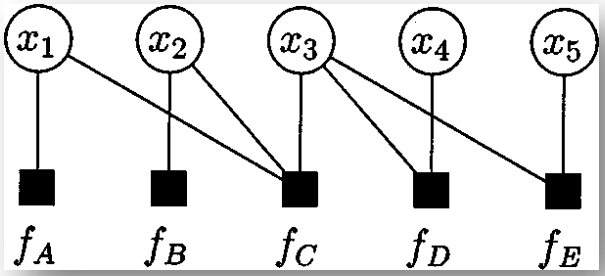


Fig. 25. ***FG para el producto*** $f_A(x_1) f_B(x_2) f_C(x_1, x_2, x_3) f_D(x_3, x_4) f_E(x_5)$ **[87]**

4.7.2 Expresiones en forma de árbol

En muchas situaciones (por ejemplo, cuando $g(x_1, \ldots, x_5)$ representa una función de probabilidad conjunta), es interesante calcular las funciones marginales $g_i(x_i)$. Se puede obtener una expresión para cada función marginal utilizando la expresión matemática anterior y aplicando la ley distributiva.

Para ilustrarlo, se escribe $g_1(x_1)$ del ejemplo anterior como:

$$g_1(x_1) = f_A(x_1)\left(\sum_{x_2} f_B(x_2)\left(\sum_{x_3} f_C(x_1, x_2, x_3)\left(\sum_{x_4} f_D(x_3, x_4)\right)\left(\sum_{x_5} f_E(x_3, x_5)\right)\right)\right)$$

O bien, utilizando una notación sumatorio:

$$g_1(x_1) = f_A(x_1) \times \sum_{\sim\{x_1\}} \left(f_B(x_2) f_C(x_1, x_2, x_3) \times \left(\sum_{\sim\{x_3\}} f_D(x_3, x_4) \right) \times \left(\sum_{\sim\{x_3\}} f_E(x_3, x_5) \right) \right)$$

Del mismo modo, se puede expresar $g_3(x_3)$ de la siguiente manera:

$$g_3(x_3) = \left(\sum_{\sim\{x_3\}} f_A(x_1) f_B(x_2) f_C(x_1, x_2, x_3) \right) \times \left(\sum_{\sim\{x_3\}} f_D(x_3, x_4) \right) \times \left(\sum_{\sim\{x_3\}} f_E(x_3, x_5) \right)$$

Las expresiones aritméticas como las dos últimas, se representan por los llamados "árboles de expresión" [110, Sec. 8.3], en los que los vértices internos (es decir, los vértices con descendientes) representan a los operadores aritméticos (por ejemplo, suma, multiplicación…) y vértices de tipo hoja (es decir, los vértices sin descendientes) representan las variables o constantes. Cuando los operadores en un árbol de expresión están restringidos a que sean completamente simétricos en sus operandos (por ejemplo, multiplicación y suma), no es necesario ordenar los vértices para evitar la ambigüedad al interpretar la expresión representada por el árbol.

Se considerará a lo largo del estudio, que los vértices de tipo hoja representan a las funciones, no sólo variables y constantes. Las sumas y productos de los árboles de expresión combinan sus operandos de forma en que las funciones se suman y multiplican. Por ejemplo, la figura Fig. 26 a) representa de manera inequívoca la expresión de notación sumatorio de $g_1(x_1)$, y la figura Fig. 27 a) la expresión de notación sumatorio de $g_3(x_3)$. Los operandos mostrados en estas figuras son las funciones productos y sumatorio, que tienen varias funciones locales como argumentos.

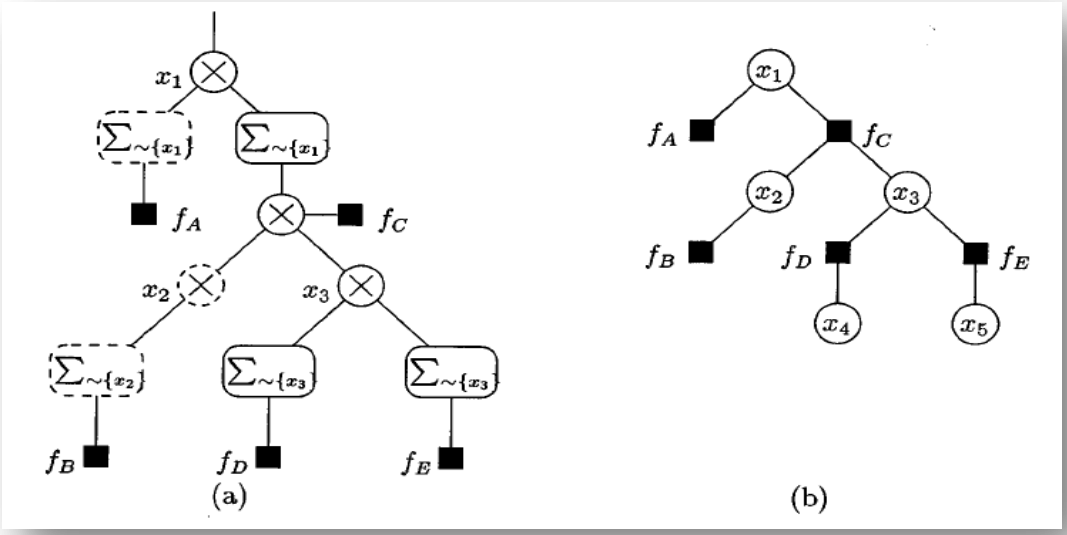


Fig. 26. ***a) Representación en árbol del sumatorio de $g_1(x_1)$ b) FG de la Fig. 13 como vértice raíz x_1*** **[87]**

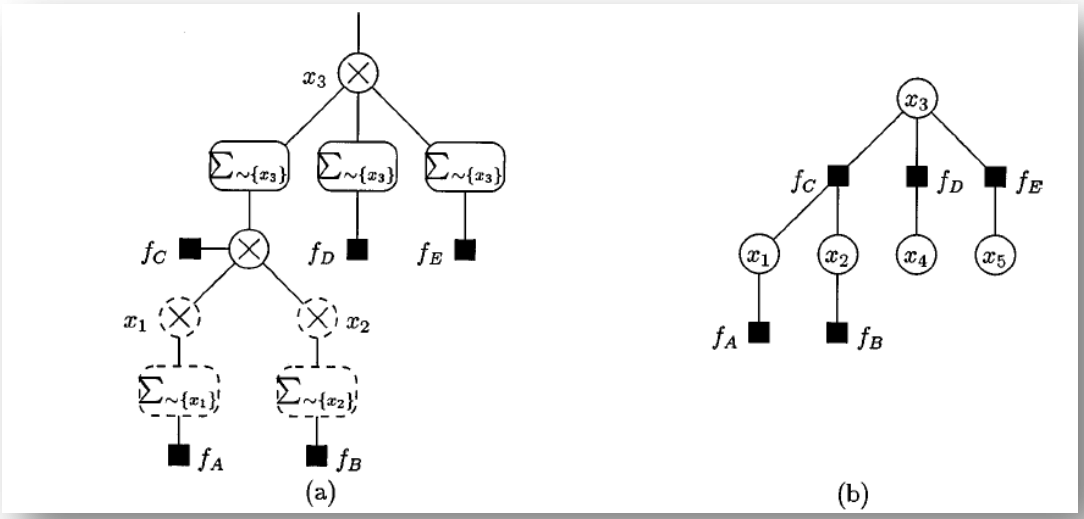


Fig. 27. ***a) Representación en árbol del sumatorio de $g_3(x_3)$ b) FG de la Fig. 14 como vértice raíz x_3*** **[87]**

En las figuras Fig. 26 y Fig. 27 b), se representa el repintado de la figura Fig. 25 teniendo en cuenta en este caso que la raíz del FG un vértice diferente. Esto es posible porque la función global definida en la primera expresión matemática de $g_1(x_1)$ fue elegida deliberadamente de tal forma que el FG correspondiera con la estructura de un árbol. Comparando el FG con sus correspondientes representaciones en árbol de las expresiones de la función marginal, es fácil observar su correspondencia.

Formalmente, para convertir una función $g(x_1, \dots, x_n)$ representada por su FG en un árbol de expresión $g_i(x_i)$, es necesario dibujar el FG como un árbol con un vértice raíz x_i. Cada nodo v en el FG tiene definido claramente un nodo padre, conocido como nodo

vecino a través del cual sólo existe un único camino desde v a x_i. Se reemplaza cada nodo variable en el FG por un operador de producto. Se reemplaza cada nodo factor en el FG por un operador multiplicador por f, y entre un nodo factor f y su nodo padre x, se inserta el operador sumatorio $\sum_{\sim\{x\}}$. Estas transformaciones locales quedan ilustradas en la figura Fig. 28 a) por un nodo variable, en la figura Fig. 28 b) por un nodo factor f con un nodo padre x. Los productos que tienen uno o ningún operando actúan como operadores identidad, o bien pueden omitirse en el árbol si son nodos hoja. Un operador sumatorio aplicado a una función con un único argumento se considera también como operación trivial, por lo que también será omitida. Aplicando esta transformación al árbol de la figura Fig. 26 y Fig. 27 a), se obtiene el árbol de expresiones de la figura Fig. 26 y Fig. 27 b). Las operaciones triviales se indican con líneas discontinuas en estos dibujos.

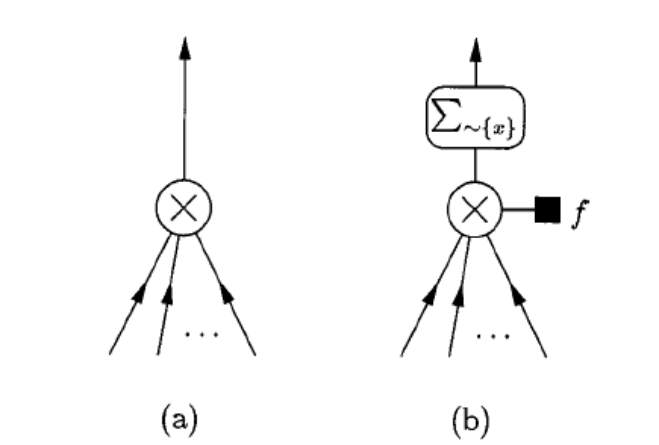


Fig. 28. ***Sustituciones locales que transforman el FG sin ciclos en un árbol de expresión para una función marginal en a) un nodo variable b) un nodo factor*** **[87]**

4.7.3 Cálculo de una función individual marginal

Cada árbol de expresión representa un algoritmo para el cómputo de cada expresión. Se podría describir el algoritmo como un procedimiento recursivo de "arriba hacia abajo" que se inicia en el vértice raíz y evalúa cada subárbol que desciende de la raíz, combinando los resultados como dictamina el operador en la raíz. De manera equivalente, se prefiere para describir el algoritmo como un procedimiento de "abajo hacia arriba" que comienza en las hojas del árbol con cada operador vértice combinando sus operandos y enviando el resultado como un operando a su padre.

En lugar de trabajar con el árbol de expresión, es más sencillo y directo describir este tipo de algoritmos de marginalización en términos de correspondencia con el FG. Para entender mejor este tipo de algoritmos, ayuda imaginarse que hay un procesador asociado a cada vértice del FG, y que los bordes de dicho grafo representan canales por los cuales estos procesadores pueden comunicarse. Los mensajes enviados entre procesadores son simplemente descripciones apropiadas de funciones marginales.

Por consiguiente, se podrá ahora introducir el concepto de algoritmo de paso de mensajes, que calcula, para un único valor de i, la función marginal $g_i(x_i)$ en un grafo libre de ciclos, donde x_i se considera el vértice raíz.

El cálculo comienza en las hojas del FG. Cada nodo variable hoja envía un mensaje de función identidad a su nodo padre, y cada nodo factor hoja f envía una descripción de f a su padre. Cada vértice espera a los mensajes de todos sus hijos antes de calcular el mensaje que se enviará a su padre. Este cálculo se realiza de acuerdo a la transformación que se muestra en la figura Fig. 28, es decir, un nodo variable simplemente envía el producto de los mensajes recibidos de sus hijos, mientras que un nodo factor f con su nodo padre x forma el producto de f con los mensajes recibidos de sus hijos, y después opera el resultado con el operador sumatorio $\sum_{\sim\{x\}}$. La utilización de "producto de mensajes" hace referencia al correspondiente producto de funciones. Si los mensajes son parametrizaciones de funciones, entonces el mensaje resultado es la parametrización de la función producto, y no necesariamente el producto numérico de los mensajes. Del mismo modo, el operador sumatorio se aplica a las funciones, y no necesariamente a los mensajes mismos.

Las operaciones terminan en el nodo raíz x_i, donde la función marginal $g_i(x_i)$ se obtiene como producto de todos los mensajes recibidos en x_i.

Es importante señalar, que un mensaje enviado por la arista $\{x, f\}$, ya sea desde el nodo variable x al nodo factor f, o viceversa, es una función con un único argumento cuya variable está asociada a la arista dada. Esto se deduce ya que, en cada nodo de factores, las operaciones sumatorio se realizan siempre por la variable asociada con la arista sobre la que se envía el mensaje. Asimismo, en un nodo variable, todos los mensajes son funciones de esa variable, y también lo es cualquier producto de estos mensajes.

El mensaje transmitido en una arista durante las operaciones del algoritmo paso de mensajes puede ser interpretado de la siguiente manera. Si $e = \{x, f\}$ es una arista en el árbol, donde x es un nodo variable y f es un nodo factor, se demuestra que el mensaje enviado en e durante las operaciones del algoritmo SP es simplemente un sumatorio para x del producto de las funciones locales que descienden del vértice que origina el mensaje.

4.7.4 Cálculo de todas las funciones marginales

En muchas circunstancias, se puede estar interesado en el cómputo de $g_i(x_i)$ para más de un valor de i. Tal cálculo podría lograrse aplicando individualmente el algoritmo de paso de mensajes para cada valor deseado de i, pero este enfoque es poco probable que sea eficiente, ya que muchos subcómputos llevados a cabo para diferentes valores de i tendrán el mismo resultado. Calculando $g_i(x_i)$ para todo i simultáneamente puede ser eficiente ya que se logra superponer en un solo FG todos los posibles casos del algoritmo de paso de mensajes. Ningún vértice en particular se toma como un vértice raíz, así que no hay una relación fija padre/hijo entre los vértices vecinos. En su lugar cada vecino w de cualquier vértice dado v se considera como padre de v. El mensaje transmitido desde v a w se calcula con el algoritmo de paso de mensajes, por ejemplo, como si w fuese de hecho el padre de v y todos los demás vecinos de v fuesen sus hijos.

Al igual que en el algoritmo de paso de mensajes, el mensaje transmitido es inicializado por las hojas. Cada vértice v permanece inactivo hasta que los mensajes gan llegado todos menos aquel enviado por la arista incidente en v. De la misma forma que en el algoritmo de paso de mensajes, una vez que estos mensajes han llegado, v es capaz de calcular un mensaje para enviarlo por la arista restante a su vecino (temporalmente considerado como el padre), como puede verse en la figura Fig. 28. Se considera que este temporal padre es un vértice w. Después de enviar un mensaje a w, el vértice v retorna al estado de reposo, esperando al mensaje de vuelta que llega desde w. Una vez que este mensaje ha llegado, el vértice es capaz de calcular y enviar mensajes a cada uno de sus vecinos (distintos de w), cada uno siendo considerado, a u vez, como un padre. El algoritmo termina cuando dos mensajes son enviados sobre cada arista, uno en cada dirección. En el nodo variable, el producto de todos los mensajes entrantes es la función marginal $g_i(x_i)$, como en el algoritmo de paso de mensajes. Este algoritmo opera mediante el cálculo de varias sumas y productos, por lo que se refiere a él como algoritmo SP.

4.7.5 Algoritmo decodificación SP

El algoritmo SP opera de acuerdo con esta simple regla:

Regla de actualización SP: *El mensaje enviado desde un nodo v a una arista e es el producto de la función local en v (o la función unidad si v es un nodo variable) con todos los mensajes recibidos en v desde otras aristas que e, sumatorio de la variable asociada con e.*

Por consiguiente, $\mu_{x\to f}(x)$ denota el mensaje enviado desde el nodo x al nodo f en la operación del algoritmo SP, $\mu_{f\to x}(x)$ denota el mensaje enviado desde el nodo f al nodo x. También, $n(v)$ denota el conjunto de vecinos asociados al nodo v en el FG. Después, como se muestra en la figura Fig. 29, los cálculos de los mensajes realizados por el algoritmo suma-producto pueden ser expresados como sigue:

Variable a función local:

$$\mu_{x\to f}(x) = \prod_{h\epsilon n(x)\setminus\{f\}} \mu_{h\to x}(x)$$

Función local a variable:

$$\mu_{f\to x}(x) = \sum_{\sim\{x\}} f(X) \prod_{y\epsilon n(f)\setminus\{x\}} \mu_{y\to f}(y)$$

donde $X = n(f)$ es el conjunto de argumentos de la función f.

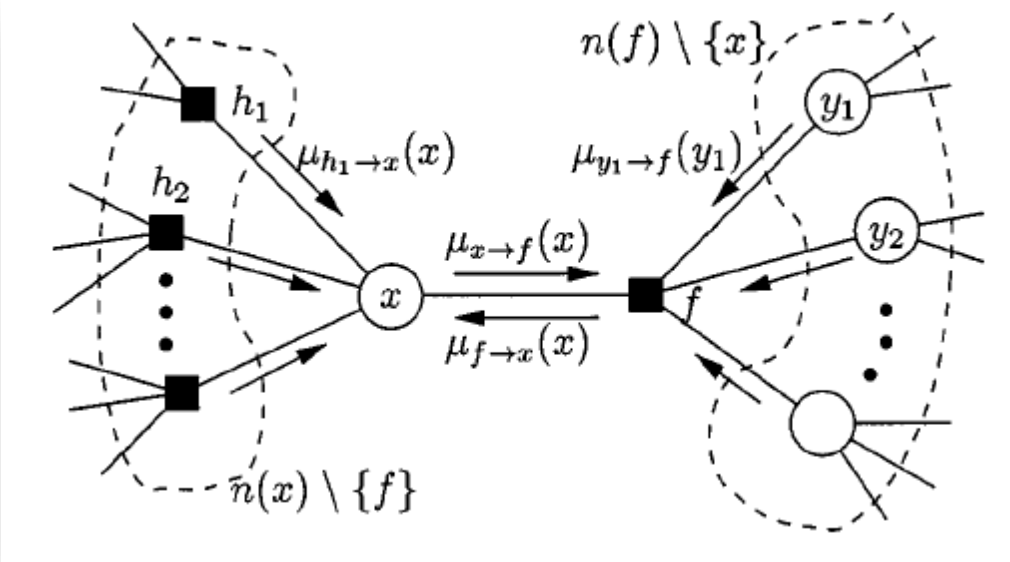


Fig. 29. ***Fragmento de un FG, muestra las reglas de actualización del algoritmo SP*** **[87]**

La regla de actualización en un nodo variable x, particularmente toma de forma sencilla la expresión dada por "Variable a función local" ya que no incluye una función local, y el sumatorio por x del producto de funciones de x es simplemente una multiplicación. Por otro lado, la regla de actualización en un nodo función local dada por "Función local a variable", en general, envuelve multiplicaciones de funciones no triviales, seguidas por la aplicación del operador sumatorio.

También se puede observar que los nodos variables de grado dos no realizan operaciones: un mensaje que llega por una arista (entrante) es simplemente transferido a otra arista (saliente).

Se quiere definir la naturaleza de los mensajes y las reglas simplificadas de actualización de los mismos dentro de una nomenclatura implementable. Por ello, se definirá para los nodos variable un alfabeto binario, un mensaje $\mu(x)$ será una función del nodo variable adyacente x, que podrá tener únicamente dos valores $\mu(0)$ y $\mu(1)$. Cuando los mensajes son funciones de probabilidad se produce que $\mu(0) + \mu(1) = 1$, entonces enviar $\mu(0)$ o $\mu(1)$ es equivalente a enviar la función $\mu(x)$. Equivalentemente, se puede enviar una combinación de $\mu(0)$ y $\mu(1)$, como por ejemplo, $\mu(1) - \mu(0)$, $\frac{\mu(1)}{\mu(0)}$ o $\log\frac{\mu(1)}{\mu(0)}$. La última cantidad $\log\frac{\mu(1)}{\mu(0)}$, o lo que es lo mismo la función de probabilidad $\log\frac{P(1)}{P(0)}$, se conoce con el nombre de LLR. El término LLR es el más extendido para el paso de mensajes debido a que sus reglas de actualización son simples y computacionalmente fáciles de implementar, además permite representar los valores de probabilidad que se encuentran muy próximos a cero o cercanos a 1 sin producir errores en la precisión.

En adelante, se cambiará la nomenclatura para los mensajes (μ) por m para distinguir entre las reglas de paso de mensajes generales frente a las reglas de paso de mensajes de tipo LLR. En el caso de estas últimas reglas hay que tener en cuenta que un mensaje ya no es una función, sino que pertenece al cuerpo de los números reales.

La regla de actualización de un nodo de chequeo de paridad c sigue la siguiente expresión:

$$m_{c\to v} = 2\tanh^{-1}\left(\prod_{y\epsilon n(c)-\{v\}} \tanh\left(\frac{m_{y\to v}}{2}\right)\right)$$

Siendo:

- $m_{a\to b}$: representa el mensaje tipo LLR enviado desde el nodo a al nodo b.
- $n(a)$: representa el conjunto de vecinos del nodo a en el grafo.

La regla de actualización de un nodo variable v, en este caso sigue la siguiente expresión:

$$m_{v\to c} = m_0(v) + \sum_{h\epsilon n(v)-\{c\}} m_{h\to v}$$

Siendo:

- $m_0(v)$ es la verosimilitud logarítmica del mensaje asociado a v. Si v no es un nodo de palabra código, este término no aparecerá. Mientras que si v es un nodo de chequeo la fórmula [111] que le corresponde será:

$$\tanh\frac{m(v\to c)}{2} = \prod_{h\neq c} tanh\frac{m(h\to v)}{2}$$

Cuando un mensaje de tipo LLR es positivo significa que $p(1) > p(0)$ y mientras la magnitud de este mensaje aumenta, el mensaje comienza a ser más fiable. En el algoritmo de SP (y otros algoritmos de paso de mensaje), los mensajes que un nodo variable envía a sus nodos de chequeo vecinos representan una presunción de sus valores conjuntos con una medida de fiabilidad. Del mismo modo, el mensaje que un nodo de control envía a un nodo variable vecino es una presunción sobre el valor conjunto del nodo variable con una cierta fiabilidad.

En un decodificador SP, un nodo variable recibe todas las presunciones de los nodos de chequeo vecinos que tiene sobre ellos. Los nodos variable procesan estos mensajes (en este caso es una simple suma) y envía los presunciones actualizadas sobre sí mismo de vuelta a los nodos de chequeo. Se puede entender que la fiabilidad de los mensajes de un nodo variable aumenta a medida que recibe una serie de presunciones acerca de si mismo. Este hecho es muy similar a los códigos de repetición, que reciben múltiples presunciones durante un solo bit procedente del canal y por lo tanto, es capaz de hacer una toma de decisiones más fiable.

La principal tarea de un nodo de chequeo es obligar a sus vecinos a satisfacer una paridad par. Así, para un nodo variable vecino v, el nodo de chequeo puede procesar las presunciones de otros nodos variables vecinos sobre sí mismo y enviar un mensaje a v donde se indica la presunción de este nodo de chequeo sobre v. El signo de este mensaje

es elegido para forzar una comprobación de paridad par y su magnitud depende de la fiabilidad de los mensajes entrantes.

Por lo tanto, y de forma similar a un nodo variable, un mensaje de salida de un nodo de control se produce por el procesamiento de todas las aristas entrantes exceptuando a la que recibe el mensaje de salida. Sin embargo, a diferencia de un nodo variable, un nodo de control recibe la presunción de todos los nodos variable vecinos sobre sus propios valores. Como resultado, la fiabilidad de los mensajes de salida será incluso menor que la de los mensajes de entrada. En otras palabras, la fiabilidad de los mensajes disminuye en los nodos de control.

Así, en pocas palabras, en un nodo de chequeo se fuerza a que los nodos variable vecinos satisfagan un chequeo de paridad par a costa de perder fiabilidad en los mensajes, pero en un nodo variable se consigue reforzar esa fiabilidad. Este proceso se repite iterativamente para eliminar los errores introducidos por el canal.

4.7.6 Un ejemplo detallado

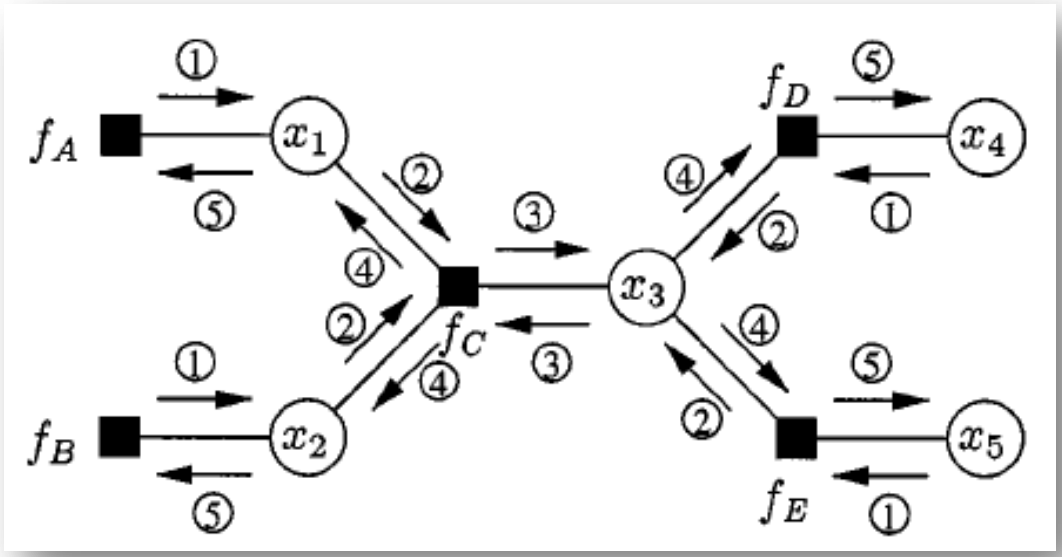


Fig. 30. ***Mensajes generados en cada paso del algoritmo SP***

La figura Fig. 30 muestra el flujo de mensajes que se generan por la aplicación del algoritmo suma-producto al FG de la figura Fig. 25. Los mensajes pueden ser generados en cinco pasos, como se indica con círculos en la figura Fig. 30. Aplicando más detalle, los mensajes se generan de la siguiente manera.

Paso 1:

$$\mu_{f_A \to x_1}(x_1) = \sum_{\sim\{x_1\}} f_A(x_1) = f_A(x_1)$$
$$\mu_{f_B \to x_2}(x_2) = \sum_{\sim\{x_2\}} f_B(x_2) = f_B(x_2)$$
$$\mu_{x_4 \to f_D}(x_4) = 1$$
$$\mu_{x_5 \to f_E}(x_5) = 1$$

Paso 2:

$$\mu_{x_1 \to f_C}(x_1) = \mu_{f_A \to x_1}(x_1)$$
$$\mu_{x_2 \to f_C}(x_2) = \mu_{f_B \to x_2}(x_2)$$
$$\mu_{f_D \to x_3}(x_3) = \sum_{\sim\{x_3\}} \mu_{x_4 \to f_D}(x_4) f_D(x_3, x_4)$$
$$\mu_{f_E \to x_3}(x_3) = \sum_{\sim\{x_3\}} \mu_{x_5 \to f_E}(x_5) f_E(x_3, x_5)$$

Paso 3:

$$\mu_{f_C \to x_3}(x_3) = \sum_{\sim\{x_3\}} \mu_{x_1 \to f_C}(x_1) \mu_{x_2 \to f_C}(x_2) f_C(x_1, x_2, x_3)$$
$$\mu_{x_3 \to f_C}(x_3) = \mu_{f_D \to x_3}(x_3) \mu_{f_E \to x_3}(x_3)$$

Paso 4:

$$\mu_{f_C \to x_1}(x_1) = \sum_{\sim\{x_1\}} \mu_{x_3 \to f_C}(x_3) \mu_{x_2 \to f_C}(x_2) f_C(x_1, x_2, x_3)$$
$$\mu_{f_C \to x_2}(x_2) = \sum_{\sim\{x_2\}} \mu_{x_3 \to f_C}(x_3) \mu_{x_1 \to f_C}(x_1) f_C(x_1, x_2, x_3)$$
$$\mu_{x_3 \to f_D}(x_3) = \mu_{f_C \to x_3}(x_3) \mu_{f_E \to x_3}(x_3)$$

$$\mu_{x_3 \to f_E}(x_3) = \mu_{f_C \to x_3}(x_3) \mu_{f_D \to x_3}(x_3)$$

Paso 5:

$$\mu_{x_1 \to f_A}(x_1) = \mu_{f_C \to x_1}(x_1)$$
$$\mu_{x_2 \to f_B}(x_2) = \mu_{f_C \to x_2}(x_2)$$
$$\mu_{f_D \to x_4}(x_4) = \sum_{\sim\{x_4\}} \mu_{x_3 \to f_D}(x_3) f_D(x_3, x_4)$$
$$\mu_{f_E \to x_5}(x_5) = \sum_{\sim\{x_5\}} \mu_{x_3 \to f_E}(x_3) f_E(x_3, x_5)$$

Final:

$$g_1(x_1) = \mu_{f_A \to x_1}(x_1) \mu_{f_C \to x_1}(x_1)$$
$$g_2(x_2) = \mu_{f_B \to x_2}(x_2) \mu_{f_C \to x_2}(x_2)$$
$$g_3(x_3) = \mu_{f_C \to x_3}(x_3) \mu_{f_D \to x_3}(x_3) \mu_{f_E \to x_3}(x_3)$$
$$g_4(x_4) = \mu_{f_D \to x_4}(x_4)$$
$$g_5(x_5) = \mu_{f_E \to x_5}(x_5)$$

En el paso de finalización, se calcula $g_i(x_i)$ como el producto de todos los mensajes dirigidos hacia x_i. Equivalentemente, puesto que un mensaje enviado hacia una arista dada es equivalente al producto de todos menos uno de todos estos mensajes, se puede calcular $g_i(x_i)$ como el producto de dos mensajes que son transmitidos (en direcciones

opuestas) a través de cualquier arista incidente en x_i. Así, por ejemplo, $g_3(x_3)$ se puede calcular de tres formas como se muestra a continuación:

$$g_3(x_3) = \mu_{f_C \to x_3}(x_3)\mu_{x_3 \to f_C}(x_3) = \mu_{f_D \to x_3}(x_3)\mu_{x_3 \to D}(x_3) = \mu_{f_E \to x_3}(x_3)\mu_{x_E \to D}(x_3)$$

Capítulo 5

Aplicación de códigos IRA sobre canal relé degradado

5.1 Presentación

Para el desarrollo de una simulación de un canal relé degradado con códigos IRA, se ha partido del estudio de Razagui y Yu [5], en el cual se plantea una aplicación práctica de la estrategia de "binning" para el canal relé desde la perspectiva del uso de codificación lineal, en la que los bits adiciones de paridad son generados en el relé para facilitar la comunicación completa entre el origen y el destino.

En [5] se plantea una característica clave del diseño de códigos LDPC de dos capas ya que son capaces de aproximarse a la capacidad de canal Gaussiano con dos diferentes SNRs y a dos diferentes tasas. Debido a que únicamente se muestran los resultados de ese diseño de códigos y no una posible implementación de la modificación de las técnicas convencionales de diseño, se ha optado por un diseño de códigos IRA que por su sencillez permite el desarrollo de la estrategia de binning.

Para demostrar la potencia de los códigos IRA en el modelado de un canal relé, principalmente se ha optado por el diseño de un canal AWGN simple con una fuente y un destino. De esta forma, se podrá hacer un estudio del comportamiento del parámetro BER para distintas configuraciones aplicando códigos IRA y LDPC regulares con diferentes SNRs.

Una vez realizado el estudio de un canal simple, se procederá a la introducción en el modelo un canal relé, aplicando las direfentes etapas de envío, codificación y decodificación establecidas por [5].

5.2 Definición del sistema completo de un canal simple Gaussiano

En esta primera parte de diseño se presentarán las distintas etapas a desarrollar para lograr la codificación en canales relés. Esta tarea tiene como misión dar una visión global de los distintos componentes lógicos del que se va a componer la simulación.

Atendiendo a la premisa de resolución de un problema: "Divide y vencerás", se ha creído conveniente subdividir el sistema completo en partes para que a la hora de de diseñar se pueda realizar de la forma más sencilla posible. Estas subdivisiones se comunicarán entre sí y tendrán una posición en el sistema global específica.

Se ha decidido llevar a cabo el teorema de separación de fuente-canal establecido por Shannon, ya que queremos proporcionar interoperabilidad a cada módulo.

Los módulos de cada subdivisión serán:

- Módulo 1: generador de la matriz de paridad
- Módulo 2: creador de la matriz generadora y codificador
- Módulo 3: generador aleatorio de mensajes
- Módulo 4: transmisor
- Módulo 5: decodificador

5.2.1 Módulo 1: generador de la matriz de paridad

Se ha realizado un desarrollo pensado únicamente en códigos bloque lineales para vectores binarios, como por ejemplo GF(2). El conjunto de palabras código válidas para un código lineal puede ser especificado por la matriz de paridad H, con M filas y N columnas. Las palabras código válidas son vectores, x, de longitud N, para los cuales $Hx = 0$, donde todas las operaciones aritméticas son realizadas en mód.-2. Cada fila de la matriz H representa un chequeo de paridad en un subconjunto de bits del vector x; todas las comprobaciones deben ser satisfactorias para que x sea una palabra código. Hay que tener en cuenta que la matriz de paridad para un código dado, es decir, para un conjunto válido de palabras código, no es única, incluso después de eliminar filas de H que son redundantes, ya que son combinaciones lineales de otras filas.

5.2.1.1 Métodos para la construcción de códigos LDPC

La simulación, que se ha implementado, se por una parte para la experimentación con códigos LDPC regulares. Estos códigos pueden ser construidos por varios métodos como hemos visto en apartados anteriores. Todos ellos incluyen un mecanismo de selección aleatoria sobre la colocación de 1s en la matriz de chequeo de paridad. Muchos de estos métodos de construcción de códigos LDPC tienen la propiedad de que el número de 1s en una columna de la matriz H es aproximadamente en media al tamaño de esta matriz de chequeo de paridad. Dada una tasa binaria del código, las matrices H verán

aumentada su dispersión a mayor longitud de la palabra código, y por consiguiente, el número de chequeos de paridad también aumentará.

Para la distribución de 1s se usa un método que se encarga de situar un número determinado de 1s en cada columna, pero además trata de mantener aproximadamente el mismo número de 1s en las filas. Inicialmente, se crean indicadores para todos los 1s que lo requieran, y se le asigna a cada fila equitativamente un número de 1s. A continuación, se asignan sucesivamente 1s a las columnas mediante selección aleatoria, sin reemplazo, únicamente con la restricción de que los 1s asignados deben encontrarse en filas distintas. Si en algún momento es imposible colocar el número de 1s requerido en las columnas restantes, un 1 es colocado en esa columna sin referencia a otras columnas, creando un posible desnivel.

5.2.1.2 Métodos para la construcción de códigos IRA

Los codificadores LDPC tienen codificadores más complejos, por lo que se ha optado también en la utilización de códigos IRA, en el que hay $u_{1,} \dots, u_k$ nodos de información y $x_1, \dots, x_r$ nodos de chequeo. Por tanto, siguiendo la teoría mostrada en el apartado de códigos IRA el número de aristas que habrá son:

$$ra = k\left(\sum_{i=2}^{J} if_k\right)$$

Para la matriz H se parte de códigos IRA sistemáticos $(k + r, k)$, siendo k el número de bits de información y r el número de bits de redundancia. Por consiguiente, el esquema que se debe cumplir con estos códigos es el siguiente [68]:

- Información $(u_1, \dots, u_k)$
- Palabra código $(x_1, \dots, x_r, u_{1,} \dots, u_k)$

En consecuencia, la tasa de codificación vendrá dada por la siguiente ecuación:

$$R = \frac{k}{k + r} == \frac{a}{\sum_i ifi + a}$$

La fórmula recursiva para la generación de los bits de paridad será la siguiente:

$$x_j = x_{j-1} + \sum_{i=1}^{a} v_{(j-1)*a+1}$$

Siendo:

- v los bits de información
- $a > 1$

El valor que tendrá a para el desarrollo de la simulación será 3, aunque se puede variar por configuración. También se fija a un valor constante todas las f_i para reducir la complejidad al codificador.

Para la creación de H_1, se ha diseñado un algoritmo de permutaciones arbitrarias en el que se van asignando de forma aleatoria tantos 1s como indique el parámetro f_i en las

columnas, así hasta cubrir las k columnas de H_1. Para cumplir con la restricción de diseño, la cual establece que cada fila de H_1 debe contener un número a de 1s, se modifica cada fila para que en el caso de que haya un número mayor o menor de 1s se distribuyan bajo las condiciones de diseño a una posición anterior.

5.2.2 Módulo 2: creador de la matriz generadora y codificador

En la utilización de códigos para el envío de mensajes, es necesario definir un mapeo de un vector s de bits con una longitud k, que representa un mensaje de la fuente para una palabra código x de longitud $N > k$. Se considerarán sólo aplicaciones lineales, que se pueden representar de la forma $x = GT$, siendo G la matriz generadora. Para un código de chequeo de paridad la matriz H, cuyas palabras código deben satisfacer $Hx = 0$, la matriz generadora debe satisfacer $HG^T = 0$. En el caso desarrollado se considera que el número de filas de la matriz de chequeo de paridad H es igual a NK, ya que se considera un caso genérico.

Se hará referencia a una codificación sistemática ya que los K bits de s se copian sin cambios a un subconjunto de N bits de x (los bits del mensaje), y el resto de bits de chequeo $M = N - K$ de x dan lugar a la palabra código. Para un código lineal siempre existe un esquema de codificación sistemático debido a que los bits de una palabra código son bits del mensaje. En general, es normal cambiar el orden de los bits de una palara código para que los primeros bits correspondan al mensaje. Las K primeras columnas de la matriz generadora KxN corresponderán a la matriz identidad.

A pesar de la generalidad del método anterior, se ha optado por una opción en la que no se supone que los bits del mensaje son los primeros, ya que los diferentes métodos de codificación prefieren diferentes lugares de posicionamiento de los bits del mensaje. En su lugar, un vector de índices con la localización de los bits del mensaje es grabado en un fichero con una representación de una porción de la matriz generadora que produce los bits de chequeo. Se puede crear más de un fichero de la matriz generadora para un único fichero de chequeo de paridad, en el que las localizaciones de los bits del mensaje pueden ser diferentes. La decodificación de un mensaje recibido en una palabra código no depende de saber cuáles son los bits del mensaje, aunque será necesario para reconstruir el mensaje original.

5.2.2.1 Representación de la matriz generadora

Para simplificar, se asumirá de aquí en adelante que los bits del mensaje se encuentran al principio de la palabra código, por lo que una palabra código se puede dividir en M bits de chequeo, con el nombre de c, seguidos por K bits del mensaje, con el nombre de s.

En el supuesto anterior, la matriz de chequeo de paridad H, se puede dividir en una matriz A de MxM que ocupan las primeras M columnas de H y en una matriz B de MxK que ocupan el resto de columnas de H. El requisito de la palabra código, x, es satisfacer todos los chequeos de paridad (es decir, $Hx = 0$), por lo que se puede escribir: $Ac + Bs = 0$. A condición de que A sea no singular, se puede concluir que $c = A^{-1}Bs$.

Puede ocurrir que A sea singular para algunas elecciones de las cuales los bits de la palabra código son bits del mensaje, pero para estas elecciones siempre existirá una matriz A no singular si las filas de H son linealmente independientes. Es posible, sin embargo, que las filas de H no sean linealmente independientes. Este es un caso excepcional y no interesante, por lo que se ha obviado del análisis.

La ecuación $c = A^{-1}Bs$ determina como deben ser los bits de chequeo, pero el cálculo real de estos bits de chequeo se puede calcular de varias maneras, aunque se ha desarrollado un único método de implementación. Cada método consiste en una representación diferente de la matriz generadora (ninguno de los métodos realiza una representación explícita de la matriz G).

Se define por tanto, una representación densa, la matriz $A^{-1}B$ de MxK, se calcula y se almacena en un formato denso en mód.-2. Un mensaje se codifica mediante el producto de los bits fuente, s, y esta matriz para obtener los bits de chequeo necesarios.

5.2.3 Módulo 3: generador aleatorio de mensajes

Este módulo proporciona las herramientas básicas para la generación de números pseudo-aleatorios, y para la generación de variables aleatorias de varias distribuciones comunes.

Todos los procedimientos de generación aleatoria utilizan la misma base de números aleatorios. Esta base se genera con una cantidad limitada de números aleatorios reales con el fin de reducir las posibilidades de los números pseudo-aleatorios no sean lo suficientemente aleatorios.

Una base de números pseudo-aleatorios se determina a través de un número entero llamado semilla, que normalmente se configura por el usuario para cada experimento. El estado del número de la base aleatoria puede ser almacenado en una estructura de tipo *rand_state*. Por ejemplo, el estado puede ser grabado en un fichero tras el final del experimento y restaurado posteriormente si se decide que se debe continuar con ese estado más tiempo.

Los métodos para la generación de variables aleatorias de varias distribuciones pueden ser encontradas en [112].

5.2.4 Módulo 4: transmisor

Una vez encontrada la palabra código que representa al mensaje de la fuente emisora, se procede al envío de la misma a través del canal, el resultado será la recepción de determinados datos procedentes de la salida con el canal. Estos datos mantendrán una relación estrecha con la palabra código enviada, no siendo la original debido al ruido aleatorio introducido por el canal. En el caso de estudio se tendrán en consideración los canales binarios sin memoria, en el que el ruido sea independiente de cada bit enviado por el canal y no afecten a los demás.

En el caso de un canal con ruido aditivo blanco (AWGN), en el que el ruido es aditivo porque la señal recibida es igual a la señal trasmitida más un ruido estadísticamente independiente de la señal, y es blanco ya que su densidad espectral de potencia es uniforme, por lo que la autocorrelación del ruido en el dominio del tiempo es cero para cualquier tiempo de offset distinto de cero, los datos recibidos serán iguales a los transmitidos más un ruido Gaussiano con media cero y desviación estándar s. La implementación seguida para el caso de estudio es que los datos enviados serán -1 para el caso de un bit 1 y 0 para el caso de un 1. En otras palabras, la distribución de la señal recibida dado un bit enviado es:

$$y \mid x = 1 \sim N(-1, s^2)$$
$$y \mid x = 0 \sim N(+1, s^2)$$

Generalmente, se asume que la desviación estándar de ruido varía con la tasa de envío de los bits, aumentando en proporción a la raíz cuadrada de la tasa de envío. La tasa de error obtenida al enviar bits sin codificar a una tasa R será la misma a la obtenida usando un código que se repite cada bit n veces a una tasa nR (suponiendo una decodificación óptima de cada bit a través de un umbral calculado con la suma de las n salida de los canales correspondientes a cada bit).

Otra forma de ver el aumento producido en s, es que cuando los bits se envían a un ritmo menor, el destino irá acumulando la salida del canal durante un largo periodo, con el resultado de que la cantidad de ruido (en relación con la señal) se verá disminuida como resultado de un promedio.

Para una mejor explicación, es común comparar los códigos de los canales AWGN en términos de tasa de error de bit y el valor de la relación señal a ruido: $\frac{E_b}{N_0} = \frac{1}{2Rs^2}$ con los que trabajan, donde $R = \frac{K}{N}$ es la tasa de código, y s^2 es la varianza de ruido en el que el código alcanza una tasa de error de bit. Por lo tanto, un código operando a una tasa menor se le permite asumir un nivel de ruido más bajo para hacer una comparación equitativa.

5.2.5 Módulo 5: decodificador

Las palabras código transmitidas son decodificadas a partir de los datos recibidos en base a la probabilidad de las posibles palabras código, que constituye la probabilidad de recibir los datos que realmente se recibieron si la palabra código en cuestión es una de las enviadas.

En el caso de estudio, como se ha mencionado anteriormente se ha establecido un modelo de canal sin memoria, en el que el ruido es independiente bit a bit. Para este tipo de canal, la probabilidad se factoriza en un producto de probabilidades para cada bit. Para efectos de la decodificación, lo único que tiene relevancia es la probabilidad relativa de que un bit sea 1 frente 0. Esta probabilidad se representa mediante un cociente de probabilidades $\frac{P(datos|bit\ es\ 1)}{P(datos|bit\ es\ 0)}$.

Para un canal de ruido aditivo blanco Gaussiano, con señales de -1 para el bit 1 y +1 para el bit 0, y una desviación estándar s, la razón de verosimilitud a favor del bit 1 cuando un dato y es recibido es:

$$exp\left(\frac{2y}{s^2}\right)$$

Es habitual considerar que pueden existir palabras códigos con la misma probabilidad a priori. Este hecho sería razonable si los mensajes de la fuente son igualmente probables (por redundancia de la fuente que se ignora, o bien por eliminar una etapa preliminar de compresión de datos).

El proceso de decodificación se puede realizar usando sólo la matriz de comprobación de paridad definida a través de las palabras códigos, sin hacer referencia a la matriz generadora definida tras asignar los mensajes provenientes de la fuente con las palabras código.

Suponiendo igual las probabilidades a priori para varias palabras código, la probabilidad de decodificar correctamente toda la palabra código se reduce a escoger la palabra código con la mayor probabilidad. Esta acción reduce al mínimo la tasa de error, pero con ello no se garantiza que la decodificación de bloque corresponda con la palabra código completa más probable, de hecho, la decodificación puede no dar lugar al conjunto de palabra código. Reducir al mínimo la tasa de error, parece ser el objetivo más sensato, sin embargo, la redundancia de bloque puede tener una gran importancia en un contexto más amplio.

Una decodificación óptima por cualquier método es imposible para códigos lineales cuando los mensajes tienen más de 20 o 30 bits de longitud. La ventaja fundamental de los códigos de baja densidad de chequeo de paridad es que son buenos a la hora del proceso de decodificación (aunque no óptimos) debido a la propagación de la probabilidad que se describe a continuación.

5.2.5.1 Decodificación a través de propagación de probabilidad

El algoritmo de propagación de probabilidad fue concebido originalmente por Robert Gallager en la década de 1960 y más tarde reinventado por David Mackay. Puede considerarse como una instancia del algoritmo SP para la inferencia en el FG, y como una instancia de la propagación de presunción de premisas en redes probabilísticas.

El algoritmo utilizado en el caso de análisis utiliza una única matriz de chequeo de paridad para el código, cuyas columnas corresponden a los bits de la palabra código, y cuyas filas corresponden a los controles de paridad, y la razón de verosimilitud para los bits deriva de los datos. Su objetivo es encontrar la probabilidad de cada bit de la palabra código transmitida, aunque los resultados del algoritmo, en general, serán sólo aproximados.

Para comenzar, la información sobre cada bit de la palabra código derivado de los datos recibidos de forma independiente de cada bit, se expresa a través de su razón de verosimilitud, la probabilidad de que el bit es 1 dividido por la probabilidad de que el bit

es 0. Esta razón de verosimilitud es igual a la razón de verosimilitud para cada bit, ya que tanto 0 como 1 se asumen igualmente probables a priori. Una vez avance el algoritmo, estas relaciones de probabilidad serán modificadas para tener en cuenta la información procedente de otros bits, siempre y cuando se cumpla con los requisitos relacionados en los que se satisfaga el chequeo de paridad. Para evitar el dos cómputo de la información, para cada bit, el algoritmo mantiene una razón de verosimilitud para cada control de paridad que afecte en esa parte, proporcionando una probabilidad de que ese bit sea 0 frente a 1 basándose únicamente en la información derivada de otros chequeos de paridad, a lo largo de los datos recibidos para el bit.

Para cada control de paridad, el algoritmo mantiene razones de verosimilitud independientes para cada bit que participa en el chequeo de paridad. Estas razones dan la probabilidad de que el chequeo de paridad se haya realizado correctamente si el bit en cuestión es 1 dividido por la probabilidad de que el chequeo se realice correctamente si el bit es 0, teniendo en cuenta que las probabilidades de otros bits que participan en el chequeo de paridad sea 1, como se deriva de las relaciones de probabilidad para esos bits con respecto a ese chequeo.

El algoritmo alterna entre recalcular la razón de verosimilitud para cada chequeo, que serán almacenadas como un campo en las entradas de la matriz de chequeo de paridad, con recalcular los ratios de verosimilitud para cada bit, que se almacenan como un campo en las entradas de la matriz reducida de la matriz de chequeo de paridad.

El proceso de recalcular la razón de verosimilitud para cada chequeo con respecto a algunos bits puede parecer un poco lento, ya que requiere todas las combinaciones posibles de valores de otros bits que participan en el chequeo que se está analizando. Afortunadamente, se ha encontrado un atajo, calculando a través:

$$t = \prod \left[\frac{1}{1+p_i} - \frac{p_i}{1+p_i} \right] = \prod \frac{2}{(1+p_i)-1}$$

Siendo:

- El productorio se realiza sobre todas la razones de probabilidad p_i de todos los bits que participan en el chequeo.
- El factor i en este producto es igual a la probabilidad del bit i sea 0 menos la probabilidad de que sea 1.

Los términos de este producto corresponden a las combinaciones posibles de los valores para los otros bits, con lo que t será la probabilidad de que el chequeo se satisfaga si el bit en cuestión es 0 menos la probabilidad de si el bit es 1. La razón de verosimilitud del chequeo con respecto al bit en cuestión puede ser calculada como $\frac{(1-t)}{(1+t)}$.

Para un chequeo en particular, el productorio anterior difiere para los diferentes bits con respecto al cual queremos calcular el cociente de probabilidad solamente en que para cada bit el factor correspondiente a ese bit se suprime.

Para calcular la razón de verosimilitud para un bit con respecto a un chequeo, se necesita multiplicar conjuntamente la razón de verosimilitud para ese bit procedente de los datos recibidos, y los valores actuales de los cocientes de probabilidad para todos los

chequeos en los que ese bit participa. Para ahorrar tiempo, estos productos se calculan mediante la combinación de productos hacia delante y hacia atrás, similar al método utilizado para la razón de verosimilitud.

Mediante la inclusión de los cocientes de probabilidad de todos los chequeos, un cálculo similar da lugar a la razón de probabilidad actual del bit 1 frente al bit 0 basándose en la información que se tiene hasta el momento. Esta relación puede considerarse un umbral que determine si la conjetura de decidir si ese bit es 1 ó 0 es la correcta.

Se espera que el algoritmo converja a un estado donde las probabilidades de bit den una decodificación casi óptima. Esto no siempre ocurre, pero el algoritmo se comparta lo suficientemente bien como para producir buenos resultados en tasas cercanas al límite teórico de Shannon.

5.3 Definición de un canal relé Gaussiano degradado

Debido a la modularidad que se ha proporcionado en el apartado anterior para los diferentes componentes que configuran un sistema completo que modela un canal AWGN, se mantendrá la funcionalidad de:

- Módulo 1: generador de la matriz de paridad
- Módulo 2: creador de la matriz generadora y codificador modificado
- Módulo 3: generador aleatorio de mensajes
- Módulo 4: transmisor
- Módulo 5: decodificador

Como se ha indicado en el Módulo 2, la codificación en el caso de los códigos IRA se realizará de tal forma que no sigue el mismo procedimiento que en los códigos LDPC, ya que se puede aplicar la fórmula recursiva de generación de bits de paridad vista en 4.6.1.

A continuación se recordarán los elementos necesarios que participan en el mecanismo de codificación IRA:

- Información $(u_1, \dots, u_k)$
- Palabra código $(u_1, \dots, u_k, x_1, \dots, x_r)$, donde $x_1, \dots, x_r$ se calcula aplicando:

$$x_j = x_{j-1} + \sum_{i=1}^{a} v_{(j-1)*a+i}$$

Únicamente, para poder implementar la estrategia DAF para un canal relé Gaussiano degradado, como se ha visto en [5], es necesario establecer una serie de pasos básicos que configuran dicha estrategia.

Antes de comenzar con el esquema iterativo de la estrategia DAF, se establece la nomenclatura utilizada para el modelado de los mensajes que serán enviados desde la fuente hasta el destino. Un mensaje se forma a partir de b bloques, $w_i \epsilon \{1,2, \dots, 2^{nR}\}$, de k bits de información. Cada bloque es aleatoriamente particionado en 2^{nR_1} bins, s_i, de tamaño $2^{n(R-R_1)}$ con $(R_1 \leq R)$. Se puede observar una representación gráfica de esta modelización del mensaje en la figura Fig. 31.

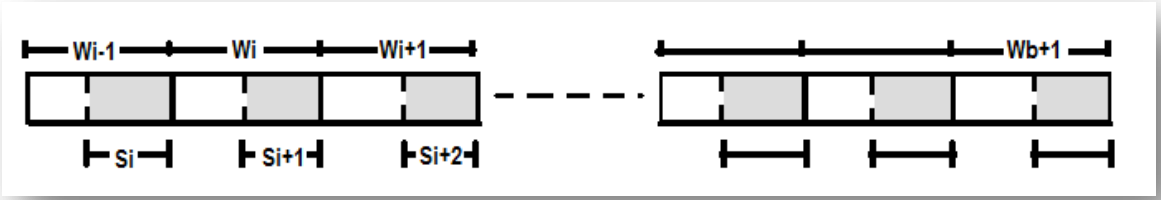


Fig. 31. ***Estructura mensaje para la estrategia DAF***

En el relé, se define: $\widehat{w}_i$ mensaje estimado de la fuente en la iteración i, $\hat{X}_1(\hat{s}_{i+1})$ palabra código del bin estimado $\hat{s}_{i+1}$ asociado a $\widehat{w}_i$.

En el destino, se define: $\widehat{\hat{s}}_i$ bin estimado gracias a la cooperación entre el relé y la fuente. Atendiendo a Fig. 32, v_{i-1} es el bin necesario para completar el mensaje w_{i-1} estimado procedente de la fuente.

Hay que tener en cuenta que la nomenclatura w_i hace referencia tanto al mensaje de la fuente en la etapa i, como al patrón de bits que lo conforman. Dicha nomenclatura también es aplicable a s_i y u_i.

Debido a que no se procedió al uso de códigos LDPC de dos capas como se indicaba en [5], será necesario utilizar un nuevo mecanismo para la etapa de codificación de la fuente y del relé. Para ello, para poder codificar los bins s_i será necesario crear la matriz de paridad H_1 asociada a la tasa R_1 y en el caso del mensaje w_i, se creará la matriz H asociada a la tasa R, sin necesidad de particionar la matriz de paridad H creada previamente para la codificación de los bloques w_i.

Considérese $X(w_i|s_i)$, palabra código enviada desde la fuente al relé, que está constituida por la suma de la palabra código $\tilde{X}(w_i)$ asociada al mensaje w_i y escalada por factor $\sqrt{\alpha P}$, y la palabra código $X_1(s_i)$ del bin s_i aplicándose en este caso un factor $\sqrt{(1-\alpha)P}$.

A lo largo de las diferentes etapas se mantendrá:

- $C_H(.)$ nomenclatura asociada a la codificación de un mensaje fuente especificado por H
- $C_{H1}(.)$ nomenclatura asociada a la codificación de un bin especificado por H_1
- $C_H^{-1}(.)$ nomenclatura asociada a la decodificación de las palabras código recibidas asociadas H aplicando el algoritmo SP.
- $C_{H_1}^{-1}(.)$ nomenclatura asociada a la decodificación de las palabras código recibidas asociadas H_1 aplicando el algoritmo SP.

5.3.1 Etapa en la fuente: iteración i

La fuente en la iteración i tiene que construir la palabra código $X(w_i|s_i)$ que estará constituida por el bin s_i perteneciente al mensaje w_{i-1} y el mensaje fuente w_i:

$$X(w_i|s_i) = \sqrt{\alpha P} * \tilde{X}(w_i) + \sqrt{(1-\alpha)P} * X_1(s_i)$$

Siendo:

- $\tilde{X}(w_i) = C_H(w_i)$, palabra código de w_i especificada por la matriz de paridad H.
- $X_1(s_i) = C_{H_1}(s_i)$, palabra código del bin s_i especificada por la matriz de paridad H_1.

Cuando el proceso de simulación se encuentre en la primera iteración en la etapa de creación del mensaje $X(w_i|s_i)$, s_i será cero debido a que no se dispone de información sobre el mensaje w_{i-1}. Será necesario alcanzar la segunda iteración del proceso para que dicho bin obtenga su valor del mensaje fuente anterior.

5.3.2 Etapa de transmisión Fuente-Relé y Fuente-Destino: iteración i

La fuente, en la iteración i, envía tanto al relé como al destino la palabra código $X(w_i|s_i)$ donde se le añade ruido aditivo Gaussiano $Z_1 \sim N(0, N_1)$. La expresión matemática final para la simulación de esta etapa quedaría de la siguiente manera:

$$Y_1 = X + Z_1 = \sqrt{\alpha P} * \tilde{X}(w_i) + \sqrt{(1-\alpha)P} * X_1(s_i) + Z_1$$

5.3.3 Etapa relé: iteración i

Una vez que se recibe Y_1 en el nodo relé, éste tiene dos misiones: decodificar $\widetilde{w}_i$ a partir de Y_1 y del bin s_i almacenado, en el nodo relé, en la iteración $i-1$ y en segundo lugar, crear el bin s_{i+1} a partir del mensaje $\widetilde{w}_i$ que se ha decodificado en la iteración i.

Para llevar a cabo la misión de decodificar $\widehat{w}_i$, en el relé, será necesario eliminar la aportación de s_i del mensaje Y_1 enviado por la fuente, ya que es conocida por el relé. Para ello, el nodo relé deberá haber almacenado previamente el bin asociado al mensaje $\widehat{w}_{i-1}$. En el caso de que el proceso se encuentre en la primera iteración, el valor de dicho bin será 0. Por consiguiente, la expresión necesaria para la simulación de dicha etapa es:

$$\widehat{w}_i = C_H^{-1}\left(Y_1 - (1-\alpha)P * \hat{X}_1(\hat{s}_i)\right)$$

En segundo lugar, debido a que ya se ha obtenido $\widehat{w}_i$, el relé puede calcular el bin $\hat{s}_{i+1}$ asociado a dicho mensaje, que a su vez será transmitido en la siguiente iteración.

$$\hat{X}_1(\hat{s}_{i+1}) = C_{H_1}(\hat{s}_{i+1})$$

En resumen, la decodificación de $\tilde{X}(w_i)$ será exitosa si y sólo si se cumple la siguiente restricción sobre la tasa R:

$$R \leq \frac{1}{2}\log\left(1+\frac{\alpha P}{N_1}\right)$$

Cabe destacar que para la utilización de códigos binarios únicamente se podrá trabajar con tasas menores que al segundo miembro de la ecuación anterior. Mientras que si partiéramos de la utilización de códigos Gaussianos, la tasa de código alcanzable será la igualdad.

5.3.4 Etapa de transmisión Relé-Destino: iteración i

En la iteración i, el relé se encarga de enviar al destino el bin codificado $\hat{X}_1(\hat{s}_i)$ con una potencia $\sqrt{\beta P}$ a través de un canal AWGN con $Z_2 \sim N(0, N_2)$.

$$Y_2 = \sqrt{\beta P} * \hat{X}_1(\hat{s}_i) + Z_2$$

5.3.5 Etapa destino: iteración i

El destino en el instante i recibe información tanto de la fuente Y_1 como del relé Y_2. Al igual que el relé, tendrá dos tareas que llevar a cabo. La primera de ellas será decodificar $\hat{s}_i$ a partir de la información disponible hasta ese momento. En segundo lugar, debido a que el destino almacena previamente Y_i' podrá reconstruir el mensaje w_{i-1} ya que además dispone de la información de $\hat{s}_i$.

Para que el proceso de reconstrucción de cada bloque del mensaje fuente, se presenta un esquema en Fig. 32 que muestra las dos secciones diferencias del mensaje que se van a ir calculando en las distintas iteraciones.

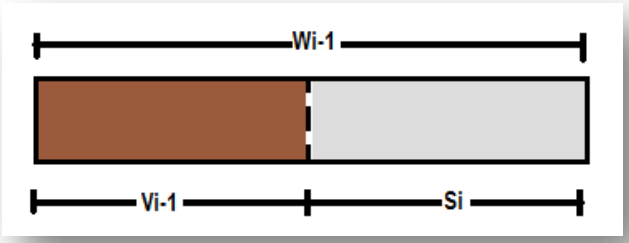


Fig. 32. ***Estructura de un bloque del mensaje para un canal relé degradado***

Para la obtención de $\hat{s}_i$ será necesario decodificar la información presente en el destino como se menciona anteriormente, por lo que la expresión que refleja esta operativa será:

$$Y_i = \sqrt{\alpha P} * \tilde{X}(w_i) + \sqrt{(1-\alpha)P} * X_1(s_i) + Z_1 + \sqrt{\beta P} * \hat{X}_1(\hat{s}_i) + Z_2$$

$$\hat{\hat{s}}_i = C_{H_1}^{-1}(Y_i)$$

En el caso de encontrarse el proceso en la primera iteración, el destino sería incapaz de obtener el bin v_{i-1} del bloque w_i ya que no se dispone de información en dicha iteración sobre Y_{i-1}. Por consiguiente, el destino se verá forzado a almacenar la información Y_i para la siguiente iteración. Debido a que se conoce la influencia de $\hat{\hat{s}}_i$ en Y_i, se ha optado que en vez de almacenar Y_i se construya Y_i'.

$$Y_i' = Y_i - \left(\sqrt{(1-\alpha)P} + \sqrt{\beta P}\right) * C_{H_1}(\hat{\hat{s}}_i)$$

Gracias a almacenar Y_i' en el destino, se podrá reconstruir el bloque completo w_i en la iteración $i+1$ ya que se dispone de v_{i-1} y $\hat{\hat{s}}_i$.

$$v_{i-1} = C_H^{-1}(Y_i'(\hat{\hat{s}}_i), \hat{\hat{s}}_i)$$

Para el proceso de decodificación v_{i-1}, el algoritmo SP utilizado para el paso de mensajes se verá amañado debido a que se conoce parte de la información contenida en la matriz de paridad H, debido a la partición en H_1. De este modo, los primeros k bits de la palabra código de cada bloque verá modificada su razón de verosimilitud en un factor dependiente de si se estima +1 o -1. Por consiguiente, se mejora la eficiencia del proceso de decodificación por paso de mensaje.

Por último, al igual que en la etapa relé, será necesario cumplir con una restricción de la tasa R_1 para que el proceso de decodificación sea satisfactorio.

$$R_1 \leq \frac{1}{2}\log\left(1 + \frac{\left(\sqrt{P_1} + \sqrt{(1-\alpha)P}\right)^2}{\alpha P + N_1 + N_2}\right)$$

5.3.6 Etapa final

Para llegar a reconstruir el mensaje completo enviado por la fuente en el destino, será necesario repetir las etapas anteriores un número de iteraciones $b+1$, siendo b el número de bloques del que se compone un mensaje.

Capítulo 6

Resultados

6.1 Establecimiento de parámetros de simulación

Para poder desplegar el conjunto de módulos para la simulación de un canal relé degradado con una estrategia DAF es necesario fijar una serie de parámetros definidos en gran detalle a lo largo del capítulo 5. A continuación, se mostrará una tabla resumen de todos ellos junto con el valor que se ha establecido para su prueba.

PARÁMETRO	VALOR
Bits de información	5000 y 30000
Tasa de código	0.5, 0.75 y 0.25
Modulación	BPSK
$\frac{E_b}{N_0}$ **(dB)**	[0, 0.03, 0.1, 0.13, 0.15, 0.18, 0.2, 0.23, 0.25, 0.28, 0.3, 0.33, 0.35, 0.38, 0.4, 0.43, 0.45, 0.48, 0.5, 0.53, 0.55, 0.58, 0.6, 0.63, 0.65, 0.68, 0.7, 0.73, 0.75, 0.78, 0.8, 0.83, 0.85, 0.88, 0.9, 0.93, 0.95, 0.98, 0.99, 1, 1.03, 1.05, 1.06, 1.08, 1.09, 1.1, 1.13, 1.15, 1.18, 1.2, 1.23, 1.25, 1.28, 1.3, 1.33, 1.35, 1.38, 1.4]
Matriz paridad	Sparse

Tabla 2. ***Parámetros definidos para simulación canal AWGN simple con códigos LDPC***

PARÁMETRO	VALOR
Bits de información	5000 y 30000
Tasa de código	0.5, 0.75 y 0.25
Modulación	BPSK
$\frac{E_b}{N_0}$ **(dB)**	[0, 0.03, 0.1, 0.13, 0.15, 0.18, 0.2, 0.23, 0.25, 0.28, 0.3, 0.33, 0.35, 0.38, 0.4, 0.43, 0.45, 0.48, 0.5, 0.53, 0.55, 0.58, 0.6, 0.63, 0.65, 0.68, 0.7, 0.73, 0.75, 0.78, 0.8, 0.83, 0.85, 0.88, 0.9, 0.93, 0.95, 0.98, 0.99, 1, 1.03, 1.05, 1.06, 1.08, 1.09, 1.1, 1.13, 1.15, 1.18, 1.2, 1.23, 1.25, 1.28, 1.3, 1.33, 1.35, 1.38, 1.4]
$a\|_{R=0.5\ y\ R=0.25}$	3
$a\|_{R=0.75}$	28
i **asociado a** $\sum_i if_i$	$3\|_{R=0.5}$ y $9\|_{R=0.25}$
$\sum_i if_i\|_{R=0.75}$	$\lambda(x)$, distribución de grados aleatoria fijando restricciones vistas en 4.6.3.

Tabla 3. ***Parámetros definidos para simulación canal AWGN simple con códigos IRA***

PARÁMETRO	VALOR
Bits de información	5000 y 30000
Tasa de código	0.5
Modulación	BPSK
Potencia (dB)	[0, 0.2, 0.4, 0.6, 0.8, 1, 1.2, 1.4, 1.6, 1.8, 2, 2.1, 2.2, 2.4, 2.6, 2.8, 3, 3.2, 3.4, 3.6, 3.8, 4, 4.2, 4.4, 4.6, 4.8, 5]
Caso $a_{R=0.5}(H)$ **y** $a_{R=0.25}(H_1)$	3
i **asociado a** $\sum_i if_i$	$3\|_{R=0.5}$ y $9\|_{R=0.25}$
α	0.868
β	0.5
N_1	1.1713
N_2	1.5457

Tabla 4. ***Parámetros definidos para simulación canal relé degradado con estrategia DAF y códigos IRA***

PARÁMETRO	VALOR
Bits de información	5000
Tasa de código	0.68
Modulación	BPSK
Potencia (dB)	[0, 0.4, 1, 1.6, 2, 2.6, 3, 3.5, 4, 4.5, 5, 5.5, 6, 6.5, 7, 7.5, 8, 8.5, 9, 9.5, 10, 10.5, 11]
Caso $A_{0.68}(H)$ **y** $A_{0.33}(H_1)$	16 y 5 respectivamente
$\sum_i if_i$	$\lambda(x)$, distribución de grados aleatoria fijando restricciones vistas en 4.6.3.
α	0.868
β	0.5

N_1	1.1713
N_2	1.5457

Tabla 5. ***Parámetros definidos para simulación canal relé degradado con estrategia DAF y códigos IRA***

6.2 Resultados caso 1 canal AWGN

Para llevar a cabo la simulación del caso 1, se parte de la elección de códigos LDPC e IRA con una configuración de 5000 bits de información, una tasa de código 0.5 y una E_b/N_0 con los valores asignados tanto en la Tabla 2 como en la Tabla 3.

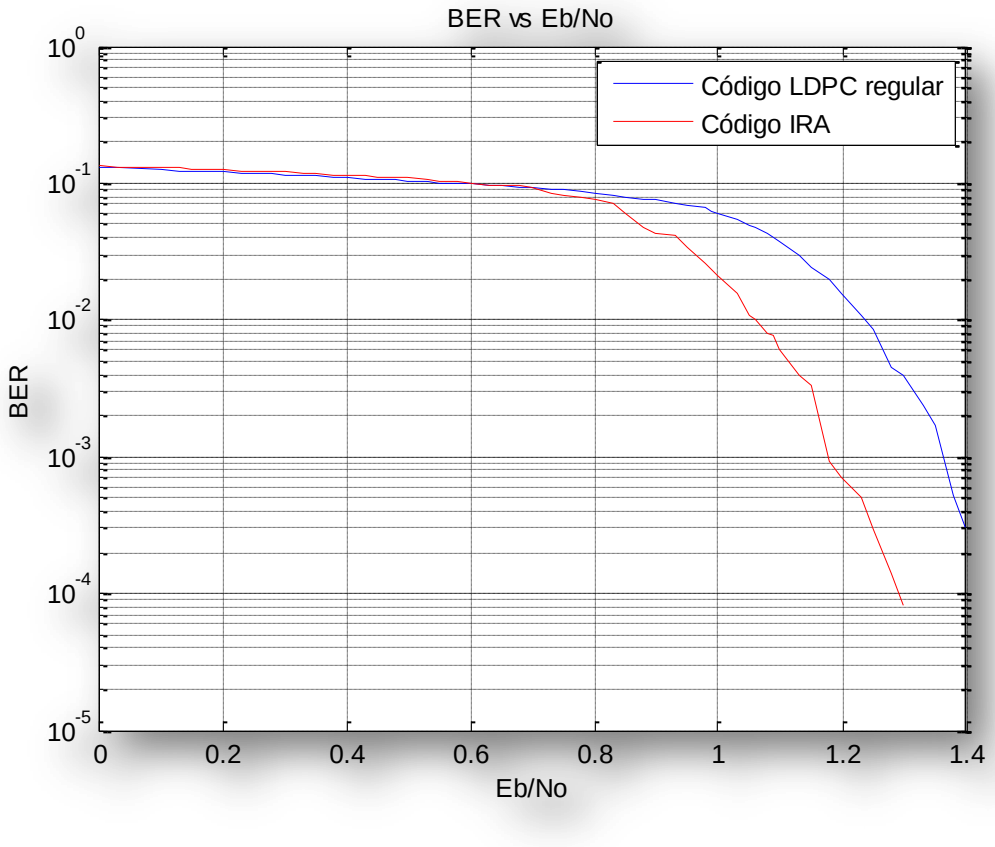


Fig. 33. ***Comparación curvas $\frac{E_b}{N_0}$ vs BER para un canal AWGN con códigos LDPC e IRA, k=5000, R=0.5***

Cómo se puede comprobar para una tasa de código de 0.5, a partir de un cierto valor de E_b/N_0, el rendimiento ofrecido por un código IRA con generación aleatoria del patrón Π es significativamente mejor que para un código regular LDPC con selección de grados también aleatoria.

6.3 Resultados caso 2 canal AWGN

Para la simulación del caso 2, partimos de códigos LDPC e IRA con 30000 bits de información, una tasa de código 0.5 y una E_b/N_0 con los valores asignados tanto en tabla Tabla 2 como en la tabla Tabla 3.

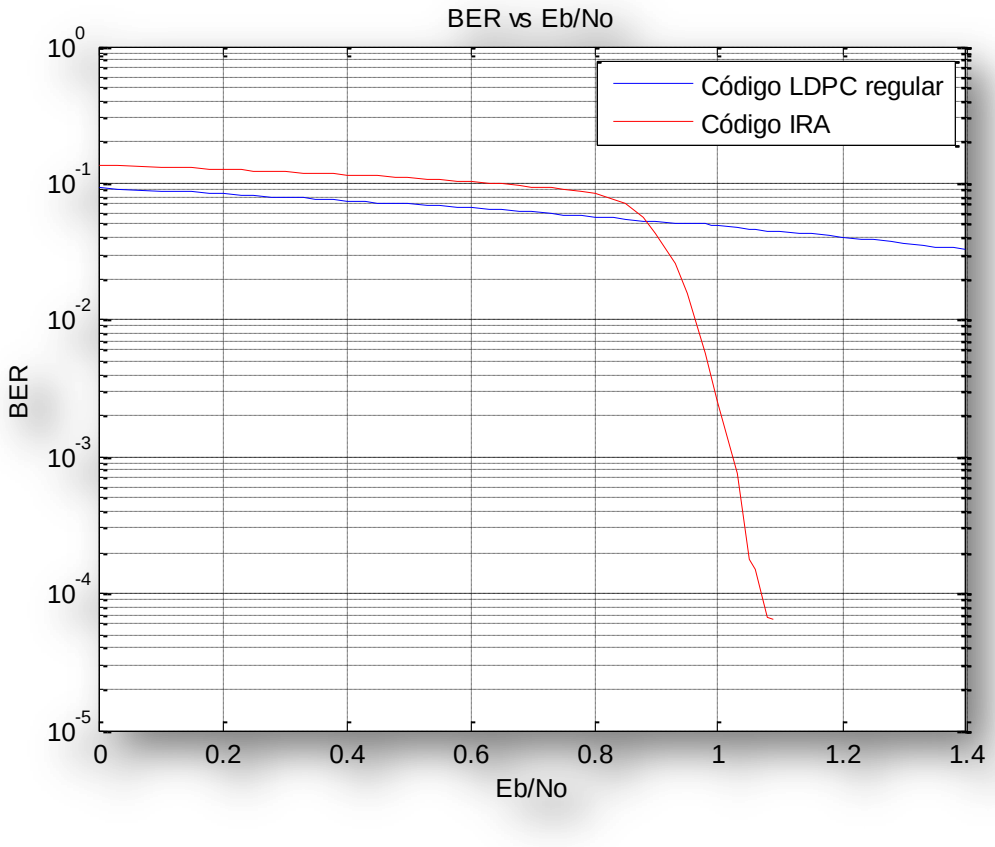


Fig. 34. ***Comparación curvas $\frac{E_b}{N_0}$ vs BER para un canal AWGN con códigos LDPC e IRA, k=30000, R=0.5***

Se observa que la curva asociada al código LDPC regular presenta una caída que no se muestra en el rango de E_b/N_0 representado ya que únicamente se pretendía ofrecer una comparación de rendimiento entre los dos códigos representados.

En la Fig. 31 se aprecia que para un valor de E_b/N_0 entorno a 0.85 dB, los códigos IRA presentan unos valores de BER mejores a los obtenidos en los códigos LDPC.

6.4 Resultados caso 3 canal AWGN

Para la simulación del caso 4, partimos de códigos LDPC e IRA con 5000 bits de información, una tasa de código 0.25 y una E_b/N_0 con los valores asignados tanto en la tabla Tabla 2 como en la tabla Tabla 3.

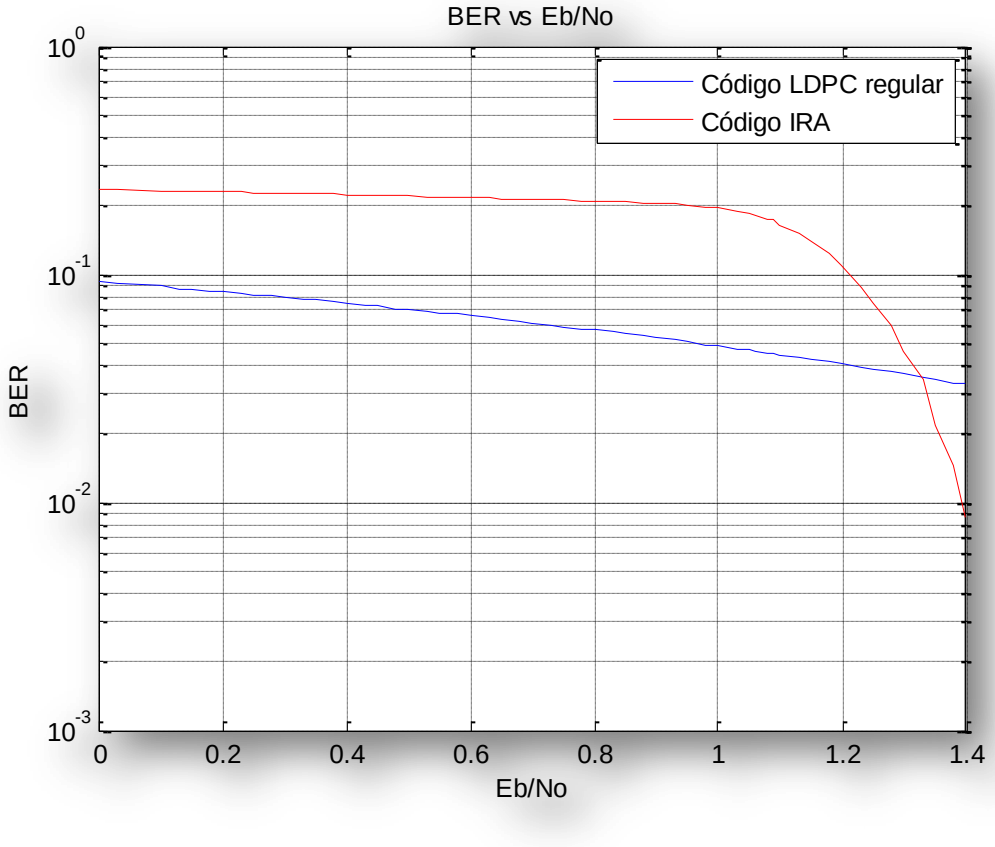


Fig. 35. ***Comparación curvas $\frac{E_b}{N_0}$ vs BER para un canal AWGN con códigos LDPC e IRA, k=5000, R=0.25***

Al igual que en el apartado 6.3, se observa que la curva asociada al código LDPC regular presenta una caída que no se muestra en el rango de E_b/N_0 representado ya que únicamente se pretendía ofrecer una comparación de rendimiento entre los dos códigos representados.

Una consecuencia inmediata al disminuir la tasa del código y que se puede observar en la Fig. 32, es que el punto de cruce entre las dos curvas se ve desplazado hasta el valor de E_b/N_0 de 1.3dB.

6.5 Resultados caso 4 canal AWGN

Para la simulación del caso 3, partimos de códigos LDPC e IRA con 5000 bits de información, una tasa de código 0.75 y una E_b/N_0 con los valores asignados tanto en la tabla Tabla 2 como en la tabla Tabla 3.

Atendiendo a la ecuación de la tasa para un código IRA sistemático $R = a/a + \sum_i i f_i$, si intentamos fijar $\sum_i i f_i$ a un valor constante como es el caso de las tasas de 0.5 y 0.25, el número de aristas salientes de los nodos de información, para $R = 0.75$, no son coincidentes con el número de aristas entrantes en los nodos de chequeo. Por consiguiente, en este caso será necesario establecer una variación de grados entre los nodos de información para poder llegar a cumplir $R = 0.75$.

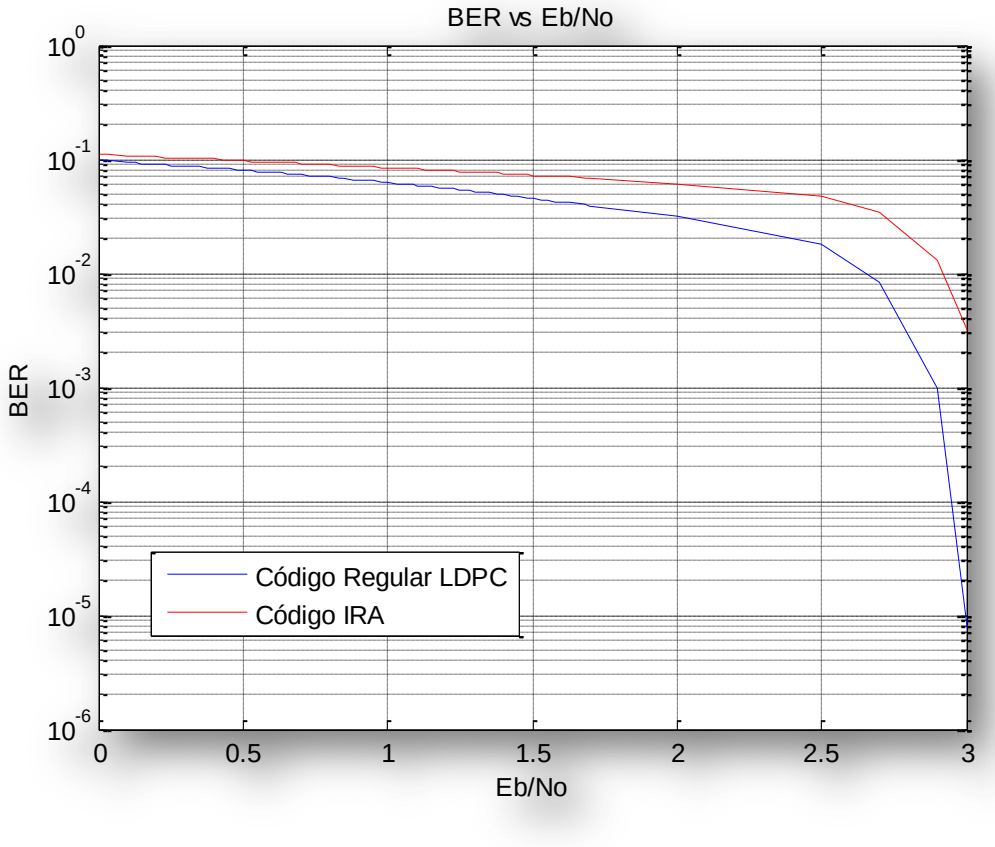


Fig. 36. ***Comparación curvas $\frac{E_b}{N_0}$ vs BER para un canal AWGN con códigos LDPC e IRA, k=5000, R=0.75***

En la Fig. 33, se observa que si se codifica con una tasa de 0.75, ambos códigos tienen una tendencia hacia valores de BER pequeños de forma más lenta que en casos anteriores.

Este sería el único caso que los códigos LDPC presentarían un mejor rendimiento que los códigos IRA para el rango de valores E_b/N_0 representados.

6.6 Resultados caso 1 canal relé degradado

Para la simulación del caso 1, partimos de códigos IRA con 5000 bits de información, una tasa de código 0.5 y el resto de valores asignados en la tabla Tabla 4.

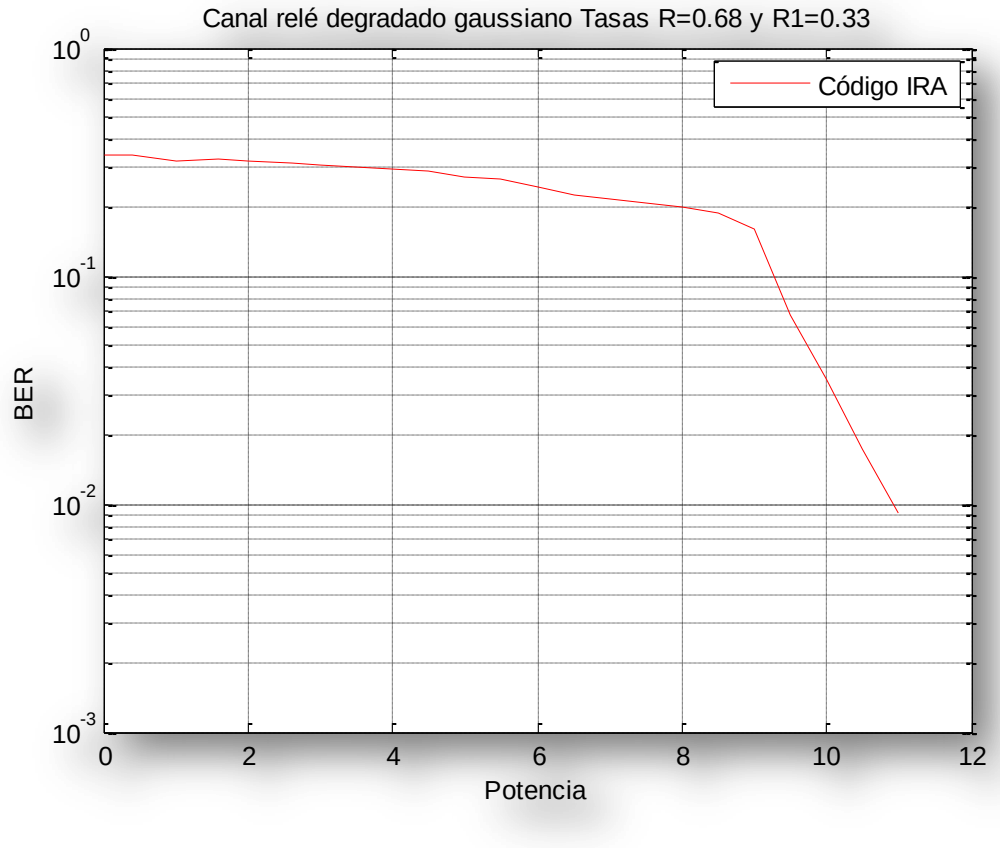


Fig. 37. ***Curva Potencia vs BER para un canal relé Gaussiano degradado con códigos IRA, k=5000, R=0.5 y R_1=0.25***

Para estabilizar estadísticamente los resultados obtenidos en esta gráfica, se ha establecido para los diferentes niveles de potencia que el mínimo número de mensajes erróneos obtenidos al finalizar todas las etapas de la estrategia DAF sea al menos un 20% del número total de mensajes enviados desde la fuente al destino.

Además, se cumplirán las restricciones de tasa de código en R^+, R^- y R_1 que se han establecido a lo largo del capítulo anterior para parámetros de la tabla Tabla 4.

$$R^+ = \frac{1}{2}\log\left(1+\frac{\alpha P}{N_1}\right) = 0.849615696070390$$

$$R^- = \frac{1}{2}\log\left(1+\frac{\alpha P}{N_1+N_2}\right) = 0.457592792707255$$

$$R_1 \leq \frac{1}{2}\log\left(1+\frac{\left(\sqrt{P_1}+\sqrt{(1-\alpha)P}\right)^2}{\alpha P+N_1+N_2}\right) = 0.442323636012968$$

Por consiguiente, habiendo escogido las tasas de codificación de en la fuente y en el relé se puede concluir que:

$$R = 0.5 < R^{+}$$
$$R_1 = 0.25 < 0.4423$$
$$R^{+} < R_1 + R^{-} = 0.75$$

Siendo:

- R la tasa de código para la obtención de la palabra código de la fuente.
- R_1 la tasa de código para la obtención de la palabra código del relé.
- $R^{+} = I(X; Y_1|X_1)$, sería la tasa alcanzable para el caso de un código Gaussiano en el enlace entre fuente-relé
- $R^{-} = I(X; Y|X_1)$, sería la tasa alcanzable para el caso general de un código Gaussiano entre la fuente y el destino
- $R_1 = I(X_1; Y)$, sería la tasa de código alcanzable en el enlace entre el relé y el destino.

En conclusión, se ha utilizado una tasa de codificación entre la fuente y el relé por debajo de la especificada por los parámetros de configuración, al igual que se cumplen las restricciones de la tasa de codificación asociada al enlace relé-destino.

6.7 Resultados caso 2 canal relé degradado

Para la simulación del caso 2, partimos de códigos IRA con 30000 bits de información, una tasa de código 0.5 y el resto de valores asignados en Tabla 4.

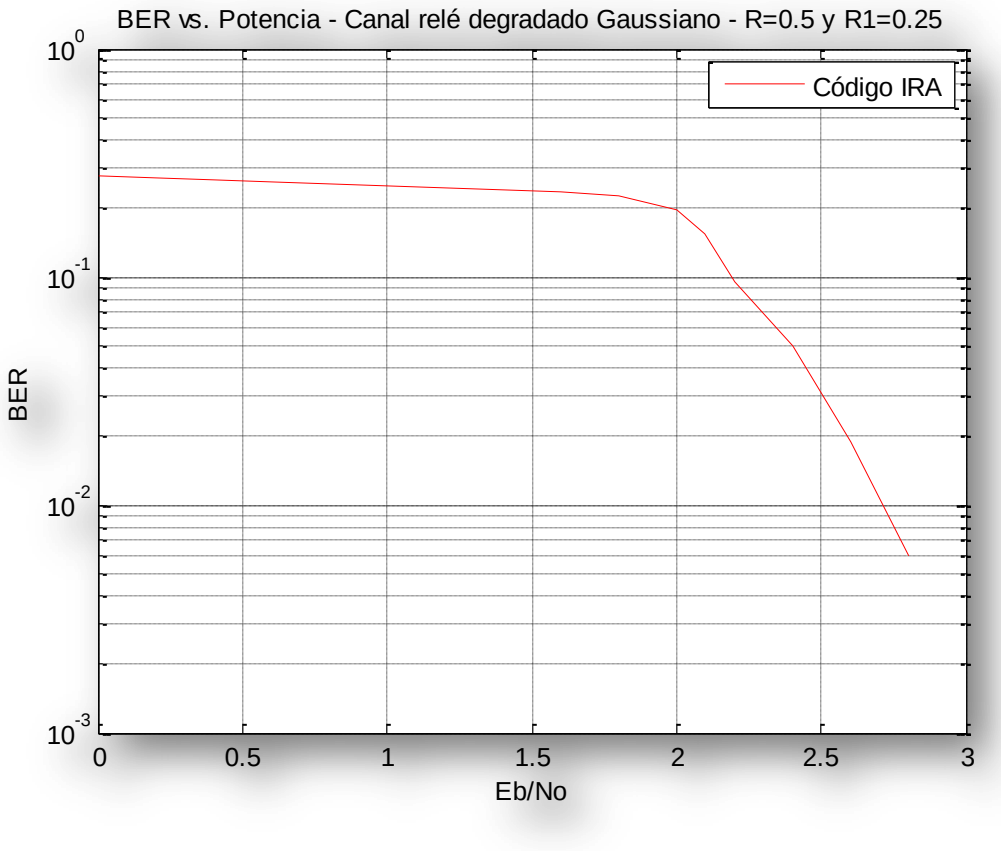


Fig. 38. ***Curva Potencia vs BER para un canal relé Gaussiano degradado con códigos IRA, k=30000, R=0.5 y R_1=0.25***

Atendiendo a los resultados mostrados en la figura Fig. 35 y comparándolos con la curva del código IRA para un canal simple Gaussiano de la figura Fig. 31, se puede concluir que la implementación de un canal relé Gaussiano presenta mejores prestaciones debido a que los valores de BER obtenidos para una potencia de ruido y señal dada son menores.

6.8 Resultados caso 3 canal relé degradado

Para la simulación del caso 3, partimos de códigos IRA con 5000 bits de información, una tasa de código 0.68 y el resto de valores asignados en la tabla Tabla 5.

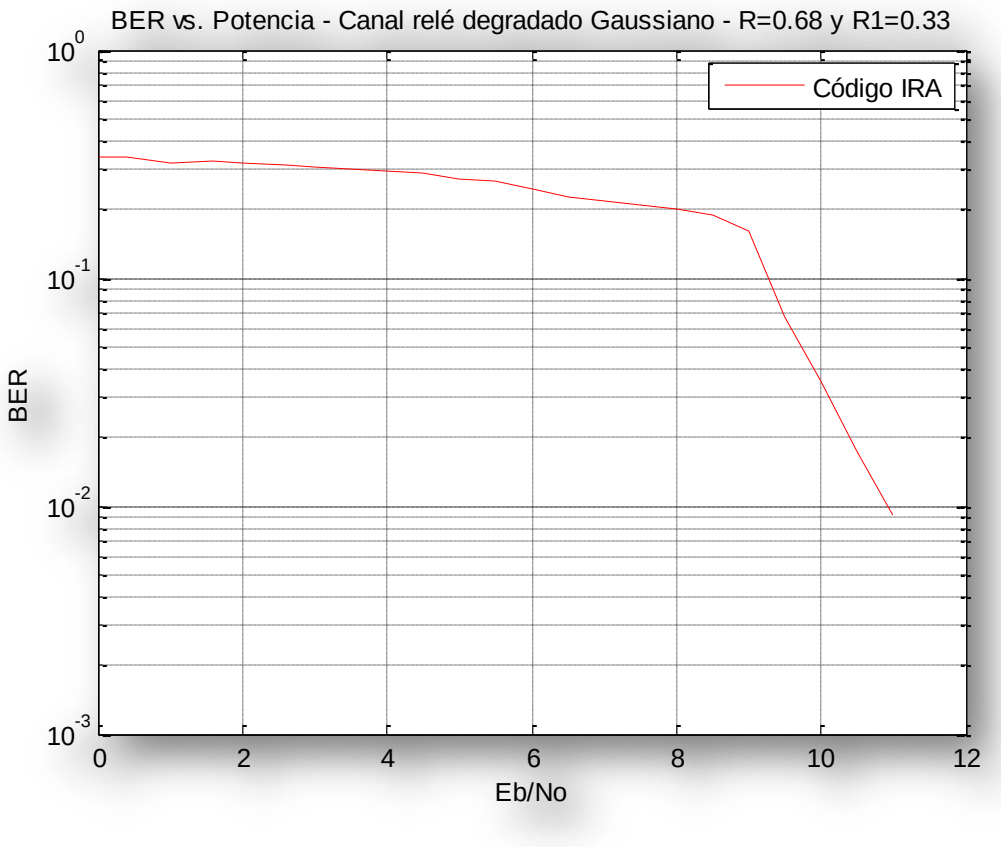


Fig. 39. ***Curva Potencia vs BER para un canal relé Gaussiano degradado con códigos IRA, k=5000, R=0.68 y R_1=0.33***

Debido a que se permiten tasas $R^+ < 0.8496$ en el sistema, se considera el caso de estudio con una tasa superior a 0.5 y menor que 0.8496.

Debido a que no se podía fijar un valor $\sum_i if_i$ constante para la tasa elegida, ha sido necesario crear una distribución de grados $\lambda(x)$ para la codificación IRA. Los criterios de implementación para esta distribución se mencionan en el apartado 4.6.3.

Se puede apreciar que debido al aumento de la tasa del código y en consecuencia, a la disminución de la redundancia, los valores de potencia para conseguir un valor de BER pequeña se verán incrementados manteniendo la misma potencia de ruido que para los apartados anteriores.

Capítulo 7

Conclusiones

El objetivo de crear una simulación basada en una implementación práctica de la estrategia de "binning" para el canal relé Gaussiano desde la perspectiva de aplicación de códigos lineales se ha conseguido gracias a introducir bits adicionales de paridad generados en el relé para facilitar la comunicación global entre la fuente y el destino. Aunque se partió del caso de utilizar códigos lineales LDPC bicapa para el proceso de codificación, se optó mejor por códigos IRA que permiten una partición de la matriz de paridad general, pudiendo así aproximarse la capacidad del canal Gaussiano a dos diferentes $SNRs$ y a dos diferentes tasas.

Además, atendiendo a los resultados de rendimiento del capítulo 6, se puede observar que hay un beneficio significativo a partir de un cierto valor de SNR y configuración de los códigos IRA frente a los tradicionalmente utilizados códigos LDPC. En segundo lugar, los resultados muestran que haciendo una distribución pesos en las columnas de H, el acumulador de los códigos IRA no sólo proporciona una ventaja en la complejidad de codificación frente a los códigos LDPC regulares, sino que introduce una ventaja en la decodificación también.

Para una posible implementación del canal relé Gaussiano ha sido necesario la aplicación del concepto de canal degradado ya que teóricamente no existía un desarrollo que proporcionase la región de capacidad con la que se podría trabajar. Esto se debe a que en el caso de no ser un canal degradado el límite superior e inferior no serían coincidentes. Por consiguiente, si el canal relé no es degradado la cooperación no podría llevarse a cabo entre los diferentes nodos que componen el canal.

Se ha podido comprobar a lo largo de la implementación del canal de estudio, que la estrategia DAF seguida no posee gran complejidad frente a la otra técnica propuesta en [4]. A pesar de que la estrategia DAF requiera de un proceso de decodificación y reenvío, no ha provocado una ralentización en la implementación final, observando que los resultados se comportan de forma esperada.

Se ha observado que gracias a la modularidad que se ha proporcionado a través de la codificación C++ de los diferentes elementos dentro del canal que se ha simulado, ha permitido introducir variaciones para los diferentes casos de estudio de forma sencilla. Lo cual permite de una forma casi inmediata la trasformación o mejora en alguno de los módulos.

Como cabía de esperar en los resultados observados en los casos para un canal simple AWGN a medida que se aumenta el número de bits de información, en la curva de BER vs. Eb/No se aprecia una mayor continuidad frente a los saltos entre valores en el caso que se disminuya ese número de bits. Otro hecho, apreciable es que aumentando la tasa de código las curvas obtenidas presentan mejores resultados, pero si esa tasa es demasiado grande la complejidad de operación aumenta. Por lo que se ha optado para la simulación de un canal relé Gaussiano degradado por una tasa de código intermedia.

Tras el diseño e implementación del canal relé Gaussiano degradado se observa que los resultados de BER obtenidos en los casos 1 y 2, responden de manera favorable tras el aumento de potencia P y en consecuencia, de P_1. Comparándolo con la simulación de un canal simple AWGN, frente a distribuciones de ruido superiores el canal relé presenta unos resultados mejores. Aunque se procediera a una optimización de los códigos IRA empleados en la codificación, posiblemente estos resultados mejorasen notablemente.

Capítulo 8

Líneas futuras

Si se hubiera querido seguir por completo la teoría expuesta en el artículo [5], hubiera sido necesario diseñar códigos LDPC reducidos de dos capas para la estrategia DAF en un canal relé Gaussiano. Por consiguiente, se propone la implementación de la estructura de códigos LDPC mostrada en Fig. 40 donde el subgrafo representa un código LDPC para canal fuente-relé. El grafo global representa un código LDPC para el destino. Se llamará a la estructura del código como código LDPC expurgado de dos capas, ya que la gráfica en general representa un subcódigo expurgado de un código de una capa más baja. Hay que tener en cuenta que la realización práctica de un código de dos capas se caracteriza por dos rangos de capacidades en SNR_+ y en SNR_-.

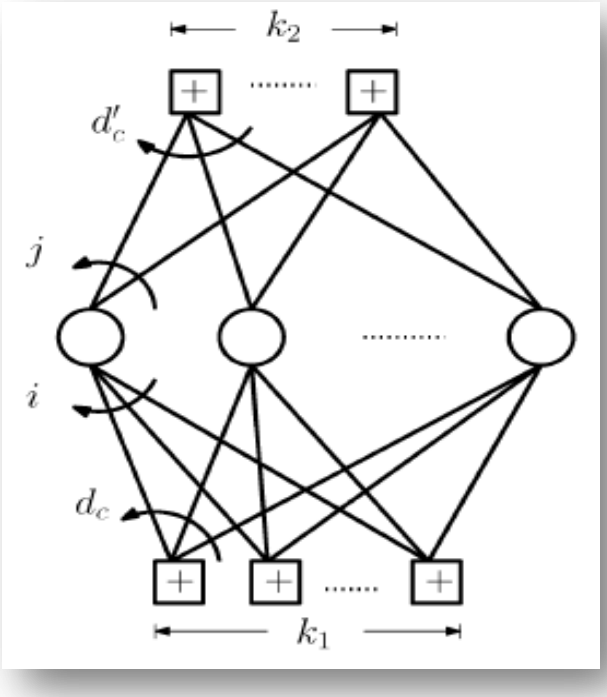


Fig. 40. ***Representación de un código expurgado de dos capas.*** **[4]**

Otra posible implementación para mejorar el comportamiento de la creación de mensajes pseudo-aleatorios sería la utilización de librerías especializadas, en vez de utilizar las librerías estándar que proporciona C. Para ello, se propone el uso de la librería IT++ para Windows ya que dispone un módulo de generación de números aleatorios.

Hasta ahora, el trabajo que se ha llevado a cabo para el desarrollo de la simulación es para un canal de un solo relé, y aunque se pueda llegar a aplicar códigos LDPC de dos capas para acercarse a la mejor tasa DAF en una configuración clásica, se puede llegar a plantear un contexto más general. Si se llegase a diseñar códigos de múltiples capas podría llegarse a implementar una red con múltiples relés. Una posible propuesta para generalizar la tasa DAF a redes con múltiples relés es imponer una ordenación lineal de los relés intermedios, y dejar que cada relé decodifique completamente el mensaje de origen con la ayuda de relés anteriores a él en dicha ordenación. De esta manera, el relé participará en la transmisión del mensaje de origen a los siguientes relés y al destino Fig. 41. Si se quiere obtener más información sobre esta línea futura se puede consultar [48] y [19].

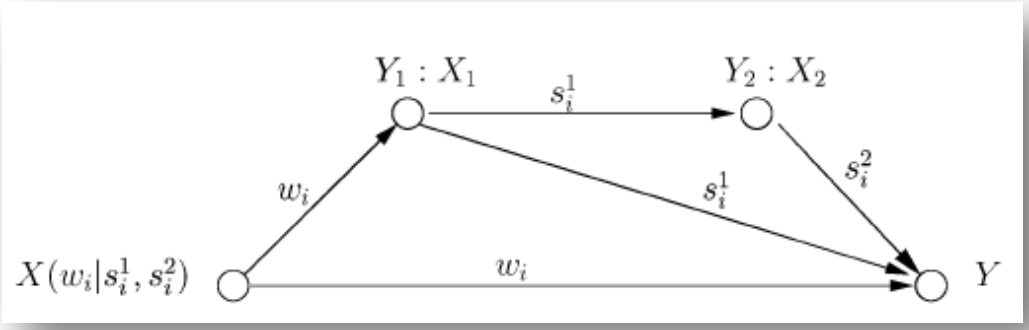


Fig. 41 ***Red de dos relés en el que el segundo facilita la transmisión de los bits de paridad desde el primer relé hasta el destino*** **[4]**

Por último, una línea futura que es lógicamente planteable sería la utilización de un canal que no fuese degradado para el caso de un canal relé Gaussiano, pero todavía no se conoce teóricamente su región de capacidad, es decir, el límite superior y el límite inferior no son coincidentes. El mismo problema se plantea para el caso de que exista más de un relé. En [113] presenta un resultado para una capacidad asintótica en una red general relé Gaussiana con relés múltiples. El término asintótica significa que el límite superior e inferior se encuentra asintóticamente cuando el número de relés tiende a infinito.

Capítulo 9

Presupuesto

9.1 Introducción

A lo largo de este capítulo se va a llevar a cabo una estimación del tiempo necesario para la ejecución y elaboración del proyecto, siendo también incluido los costes asociados a su desarrollo.

9.2 Descripción del entorno de trabajo

El equipo que sería necesario para la consecución del proyecto de ingeniería en los plazos establecidos constaría de los siguientes miembros:

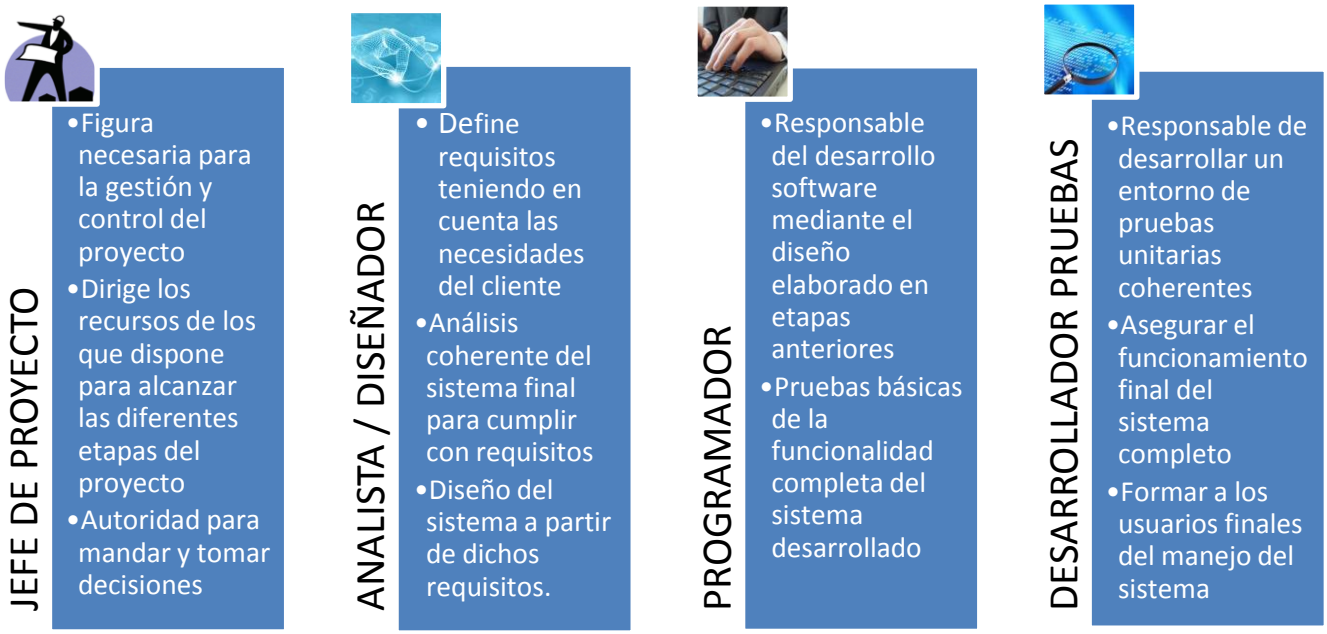


Tabla 6. Representación de los diferentes roles asociada su responsabilidad

9.3 Estimación de recursos

Para proceder con la estimación de recursos es necesario tener en cuenta el coste acarreado por la contratación de personal cualificado empleado en el proyecto como también el coste material empleado durante la consecución del mismo.

9.3.1 Coste de personal cualificado

Debido a que el proyecto a desarrollar precisa de perfiles técnicos es necesario recurrir a personal cualificado. Dentro del proyecto, cada integrante deberá adoptar un rol dependiendo del trabajo que vaya a realizar y del perfil del que disponga. En la tabla Tabla 7 se muestra una estimación del coste de cada miembro del equipo dependiente de su rol, teniendo en cuenta que los costes son importes brutos.

ROL	COSTE/HORA (€)	Nº HORAS	COSTE (€)
Jefe proyecto	100	70	7000
Analista/Diseñador	60	50	3000
Programador	35	150	5250
Desarrollador pruebas	40	60	2400
COSTE TOTAL DEL PERSONAL			17650

Tabla 7. Coste final del equipo

9.3.2 Coste Material

Para el desarrollo del proyecto se precisa tanto software, hardware como otro tipo de materiales. Como en el apartado anterior, se muestra en la tabla Tabla 8 el coste de cada elemento empleado en el diseño, desarrollo y pruebas del sistema completo. Al igual que en el caso anterior, los costes no incluyen I.V.A.

RECURSO MATERIAL	CANTIDAD	COSTE TOTAL (€)
Ordenador portátil	4	2000
Switch 16 puertos (100Mbps)	1	68,63
Cables de red RJ15 (1.6m)	16	28.16
Office Professional 2012 (2 Pcs)	2	1438
Microsoft Project Professional	1	1067
Eclipse software	3	0
Microsoft Windows 7 Professional	4	1263
Conexión ADSL	4 meses	200
COSTE TOTAL DE MATERIALES		6064.29

Tabla 8. ***Coste final de los recursos materiales***

9.3.3 Gastos indirectos

En la tabla Tabla 9 se mostrarán los gastos indirectos que han aumentado el coste final del desarrollo del proyecto. Como en apartados anteriores, los costes no incluirán I.V.A.

DESCRIPCIÓN	COSTE (€)
Limpieza instalaciones	INCLUIDO EN COSTES INDIRECTOS
Alquiler de instalaciones	INCLUIDO EN COSTES INDIRECTOS
Electricidad	INCLUIDO EN COSTES INDIRECTOS
Agua	INCLUIDO EN COSTES INDIRECTOS
Gastos seguridad	INCLUIDO EN COSTES INDIRECTOS
Gastos mantenimiento	INCLUIDO EN COSTES INDIRECTOS
COSTES INDIRECTOS (18%)	$23714.29 * (0.18) = 4268.5722$

Tabla 9. ***Costes indirectos***

9.3.4 Resumen de gastos

El presupuesto que se entregará al cliente se configura con los costes del personal, los costes materiales y los costes indirectos, siendo importante la suma de un porcentaje del margen de imprevistos, los beneficios a obtener y el I.V.A.

DESCRIPCIÓN	COSTE (€)
Costes del personal	17650
Costes materiales	6064.29
Costes indirectos	42685.722
COSTES TOTALES DEL PROYECTO	27982.8622

Tabla 10. ***Costes totales del proyecto***

El margen para los imprevistos que se aplicará al proyecto será del 11%. Por consiguiente, habrá $27982.8622 * (0.11) = 3078.1148$ € de margen para imprevistos.

El beneficio que se espera obtener para el proyecto es de un 30% del coste total más el margen para imprevistos. Por lo tanto, el beneficio obtenido será de **9318,2931 €**.

La suma del coste total del proyecto, el margen para imprevistos y el beneficio hacen una suma total de **40379.2701 €**, aplicando un 21% de IVA, el coste final del proyecto sería de **48858.916 €.**

Leganés a 29 de Octubre de 2012

El ingeniero proyectista

Fdo. Tamara Benito Matías

Glosario

AAF	*Amplified-And-Fordward*
AEP	*Asymptotic Equipartition Property*
AWGN	*Additive white Gaussian noise*
BIAWGN	*Binary Input Adittive White Gaussian Noise*
BEC	*Binary Erasure Channel*
VER	*Bit Error Rate*
BSC	*Binary symmetric channel*
CAF	*Compress-And-Forward*
DB	*DeciBel*
DE	*Density Evolution*
DAF	*Decode-and-Fordward*
DMS	*Discrete Memoryless Source*
ETSI	*Escuela Técnica Superior de Ingeniería*
EXIT	*Extrinsic Information Transfer*
FDP	*Función de Densidad de Probabilidad*
FG	*Factor Graph*
GA	*Gaussian Aproximation*
GF	*Galois Field*
HARQ	*Hybrid Automatic Repeat ReQuest*
IEEE	*Institute of Electrical and Electronics Engineers*
IRA	*Irregular Repeat-Accumulate*
IR-HARQ	*Incremental Redundancy Hybrid ARQ*
ISI	*Intersymbol Interference*
LDPC	*Low Density Parity Check*
LLR	*Log-Likelihood Ratio*
LT	*Luby Transform*
MAC	*Multiple Access Channel*
MAP	*Maximum a posteriori probability*
MPF	*Marginal Product of Functions*
OFDMA	*Orthogonal Frequency-Division Multiple Access*
RA	*Repeat-Accumulate*
SNR	*Signal-to-Noise Ratio*
SP	*Sum-Product*

Referencias

[1] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung. “Network information flow”. IEEE Transactions on Information Theory, July 2000.

[2] P. Herhold, “Cooperative relaying protocols and performances,” PhD thesis at the Technical University of Dresden, July 2005

[3] J. L. Laneman, David N. C. Tse, and G. W. Wornell, “Cooperative diversity in wireless networks: efficient protocols and outage behavior,” IEEE Trans. Information Theory, Vol. 50, No. 12, December 2004.

[4] Peyman Razaghi, “Bilayer Low-Density Parity-Check Codes for Decode-and-Fordward in Reñay Channels”, *IEEE Trans. Inform. Theory*, vol. 53, no. 10, Oct. 2007.

[5] Peyman Razagui and Wei Yu, “Bilayer Low-Density Parity-Check Codes for Decode-and –Forward in Relay Channels”. IEEE Trans. Inform. Theory, vol. 53, pp.1-17, Oct. 2007

[6] C. E. Shannon, “A mathematical theory of communication,” Bell System Technical Journal, vol. 27, pp. 379–423, 623–656, July and October 1948, (Reprinted Claude Elwood Shannon: Collected Papers, pp. 5–83, (N. J. A. Sloane and A. D. Wyner,eds.) Piscataway IEEE Press, 1993).

[7] Gerhard Kramer, “Topics Multi-User Information Theory”. Foundations and Trends in Communications and Information Theory,vol 4. 2008. 9th Chapter

[8] J. Wolfowitz. “The coding of messages subject to chance errors”. Illinois Journal of Mathematics, 1: 591-606, 1957

[9] E. C. Van der Meulen. “Three-terminal communication channels”, Adv. Appl. Prob., vol 3, pp. 120-154, 1971.

[10] ---, “Transmission of information in a T-terminal discrete memoryless channel,” PH. D. dissertation, Dep. Of Statistics, University of California, Berkeley, 1968.

[11] ---, “A survey of multi-way channels in information theory: 1961-1976”, IEEE Trans. Inform. Theory, vol. IT-23, no. 2, January 1977.

[12] H. Sato, “Information transmission through a channel with realy”, The Aloha System, University of Hawaii, Honolulu, Tech. Rep. B76-7, Mar. 1976.

[13] T. Cover and A. A. El Gamal, "Capacity Theorems for the Relay Channel". IEEE Trans. Inform. Theory, vol. IT-25, pp.1-13, Sept. 1979

[14] Piyush Gupta and P. R. Kumar. Towards an information theory of large networks: An achievable rate region. submitted to IEEE Transactions on Information Theory, 2001.

[15] Liang-Liang Xie and P. R. Kumar. A network information theory for wireless communciation: Scaling laws and optimal operation. Submitted to IEEE Transactions on Information Theory, 2002.

[16] T. Cover. Joy A. Thomas "Elements of Information Theory". Wiley series in telecommunications. 2nd Edition.

[17] G. Kramer, M. Gastpar y P. Gupta, "Cooperative strategies and capacity theorems for relay networks", *IEEE Trans. Inf. Theory*, vol. 51, no. 9, pp. 3073-3063, Sept. 2005.

[18] A. Host-Mandsen y J. Zhang, "Capacity bounds and power allocation for wireless relay channel", *IEEE Trans. Inf. Theory*, vol. 51, pp. 2020-2040, Jun. 2005.

[19] Liang-Liang Xie and P. R. Kumar, "An Achievable Rate for the Multiple-Level Relay Channel". IEEE Trans. Inform. Theory, vol. 51, pp.1-11, April 2005

[20] M. Janani, A. Hedayat, T. Hunter, and A. Nosratinia, "Coded cooperation in wireless communications: Space-time transmission and iterative decoding", IEEE Trans. Signal Process., vol. 52, no. 2, pp. 362-371, February 2004

[21] R. U. Nabar, H. Bölcskei y F. W. Kneübunhler, "Fading relay channels: Performace limits and space-time signal design," *IEEE J. Sel. Areas Commun.*, vol. 22, pp. 1099-1109.

[22] A. Sendonaris, E. Erkip y B. Aazhang, "User cooperation diversity – Part I: System description," *IEEE J. Sel. Areas Commun.*, vol. 51, pp. 1927-1938, Nov. 2003.

[23] J. N. Laneman y G. W. Wornell, "Distributed space.time-coded protocols for exploiting cooperative diversity in wireless networks," IEEE J. Sel. Areas Commun., vol. 49, pp. 2415-2425, Oct. 2003.

[24] G. Kramer, "Distributed and layered codes for relaying," in *Conf. Rec. 39th Asilimar Conf. Signals, Syst., Comp.*, Oct. 2005, pp. 1752-1756.

[25] G. Kramer, "Communication strategies and coding for relaying," *in Wireless Communications, IMA Volumes in Mathematics and its Applications*, P. Agrawal, D. M. Andrews, P. J. Fleming, G. Yin, y L. Zhang, Eds. New York: Springer-Verlag, 2007, vol. 143, pp. 163-175.

[26] M. A. Khojastepour, N. Ahmed y B. Aazhang, "Code desing for the relay channel and factor graph decoding," *in Conf. Rec. 38th Asilomar Conf. Signals, Syst., Comp.*, 2004, vol.2, pp. 2000-2004.

[27] C. Li, M. A. Khojastepour, G. Yue, X. Wang y M. Mandihian, "Performance analysis and code desing for cooperative relay channels*," in Proc. 40th Ann. Conf. Inf. Sci., Syst.*, (CISS), 2006.

[28] Z. Zheng and T. M. Duman, "Capacity-approaching turbo coding and iterative decoding for relay channels", IEEE Trans. Commun., vol. 53, no. 11, pp. 1895-1905, Nov. 2005.

[29] T. E. Hunter y A. Nosratinia, "Cooperative diversity through coding," *in Proc. IEEE Int. Symp. Inf. Theory* (ISIT), Jul. 2002, p.220.

[30] B. Zhao y M. C. Valenti, "Distributed turbo coded diversity for relay channel," *Electronics Letters*, vol. 39, pp. 786-787, May 2003.

[31] A. Chakrabarti, A. de Baynast, A. Sabharwal, and B. Aazhang, "Low density parity check codes for the relay channel," *IEEE J. Select. Areas Commun.*, vol. 25, pp. 280–291, Feb. 2007.

[32] J. Ezri and M. Gastpar, "On the performance of independently designed LDPC codes for the relay channel," *in Proc. IEEE Int. Symp. Inf. Theory* (ISIT), 2006, pp. 977–981.

[33] J. Hu and T. M. Duman, "Low density parity check codes over halfduplex relay channels," *in Proc. IEEE Int. Symp. Inf. Theory* (ISIT), 2006, pp. 972–976.

[34] J. Hu and T. M. Duman, "Low density parity check codes over wireless relay channels," *IEEE Trans. Wireless Commun.*, 2007.

[35] G. Kramer, M. Gastpar y P. Gupta, "Cooperative strategies and capacity theorems for relay networks", *IEEE Trans. Inf. Theory*, vol. 51, no. 9, pp. 3073-3063, Sept. 2005.

[36] M. Luby, "LT codes," *in Proc. The 43rd Annual IEEE Symposium on Foundations of Computer Science*, Nov. 2002, pp. 271-280.

[37] A. Shokrollahi, "Raptor codes," *IEEE Trans. Inf. Theory*, vol. 52, no. 5, pp. 2551–2567, Jun. 2006.

[38] O. Etesami and A. Shokrollahi, "Raptor codes on binary memoryless symmetric channels*," IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2033–2051, Jun. 2006.

[39] M. R. Yazdani and A. H. Banihashemi, "On construction of rate-compatible low-density parity-check codes," *IEEE Commun. Lett.*, vol. 8, no. 3, pp. 159–161, Mar. 2004.

[40] J. Kim, W. Hur, A. Ramamoorthy, and S. Mclaughlin, "Design of rate compatible irregular LDPC codes for incremental redundancy hybrid ARQ systems," *in Proc. IEEE Int. Symp. Inf. Theory* (ISIT), Jul. 2006, pp. 1139–1143.

[41] J. Kim, A. Ramamoorthy, and S. Mclaughlin, "Design of efficiently encodable rate-compatible irregular LDPC codes," *in Proc. IEEE Int. Commun. Conf.* (ICC), Jun. 2006, vol. 3, pp. 1131–1136.

[42] J. Ha, J. Kim, and S. W. McLaughlin, “Rate-compatible puncturing of low-density parity-check codes,” *IEEE Trans. Inf. Theory*, vol. 50, no. 11, pp. 2824–2836, Nov. 2004.

[43] E. Soljanin, N. Varnica, and P. Whiting, “Incremental redundancy hybrid ARQ with LDPC and raptor codes,” *in Proc. IEEE Int. Symp. Information Theory*, Adelaide, Australia, Sep. 2005, pp. 995–999.

[44] B. Zhao and M. C. Valenti, “Practical relay networks: A generalization of hybrid-ARQ,” *IEEE J. Sel. Areas Commun.*, vol. 23, no. 1, pp. 7–18, Jan. 2005.

[45] S. Sesia, G. Caire, and G. Vivier, “Incremental redundancy hybrid ARQ schemes based on low-density parity-check codes,” IEEE Trans. Commun., vol. 47, no. 8, pp. 1311–1321, Aug. 2004.

[46] T. M. Cover ay S. K. Leung-Yan-Cheong. “A rate region for the multiple-access channel with feedback” submitted to IEEE Trans. Inform. Theory, 1976.

[47] T. J. Richardson y R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 599-618, Feb. 2001.

[48] Gerhard Kramer, Michel Gastpar and Piyush Gupta, “Cooperative Strategies and Capacity Theory for Relay Networks”. IEEE Trans. Inform. Theory, vol. 51, pp.1-27, Sept. 2005

[49] A. Wyner, “Recent results in the Shannon theory,” *IEEE Trans. Inf. Theory*, vol. 20, no. 1, pp. 2–10, Jan. 1974.

[50] S. S. Pradhan and K. Ramchandran, “Distributed source coding using syndromes (DISCUS): Design and construction,” *IEEE Trans. Inf. Theory*, vol. 49, no. 3, pp. 626–643, Mar. 2003.

[51] N. Wiberg, H.-A. Loeliger, and R. Kötter, “Codes and iterative decoding on general graphs,” Eur. Trans. Telecomm., vol. 6, pp. 513–525, Sept. / Oct. 1995.

[52] J. C. Willems, “Models for Dynamics,” in Dynamics Reported, Volume 2, U. Kirchgraber and H. O.Walther, Eds. New York:Wiley, 1989, pp. 171–269.

[53] R. M. Tanner. "A recursive approach to low complexity codes," *IEEE Trans, Inform. Theory*, vol. 27, no. 5, pp. 533-547, Sept. 1981.

[54] R. G. Gallager. “Low-Density Parity-Check Codes”. Cambridge, MA: M.I.T. Press, 1963

[55] -----. “The generalized distributive law,” IEEE Trans. Inform. Theory, vol. 46, pp. 325–343, Mar. 2000.

[56] V. Isham, “An introduction to spatial point processes and Markov random fields,” Int. Stat. Rev., vol. 49, pp. 21–43, 1981.

[57] R. Kindermann and J. L. Snell, Markov Random Fields and Their Applications. Providence, RI: Amer. Math. Soc., 1980.

[58] C. J. Preston, Gibbs States on Countable Sets. Cambridge, U.K.: Cambridge Univ. Press, 1974.

[59] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible*. Inference. San Mateo, CA: Morgan Kaufmann Publishers, 1988.

[60] F. V. Jensen, An Introduction to Bayesian Networks. New York: Springer-Verlag, 1996.

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

[62] B. J. Frey, Graphical Models for Machine Learning and Digital Communication. Cambridge, MA: MIT Press, 1998.

[63] F. R. Kschischang y B. J. Frey, "Iterative decoding of compound codes by probability propagation in graphical models," *IEEE J. Select. Areas Commun.*, vol. 16, no 2, pp. 219-230, Feb.1998

[64] D. J. C. MacKay and R. M. Neal, "Good codes based on very sparse matrices," in Cryptography and Coding. 5th IMA Conference (Lecture Notes in Computer Science), C. Boyd, Ed. Berlin, Germany: Springer,1995, vol. 1025, pp. 100–111.

[65] D.J.C MacKay. "Good error-correcting codes based on very sparse matrices", IEEE Trans. Inform. Theory, vol. 45, pp. 399-431, March 1999

[66] R. J. McEliece, D. J. C. Mackay. Y J.-F. Cheng, "Turbo decoding as an instance of Pearl´s belief propagation algorithm," *IEEE J. Select. Areas Commun.*, vol16, no 2, pp. 140-152, Feb. 1998.

[67] G. D. Forney Jr., "On iterative decoding and the two-way algorithm," in Proc. Int. Symp. Turbo Codes and Related Topics, Brest, France, Sept. 1997.

[68] Bravo Santos, Ángel. Técnicas de Tratamiento de Señal en Codificación de Canal. Abril 2010

[69] N. Wiberg. "Codes and decoding on general graphs," Ph. D. dissertation, Linköping University, Suiza, 1996. [Link]. Disponible: http://www.it.isy.liu.se/publikationer/LIU-TEK-THESIS-440.pDAF

[70] R. L. Graham, D. E. Knuth, and O. Patashnik, Concrete Mathematics. Reading, MA: Addison-Wesley, 1989.

[71] C. Berrou, A. Glavieux, y P. Thitimajshima. "Near Shannon limit error-correcting coding and decoding: Turbo codes," en Proc. *IEEE International Conferenence on Comunications*, Ginebra, Suiza, 1993, pp. 1064-1070.

[72] Masoud Ardakani, "Eficient Analysis, Design and Decoding of Low-Density Parity-Check Codes", PH. D. dissertation, Dep. Of Electrical and Computer Engineering, University of Toronto.

[73] T. Richardson and R.Urbanke. “Modern coding Theory”. Cambrige University Press. Oct. 2007. pp.382-426.

[74] M. Sipser y D. A. Spielman, "Expander codes," *IEEE Trans. Inform. Theory*, vol 42, no. 6, pp. 1710-1722, Nov. 1996.

[75] T. J. Richardson, M. A. Shokrollahi, y R. L. Urbanke. "Design of capacity-approaching irregular low-density parity-check codes," IEEE Trans. Inform. Theory, vol. 47, no. 2, pp. 619-637, Feb. 2001.

[76] D. Divsalar, H. Jin, R. J. McEliece, "Coding theorems for 'turbo-like' codes," in *Proc. 36th Allerton Conference on Communications, Control, and Computing*, Monticello, Illinois, 1998, pp. 201-210.

[77] L. Ping y K. Y. Wu, "Concatenated tree codes: a low-complexity high-performance approach," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 791-799, Feb. 2001.

[78] B. J. Frey y D. J. C. Mackay, "Irregular turbo codes," *in Proc. 37th Allerton Conference on Communications, Control, and Computing*, Allerton House, Illinois, 1999.

[79] H. Jin, A. Khandekar y R. McEliece, "Irregular repeat-accumulate codes," in *Proc. 2nd International Symposium on Turbo Codes and Related Topics*, Brest, France, 2000, pp. 1-8.

[80] A. Shokrollahi, "New Sequence of linear time erasure codes approaching the channel capacity," in Proc. The iInternational Symposium onm Applied Algebra, Algebraic Algorithms and Error-Correcting Codes, no. 1719, Lecture Notes in Computer Science, 1999, pp. 65-67.

[81] --, "Capacity-achieving sequences," *IMA Volumes in Mathematics and its Applications*, vol. 123, pp.153-166, 2000.

[82] H. El Gamal y A. R. Hammons, "Analyzing the turbo codecoder using the Gaussian approximation," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 671-686. Feb. 2001.

[83] P. Rusmevichientong y B. V. Roy, "An analysis of belief propagation on the turbo decoding graph with gaussian densities," IEEE Trans. Inform. Theory, vol. 47, no. 2, pp. 745-765, Feb. 2001

[84] S.-Y. Chung, T. J. Richardson, y R. L. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian aproximation," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 657-670, Feb. 2001.

[85] *Digital Video Broadcasting (DVB)*, European Standard (Telecommunications series) Std. ETSI EN 302 307 V1.1.1, 2004

[86] *Optional B-LDPC coding for OFDMA PHY*, IEEE Std. IEEE 802.16e-04/78, 2004.

[87] F. R. Kschischang, B. J. Frey, y H.-A. Loeliger, "Factor graphs and the sum-product algorithm," IEEE Trans. Inform. Theory, vol. 47, no. 2, pp. 498-519.

[88] S. M. Aji, G. B. Horn, y R. J. McEliece, "Iterative decoding on graphs whith a single cyle", *in Proc. IEEE International Symposium on Information Theory*, 1998, p. 276.

[89] Y. Weiss y W. T. Freeman, "On the optimality of solutions of the max-product belief- propagation algorithm in arbitrary graphs," *IEEE Trans. Inform*. Theory, vol. 47, no. 2, pp.736-744, Feb. 2001.

[90] B. J. Frey y D. J. C. Mackay, "A revolution: Belief propagation in graphs with cycles," in *Proc. Conference on Advances in Neural Informaction Processing Systems 10*, Denver, Colorado, Julio 1998, pp. 479-485.

[91] J. Yedidia, W. T. Freeman, y Y. Weiss, "Understanding belief propagation and its generalizations," Mitsubishi Electrie Research Laboratories, Tech. Rep. TR2001-22, Nov. 2001.

[92] F. R. Kschischang, "Codes defined on graphs," IEEE Commun. Mag., vol. 41, no. 8, pp. 118-125, Agosto 2003.

[93] L. C. Perez, J. Seghers, y D. J. Costello Jr., "A distance spectrum interpretation of turbo codes", *IEEE Trans. Inform. Theory*, vol. 42, no. 6, pp. 1698-1709, Nov. 1996.

[94] D.J.C MacKay and R.M. Neal. "Near Shannon limit performance of low density parity check codes", IEEE Electronics Letters, vol. 32, nº 18, pp. 1645-1655, 29th August 1996.

[95] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman, and V. Stemann, "Practical loss-resilient codes," in Proc. 29th ACM Symp. Theory of Computing (STOC), 1997, pp. 150–159.

[96] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman, "Efficient erasure correcting codes," IEEE Trans. Inform. Theory, vol. 47, pp. 569–584, Feb. 2001.

[97] N. Varnica and A. Kav˘cic´, "Optimized LDPC codes for partial response channels," in Proc. IEEE Int. Symp. Information Theory (ISIT 2002), Lausanne, Switzerland, June/July 2002, p. 197.

[98] X. Ma, N. Varnica, and A. Kav˘cic´, "Matched information rate codes for binary ISI channels," in Proc. IEEE Int. Symp. Information Theory (ISIT 2002), Lausanne, Switzerland, June/July 2002, p. 269.

[99] B. M. Kurkoski, P. H. Siegel, and J. K. Wolf, "Joint message-passing decoding of LDPC codes and partial-response channels," IEEE Trans. Inform. Theory, vol. 48, pp. 1410–1422, June 2002.

[100] M. G. Luby, M.Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Analysis of low density codes and improved designs using irregular graphs," *in Proc. 30th Ann. ACM Symp. Theory of Computing*, 1998, pp. 249–258.

[101] S. ten Brink, “Convergence behavior of iteratively decoded parallel concatenated codes,” *IEEE Trans. Commun.*, vol. 49, no. 10, pp. 1727–1737, Oct. 2001.

[102] S. ten Brink, “Designing iterative decoding schemes with the extrinsic information transfer chart,” AEÜ Int. J. Electron. Commun., vol. 54, no. 6, pp. 389–398, Dec. 2000.

[103] J. Boutros and G. Caire, “Iterative multiuser joint decoding: Unified framework and asymptotic analysis,” IEEE Trans. Inform. Theory, vol. 48, pp. 1772–1793, July 2002.

[104] F. Lehmann and G. M. Maggio, “An approximate analytical model of the message passing decoder of LDPC codes,” in Proc. IEEE Int. Symp. Information Theory (ISIT 2002), Lausanne, Switzerland, June/July 2002, p. 31.

[105] M. Ardakani and F. R. Kschischang, “Designing irregular LPDC codes using exit charts based on message error rate,” in Proc. IEEE Int. Symp. Information Theory (ISIT 2002), Lausanne, Switzerland, June/July 2002, p. 454.

[106] J. Boutros, G. Caire, E. Viterbo, H. Sawaya, and S. Vialle, “Turbo code at 0.03 dB from capacity limit,” in Proc. IEEE Int. Symp. Information Theory (ISIT 2002), Lausanne, Switzerland, June/July 2002, p. 56.

[107] H. Jin, “Analysis and design of turbo-like codes,” Ph.D. dissertation, Calif. Inst. Technol., Pasadena, 2001.

[108] H. Jin, Personal communication, Jan. 2006.

[109] Yifei Zhang and William E. Ryan, “Structured IRA Codes: Performance Analysis and Construction”.

[110] K. H. Rosen, Discrete Mathematics and its Applications, 4th ed. New York: WCB/McGraw-Hill, 1999.

[111] G. Dueck. “The capacity region of the two-way channel can exceed the inner bound”. Inform. Contr., 40: 258-266, 1979.

[112] L. Devroye, “Non-Uniform Random Variable Generation”. New York. 1986.

[113] M. Gastpar and M. Vetterli, “On the Capacity of Wireless Networks: the Relay Case,” in IEEE INFOCOM, vol. 3, pp. 1577-1586, 2002.

[114] A. Shokrollahi, “Capacity-achieving sequences,” *in Codes, Systems, and Graphical Models: IMA Volumes in Mathematics and its Applications*, B. Marcus and J. Rosenthal, Eds. New York: Springer-Verlag, 2000, vol. 123, pp. 153–166.

[115] A. Shokrollahi, “New sequences of linear time erasure codes approaching the channel capacity,” *in Proc. 13th Int. Symp. Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, 1999, pp. 65–76.

[116] P. Oswald and A. Shokrollahi, “Capacity-achieving sequences for the erasure channel,” *IEEE Trans. Inf. Theory*, vol. 48, no. 2, pp. 2017–3028, Dec. 2002.

[117] A. Roumy, S. Guemghar, G. Caire, and S. Verdu, “Design methods for irregular repeat-accumulate codes,” IEEE Trans. Inform. Theory, vol. 50, no. 8, pp. 1711–1727, August 2004.