



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE
TELECOMUNICACIÓN

INGENIERÍA DE TELECOMUNICACIÓN-ITIS

PROYECTO FIN DE CARRERA

ALGORITMO DE ENCAMINAMIENTO JERÁRQUICO PARA WSN
EXTENSAS BASADO EN CODIFICACIÓN DE RED Y LDPC

Autor: Juan Carlos Vázquez Arranz

Tutor: Eduardo Morgado Reyes

Curso académico 2011/2012

TÍTULO: ALGORITMO DE ENCAMINAMIENTO JERÁRQUICO PARA
WSN EXTENSAS BASADO EN CODIFICACIÓN DE RED Y
LDPC

AUTOR: JUAN CARLOS VÁZQUEZ ARRANZ

TUTOR: EDUARDO MORGADO REYES

La defensa del presente Proyecto Fin de Carrera se realizó el día
siendo calificada por el siguiente tribunal:

PRESIDENTE:

SECRETARIO:

VOCAL:

Habiendo obtenido la siguiente calificación:

CALIFICACIÓN:

Presidente

Secretario

Vocal

“Our virtues and our failings are inseparable, like force and matter.
When they separate, man is no more. ”

Nikola Tesla

Agradecimientos

En primer lugar quiero agradecer a la Universidad Rey Juan Carlos la oportunidad que me ha brindado durante estos años y que ha ayudado a formarme académicamente y como persona. Además, agradezco de una manera especial a mi tutor, Eduardo Morgado Reyes, por los consejos que me ha dado en estos últimos años y porque sin su apoyo, dedicación y sobre todo paciencia no habría podido nunca realizar este Proyecto. También quiero dedicar unas palabras a los usuarios con los que he compartido el cluster de CUDA, ya que saben como yo lo que puede llegar a tardar una simulación y lo rápido que puede llegar a ir ese servidor.

No pueden faltar en estas líneas mis padres, que me han ayudado y me han dado su apoyo en todo lo que he necesitado. Soy lo que soy gracias a vosotros.

También se merecen una mención especial mis compañeros de fátiga a lo largo de estos últimos años, en especial Ángela por estar siempre a mi lado, soportándome y sabiendo cómo animarme en todo momento. Borja, Cervi, Elena, Davor, Lore, David, Alberto, María y otros tantos compañeros a los que podría nombrar, os agradezco los buenos e inolvidables momentos que hemos pasado tanto en clase como fuera de ella. Finalmente también quiero acordarme de Raúl, por ayudarme en mi carrera profesional, y de Luis, por apoyarme desde el instituto.

Resumen

En los últimos años, ha habido un fuerte desarrollo de dispositivos que usan sensores para la monitorización y, gracias a ello, cada vez los nodos sensores son más pequeños y baratos. Esto ha propiciado la aparición de WSN (*Wireless Sensor Network*; Redes de Sensores Inalámbricos) extensas, que pueden englobar un alto número de sensores que recogen información y la comunican de manera inalámbrica hasta un nodo central. Uno de los problemas que surgen en este tipo de redes es que los algoritmos de encaminamiento, que permiten llevar la información desde el nodo origen al nodo destino, no pueden evitar que aumente la tasa de error a medida que aumenta el número de saltos necesarios para alcanzar el nodo destino.

El propósito de este Proyecto Fin de Carrera consiste en proponer un algoritmo jerárquico que busque reducir la BER (*Bit Error Rate*; Probabilidad de Error de Bit) en WSN densas y extensas y estudiar sus prestaciones. Para ello, se va a emplear una estructura de red de tipo árbol utilizando dos niveles, reduciendo así el número de saltos a dos. Además, en ambos niveles se va a utilizar una técnica de codificación de canal distribuida basada en Codificación de Red con códigos LDPC (*Low Density Parity Check codes*; códigos de Chequeo de Paridad de Baja Densidad). Una vez propuesto el algoritmo, se analizan sus prestaciones a partir de la obtención, mediante simulaciones, de las curvas BER frente a SNR (*Signal to Noise Ratio*; Relación Señal a Ruido). Dichas curvas se obtienen para una serie de escenarios posibles, simulados por medio de la herramienta *MATLAB*, modificando los valores de una serie de parámetros y estudiando cómo afectan a dichas curvas, con el objetivo de obtener valores adecuados para los distintos escenarios.

Las prestaciones del algoritmo propuesto se comparan con el caso sin codificar y con otra alternativa de Codificación de Red y códigos LDPC en escenarios multisalto que fue estudiada previamente en el Departamento de Teoría de la Señal y Comunicaciones de la Universidad Rey Juan Carlos. Los resultados de las simulaciones permiten comprobar que el algoritmo propuesto en este Proyecto ofrece una importante ganancia de codificación respecto a los citados casos, consiguiendo una reducción significativa de la BER extremo a extremo en la WSN.

Índice general

Agradecimientos	I
Resumen	III
1. Introducción	1
1.1. Motivación	1
1.2. Objetivos	6
1.3. Materiales utilizados	7
1.4. Planificación del Proyecto	8
1.5. Estructura de la memoria	10
2. Fundamentos de las Técnicas Empleadas	11
2.1. Codificación de Red	11
2.2. Códigos LDPC	15
3. Descripción del algoritmo propuesto	19
3.1. Descripción del escenario y del algoritmo de base	20
3.1.1. Codificación LDPC	23
3.1.2. Decodificación LDPC: Algoritmo Suma-Producto	25
3.1.3. Ejemplo de Codificación de Red con LDPC	28
3.2. Algoritmo jerárquico basado en Codificación de Red con LDPC	34
4. Resultados	43
4.1. Parámetros que afectan a nuestro sistema	45
4.1.1. Influencia del número de iteraciones	45
4.1.2. Influencia del parámetro d	48
4.1.3. Influencia del número de nodos de la red	49
4.2. Análisis del Primer Nivel	51
4.2.1. Agrupaciones de 50 nodos	52
4.2.2. Agrupaciones de 100 nodos	53
4.2.3. Agrupaciones de 200 nodos	54
4.2.4. Agrupaciones de 300 nodos	55
4.2.5. Agrupaciones de 500 nodos	56

4.2.6. Mejores opciones para el primer nivel	57
4.3. Análisis del Segundo Nivel	59
4.3.1. Caso ideal en el Segundo Nivel	61
4.3.2. Segundo Nivel: 5 clusters	62
4.3.3. Segundo Nivel: 15 clusters	64
4.4. Análisis del sistema completo	66
5. Conclusiones y trabajos futuros	73
5.1. Conclusiones	73
5.2. Lineas Futuras	75
Lista de Acrónimos	77
Bibliografía	81

Índice de figuras

1.1. Ilustración de Marconi junto a la primera radio	2
1.2. Ejemplo de una red mallada de sensores	3
1.3. Estructura de un nodo de una WSN.	4
1.4. Topologías de red en una WSN	5
1.5. Interfaz de MATLAB.	7
1.6. Diagrama de Gantt con la planificación del Proyecto.	10
2.1. Ejemplo de nodo de retransmisión.	12
2.2. Ejemplo de Codificación de Red aprovechando las capacidades del canal.	14
2.3. Modelos de red de retransmisión	14
2.4. Ejemplo de grafo bipartido asociado a una matriz de chequeo H	17
3.1. Ejemplo de fase de envío con 5 nodos.	29
3.2. Ejemplo de fase de reenvío para 5 elementos y $d=2$	32
3.3. BER extremo a extremo para un radio de cobertura de $0.9R_0$	35
3.4. Ganancias de codificación para diferentes radios de cobertura	36
4.1. Regiones de la curva BER-SNR para un código LDPC	44
4.2. Curva BER-SNR para distintas iteraciones del algoritmo de decodificación	46
4.3. Tiempo de decodificación para distintas iteraciones	47
4.4. Curva BER-SNR para distintos valores del <i>parámetro</i> d con 200 nodos	49
4.5. Curva BER-SNR para diferentes conjuntos de nodos	50
4.6. Curva BER-SNR para distintos valores de d en agrupaciones de 50 nodos	53
4.7. Curva BER-SNR para distintos valores de d en agrupaciones de 100 nodos	54
4.8. Curva BER-SNR para distintos valores de d en agrupaciones de 200 nodos	55
4.9. Curva BER-SNR para distintos valores de d en agrupaciones de 300 nodos	56
4.10. Curva BER-SNR para distintos valores de d en agrupaciones de 500 nodos	57
4.11. Curva BER-SNR para los mejores resultados del primer nivel	58
4.12. Caso ideal segundo nivel	61
4.13. Segundo Nivel 5 clusters con $d=12$	63
4.14. Segundo Nivel 5 clusters con $d=13$	63
4.15. Segundo Nivel 15 clusters con $d=12$	65
4.16. Segundo Nivel 15 clusters con $d=13$	65

4.17. Análisis completo para 5 clusters	67
4.18. Análisis completo para 15 clusters	68
4.19. Análisis comparativo para 5 clusters con y sin codificación	69
4.20. Análisis comparativo para 15 clusters con y sin codificación	69

Índice de cuadros

3.1. Ejemplo de símbolos recibidos correctamente en el segundo nivel.	39
3.2. Ejemplo de respuestas en la fase de reenvío en el segundo nivel.	40
3.3. Formación de la matriz de paridad P para el segundo nivel.	41
4.1. Grados de dispersión para $d=6$ y distinto número de nodos.	51
4.2. Grados de dispersión para 50 nodos y distinto valores del <i>parámetro</i> d . . .	52
4.3. Grados de dispersión para 100 nodos y distinto valores del <i>parámetro</i> d . . .	54
4.4. Grados de dispersión para 200 nodos y distinto valores del <i>parámetro</i> d . . .	55
4.5. Grados de dispersión para 300 nodos y distinto valores del <i>parámetro</i> d . . .	56
4.6. Grados de dispersión para 500 nodos y distinto valores del <i>parámetro</i> d . . .	57
4.7. Grados de dispersión para las mejores curvas del primer nivel.	58

Capítulo 1

Introducción

El presente capítulo supone una introducción al trabajo realizado en este Proyecto Fin de Carrera. Este capítulo se estructura en cinco secciones: En la Sección 1.1 se va a explicar en qué consiste una red inalámbrica de sensores. Además, se introduce la motivación que lleva a elaborar el presente Proyecto Fin de Carrera. En la Sección 1.2 se tratan los objetivos que va a cumplir este trabajo, respondiendo a las necesidades expuestas en la primera sección de este capítulo. En la Sección 1.3 se detallan las herramientas utilizadas que han facilitado la realización de este Proyecto. La Sección 1.4 expone la planificación que se ha llevado a cabo para poder realizar este trabajo. Finalmente, en la Sección 1.5 se resumen los contenidos que van a tratarse en el resto de capítulos de este trabajo.

1.1. Motivación

Cada día se vuelve más habitual el uso de tecnologías inalámbricas en todos los ámbitos, desplazando a los tradicionales cables. El propio Nikola Tesla fue uno de los precursores de las tecnologías inalámbricas cuando en 1893 realizó una demostración de transmisión inalámbrica en la Convención de la Asociación Nacional de Alumbrado Eléctrico de St. Louis [Cheney, 2001]. Pocos años después, en 1896, Guglielmo Marconi registraba en Londres la primera patente de un telégrafo inalámbrico, utilizando para ello varias patentes del propio Tesla. Desde entonces, los avances en el campo de las comunicaciones inalámbricas han sido numerosos, especialmente a partir de los años 90. La evolución de los equipos de transmisión y recepción han permitido que se redujera su tamaño y su precio, convirtiéndolos en un reclamo comercial. Los usuarios pronto se dieron cuenta de la gran comodidad que ofrecía esta nueva tecnología para sustituir o incluso complementar a la ya implantada tecnología cableada. Esto hizo que numerosas empresas vieran una gran oportunidad de negocio e invirtieran en este sector, permitiendo el gran



Figura 1.1: Ilustración de Marconi junto a la primera radio [Westerop, 2010].

desarrollo que tenemos hoy en día.

Una red inalámbrica es un conjunto de nodos o elementos comunicados entre sí por ondas de radio. Los primeros trabajos publicados sobre redes inalámbricas tratan de 1979, cuando IBM publicó los resultados de un experimento realizado en Suiza en el que se pretendía establecer en una fábrica una red local conectada por medio de tecnología infrarroja [Gfeller and Bapst, 1979]. Desde entonces, estas redes han evolucionado hasta llegar a implantarse en nuestros hogares. Actualmente, las redes inalámbricas son muy diversas, y abarcan desde conexiones entre pocos dispositivos, como puede hacerse con conexiones Bluetooth, hasta las grandes redes de telefonía móvil.

En cuanto a su topología, las redes inalámbricas presentan topologías diferentes a las de las redes cableadas. Este hecho se debe a que el medio en el que se propagan las ondas de radio es el propio aire y, por lo tanto, es un medio compartido y con distintas características que el cable. Por ello, para una comunicación inalámbrica únicamente son necesarios dos dispositivos que puedan emitir y recibir ondas de radio.

Las WSN (*Wireless Sensor Network*; Redes de Sensores Inalámbricos) son un tipo de red compuesta por una cantidad variable de elementos (denominados nodos) que están equipados con sensores y que envían información de manera inalámbrica. El gran desarrollo de la tecnología durante las últimas décadas ha permitido que los circuitos electrónicos sean cada vez más pequeños y más baratos y esto ha propiciado que estas redes estén actualmente en pleno auge. Tal es su importancia que el MIT (*Massachusetts Institute of Technology*; Instituto de Tecnología de Massachusetts) publicó una lista en Febrero de 2003 sobre las 10 tecnologías emergentes que cambiarían el mundo y las WSN ocupaban el primer lugar [Werff, 2003].

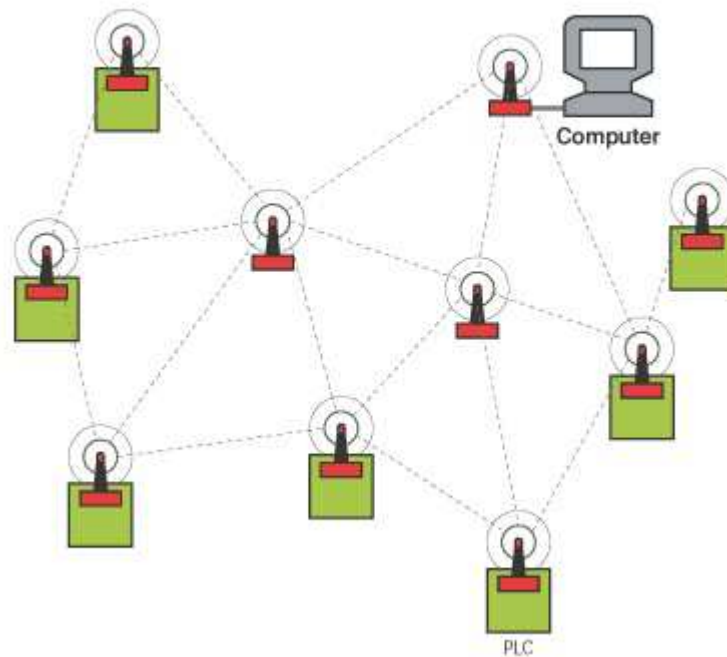


Figura 1.2: Ejemplo de una red mallada de sensores [Poor, 2003].

El principal uso de las WSN es el de la monitorización, ya que los nodos que las componen se encuentran equipados con sensores. Uno de los mayores logros de estas redes es la sencillez de los nodos que la integran y que permite desplegar redes en lugares remotos que integren a un gran número de nodos por un precio razonablemente bajo. Por lo general, estos nodos requieren poca capacidad de procesamiento, ya que el procesamiento se suele realizar en otro equipo preparado para este fin. Al realizarse pocos cálculos dentro del nodo se reduce el consumo de energía del mismo, lo que convierte a estos equipos en ideales para su funcionamiento en lugares remotos o de difícil acceso a la red eléctrica y permite su utilización en dispositivos portables que requieran de una cierta autonomía.

Cada nodo de los que conforman una WSN suele estar compuesto de los siguientes elementos, como se aprecia en la Figura 1.3:

- Una unidad de sensado, encargada de convertir una magnitud física o química (normalmente se trata de un parámetro ambiental) en un impulso eléctrico.
- Un microcontrolador, que dirige el funcionamiento del nodo y realiza un preprocesado de los datos adquiridos. Incorporan una pequeña memoria y suelen estar programados en lenguajes de bajo nivel que optimizan sus recursos.
- Un transmisor/receptor radio, que es capaz de emitir o recibir una onda de radiofrecuencia. Actualmente, los protocolos de comunicación más utilizados son ZigBee

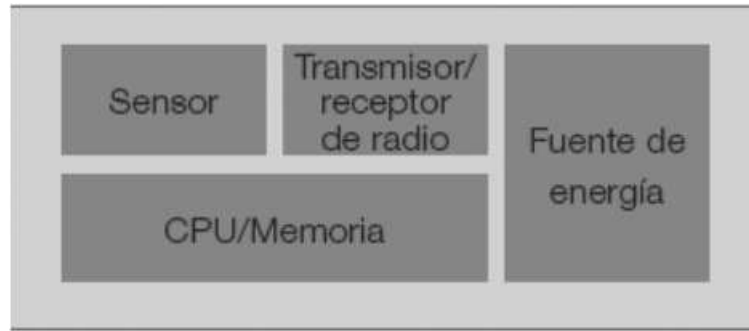


Figura 1.3: Estructura de un nodo de una WSN.

y Bluetooth y permiten una velocidad de transmisión cercana a 1 Mbps.

- Una batería, que proporciona energía a los distintos elementos del nodo. Normalmente se trata de pequeñas pilas y se busca optimizar su consumo en función de su aplicación.

Las redes de sensores inalámbricos pueden localizarse a lo largo de áreas muy extensas, pudiendo incluir un gran número de nodos. Esto provoca que muchos nodos no tengan conexión directa con la unidad encargada de procesar los datos, por lo que se suelen utilizar algoritmos de encaminamiento multisalto para garantizar que la información pueda llegar desde un nodo origen a un nodo destino. En estos algoritmos se establecen una serie de rutas en las que los nodos de la red actúan como repetidores, replicando la información que reciben con el fin de que la información pueda alcanzar el nodo destino. El problema de este tipo de algoritmos es que según aumenta el número de saltos necesarios para alcanzar el destino aumenta también la probabilidad de que los datos que se envíen lleguen erróneos a su destino [Morgado et al., 2010], lo que comúnmente se conoce como BER (*Bit Error Rate*; Probabilidad de Error de Bit) si se miden los bits que llegan erróneos, o SER (*Symbol Error Rate*; Probabilidad de Error de Símbolo) si se hace lo mismo con los símbolos transmitidos.

Actualmente, en las WSN predominan tres tipos de topologías [Instruments, 2009] en función de cómo enlazan unos nodos con otros: la topología de estrella, la topología de cluster y la topología mallada.

- En la topología de estrella (en inglés *star*) todos los nodos tienen conexión directa con el nodo destino al que va dirigida la información recogida por la red de sensores.
- La topología de cluster o también llamada de árbol (en inglés *cluster/tree*), es un tipo de topología jerárquica en el que la red se divide en pequeños clusters o agrupaciones de nodos, en las que todos los nodos de ese cluster transmiten la información a un nodo raíz o cabeza de cluster. Posteriormente, este nodo retransmite la información

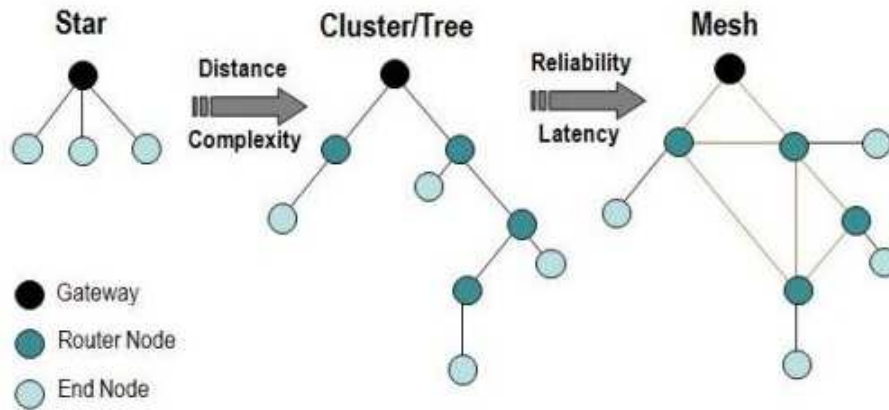


Figura 1.4: Topologías de red en una WSN [Instruments, 2009].

a su nodo raíz y así hasta llegar a la unidad de control.

- En la topología mallada (en inglés *mesh*) se aprovechan las conexiones entre los nodos de la red para llegar a la unidad de control, pudiéndose elaborar distintos caminos en función de la disposición de los nodos de la red en el momento de establecer la comunicación. Esta topología es más robusta que la topología de cluster, ya que todos los nodos tienen la misma importancia y el fallo de un nodo no afecta al funcionamiento del resto de nodos, pero también es más compleja de gestionar y puede incrementar la latencia en las transmisiones.

El estudio de las WSN es una de las distintas líneas de investigación que se están llevando a cabo en el departamento de Teoría de la Señal y Comunicaciones de la Universidad Rey Juan Carlos. Debido a la importancia que han cobrado las tecnologías inalámbricas en los últimos años que ha surgido un grupo en el propio departamento denominado CECI (Centro Experimental de Comunicaciones Inalámbricas) y cuyo objetivo es el análisis de los sistemas actuales y la creación de nuevos servicios y tecnologías para comunicaciones inalámbricas.

El trabajo desarrollado en el presente Proyecto se enmarca dentro de los trabajos de investigación del Departamento sobre las prestaciones en redes ad hoc inalámbricas y, más concretamente, en WSN. En investigaciones anteriores [Morgado, 2009] se exploró la combinación de técnicas de Codificación de Red con códigos LDPC (*Low Density Parity Check codes*; códigos de Chequeo de Paridad con Baja Densidad) con el objetivo de reducir la BER en WSN extensas y densas que engloben un gran número de nodos. Dichas investigaciones demostraron que la mejoría de prestaciones desaparecía al incrementarse las rutas multisalto dentro de la WSN, por lo que este Proyecto nace con la idea de investigar la jerarquización de la WSN como alternativa para poder utilizar Codificación de Red y códigos LDPC distribuidos de forma que se reduzca la BER extremo a extremo dentro de la WSN. Se va a proponer un algoritmo basado en una jerarquización de dos

niveles y se van a evaluar las prestaciones de dicho algoritmo por medio de simulaciones que modelan el comportamiento del algoritmo y el escenario a analizar.

La motivación que ha llevado al proyectante a elegir el presente Proyecto ha sido su interés por trabajar en el campo de las WSN, un campo en alza en el que se permitía aplicar los conocimientos adquiridos a lo largo de la carrera en el campo de la teoría de la señal, que es el que mejor se adapta a los gustos del proyectante. De esta manera, el proyectante podía participar en un Proyecto de investigación, familiarizándose con este entorno y aprendiendo técnicas novedosas, como puede ser la Codificación de Red.

1.2. Objetivos

Este Proyecto supone la continuación del trabajo realizado en [Morgado, 2009]. El objetivo de este Proyecto es el de presentar un algoritmo de encaminamiento jerárquico que busque mejorar la BER en una red WSN extensa y densa. Para ello se propone un algoritmo basado en dos niveles de jerarquía, dividiendo los nodos en agrupaciones que engloben un número fijo de nodos de manera que los nodos puedan establecer vecindades y que tenga un enlace directo con un nodo cabeza de cluster. De esta manera, la comunicación se basa en dos niveles: el primer nivel corresponde al envío de los datos de los nodos de la agrupación al nodo cabeza de cluster y el segundo nivel a la comunicación entre los nodos cabeza de cluster y el nodo central encargado del procesado de los datos.

Para reducir la BER dentro de cada nivel, se van a emplear técnicas de codificación basadas en Codificación de Red y códigos LDPC. Se simularán una serie de escenarios con el objetivo de comparar varios casos posibles y obtener los parámetros para los que nuestro algoritmo ofrece mejores resultados. Se busca poder diseñar una WSN basándose en el algoritmo propuesto, obteniendo para ello el número adecuado de clusters y el valor adecuado de los parámetros que intervienen en el sistema. Este análisis se va a realizar comparando distintas curvas que presenten la BER frente a distintos valores de SNR (*Signal Noise Ratio*; Relación Señal a Ruido) para cada escenario.

Con esto, los objetivos del Proyecto se pueden resumir en:

- Estudio teórico previo sobre las técnicas de Codificación de Red y códigos LDPC.
- Familiarización con la herramienta de simulación y con el algoritmo de base.
- Implementación del algoritmo principal.

- Análisis de los resultados y obtención de los parámetros adecuados para cada escenario.
- Extracción de conclusiones a partir de los resultados obtenidos.

1.3. Materiales utilizados

R. E. Shannon definió en qué consistía una simulación como: “el proceso de diseñar un modelo de un sistema real y realizar experimentos con este modelo con el propósito de entender el desarrollo de un sistema y/o evaluar varias estrategias para el funcionamiento del sistema.” [Shannon, 1998]

Para realizar este Proyecto se han generado una serie de simulaciones de un algoritmo basado en una red de sensores inalámbricos con el fin de reducir la probabilidad de error de bit. Para las simulaciones se ha utilizado el entorno de simulación *MATLAB* funcionando sobre un servidor ubicado en el departamento de Teoría de la Señal y Comunicaciones de la Universidad Rey Juan Carlos. Este servidor, llamado CUDA, se encuentra equipado con 8 procesadores GPU NVIDIA Tesla C2050, dos procesadores Intel Xeon Quad y 24 GB de RAM, además de otros 24 GB de RAM dedicadas a los procesadores GPU. En total, el rendimiento del servidor es de 4 Teraflops en operaciones de doble precisión. El código generado para este Proyecto se ha adaptado de manera que se puedan aprovechar las prestaciones de dicho servidor.

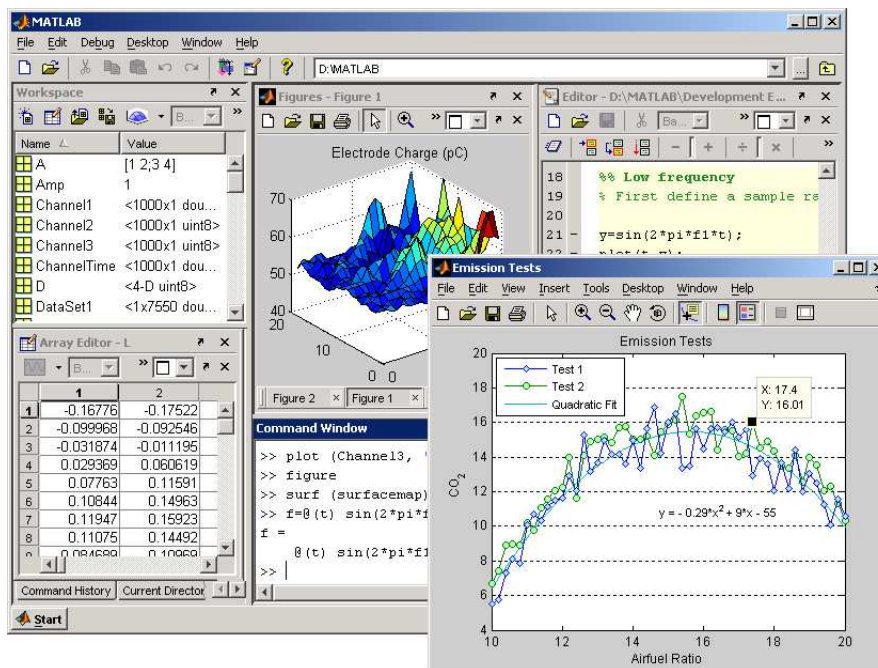


Figura 1.5: Interfaz de MATLAB.

MATLAB es una herramienta desarrollada por la empresa *The MathWorks* y está compuesta por un lenguaje de programación de alto nivel y un IDE (*Integrated Development Environment*; Entorno de Desarrollo Integrado) especializado en cálculo matemático. La primera versión de *MATLAB* data de 1984 y fue creada por Creeve Moler, aunque el lenguaje de programación M fue creado en los años 70 para ofrecer un sencillo acceso a las librerías de matrices LINPACK. Además de la herramienta principal se incluyen varias herramientas denominadas *Toolbox* especializadas en distintos dominios de trabajo, como son el procesamiento en paralelo, el procesamiento de señales y comunicaciones, sistemas de monitorización y control o dominios matemáticos y estadísticos.

El nombre de la herramienta, *MATLAB*, es un acrónimo de *MATrix LABoratory* (laboratorio de matrices) y en su propio nombre se muestra cuál es el elemento básico de almacenamiento que usa: la matriz. Actualmente, *MATLAB* se usa para numerosas aplicaciones entre las que destacan el procesado de señales e imágenes, la simulación y modelado de sistemas, el desarrollo de algoritmos y el análisis y visualización de datos. Además, se incluyen funciones para integrar sus algoritmos con aplicaciones y lenguajes externos, como C/C++, FORTRAN o Java.

1.4. Planificación del Proyecto

En esta sección se explican las tareas que se han llevado a cabo para la planificación de este Proyecto. En la Figura 1.6 se representa por medio de un diagrama de Gantt cómo se han organizado esas tareas a lo largo del año. Nótese que algunas de las tareas se han solapado en el tiempo, algunas porque se han podido realizar en paralelo y otras porque son subtareas de otra y se han marcado en azul más claro. A continuación, se explican las tareas que se han llevado a cabo y su duración aproximada en meses:

1. **Documentación (Estudio previo)**. Duración de 4 meses. En esta fase se realiza una documentación con el fin de familiarizarse y comprender las técnicas que se van a utilizar para el algoritmo presentado.
2. **Estudio del Algoritmo de Base**. Duración de 2 meses. Esta fase consiste en el estudio y la implementación por medio de *MATLAB* de un sistema que simule la transmisión inalámbrica utilizando Codificación de Red con códigos LDPC.
3. **Replicación de resultados del Algoritmo Base**. Duración de 1 mes. Esta fase es una subtarea de la tarea llamada “Estudio del Algoritmo de Base” y consiste en la implementación por medio de *MATLAB* del sistema que simule la transmisión inalámbrica utilizando Codificación de Red con códigos LDPC y que replique los

resultados obtenidos en [Morgado, 2009].

4. **Herramientas de Simulación.** Duración de 2 meses. Esta fase consiste en la familiarización con las técnicas necesarias para poder implementar por medio de *MATLAB* un sistema de transmisión inalámbrica y la codificación de códigos LDPC.
5. **Desarrollo del Algoritmo propuesto.** Duración de 7 meses. Esta fase consiste en la implementación, simulación y análisis de los resultados obtenidos por medio de *MATLAB* del algoritmo propuesto en este Proyecto.
6. **Desarrollo y Simulaciones Primer Nivel.** Duración de 4 meses. Esta fase es una subtarea de la tarea llamada “Desarrollo del Algoritmo propuesto” y consiste en la implementación, simulación y análisis de los resultados obtenidos para el primer nivel del algoritmo propuesto en este Proyecto.
7. **Desarrollo y Simulaciones Segundo Nivel.** Duración de 4 meses. Esta fase es una subtarea de la tarea llamada “Desarrollo del Algoritmo propuesto” y consiste en la implementación, simulación y análisis de los resultados obtenidos para el segundo nivel del algoritmo propuesto en este Proyecto.
8. **Simulaciones Algoritmo Completo.** Duración de 2 meses. Esta fase es una subtarea de la tarea llamada “Desarrollo del Algoritmo propuesto” y consiste en la simulación y análisis de los resultados obtenidos para el sistema completo del algoritmo que se propone en este Proyecto.
9. **Extracción de Conclusiones.** Duración de 2 meses. Esta fase consiste en la extracción de conclusiones a partir de los resultados obtenidos para el algoritmo que se ha propuesto en este Proyecto.
10. **Elaboración de la Memoria.** Duración de 5 meses. Esta fase corresponde a la elaboración de la memoria del presente Proyecto Fin de Carrera.
11. **Familiarización con LaTeX.** Duración de 1 mes. Esta fase es una subtarea de la tarea llamada “Elaboración de la Memoria” y consiste en la familiarización con la herramienta para la composición de textos LaTeX, con la que se redacta la presente memoria.
12. **Redacción de la Memoria.** Duración de 5 mes. Esta fase es una subtarea de la tarea llamada “Elaboración de la Memoria” y consiste en la redacción de la presente memoria.

ID	Tareas	2011						2012					
		Junio	Julio	Agosto	Septiembre	Octubre	Noviembre	Diciembre	Enero	Febrero	Marzo	Abril	Mayo
1	Documentación (estudio previo)												
2	Estudio de Algoritmo de Base												
3	Replicación de Resultados Algoritmo Base												
4	Herramientas de Simulación												
5	Desarrollo del Algoritmo propuesto												
6	Desarrollo y Simulaciones Primer Nivel												
7	Desarrollo y Simulaciones Segundo Nivel												
8	Simulaciones Algoritmo Completo												
9	Extracción de Conclusiones												
10	Elaboración de la Memoria												
11	Familiarización con LaTeX												
12	Redacción de la Memoria												

Figura 1.6: Diagrama de Gantt con la planificación del Proyecto.

1.5. Estructura de la memoria

La estructura de la presente memoria consta de un total de cinco capítulos. El resto de capítulos de la misma quedan de la siguiente manera:

- En el Capítulo 2 se describen dos de los conceptos fundamentales que se van a tratar en este Proyecto. el capítulo comienza explicando en qué consisten la Codificación de Red y sus usos. Posteriormente, se analizan las características de los códigos LDPC, para terminar estudiando cómo se pueden combinar con la Codificación de Red.
- En el Capítulo 3 se describe el algoritmo propuesto. Primeramente se muestra el escenario sobre el que se van a realizar las simulaciones, incluyendo la metodología para realizar la codificación y decodificación del algoritmo. Finalmente se explica en qué consiste el algoritmo jerárquico que se propone y cómo afectan ambos niveles al mismo.
- En el Capítulo 4 se analizan los resultados de las simulaciones realizadas a partir del algoritmo propuesto, tanto para los dos niveles por separado como para el resultado final a través de ambos. Finalmente se comparan los resultados obtenidos con los casos de no utilizar codificación y utilizar codificación LDPC en redes multisalto.
- En el Capítulo 5 se exponen las conclusiones derivadas de los resultados obtenidos en el capítulo anterior y se proponen posibles líneas de investigación o desarrollo sobre este Proyecto.

Capítulo 2

Fundamentos de las Técnicas Empleadas

El algoritmo que se propone en este Proyecto es un algoritmo jerárquico que utiliza Codificación de Red con LDPC para reducir la BER en WSN extensas. Para entender cómo funciona este algoritmo es necesario explicar los dos conceptos principales que intervienen en dicho algoritmo: la Codificación de Red y los códigos LDPC. La ventaja de utilizar códigos LDPC consiste en que se pueden combinar con la Codificación de Red para ofrecer una codificación distribuida. En el Capítulo 3, donde se explica la estructura y funcionamiento del algoritmo propuesto, se profundiza acerca de cómo se combinan ambos conceptos para realizar dicha codificación.

El presente capítulo se divide en dos secciones: En la primera sección se expone en qué consiste la Codificación de Red y qué ventajas ofrece emplear este tipo de codificación en una red que englobe muchos nodos. En la segunda sección se explica qué son los códigos LDPC y su funcionamiento.

2.1. Codificación de Red

Para entender mejor en qué consiste la Codificación de Red es necesario conocer algo más acerca de cómo cooperan los distintos elementos en una red. Se entiende por Comunicación Cooperativa a aquella en la que los elementos de una red colaboran optimizando los recursos de los que disponen con el objetivo de mejorar las prestaciones de dicha red. Esta cooperación se vuelve más necesaria a medida que aumenta el tamaño de una red. Se pueden identificar dos tipos de cooperación [Fitzek and Katz, 2006]:

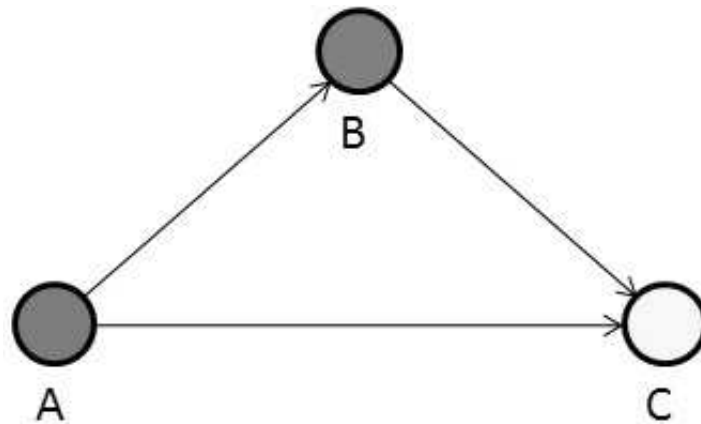


Figura 2.1: Ejemplo de nodo de retransmisión.

- La cooperación implícita (o pasiva) se caracteriza por que los elementos interactúan sin ningún marco preestablecido entre ellos.
- La cooperación explícita se caracteriza por aprovechar las características de un escenario concreto en el que los nodos interactúan para compartir una serie de recursos.

Gracias a la cooperación explícita, se puede pasar de tener un enlace directo entre el origen y el destino a tener una serie de enlaces a lo largo de una red de nodos entre el nodo origen y el nodo destino. En este escenario ya no se pueden analizar las prestaciones de un enlace directo, sino que el estudio debe realizarse entre los dos extremos de la comunicación, apareciendo así el concepto de extremo a extremo, que tiene en cuenta los distintos saltos que se dan hasta alcanzar el destino.

El ejemplo más sencillo de Comunicación Cooperativa es el que utiliza un nodo que retransmite la información entre el origen y el destino, como se muestra en el esquema de la Figura 2.1. En este ejemplo, el nodo *A* puede transmitir su información hacia el nodo *C* por medio del enlace directo que hay entre ellos o puede transmitirla a través del nodo *B*, que retransmite la información recibida por parte del nodo *A* hasta llegar al nodo *C*. Este segundo camino a través del nodo *B* se puede utilizar en el momento que se caiga el enlace directo para mantener la conexión entre los nodos *A* y *C* o para enviar información de manera simultánea por ambos caminos, permitiendo enviar una cantidad mayor de información que si sólo se utilizase un camino en el mismo tiempo.

La Codificación de Red (en inglés Network Coding) [Ahlsvede et al., 2000] es un caso especial de Comunicación Cooperativa donde se explota la topología de una red para aumentar el rendimiento de la misma. Los propios nodos de la red son los que codifican la transmisión, permitiendo que partes del mensaje que se transmite a su destino puedan ir por distintos caminos, aumentando así la capacidad del sistema en redes *multicast* [Ahlsvede et al., 2000] [Jaggi et al., 2003]. Otra posibilidad es que se utilicen los distintos

caminos para reducir el número de errores en la transmisión al enviar varias copias de los mismos mensajes por diferentes caminos hacia su destino[Guo et al., 2007].

Un buen ejemplo de este tipo de codificación es el que se ilustra en la Figura 2.2 para una transmisión *multicast* en la que se busca aumentar la capacidad del sistema. En esta figura se presenta una red compuesta por un total de 7 nodos en los que el nodo identificado como S transmite dos símbolos, $b1$ y $b2$, a dos nodos destino: Y y Z . El resto de nodos de la red retransmiten la información de manera que pueda llegar a ambos destinos. En un sistema tradicional en el que no se utilizara Codificación de Red se establecería un camino hacia cada destino y se transmitiría primeramente un símbolo y después el otro, durando en total la transmisión el doble de tiempo que si se transmitiera un único símbolo.

En la Figura 2.2 se ha utilizado Codificación de Red para reducir el tiempo que se tarda en transmitir y, por lo tanto, aumentar la capacidad del sistema. El nodo S dispone de dos caminos por los que puede transmitir, uno hacia el nodo T y otro hacia el nodo U , y por lo tanto puede enviar por separado $b1$ hacia el nodo T y $b2$ hacia el nodo U . Los nodos T y U retransmiten el símbolo que han recibido hacia los nodos Y y Z , respectivamente. A su vez, como los nodos T y U tienen un enlace directo cada uno con el nodo W , el nodo T le transmite el símbolo $b1$ y el nodo U le transmite el símbolo $b2$. Así, en este momento, el nodo Y ya ha recibido el símbolo $b1$, mientras que el nodo Z ha recibido el símbolo $b2$. En el siguiente paso el nodo W elabora una codificación que consiste en la combinación de los dos símbolos que ha recibido en el paso anterior. En este caso, dicha combinación consiste en la suma binaria de ambos símbolos, con lo que el nodo W transmite el símbolo $b1 \oplus b2$ al nodo X . El nodo X retransmite a los nodos Y y Z el símbolo que corresponde a $b1 \oplus b2$, con lo que estos dos nodos tendrán en total uno de los símbolos y la suma binaria del símbolo que ha recibido junto con el otro símbolo. Para decodificar el mensaje que ha recibido y obtener el símbolo que le falta es suficiente con aplicar una puerta lógica XOR al símbolo recibido en primera instancia con el símbolo codificado. Así, el nodo Y al aplicar una puerta XOR a su símbolo $b1$ con el símbolo codificado $b1 \oplus b2$ obtiene el símbolo $b2$. Lo mismo sucede para el nodo Z , que al aplicar una puerta XOR a al símbolo que recibió antes $b2$ con el símbolo codificado $b1 \oplus b2$ obtiene el símbolo que le faltaba $b1$.

De esta manera se ve como la Codificación de Red ha permitido aumentar la capacidad de la red presentada en la Figura 2.2, transmitiéndose en aproximadamente el mismo tiempo dos símbolos en vez de uno. También se podría haber utilizado la red para reducir la probabilidad de que el símbolo recibido sea erróneo, repitiendo el mismo esquema y enviando el otro dato por la otra rama. Así se reciben los dos símbolos y una combinación de ellos que sirve de chequeo de paridad, a costa de reducir la capacidad del sistema. Sobre esta idea se profundizará en el Capítulo 3, donde se presenta el algoritmo que busca reducir la BER en WSN extensas.

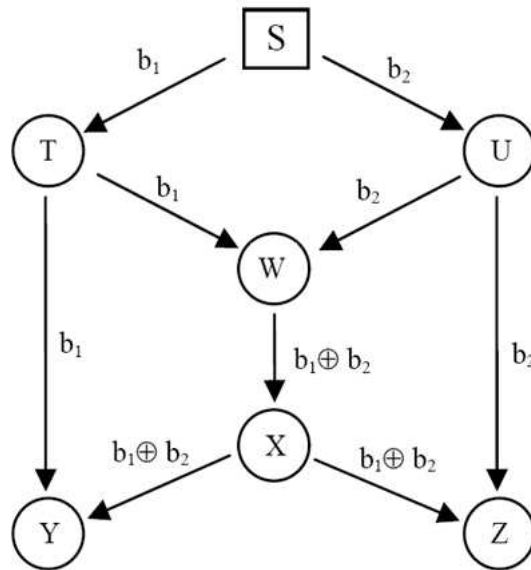


Figura 2.2: Ejemplo de Codificación de Red aprovechando las capacidades del canal.

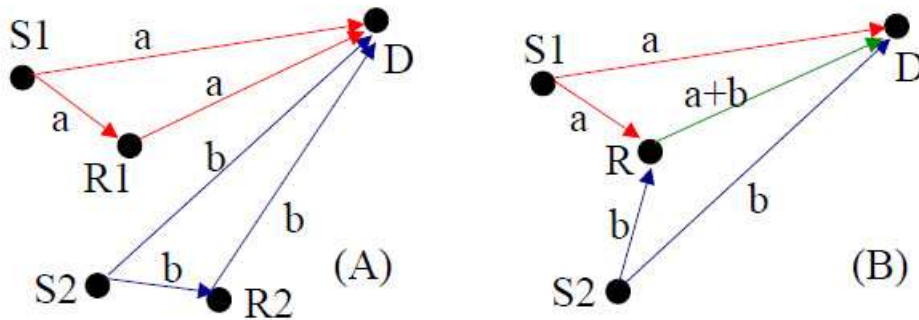


Figura 2.3: Modelos de red de retransmisión: modelo convencional(A) y modelo con Codificación de Red (B) [Bao and Li, 2005].

En este Proyecto se va a analizar un esquema de Codificación de Red destinado a WSN extensas y densas [Quek et al., 2007] [Rachuri and Murthy, 2009] [Panda et al., 2010]. Este tipo de redes están compuestas por un gran conjunto de nodos que abarcan un área relativamente pequeño y cuyos nodos recogen información del entorno y la envían de manera periódica hacia uno o varios destinos comunes. El uso de antenas isótropas permite que los nodos establezcan vecindades, al encontrarse en el radio de cobertura de otros nodos. En este tipo de redes no se puede garantizar que todos los nodos tengan una conexión directa con el nodo destino, por lo que es muy habitual que el resto de nodos colaboren para que la información original llegue a su destino por medio de retransmisiones.

El esquema de Codificación de Red que se va a utilizar como base del desarrollo es el ANCC (*Adaptive Network Coded Cooperation*) [Bao and Li, 2005] [Bao and Li, 2008],

que se adapta de manera precisa a una WSN extensa y densa. En este esquema se propone combinar la red en forma de grafo (*network-on-graph*), que describe la topología instantánea de una red, con los códigos basados en grafos (*codes-on-graph*) y que se puedan adaptar a la naturaleza ad-hoc de una red inalámbrica, como pueden ser los códigos LDPC. Así, una red que se pueda representar mediante de un grafo podría conseguir por medio de una codificación mejorar alguna de sus prestaciones, como podrían ser reducir el número de errores. La idea básica es la de aprovechar la denominada Diversidad de Cooperación (en inglés *Cooperative Diversity*), que consiste en una diversidad espacial que se aprovecha de la independencia de los distintos canales que hay entre el origen y el destino. La Figura 2.3 representa el caso más básico de esta cooperación. La imagen (A) representa el típico esquema de repetición para dos nodos que envían sus símbolos a un destino común, cada uno con su propio nodo repetidor. Una manera más inteligente de realizar este esquema es utilizar un nodo repetidor común para ambos nodos y que en vez de retransmitir los símbolos recibidos, transmita un símbolo codificado que incluya la información de ambos símbolos, como es la suma binaria de ambos: $a \oplus b$. De esta manera se ve que la información recibida en ambas imágenes es equivalente.

2.2. Códigos LDPC

Los códigos LDPC son un tipo de códigos de corrección de errores propuestos por Robert Gallager en 1963 [Gallager, 1963]. Estos códigos se caracterizan por tener una matriz generadora G poco densa, es decir, una matriz generadora binaria que contiene un alto número de ceros y por lo tanto un número bajo de unos. Gallager definió los códigos LDPC como un código $C_{LDPC}(n,j,k)$ siendo n la longitud de las palabras código (en inglés *codeword*), j el número de unos en por columna y k el número de unos por fila. En estos códigos, todas las palabras código que se pueden formar a partir de la misma matriz generadora se encuentran a la misma distancia unas de otras [Peterson, 1961].

Inicialmente, los códigos LDPC no tuvieron mucho éxito y cayeron en el olvido, debido a que requerían una alta capacidad de cómputo para su decodificación y la capacidad de los ordenadores de la época era bastante limitada. Apenas autores como Tanner trataron de conseguir una implementación válida de los códigos de Gallager [Tanner, 1981]. La aparición de los turbocódigos en 1993 [Berrou et al., 1993] y los avances en computabilidad provocaron que la comunidad científica re-descubriera estos códigos a mediados de los 90 [MacKay and Neal, 1996] [MacKay, 1999].

La idea de cualquier código de corrección de errores es la de transformar un mensaje formado por un número concreto de bits en una palabra código válida. En la codificación se introduce una fuerte redundancia de manera que las palabras código sean lo

suficientemente diferentes unas de otras. Esto permite que si una palabra transmitida se corrompe durante la transmisión por culpa del canal, el decodificador pueda disponer de la información necesaria para poder así recuperar el mensaje original.

La codificación en los códigos LDPC es bastante similar a la de cualquier código basado en bloques, con el único requisito de que la matriz de chequeo de paridad H , que se construye a partir de la matriz generadora G , sea dispersa [Richardson and Urbanke, 2001]. Sin embargo, la gran diferencia entre ellos reside en la forma en la que se decodifican. Los códigos de bloque clásicos se decodifican con algoritmos de máxima verosimilitud y por lo que por lo general necesitan de códigos con una longitud muy pequeña. Para decodificar un código LDPC se utilizan algoritmos iterativos basados en la representación gráfica de la matriz de chequeo de paridad.

Originalmente, Gallager definió los códigos LDPC como unos códigos cuya matriz generadora G tiene todas sus columnas y filas con un número constante de unos j y k respectivamente. Esto es lo que se conoce como un código LDPC regular. Posteriormente se ha descubierto que usar códigos LDPC irregulares, donde el número de unos en cada fila o columna de la matriz generadora no es constante, también permite aproximar la capacidad de un canal gaussiano cerca del límite de Shannon [Richardson and Urbanke, 2001].

Es posible asociar la matriz de chequeo de paridad H a un grafo bipartido, conocido como grafo de Tanner [Tanner, 1981], en un código LDPC, como se ve en la Figura 2.4. Este grafo, contiene dos conjuntos de nodos: los nodos variable, que se asocian con cada columna de la matriz H , y los nodos de chequeo, que se asocian con las filas. El grafo se construye de manera que cada entrada no nula de la matriz H conecte el nodo variable con el nodo de chequeo que cruza. Un ciclo en un grafo de Tanner es una secuencia de nodos conectados que empieza y acaba en el mismo nodo y que incluye una única vez a cada uno de los nodos. Lo que se busca en este tipo de códigos es que la longitud de un ciclo (entendiendo por longitud el número de aristas por las que pasa el ciclo) sea lo mayor posible, ya que se asocia el tamaño del ciclo al rango de la matriz de chequeo de paridad. A mayor rango, mayor independencia de los vectores que lo forman y esto beneficia a la decodificación de los códigos, ya que ofrece una mayor capacidad de detección y corrección de errores [Tian et al., 2003].

En la Figura 2.4 se ilustra un ejemplo de un grafo de bipartido para la matriz de chequeo H :

$$H = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 \end{pmatrix} \quad (2.1)$$

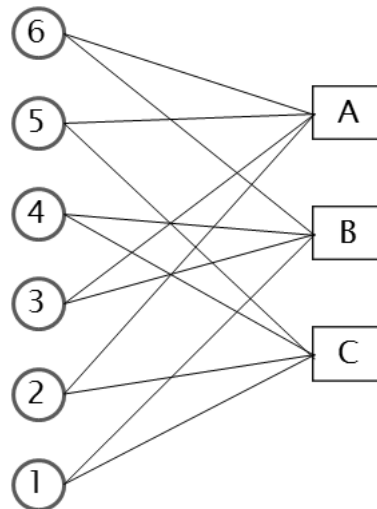


Figura 2.4: Ejemplo de grafo bipartido asociado a la matriz de chequeo H (2.1).

Los círculos se corresponden con los nodos variable y han sido numerados del 1 al 6 en función de la columna que representaban. De igual manera, los nodos de chequeo están representados por cuadrados y las letras A , B y C se corresponden con las filas 1, 2 y 3, respectivamente. Nótese que todos los nodos variable y todos los nodos de chequeo tienen el mismo número de aristas. Este hecho sólo se da en caso de que sea un código de chequeo de paridad regular. Como se puede apreciar, este ejemplo no corresponde a una matriz generadora de un código LDPC, ya que no se trata de una matriz dispersa, pero es suficiente para entender cómo se representa el grafo de Tanner a partir de una matriz de chequeo de paridad y viceversa.

En el Capítulo 3, donde se presenta el algoritmo propuesto para este Proyecto, se profundiza en el método para realizar la codificación y la decodificación de un código LDPC irregular. Como se ha expuesto anteriormente, para la codificación se utilizan las mismas técnicas que para la de cualquier código de bloques. Sin embargo, la decodificación es un poco más complicada y se utilizan algoritmos iterativos.

El algoritmo más utilizado para la decodificación de códigos LDPC es el algoritmo Suma-Producto o algoritmo de propagación de creencias [MacKay, 1999]. Dicho algoritmo es un algoritmo iterativo que estima la probabilidad a posteriori de los símbolos a partir de la matriz de chequeo de paridad, los símbolos recibidos y las verosimilitudes de los símbolos que atraviesan el canal.

En [Arnone, 2008] se estudia el funcionamiento de varios decodificadores de códigos LDPC para su implementación en dispositivos de lógica programable, de donde se extraen las siguientes conclusiones:

- **Decodificador Suma-Producto:** También conocido como Mackay-Neal, es la for-

ma más básica de implementar la decodificación iterativa de los códigos LDPC. Emplea gran cantidad de productos y cocientes, asumiendo que se trabaja con aritmética de punto flotante y con precisión infinita. En este documento se desaconseja su uso cuando se quiere implementar en forma circuital debido sobre todo a los efectos del error de cuantización. Por lo general, para la implementación se utilizan simplificaciones y aproximaciones del mismo.

- **Decodificador Paralelo:** En este decodificador cada nodo, ya sea variable o de chequeo, se implementa de forma individual como una unidad y se conecta al resto de unidades pasándose mensajes en función del grafo que representa dicho código. Su implementación es compleja y se especializa para matrices de chequeo de paridad pequeñas, donde se obtienen altas velocidades de decodificación.
- **Decodificador basado en la partición de la matriz de chequeo de paridad H :** La idea principal es la de subdividir la matriz de chequeo de paridad en varias submatrices que se repiten de manera periódica. Su principal limitación es la de que dichas matrices deben cumplir la condición de repetición periódica.

Capítulo 3

Descripción del algoritmo propuesto

En este capítulo se va a explicar el proceso que se ha llevado a cabo para diseñar el algoritmo propuesto para este Proyecto. Este algoritmo tiene como objetivo reducir la BER extremo a extremo en WSN extensas. Para ello se va a utilizar un esquema que realice una Codificación de Red, empleando para ello códigos LDPC.

El esquema de una WSN es, por lo general, un esquema centralizado en el que los nodos envían la información que recogen a un nodo destino, encargado de almacenar y procesar esos datos. Normalmente, en una WSN los nodos que originan los datos no pueden transmitir directamente la información a su destino, ya que se encuentran lejos de su área de cobertura, por lo que necesitan la cooperación del resto de nodos para encaminar la información hasta que llegue a su destino. Como consecuencia aparecen algoritmos multisalto, que aumenta la BER extremo a extremo a medida que aumenta el número de saltos necesarios para alcanzar el destino [Morgado et al., 2010].

Tradicionalmente, la Codificación de Red se ha utilizado para aumentar la capacidad de la red en transmisiones *multicast* [Ahlsvede et al., 2000] [Jaggi et al., 2003], pero nuestro análisis se centrará en utilizar la Codificación de Red para reducir la BER, aprovechando para ello la diversidad que pueden ofrecer los nodos al escuchar los mensajes que transmiten sus vecinos en una red densa, de manera similar a la que se usó en [Guo et al., 2007]. Para ello se va a utilizar una versión de los códigos LDPC presentados por Gallager [Gallager, 1963] con un esquema de codificación ANCC [Bao and Li, 2005] [Bao and Li, 2008]. Los códigos LDPC son unos potentes códigos encargados de corregir errores y con ellos se consigue que la red sea la encargada de codificar la información transmitida, lo que nos va a permitir utilizarla para reducir así la BER extremo a extremo [Morgado, 2009]. Además, como la BER aumenta con el número de saltos necesarios para alcanzar el destino, se propone un esquema que reduzca el número de saltos a dos, estableciendo así una jerarquía basada en dos niveles.

El presente capítulo se ha estructurado en dos secciones: en la primera sección se presenta el escenario sobre el que se va a emplear el algoritmo propuesto en este Proyecto, haciendo una breve descripción de los elementos y conceptos más importantes de los que consta dicho algoritmo. En la segunda sección se presenta la necesidad de elaborar un algoritmo jerárquico a partir de lo visto en el apartado anterior debido a las limitaciones que aparecen al aplicar el esquema anterior en una red multisalto.

3.1. Descripción del escenario y del algoritmo de base

El algoritmo que se presenta en este Proyecto se adapta a una WSN extensa y densa [Quek et al., 2007]. Una WSN es un tipo de red inalámbrica compuesto por un gran número de nodos que recogen información del entorno y la envían a uno o varios nodos centrales, encargados de almacenar y procesar los datos. Se considera una WSN extensa y densa cuando engloba un gran número de nodos en una extensión pequeña. En nuestro caso, consideramos una WSN extensa cuando contiene más de 1000 nodos. Estos nodos cooperan para garantizar la transmisión de cualquier nodo a su destino. El hecho de ser una red densa implica que cada nodo se encuentra en el radio de cobertura de varios nodos, pudiendo así establecer vecindades.

Uno de los principales problemas que tienen las comunicaciones inalámbricas es que el aire es un canal que dificulta bastante las transmisiones, aumentando notablemente el número de errores respecto a las redes cableadas. Las limitaciones de los propios nodos que forman una WSN hacen que sea necesaria la cooperación entre ellos para garantizar que la información emitida por cualquier nodo pueda llegar a su destino. Como se explicó en el capítulo anterior, debido a dicha cooperación, se pasa de tener un enlace directo entre el origen y el destino a tener una serie de enlaces a lo largo de una red de nodos entre el origen y el destino, con lo que el análisis pasa a realizarse entre los dos extremos de la comunicación.

Los nodos de una red inalámbrica de sensores son cada uno de los elementos que conforman dicha red. Estos nodos están compuestos al menos de un sensor, una micro-controlador, una fuente de energía y un transmisor/receptor radio. Además, dependiendo de su funcionalidad, pueden disponer de otras unidades en su interior aparte de las anteriormente citadas.

Actualmente, el mercado de la tecnología tiende a centralizar la complejidad de las redes, simplificando en la medida de lo posible en las WSN las labores de los nodos. Esto permite que el coste de un nodo dedicado al sensado sea muy bajo (actualmente algunos nodos apenas cuestan un dólar) y fácilmente programable, lo que abarata también el

despliegue de una red que incorpore un gran número de nodos, como es el caso de las WSN extensas. Además, el que se simplifiquen las funciones del nodo también permite optimizar el consumo de batería, al no necesitar de largos periodos para el procesamiento de datos. El hecho de que un nodo tenga un consumo óptimo de su batería implica que se pueda alargar su vida útil, lo que ayuda en el mantenimiento de una red, ya que aumenta el tiempo que tarda un nodo en estar inactivo por falta de energía. A su vez, el que se aumente la duración de las baterías también reduce los costes de amortización, requiriendo realizar un menor número de sustituciones porque pueda haber nodos que no estén operativos por culpa de falta de energía.

Con el objetivo de simplificar las labores de los nodos, el modo de acceso al medio que se va a utilizar será un acceso TDMA (*Time Division Multiple Access*; Acceso Múltiple por División en Tiempo). En un medio compartido, como es el aire en nuestro caso, es muy importante elaborar un método que garantice que todos los elementos que comparten ese medio puedan transmitir. Además se debe reducir lo máximo posible el hecho de que se produzcan interferencias entre los distintos elementos que conforman la red. En nuestro caso, nuestro escenario está formado por un gran número de nodos que quieren transmitir información hacia un destino común. Al utilizar TDMA, a cada nodo se le asigna un espacio o *slot* de tiempo determinado para poder enviar sus datos, mientras que el resto de nodos permanecen en silencio durante ese espacio de tiempo. La motivación para emplear esta técnica de multiplexación y no otra es que nuestro algoritmo requiere que todos los nodos sean capaces de transmitir y recibir datos procedentes de otros nodos y TDMA es la forma más sencilla de implementar esto. Otras técnicas, como FDMA (*Frequency Division Multiple Access*; Acceso Múltiple por División en Frecuencia) o CDMA (*Code Division Multiple Access*; Acceso Múltiple por División en código) podrían introducir una mayor complejidad a nuestros nodos.

Cuando se utiliza TDMA en una WSN, la información que transmite cada nodo puede ser escuchada por cualquier otro nodo que se encuentre en sus inmediaciones. Por esta razón se puede utilizar un esquema de Codificación de Red ANCC que se presentó en [Bao and Li, 2005] y [Bao and Li, 2008]. Este esquema busca emparejar la topología de red instantánea con el grafo de codificación de canal. Para que un esquema ANCC funcione correctamente se requiere que los canales entre los nodos sean ortogonales, y esto se puede conseguir utilizando TDMA.

Inicialmente, no se contempló el uso de este esquema en redes WSN donde pudieran aparecer rutas mutisalto, pero se pueden realizar una serie de asunciones que lo posibilitan [Morgado, 2009]. La idea del esquema ANCC consiste en que cada nodo escuche los datos transmitidos por el resto de nodos y, posteriormente, pueda enviar una parte de dicha información al nodo destino, que en nuestro caso se tratará de una combinación lineal de los datos transmitidos por algunos de sus nodos vecinos. Esto equivale a una codificación de canal distribuida de manera que los propios nodos de la red son los que construyen

la matriz generadora de un código LDPC, ya que cada uno participa con dos símbolos en la comunicación: el símbolo que transmite con la información propia de ese nodo y el símbolo que corresponde a la combinación de la información recibida de otros nodos. Estos dos símbolos se corresponden con dos columnas de la matriz generadora. El nodo destino es capaz de decodificar la información procedente del conjunto de nodos tras las dos fases de la comunicación (envío de datos propios de cada nodo y reenvío de datos ajenos a cada nodo) gracias a la matriz de chequeo de paridad.

En este esquema ANCC se van a emplear un tipo especial de códigos LDPC: los códigos LDGM (*Low Density Generator Matrix*; Matriz Generadora de Baja Densidad) [Frias and Zhong, 2003]. Los códigos LDGM se caracterizan porque su codificación resulta muy sencilla, lo que facilita realizar una codificación distribuida. En estos códigos la matriz generadora G se construye al concatenar la matriz identidad I_N y una matriz dispersa P , resultado de la elección de la información que combina cada nodo. En este tipo de codificación la tasa de codificación es de $1/2$, ya que por cada bit de información se envía otro bit de chequeo de paridad.

En el caso de una WSN, el desvanecimiento del canal se puede modelar de acuerdo a una distribución Rayleigh que varía a lo largo del tiempo, lo que permite que el estado del canal no dependa de la distancia y, por lo tanto, nodos situados a la misma distancia del transmisor puedan ver canales distintos. Este es un modelo de canal recomendado para propagación de ondas de radio a través de la troposfera y la ionosfera y para ambientes urbanos con gran densidad de edificios, aunque también se suele aplicar cuando no hay línea de visión directa entre el transmisor y el receptor, ya que incluye los efectos del multicamino [Biglieri et al., 1998] [Proakis, 2000].

En este tipo de redes, el estado del canal entre los nodos tiene un papel fundamental, puesto que las ecuaciones de paridad se eligen en función de las condiciones en las que cada nodo recibe los símbolos procedentes del resto de nodos de la red. De esta manera, al tratarse de canales independientes, las respuestas que elaboran los nodos no están correladas y, por lo tanto, se reduce la posibilidad de que varios nodos elaboren respuestas iguales. Esto provoca que las columnas de la matriz generadora no se encuentren correladas, manteniendo las buenas prestaciones de los códigos LDPC [Morgado, 2009].

Gallager demostró en [Gallager, 1963] que en los códigos LDPC la probabilidad de error decrece exponencialmente con la longitud de la palabra código y la distancia entre palabras código aumenta con la longitud, con lo que utilizar matrices de chequeo de paridad de gran tamaño, como sucede cuando engloban un gran número de nodos, ofrece unas buenas prestaciones.

En el capítulo anterior se definió los códigos LDPC como un código $C_{LDPC}(n, j, k)$ donde n es la longitud de las palabras código, j es el número de unos en por columna y k

es el número de unos por fila. La matriz generadora G de un código LDPC debe ser una matriz muy dispersa. Se define una matriz dispersa [Richardson and Urbanke, 2001] como aquella matriz (generalmente binaria) en la que la gran mayoría de sus valores son ceros. Así, en nuestro escenario, una matriz muy dispersa será aquella en la que $n \gg j, k$.

El esquema ANCC consta de tres fases diferenciadas:

Fase de difusión Durante esta fase, cada uno de los N nodos que conforman la red transmite su símbolo en el *slot* asignado. El nodo central y el resto de nodos de la red permanecen en silencio a lo largo de dicho *slot*, permitiéndoles recibir y almacenar dicho símbolo. Al mismo tiempo, los nodos de la red analizan las condiciones en las que se ha recibido cada uno de los símbolos a partir del estado del canal durante la recepción de cada símbolo. Esta información será utilizada durante la fase de reenvío.

Fase de Reenvío En esta fase cada uno de los N nodos de la red envía al nodo central una combinación lineal de algunos de los símbolos que ha recibido provenientes del resto de nodos en la fase anterior. A partir de la información recogida en la fase de difusión, cada nodo de la red elige a los d símbolos que se han recibido con mayor veracidad y que participarán en dicha combinación, que consistirá en la suma binaria de esos símbolos. Una vez obtenido el símbolo con la información que se va a reenviar, cada nodo lo transmite en su *slot* asignado hacia el nodo central. En la Sección 3.1.1 se explica más detenidamente como se elabora la codificación en dicho algoritmo.

Fase de decodificación Una vez el nodo central ha recibido todos los símbolos de las dos fases anteriores, se procede a decodificar dicha información. Para ello se utiliza el algoritmo Suma-Producto, que se explicará más detenidamente en la Sección 3.1.2.

3.1.1. Codificación LDPC

La codificación de los códigos LDPC no se diferencia de la codificación de cualquier código de bloque. A continuación se resumen los vectores y matrices que intervienen en la codificación de nuestro algoritmo:

Vector de información \mathbf{u} Es el vector compuesto por los N bits de información, cada uno de ellos procedente de cada uno de los nodos que transmiten la información:

$$u = [u_1, u_2, u_3, \dots, u_N] \quad (3.1)$$

Matriz generadora G Es una matriz de tamaño $N \times 2N$ que se utiliza para generar las palabras codificadas. Esta matriz está definida por la matriz identidad I_N y la matriz P , que es la encargada de generar los bits de paridad presentes en las palabras codificadas. La forma de obtener la matriz P depende de las ecuaciones de paridad que se quieran utilizar. En la Sección 3.1.3 se ve un ejemplo de este algoritmo y de cómo se pueden obtener dichas ecuaciones de paridad.

$$G = [I_N P] \quad (3.2)$$

Vector de información codificada c Es el vector que resulta de codificar el vector de información u con la matriz generadora G . Tiene longitud $2N$ de manera que los primeros N bits corresponden a los bits de información originales y los N últimos bits corresponden a combinaciones lineales (definidas por la matriz P) de los bits de información.

$$c = uG = [c_1, c_2, \dots, c_N, c_{N+1}, c_{N+2}, \dots, c_{2N}] \quad (3.3)$$

$$[c_1, c_2, \dots, c_N] = [u_1, u_2, \dots, u_N] \quad (3.4)$$

$$[c_{N+1}, c_{N+2}, \dots, c_{2N}] = [u_1, u_2, \dots, u_N]P \quad (3.5)$$

Matriz de chequeo de paridad H Es la matriz de chequeo de paridad para el código sistemático generado por G , que tendrá dimensiones $N \times 2N$. La matriz H se construye de manera que el producto entre los vectores fila de la matriz G con los de la la matriz H sean ortogonales. De manera que:

$$H = [P^T I_N] \quad (3.6)$$

$$G \cdot H^T = 0 \quad (3.7)$$

Donde P^T y H^T son las matrices transpuestas de P y H respectivamente.

Vector de síndrome s Este vector es el resultado de chequear una palabra con la matriz de chequeo de paridad H . Al realizarse la decodificación por síndrome, debe cumplirse que el síndrome de una palabra codificada válida c debe ser el vector nulo, como se deduce de:

$$s = c \cdot H^T \quad (3.8)$$

Sustituyendo y aplicando (3.3) y (3.7) se obtiene que:

$$s = u \cdot G \cdot H^T = u \cdot 0 = 0 \quad (3.9)$$

Con lo que se deduce que un vector x corresponde a una palabra código si se cumple que:

$$x \cdot H^T = 0 \quad (3.10)$$

3.1.2. Decodificación LDPC: Algoritmo Suma-Producto

Para decodificar los códigos LDPC generados en nuestro algoritmo se va a utilizar el algoritmo Suma-Producto definido en [MacKay, 1999] para poder aplicarlo a ANCC. Se trata de un algoritmo iterativo que estima la probabilidad a posteriori de los símbolos a partir de la matriz de chequeo de paridad, los símbolos recibidos y las verosimilitudes de los símbolos que atraviesan el canal. El algoritmo es apropiado para un canal binario con ruido independiente. En este algoritmo de decodificación se distinguen varias etapas:

Inicilización Se definen dos conjuntos de datos: El conjunto $L(m)$ corresponde con los nodos que han utilizado el dato que envía el nodo m . De la misma manera, el conjunto $M(l)$ corresponde con los nodos que han participado en la respuesta elaborada por el nodo l . Se definen dos matrices, la matriz Q y la matriz la matriz R , cuyos elementos se van actualizando a partir de la información de los conjuntos $L(m)$ y $M(l)$. El valor Q_{ml}^y corresponde a la probabilidad de que el bit l del vector x que se analiza tenga el valor $y \in \{0, 1\}$ a partir de la información que ofrece el resto de nodos, sin incluir al nodo m .

El valor R_{ml}^y corresponde a la probabilidad del bit m del vector x una vez fijado que el valor de l es y .

Se fijan las verosimilitudes p_l^0 y p_l^1 como la probabilidad de que el bit l del vector x sea 0 ó 1, respectivamente. Para todos los elementos l, m que intervienen en la matriz de chequeo de paridad H se inicializan los valores de Q_{ml}^0 y Q_{ml}^1 a p_l^0 y p_l^1 respectivamente.

Paso horizontal se define la matriz δQ_{ml} como la diferencia entre Q_{ml}^0 y Q_{ml}^1 :

$$\delta Q_{ml} = Q_{ml}^0 - Q_{ml}^1 \quad (3.11)$$

$$\delta Q_{ml} = Q_{ml}^0 - Q_{ml}^1 \quad (3.12)$$

Se obtienen los valores de R_{ml}^0 y R_{ml}^1 :

$$\delta R_{ml} = R_{ml}^0 - R_{ml}^1 \quad (3.13)$$

$$\delta R_{ml} = \prod_{l' \in L(m) \setminus l} \delta Q_{ml} \quad (3.14)$$

$$\delta R_{ml}^0 = \frac{1 + \delta R_{ml}}{2} \quad (3.15)$$

$$\delta R_{ml}^1 = \frac{1 - \delta R_{ml}}{2} \quad (3.16)$$

Paso vertical En este paso se toman los valores de R_{ml}^0 y R_{ml}^1 para actualizar los valores de Q_{ml}^0 y Q_{ml}^1 . Para cada l se calcula:

$$Q_{ml}^0 = \alpha_{ml} p_l^0 \prod_{m' \in M(l) \setminus m} R_{m'l}^0 \quad (3.17)$$

$$Q_{ml}^1 = \alpha_{ml} p_l^1 \prod_{m' \in M(l) \setminus m} R_{m'l}^1 \quad (3.18)$$

Donde la constante α_{ml} se elige de manera que cumpla la siguiente condición:

$$\delta Q_{ml}^0 + Q_{ml}^1 = 1 \quad (3.19)$$

Para obtener los valores de α se define otra nueva matriz:

$$C_{ml}^y = p_l^y \prod_{m' \in M(l) \setminus m} R_{m'l}^y \quad (3.20)$$

Con lo que Q_{ml}^y se puede expresar como:

$$Q_{ml}^y = \alpha_{ml} C_{ml}^y \quad (3.21)$$

Como se debe cumplir la ecuación (3.19), se deduce que el valor de α_{ml} es:

$$\alpha_{ml} = \frac{1}{C_{ml}^0 + C_{ml}^1} \quad (3.22)$$

Decodificación A partir de los valores obtenidos en los pasos anteriores se crea un vector estimado \hat{x} decodificado. Este vector se crea de manera que:

$$\hat{x}_l = Q_l^1 > 0,5 \quad (3.23)$$

Con lo que los valores de \hat{x} serán 1 si cumple que $Q_l^1 > 0,5$ y 0 si no lo cumple. Una vez construido el vector \hat{x} se debe verificar que cumple la ecuación (3.8) y su síndrome es el vector nulo:

$$\hat{x} \cdot H^T = 0 \quad (3.24)$$

Si se cumple dicha condición, el algoritmo dará como salida el vector \hat{x} . En caso

contrario se realizarán nuevamente los pasos horizontal, vertical y decodificación del algoritmo hasta que se cumpla o que se agote el número máximo de iteraciones para el mismo.

3.1.3. Ejemplo de Codificación de Red con LDPC

En las Figuras 3.1 y 3.2 se ve un ejemplo de cómo funciona el esquema ANCC que utiliza Codificación de Red con códigos LDGM. En este ejemplo se han empleado cinco elementos y con $d=2$ para la fase de reenvío. Aunque en este ejemplo no se está utilizando realmente códigos LDGM, porque para ello la matriz generadora debe ser poco densa, puede ilustrar de manera bastante clara los pasos que se siguen en la codificación de este tipo de códigos.

En la Figura 3.1 se ve cómo se comportan los nodos en la fase de difusión y los datos que almacenan. En nuestro ejemplo hay 5 nodos, etiquetados como A , B , C , D y E y un nodo central que es el que debe recibir dicha información, en color marrón.

En la imagen 1 de la Figura 3.1 se muestra la transmisión del nodo A . El nodo A transmite su símbolo y tanto el nodo central como el resto de nodos se mantienen en silencio y escuchando el símbolo a que el nodo A ha transmitido. El resto de nodos analiza la información recibida y verifican si el símbolo se considera bien recibido o no. Si el nodo considera que se ha recibido correctamente se procede a almacenar el símbolo que ha enviado el nodo A . En caso contrario se descarta el símbolo. En nuestro ejemplo, los nodos que han recibido correctamente el dato son los nodos B , D y E , y se ha representado con un circulito al lado de cada nodo, como se ve en la imagen 2. Además, El nodo central también almacena el dato que envió el nodo A .

En la imagen 2 de la Figura 3.1 se aprecia la transmisión del nodo B . Al igual que se ha explicado para el nodo A , el nodo B transmite su símbolo, mientras el resto de nodos de la red escuchan el canal y almacenan el símbolo si estiman que se ha recibido correctamente. En este caso, son los nodos A y E los que almacenan el símbolo, como se ve en la imagen 3. El nodo central almacena también el símbolo.

Estos pasos se repiten para la transmisión de los símbolos del resto de nodos (C , D y E) de la red, como se muestra en las imágenes 3, 4 y 5 de la Figura 3.1. Finalmente, en la imagen 6 de la Figura 3.1 se puede ver el resultado de los símbolos que ha almacenado cada uno de los nodos:

- El nodo central ha recibido todos los símbolos (a , b , c , d y e).

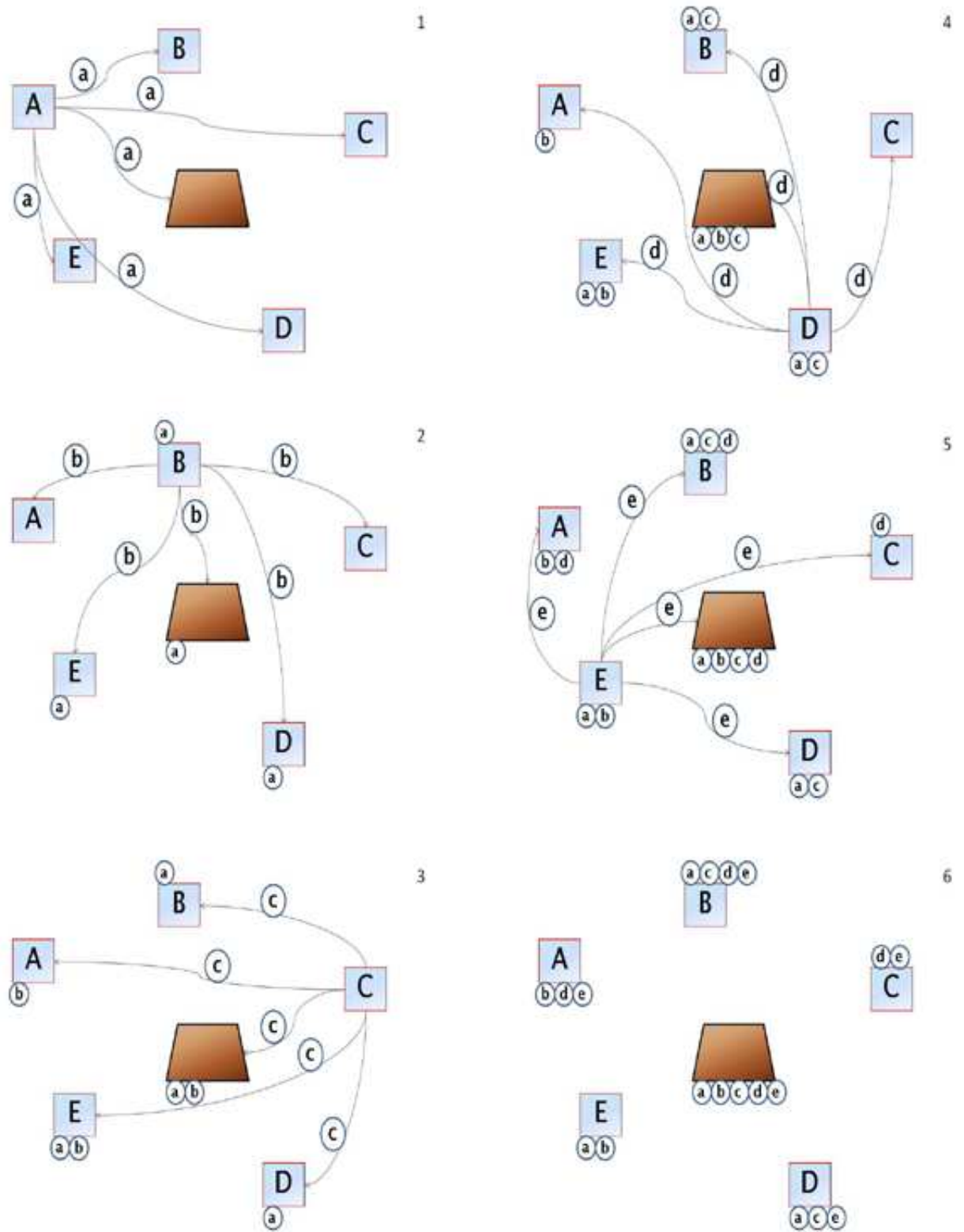


Figura 3.1: Ejemplo de fase de envío con 5 nodos.

- El nodo A ha recibido correctamente los símbolos b , d y e .
- El nodo B ha recibido correctamente los símbolos a , c , d y e .
- El nodo C ha recibido correctamente los símbolos d y e .
- El nodo D ha recibido correctamente los símbolos a , c y e .
- El nodo E ha recibido correctamente los símbolos a y b .

Con lo que el vector de información u compuesto por los símbolos que han transmitido todos los nodos queda:

$$u = \left(a \ b \ c \ d \ e \right)$$

Una vez todos los nodos han transmitido su símbolo comienza la fase de reenvío. Primeramente, como se ve en la imagen 1 de la Figura 3.2, cada nodo elige los d símbolos que mejor ha recibido en la fase anterior. Podría darse el caso en que uno de los nodos tenga menos de d nodos recibidos correctamente. En ese caso, ese nodo transmitiría únicamente los nodos que haya recibido de una manera correcta, quedando la matriz generadora G con un número por debajo de los d unos en la columna correspondiente. En el ejemplo que nos ocupa tenemos que cada nodo ha elaborado las siguientes salidas:

- El nodo A ha recibido b , d y e y la respuesta que elabora será $\mathbf{b} \oplus \mathbf{d}$.
- El nodo B ha recibido a , c , d y e y la respuesta que elabora será $\mathbf{c} \oplus \mathbf{e}$.
- El nodo C ha recibido d y e y la respuesta que elabora será $\mathbf{d} \oplus \mathbf{e}$.
- El nodo D ha recibido a , c y e y la respuesta que elabora será $\mathbf{a} \oplus \mathbf{e}$.
- El nodo E ha recibido a , b y e y la respuesta que elabora será $\mathbf{a} \oplus \mathbf{b}$.

Donde el símbolo \oplus representa la suma binaria, con lo que la expresión $a \oplus b$ representa la suma binaria de los símbolos a y b . El símbolo resultante de esta suma binaria es el que enviará cada nodo en la fase de reenvío. El comportamiento de la fase de reenvío se muestra en la Figura 3.2 y es bastante similar al que tenían los nodos en la fase de difusión. Cada nodo emite el nuevo símbolo que ha elaborado mientras el resto de símbolos permanecen en silencio. La principal diferencia entre las dos fases es que en esta segunda fase ya no es necesario que los nodos escuchen los símbolos que transmite el resto de

nodos. En esta segunda fase únicamente el nodo central debe escuchar y almacenar los símbolos.

El proceso que se representa en la Figura 3.2 corresponde a la fase de reenvío y es el siguiente:

Primeramente, todos los nodos de la red preparan el símbolo que van a enviar en la fase de reenvío. Posteriormente, el nodo A envía el símbolo correspondiente a $b \oplus d$ junto con las cabeceras necesarias para decodificarlo en el nodo central, como se representa en la imagen 1 de la Figura 3.2. El nodo central recibe y almacena la información recibida. Estos pasos se repiten para los nodos B , C , D y E , que transmiten sus símbolos correspondientes en las imágenes 2, 3, 4 y 5 de la Figura 3.2, respectivamente.

Finalmente, en la imagen 6 de dicha figura, se ve como el nodo central ha recibido todos los datos correspondientes a ambas fases. Así, tras estas dos fases, en nuestro ejemplo y a partir de las ecuaciones (3.4) y (3.5) se tiene que el vector de información codificada c resultante es:

$$c = [a, b, c, d, e, b \oplus d, c \oplus e, d \oplus e, a \oplus e, a \oplus b]$$

A partir de los datos anteriores sobre los nodos que transmiten y los que va a combinar cada nodo en la fase de reenvío, el nodo central puede construir la matriz generadora G . Para ello se necesita la matriz que expresa las ecuaciones de paridad P . Esta matriz P se construye a partir de la información de la fase de reenvío. Cada columna y fila se corresponde con cada uno de los nodos que integran la red. Un elemento p_{ij} tendrá valor 1 si el nodo que corresponde a la fila i participa en la suma binaria de la respuesta que emite el nodo que corresponde a la columna j , siendo 0 en caso de que no participe. Con ello, la matriz P resultante para el ejemplo anterior es:

$$P = \begin{pmatrix} \overbrace{0}^A & \overbrace{0}^B & \overbrace{0}^C & \overbrace{1}^D & \overbrace{1}^E & \}A \\ 1 & 0 & 0 & 0 & 1 & \}B \\ 0 & 1 & 0 & 0 & 0 & \}C \\ 1 & 0 & 1 & 0 & 0 & \}D \\ 0 & 1 & 1 & 1 & 0 & \}E \end{pmatrix}$$

Donde en la parte superior se ha etiquetado el nodo al que corresponde cada columna y en la parte derecha el nodo al que corresponde la fila, siendo así, por ejemplo, un uno el valor del elemento correspondiente a la fila B y la columna A si el símbolo que transmite

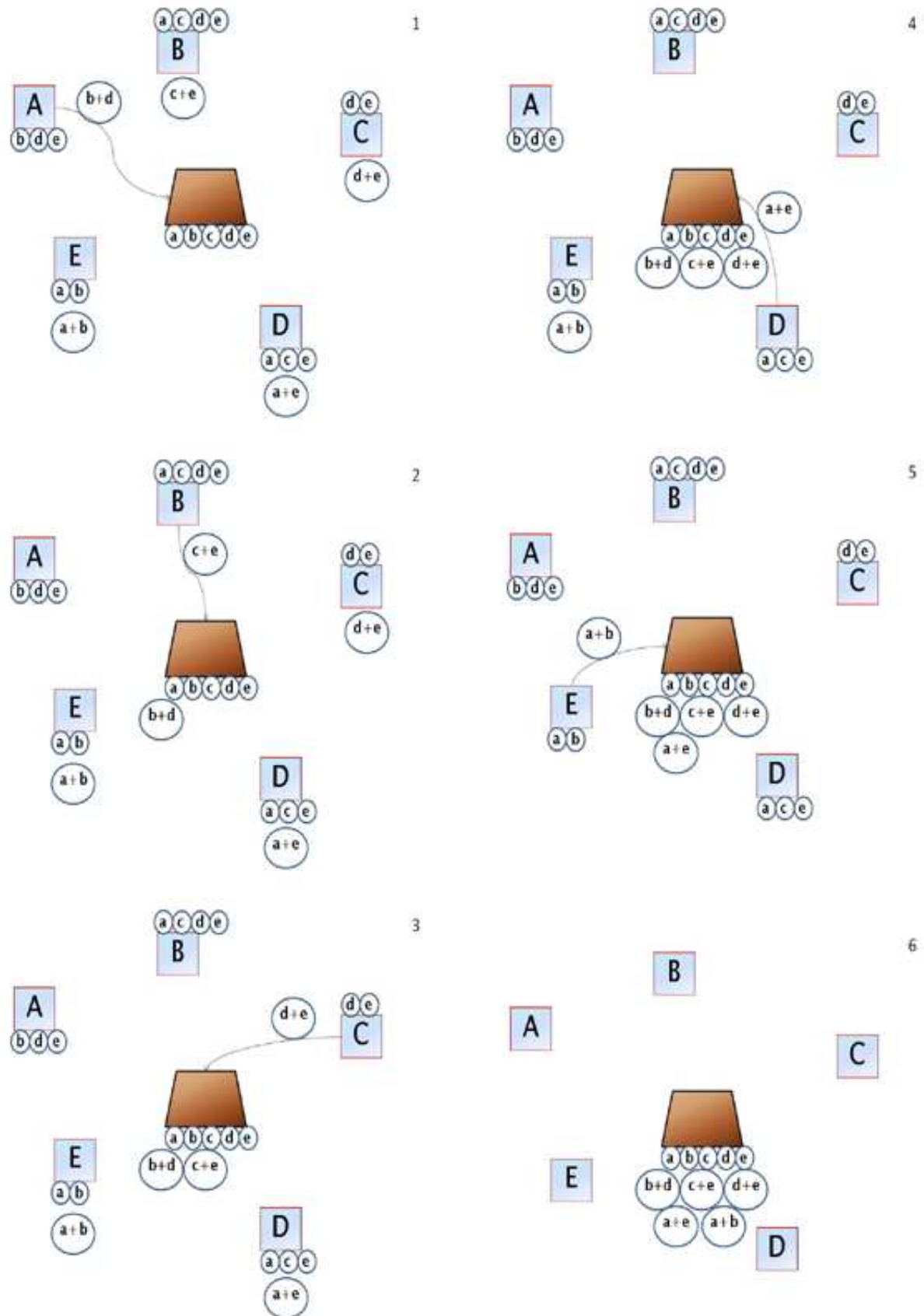


Figura 3.2: Ejemplo de fase de reenvío para 5 elementos y $d=2$.

el nodo A en la fase de reenvío lleva información del símbolo que transmitió el nodo B , como sucede en este caso. De acuerdo con la ecuación (3.2) y la matriz P calculada, se puede construir la matriz G encargada de generar este código, que queda de la siguiente manera:

$$G = \begin{pmatrix} \overbrace{1}^A & 0 & 0 & 0 & 0 & \overbrace{0}^A & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

La ecuación (3.2) define que la matriz generadora G se puede dividir en dos mitades. La primera mitad corresponde a una matriz identidad I_N de tamaño N . Esta matriz identidad representa la fase de difusión, en la que cada nodo de la red transmite su dato, y esto se representa con un 1 para cada elemento g_{ii} donde $i \in \{1, N\}$. La segunda mitad de la matriz G corresponde a la matriz P , que define las ecuaciones de paridad, como se ha explicado anteriormente. De esta manera, se aprecia que en la matriz generadora G a cada nodo le corresponden dos columnas, la columna i para la transmisión en la fase de difusión y la columna $N+i$ para la transmisión en la fase de reenvío, lo que se corresponde con una codificación distribuida, como se muestra en el ejemplo anterior, donde se etiqueta que la primera y la sexta columna corresponden a las dos veces que transmite el nodo A , una en la fase de difusión y otra en la fase de reenvío.

Además, a partir de la matriz P el nodo central puede construir la matriz H de chequeo de paridad, como se indica en la ecuación (3.6):

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

Esta matriz de chequeo de paridad H será utilizada por el algoritmo Suma-Producto explicado en la Sección 3.1.2 para obtener el vector de información u transmitido.

3.2. Algoritmo jerárquico basado en Codificación de Red con LDPC

En la sección anterior se han presentado las bases del algoritmo ANCC que utiliza Codificación de Red con códigos LDPC en WSN extensas basadas en un único nivel. Todos los nodos que se presentaban en la red alcanzaban el nodo destino en un único salto. Este esquema facilitaba y simplificaba bastante el análisis, pero este esquema es difícil de representar en un entorno real y pueden aparecer varios saltos para alcanzar el nodo destino, y esto depende, entre otros factores, del número de nodos, la distancia entre ellos y la tecnología empleada.

Como se demostró en [Morgado, 2009], a medida que aumenta el número de nodos que requieren de más de un salto para alcanzar el destino, los beneficios de usar este esquema con códigos LDPC disminuyen, con lo que se hace necesaria una estrategia que minimice ese número de saltos tanto como sea posible. En la Figura 3.3 se representan las curvas BER-SNR obtenidas para una WSN de 1000 nodos en los casos de usar codificación de Red con LDPC y sin codificar utilizando un radio de cobertura de $0.9R_0$, donde R_0 es la distancia a la que se encuentra el nodo más alejado del centro. Los nodos que se encuentren a una distancia mayor a $0.9R_0$ del centro deben realizar dos saltos para llegar a él. En esta figura, se compara el caso codificado con la transmisión directa sin codificar y con la transmisión sin codificar en la que los nodos situados a una distancia mayor a $0.9R_0$ del centro.

Se define el Punto de Mejora como el valor de SNR a partir del cuál se obtienen valores menores de BER utilizando Codificación de Red que sin utilizar codificación. De esta manera, en la Figura 3.3 se ve que se obtienen mejores valores de BER a partir de los 6dB, con lo que el punto de mejora se sitúa en ese valor de SNR. Se define la Ganancia de Codificación como la diferencia en dB de valores de SNR para los que se obtiene la misma BER en ambos esquemas. Así, si se calcula la Ganancia de Codificación en la Figura 3.3 para el valor de $BER=2 \cdot 10^{-3}$ se puede obtener que la Ganancia de Codificación es superior a los 10 dB al utilizar Codificación de Red.

La Figura 3.4 representa las ganancias de codificación obtenidas utilizando Codificación de Red con LDPC para diferentes radios de cobertura, y que van desde el caso en que haya transmisión directa ($R_N/R_0=1$) al caso en que la información procedente de todos los nodos deba dar dos saltos para alcanzar su destino ($R_N/R_0=0.5$). En dicha gráfica se ve que, según aumenta el número de nodos que deben dar dos saltos para alcanzar el destino, la ganancia de codificación es menor. Además, los resultados muestran que al aumentar el número de saltos para alcanzar el destino la ganancia de codificación se reduce, con lo que uno de los objetivos a la hora de diseñar una red es el de reducir el

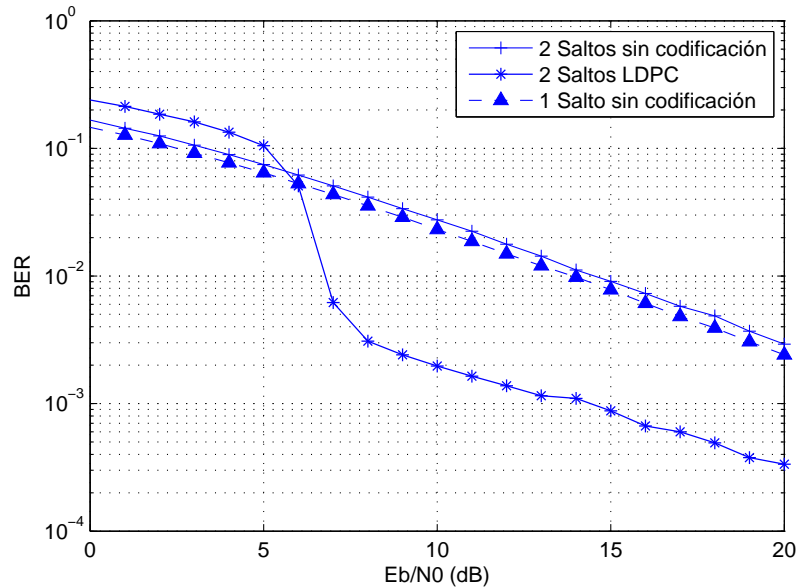


Figura 3.3: BER extremo a extremo para un radio de cobertura de $0.9R_0$. Se comparan los casos de transmisión directa, transmisión sin codificación con dos saltos para una red de 1000 nodos donde los nodos que se encuentren a distancias superiores a $0.9R_0$ del centro y Codificación de Red con LDPC con dos saltos para los nodos a distancias del centro superiores a $0.9R_0$.

número de saltos para alcanzar el destino lo máximo posible.

Por esta razón se presenta en este Proyecto un algoritmo jerárquico basado en dos niveles. La estrategia consiste en dividir el conjunto total de nodos en varios clusters o agrupaciones que engloben un alto número de nodos cada uno. Para garantizar el funcionamiento del sistema, los cluster se eligen de manera que todos los elementos del mismo cluster puedan tener un enlace directo con el nodo de mayor importancia de la agrupación, al que denominaremos nodo cabeza de cluster (en inglés *cluster head*). Este nodo se encargará de ser el nodo destino de las transmisiones realizadas por el resto de nodos del cluster, siendo el elemento que recibe y decodifica toda la información de dicha agrupación.

Los clusters constituyen el primer nivel y responden al esquema de ANCC presentado en la sección anterior. El segundo nivel está formado por los nodos cabeza de cluster, que envían toda la información procedente de los nodos de su agrupación al nodo central. Como se mostró en [Gallager, 1963] los códigos LDPC tienen buenas prestaciones cuando la matriz generadora es muy dispersa, lo que obliga a utilizar palabras código de gran tamaño y, por lo tanto, matrices generadoras de gran tamaño. Esto implica que los clusters engloben un gran número de nodos, con lo que el número de clusters, y por lo tanto de nodos cabeza de cluster, va a ser muy bajo.

Al disponerse en el segundo nivel de un número muy reducido de nodos cabeza

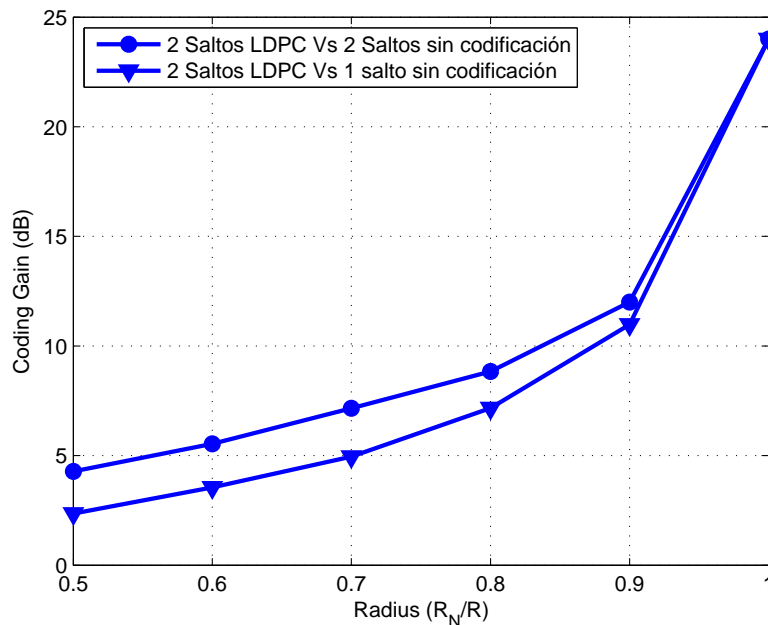


Figura 3.4: Valores de ganancia de codificación obtenidas para diferentes radios de cobertura. Se compara la ganancia de codificación empleando Codificación de Red con LDPC respecto a la transmisión directa con un único salto y con dos saltos.

de cluster, si se utilizase directamente el mismo esquema que en el nivel anterior, se obtendrían matrices generadoras que no podrían ser muy dispersas y por lo tanto el esquema ANCC no va a ofrecer unos buenos resultados.

Para garantizar el buen comportamiento de nuestro algoritmo se propone para el segundo nivel hacer una modificación a la matriz generadora, de manera que cada nodo cabeza de cluster envíe todos los símbolos que ha recibido de su agrupación y todos ellos participen en la construcción de dicha matriz generadora como elementos independientes, ya que sus aportaciones a la matriz generadora se consideran independientes aunque todos sus símbolos procedan del mismo nodo.

Cada nodo cabeza de cluster emite cada uno de los símbolos de su agrupación, mientras que el resto de nodos cabeza de cluster escuchan la transmisión y almacenan esos símbolos si se han recibido correctamente (de la misma manera que se hacía para el nivel anterior), guardando información acerca del nodo que lo transmitió y el símbolo que correspondía. Posteriormente, cada nodo cabeza de cluster “simula” que está compuesto por N nodos independientes, que se corresponde con cada uno de los símbolos del nivel anterior. De esta manera, denominaremos nodo X_y al que corresponde a lo que se escucharía desde el símbolo $y \in \{1, N\}$ del nodo cabeza de cluster X . Después, por medio de un algoritmo aleatorio, para cada X_y se eligen d símbolos de entre los que el nodo cabeza de cluster X ha recibido correctamente y, a partir de esos símbolos elegidos, se elabora la respuesta que cada uno de esos nodos X_y enviará en la fase de reenvío. La respuesta,

igual que ocurría en el primer nivel, consiste en la combinación de los d símbolos elegido para cada nodo X_y . Este hecho nos permite construir una matriz generadora de tamaño $Nk \times 2Nk$, siendo N el número de símbolos que transmite cada nodo cabeza de cluster y k el número de cluster en que se divide nuestra red.

El nodo central recibe y decodifica toda la información como si cada símbolo hubiera sido emitido por un nodo independiente, con lo que el esquema de decodificación es exactamente igual al que se usaba en el nivel anterior y que se explica en la Sección 3.1.2.

En este segundo nivel es aun más importante el estado del canal que en el primero. El esquema ANCC se basa en que los canales entre los nodos sean independientes, y esto facilita que las columnas de la matriz generadora sean linealmente independientes. En el caso del segundo nivel, cada nodo cabeza de cluster envía un total de N datos, procedentes de los nodos de su agrupación y esto provoca que no se pueda garantizar que las columnas de la matriz generadora sean independientes.

Para analizar cómo afecta el canal, vamos a recurrir al tiempo de coherencia (T_c), que se define como el tiempo durante el cual un canal se mantiene aproximadamente constante. Como en este nivel tenemos a pocos nodos que tienen que transmitir muchos datos, no se puede garantizar que los canales varíen entre las transmisiones de varios símbolos consecutivos. Es por ello por lo que el tiempo de coherencia juega un papel fundamental a la hora de diseñar la matriz generadora, ya que si el tiempo de coherencia es muy alto la matriz generadora correspondiente tendrá pocas columnas linealmente independientes. Como en nuestro algoritmo nos interesa que los canales sean lo más independientes posibles y el tiempo de coherencia no se puede modificar, en este segundo nivel aparece el concepto de transmisión a ráfagas. Se entiende por ráfaga al conjunto de símbolos que cada nodo transmite de manera consecutiva. Una ráfaga tiene una duración T_R cuando incluye R símbolos transmitidos. Los nodos se turnan para transmitir cada uno R símbolos de manera que, si intervienen k nodos, el tiempo de coherencia T_c sea menor que kT_R y, por lo tanto, la próxima vez que le toque transmitir al mismo nodo el canal habrá cambiado, ofreciendo datos en un canal distinto del de la anterior ráfaga.

En la práctica, nos interesa que el tiempo de coherencia tenga la menor duración posible para que así cada ráfaga englobe muy pocos símbolos de manera que el número de símbolos que se encuentren correlados por el mismo canal sea lo menor posible y, por lo tanto, las columnas de la matriz generadora del código serán más independientes.

De esta manera, el tamaño de las ráfagas debe ser tal que se cumpla la siguiente ecuación:

$$T_R \geq \frac{T_c}{N_{cluster}} \quad (3.25)$$

Donde $N_{cluster}$ representa el número de nodos cabeza de cluster que intervienen, T_R es la duración de la ráfaga y T_c es el tiempo de coherencia del canal. Si se sustituye la expresión para hallar la duración del tamaño de la ráfaga en la ecuación (3.25) se tiene que:

$$R \geq \frac{T_c}{N_{cluster} \cdot T_s} \quad (3.26)$$

Donde R es el número de símbolos por ráfaga y T_s es el tiempo que se tarda en transmitir un símbolo.

Para entender de manera más clara el funcionamiento de los nodos en este segundo nivel, se propone el siguiente ejemplo: supongamos que tenemos en total cuatro agrupaciones en el primer nivel y, por lo tanto, cuatro nodos cabeza de cluster que van a transmitir la información a un nodo central. Si cada agrupación incluye un total de cuatro nodos tenemos el siguiente esquema:

- El nodo A es el nodo cabeza de cluster de la agrupación A, transmite la información de los nodos A_1 , A_2 , A_3 y A_4 , que la denotaremos como a_1 , a_2 , a_3 y a_4 , respectivamente.
- El nodo B es el nodo cabeza de cluster de la agrupación B, transmite la información de los nodos B_1 , B_2 , B_3 y B_4 , que la denotaremos como b_1 , b_2 , b_3 y b_4 , respectivamente.
- El nodo C es el nodo cabeza de cluster de la agrupación C, transmite la información de los nodos C_1 , C_2 , C_3 y C_4 , que la denotaremos como c_1 , c_2 , c_3 , y c_4 , respectivamente.
- El nodo D es el nodo cabeza de cluster de la agrupación D, transmite la información de los nodos D_1 , D_2 , D_3 y D_4 , que la denotaremos como d_1 , d_2 , d_3 y d_4 , respectivamente.

Supongamos ahora que cada nodo transmite ráfagas de 2 símbolos y que el tiempo de coherencia está por debajo de los $8 T_s$, con lo que después de que los 4 nodos transmitan su ráfaga, se puede considerar que el canal ha variado, ya que se cumple la ecuación (3.26). En caso de no cumplirse esta ecuación, los nodos deberían transmitir ráfagas con

un número mayor de símbolos por ráfaga con el fin de garantizar la variabilidad del canal.

A continuación se detallan ordenadamente los pasos que se siguen en esta transmisión:

- El nodo A comienza la transmisión de una ráfaga con los dos primeros símbolos procedentes de su cluster: a_1 y a_2 . El resto de nodos de la red se mantienen en silencio escuchando la transmisión del nodo A y almacenan ambos símbolos si los ha recibido de manera correcta. A su vez, el nodo central almacena ambos símbolos.
- Después del nodo A transmite el nodo B los dos primeros símbolos de su agrupación: b_1 y b_2 . El nodo central almacena ambos símbolos y el resto de nodos de la red escuchan y almacenan los símbolos si han sido recibidos correctamente. Este mismo paso se repite para el resto de nodos de la red, los nodos C y D con la transmisión de sus dos primeros símbolos: c_1 y c_2 , y d_1 y d_2 , respectivamente.
- Una vez ha transmitido el último nodo de la red sus primeros símbolos, vuelve a transmitir de nuevo el primer nodo de la red, que en nuestro caso es el nodo A . Como el tiempo transcurrido desde que comenzó la transmisión del nodo A ($8 T_s$) cumple con la ecuación (3.26), se entiende que el canal ha cambiado. El nodo A transmite los dos siguientes símbolos, que en este caso son los símbolos a_3 y a_4 . El nodo central almacena el dato recibido y el resto de nodos de la red escucha y almacena los símbolos si se han recibido correctamente.
- De la misma manera vuelven a transmitir todos los nodos de la red hasta que hayan transmitido todos sus símbolos. En ese momento, el nodo central habrá recibido todos los símbolos procedentes de los nodos cabeza de cluster y todos los nodos de la red tendrán información suficiente acerca de lo que transmitieron los otros nodos de la red para elaborar una respuesta en la fase de reenvío.

Podemos suponer que la información que han recopilado el resto de elementos de la red con los símbolos que han recibido correctamente es la que se resume en el Cuadro 3.1.

Nodo	Mejores símbolos escuchados
Nodo A	$b_1, b_2, b_3, c_2, c_3, c_4, d_1$ y d_2
Nodo B	$a_2, a_3, c_1, c_3, c_4, d_2$ y d_3
Nodo C	$a_1, a_4, b_1, b_3, b_4, d_1$ y d_4
Nodo D	$a_1, a_3, a_4, b_2, b_3, c_1, c_3, c_4$

Cuadro 3.1: Ejemplo de símbolos recibidos correctamente en el segundo nivel.

Una vez se han enviado todos los símbolos correctamente comienza la fase de reenvío. Supongamos ahora que el valor del parámetro d para nuestro algoritmo es $d = 3$, es decir, que cada respuesta que envía en la fase de reenvío es la suma binaria de tres símbolos recibidos. Cada uno de los nodos (A, B, C y D) debe elaborar cuatro respuestas, una por

cada uno de los símbolos que recibió cada nodo cabeza de cluster dentro de su agrupación. La respuesta correspondiente a cada símbolo se elabora eligiendo de manera aleatoria tres de los símbolos que se recibieron correctamente.

Así, un ejemplo de las respuestas posibles que elaboran cada uno de los nodos asociadas a cada símbolo, podrían ser las que se exponen en el Cuadro 3.2.

Si atendemos a las ecuaciones de paridad que se representan en el Cuadro 3.2, se puede construir la matriz de paridad P . Para ello se va a utilizar el Cuadro 3.3, que sirve para explicar cómo se forma la matriz de paridad P . En esta tabla, las filas corresponden a los símbolos transmitidos en la fase de difusión de este segundo nivel, mientras que las columnas corresponden a cada una de las respuestas que se envía en la fase de reenvío. De esta manera, si el elemento que corresponde a la columna A_R^1 y la fila b_1 es un 1 significa que en la primera respuesta que envía el nodo cabeza de cluster A interviene el símbolo b_1 .

Nodo	Respuesta
A_R^1	$b_1 \oplus b_3 \oplus c_2$
A_R^2	$b_2 \oplus c_3 \oplus d_2$
A_R^3	$c_2 \oplus c_4 \oplus d_1$
A_R^4	$b_1 \oplus c_3 \oplus c_4$
B_R^1	$a_2 \oplus c_1 \oplus c_4$
B_R^2	$a_3 \oplus c_3 \oplus d_3$
B_R^3	$c_3 \oplus c_4 \oplus d_2$
B_R^4	$a_2 \oplus c_3 \oplus d_2$
C_R^1	$a_1 \oplus b_1 \oplus b_4$
C_R^2	$a_4 \oplus d_1 \oplus d_4$
C_R^3	$b_3 \oplus b_4 \oplus d_1$
C_R^4	$a_1 \oplus b_3 \oplus d_4$
D_R^1	$a_1 \oplus a_3 \oplus a_4$
D_R^2	$a_4 \oplus b_2 \oplus c_4$
D_R^3	$a_3 \oplus b_3 \oplus c_1$
D_R^4	$a_1 \oplus b_3 \oplus c_4$

Cuadro 3.2: Ejemplo de respuestas en la fase de reenvío en el segundo nivel.

	A_R^1	A_R^2	B_R^1	B_R^2	C_R^1	C_R^2	D_R^1	D_R^2	A_R^3	A_R^4	B_R^3	B_R^4	C_R^3	C_R^4	D_R^3	D_R^4	
a_1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0	1
a_2	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0
b_1	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0
b_2	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
c_1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0
c_2	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
d_1	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0
d_2	0	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
a_3	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	1	0
a_4	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0
b_3	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
b_4	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0
c_3	0	1	0	1	0	0	0	0	0	1	1	1	0	0	0	0	0
c_4	0	0	1	0	0	0	0	1	1	1	1	0	0	0	0	0	1
d_3	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
d_4	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0

Cuadro 3.3: Formación de la matriz de paridad P para el segundo nivel.

Con ello, la matriz P queda de la siguiente manera:

$$P = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Una vez construida la matriz de paridad P se construye la matriz generadora G y de chequeo de paridad H de acuerdo a las ecuaciones (3.2) y (3.6), de la misma manera que se hacía para el primer nivel.

De esta manera, se muestra por medio del ejemplo anterior cómo se ha solventado el problema de aplicar el esquema ANCC en el segundo nivel, en el que pocos nodos intervienen en la codificación y en el que, de haber aplicado el mismo esquema que para el primer nivel, las matrices generadoras de los códigos LDPC habrían sido poco dispersas, ofreciendo así malas prestaciones.

Capítulo 4

Resultados

En este capítulo se van a analizar los resultados obtenidos para el algoritmo diseñado y descrito en este Proyecto, basado en un algoritmo de jerarquización usando Codificación de Red con códigos LDPC con el objetivo de reducir la BER en WSN extensas y densas. Para ello se ha utilizado la herramienta de simulación *MATLAB*, recreando un conjunto de escenarios en los que se han modificado varios parámetros con el fin de analizar cómo afectan dichos parámetros a la curva BER frente a SNR. Además, en este Proyecto se busca obtener los valores adecuados de dichos parámetros.

Los principales parámetros que se van a analizar para este algoritmo son:

- El número de iteraciones del algoritmo Suma-Producto. Se analizará para un ejemplo concreto la mejora que supone tener un mayor número de iteraciones en el algoritmo de decodificación Suma-Producto.
- El *parámetro d*. Se probarán varias opciones para ver cómo afecta el número de nodos que intervienen como combinación lineal en la respuesta que envía cada nodo durante la fase de reenvío a la BER obtenida en un esquema ANCC.
- El número de nodos por agrupación. En este caso se analizará principalmente cómo afecta a la BER obtenida el número de nodos que intervienen en la codificación LDPC distribuida y se presentarán los resultados para distintas opciones. Después, se analizarán las repercusiones que esto tiene en el segundo nivel con un número de nodos cabeza de clúster más pequeño y teniendo que enviar los datos cosechados en el primer nivel.

El método que se va a seguir para determinar que un escenario es mejor que otro se basa en la inspección de sus curvas BER frente a SNR, eligiendo la que obtenga un

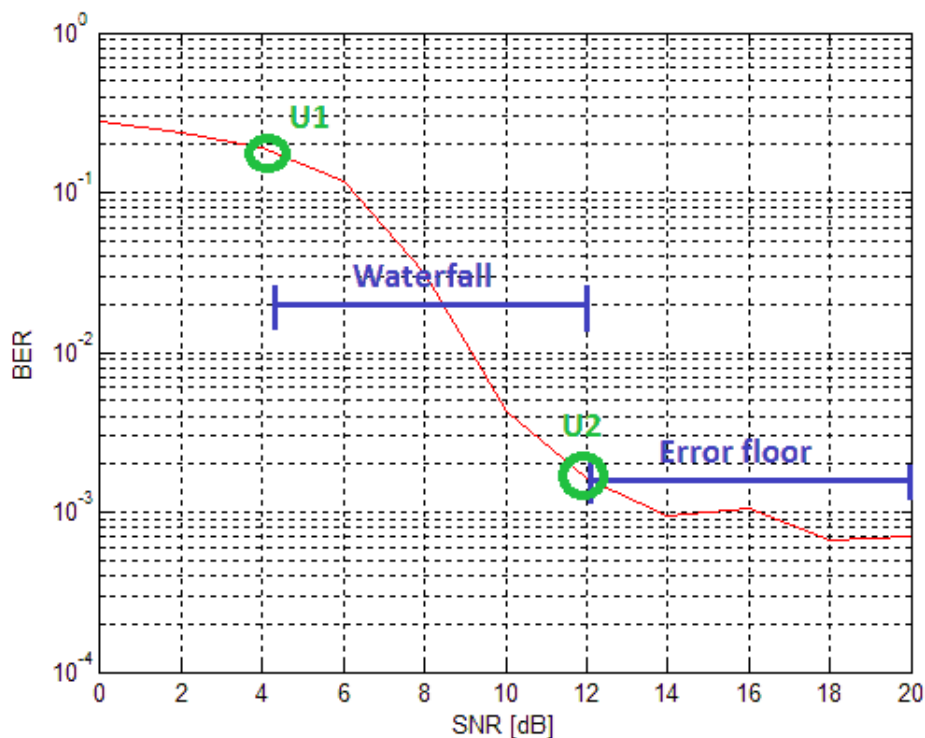


Figura 4.1: Regiones de la curva BER-SNR para un código LDPC.

mejor compromiso entre los valores de BER obtenidos a valores de SNR altas y bajas. La curva BER-SNR para un código LDPC presenta una curva muy característica, que se representa en la Figura 4.1 y en la que se pueden distinguir tres regiones [Proakis, 2000] [Tian et al., 2003]:

- La primera región se da para valores bajos de SNR. En esta región la curva BER-SNR cae con poca pendiente hasta alcanzar un cierto umbral, al que denominaremos u_1 . En esta región se obtiene unos niveles mayores de BER al utilizar codificación LDPC que si no se utilizase.
- La segunda región se denomina región de *Waterfall* y se caracteriza porque la curva BER-SNR tiene una fuerte pendiente hasta que alcanza un nuevo umbral, al que denominaremos u_2 y a partir del cual la pendiente se reduce.
- A la tercera región se la conoce como *Error Floor* y aparece a valores altos de SNR. Esta región se caracteriza porque su pendiente vuelve a ser similar a la que se daba en la primera región.

Los estudios realizados muestran que existe una relación entre la posición del umbral u_1 , la pendiente en la región de *Waterfall* y los valores de BER obtenidos en la región de *Error Floor*. Esto implica que, si se utiliza codificación LDPC y el umbral u_1 de la

curva BER-SNR se da a valores mayores de SNR, la pendiente en la región de *Waterfall* será mayor y se dará para un rango mayor de SNR que en otra curva en la que el umbral u_1 se sitúe a valores menores de SNR. Además, los valores de BER obtenidos en la región de *Error Floor* serán menores cuanto mayores sean los valores de SNR a los que aparece el umbral u_1 .

Por este motivo, el objetivo es hallar un compromiso entre que este umbral u_1 se sitúe a los valores de SNR menores posibles para que los valores de BER obtenidos en la región de *Error Floor* sean mejores y que dicha región aparezca para valores de SNR razonables en una comunicación inalámbrica.

Este capítulo se va a estructurar en cuatro secciones: La primera sección analiza cómo afectan los distintos parámetros que se van a analizar a nuestro sistema. En la segunda sección se analizan varios escenarios posibles para el primer nivel, variando el número de nodos y el valor del parámetro d . En la tercera sección se analiza en dos escenarios con distinto número de clusters cómo afecta el tamaño de las ráfagas para distintos valores de d . En la cuarta sección se analiza y compara el sistema completo para los mejores escenarios de los analizados en las secciones anteriores.

4.1. Parámetros que afectan a nuestro sistema

En esta sección se va a analizar cómo afectan la variación en los valores de algunos parámetros aplicados a escenarios concretos, con el fin de obtener los mejores valores.

4.1.1. Influencia del número de iteraciones

Primeramente se va a analizar cómo afecta el número de iteraciones del algoritmo de decodificación Suma-Producto a la curva BER frente a SNR. El objetivo es el de estudiar las ventajas que puede ofrecer utilizar un número mayor de iteraciones para dicho algoritmo de cara a la implementación en un escenario real.

Como se comentó en el Capítulo 3, el algoritmo Suma-Producto es un algoritmo que realiza una serie de operaciones sobre los símbolos recibidos con el fin de que puedan cumplir con la condición del síndrome que se muestra en la ecuación (3.8). En el caso en el cuál el vector provisional no cumpla con dicha ecuación, se actualizan los valores y se realizan nuevas operaciones sobre ellos. Este paso se puede repetir de manera infinita hasta conseguir que se cumpla la condición de síndrome, con lo que se vuelve necesario acotar dicho paso a un número limitado de iteraciones.

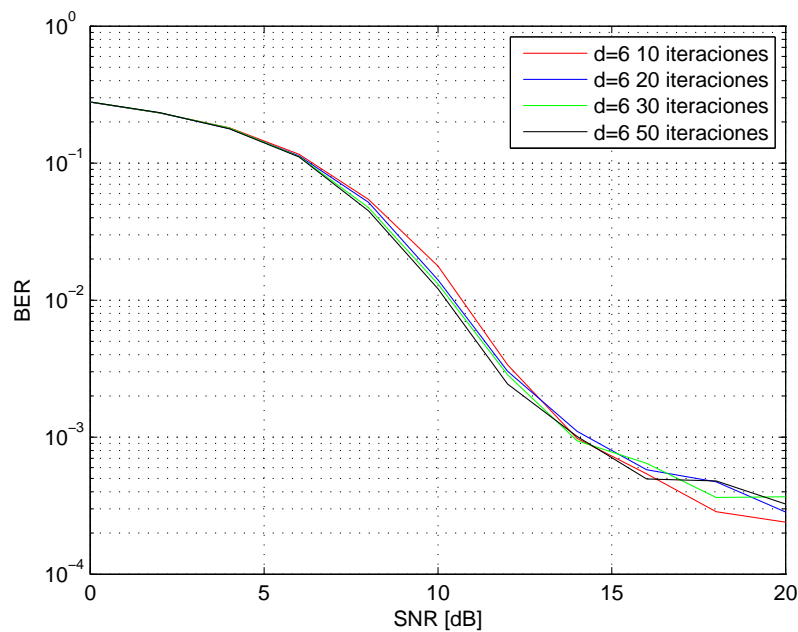


Figura 4.2: Curva BER-SNR obtenida para una codificación LDPC distribuida (primer nivel del algoritmo presentado) en una WSN de 100 nodos. Las combinaciones de la fase de reenvío son de $d=6$ nodos y se varía el número máximo de iteraciones del algoritmo de decodificación tomando los valores 10, 20, 30 y 50.

El objetivo de esta sección es analizar si utilizar un gran número de iteraciones mejora de manera significativa los valores de BER obtenidos, como para compensar el retraso de tiempo que produce durante la decodificación.

La Figura 4.2 representa las distintas curvas obtenidas en las simulaciones para el caso de una agrupación de 100 nodos en la que el valor del *parámetro* d es 6, utilizando distintos números de iteraciones máximas en el algoritmo de decodificación.

Observando los resultados se aprecia que al aumentar el número de iteraciones del algoritmo Suma-Producto se produce una ligera mejora en las curvas BER. Sin embargo, aumentar el número de iteraciones repercute negativamente al tiempo que tarda el sistema en decodificar los símbolos.

La Figura 4.3 muestra el tiempo que se tarda en decodificar los códigos LDPC utilizando el algoritmo Suma-Producto por medio de la herramienta *Profiler* de *MATLAB*. Las medidas se han tomado para un número bajo de datos y para 10, 20, 30 y 50 iteraciones del algoritmo de decodificación. En dicha figura se ve que en el caso de tener 20 iteraciones, el tiempo que tarda en decodificar es un 75 % superior al que tardaría usando 10 iteraciones. Este tiempo es aun mayor al aumentar el número de iteraciones a 30 (cerca del 200 % superior al tiempo que tarda con 10 iteraciones) y a 50 (cerca del 250 % superior al tiempo que tarda con 10 iteraciones).

Profile Summary 10 iteraciones

Generated 04-Jun-2012 03:11:34 using cpu time.

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
RayleighConCodFuncion	1	20.264 s	0.334 s	
Decod_LDPC	130	19.126 s	19.126 s	

Profile Summary 20 iteraciones

Generated 04-Jun-2012 02:57:24 using cpu time.

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
RayleighConCodFuncion	1	36.947 s	1.908 s	
Decod_LDPC	130	33.782 s	33.782 s	

Profile Summary 30 iteraciones

Generated 04-Jun-2012 02:59:47 using cpu time.

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
RayleighConCodFuncion	1	60.025 s	0.436 s	
Decod_LDPC	130	58.547 s	58.547 s	

Profile Summary 50 iteraciones

Generated 04-Jun-2012 03:07:50 using cpu time.

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
RayleighConCodFuncion	1	69.601 s	0.332 s	
Decod_LDPC	130	68.442 s	68.442 s	

Figura 4.3: Tiempo que tarda la función que decodifica por medio del algoritmo Suma-Producto medido con la herramienta *Profiler* de *MATLAB* para una codificación LDPC distribuida en una WSN de 100 nodos, con un número de combinaciones en la fase de reenvío de $d=6$ y en el que el número de iteraciones del algoritmo de decodificación es de 10, 20, 30 y 50 iteraciones.

Se puede concluir que aumentar el número de iteraciones no ofrece una ventaja significativa a nuestro algoritmo, ya que aumenta notablemente el tiempo que dura la fase de decodificación. Por ello de aquí en adelante se usarán un máximo de 10 iteraciones en dicho algoritmo para todos los decodificadores, ya que es uno de los valores más bajos para los que ofrece unos buenos resultados y se tarda relativamente poco tiempo en decodificar.

4.1.2. Influencia del parámetro d

Lo siguiente que se va a analizar es cómo influye el valor del *parámetro d* y qué valores de este parámetro son válidos para nuestro estudio. El *parámetro d* corresponde al número de símbolos procedente de los nodos de la agrupación que intervienen en la respuesta de cada uno de los nodos durante la fase de reenvío. Un mayor valor de este parámetro implica que el símbolo transmitido incluye información sobre un mayor número de nodos, pero un número muy alto puede implicar que la matriz generadora deje de ser muy dispersa, aumentando la probabilidad de que aparezcan ciclos cortos. Esto provoca que la codificación LDPC deje de tener buenas propiedades y las curvas BER-SNR resultantes pierdan pendiente en la región de *Waterfall*. Por lo tanto, se necesita obtener un rango en el cuál el *parámetro d* sea lo suficientemente bajo para que la matriz generadora sea muy dispersa, pero aportando suficiente información del resto de nodos de la red.

De esta manera, el objetivo es encontrar el rango de funcionamiento óptimo para el *parámetro d* , que ofrezca un buen compromiso entre que el umbral u_1 se sitúe a valores de SNR menores y que los valores de BER obtenidos en la región de *Error Floor* sean menores.

La Figura 4.4 representa las curvas que se han obtenido en las simulaciones para el caso de una agrupación de 200 nodos, donde el valor del *parámetro d* varía entre 3 y 20, utilizando como máximo 10 iteraciones en el algoritmo de decodificación. En esta figura se aprecia que para valores de este parámetro que se encuentren entre 5 y 13 se dan buenos resultados al utilizar codificación con LDPC.

Al inspeccionar las curvas BER frente a SNR de la Figura 4.4, se ve que para valores del *parámetro d* inferiores a 5 la región de *Waterfall* es muy corta y comienza a valores de SNR muy bajos, provocando que los valores de BER obtenidos sean bastante altos en la región de *Error Floor*. También se aprecia que para valores del *parámetro d* de entre 13 y 16 se da una mayor caída de la BER, pero a valores de SNR demasiado altas. Si se tiene en cuenta que el rango de valores de SNR típico en comunicaciones inalámbricas está por debajo de los 25 dB, no tiene sentido utilizar valores del *parámetro d* que produzcan muy buenos resultados a valores de SNR tan altas, como ocurre para esos valores. Además, para valores del *parámetro d* superiores a 17 se ve que la pendiente de la curva en la zona

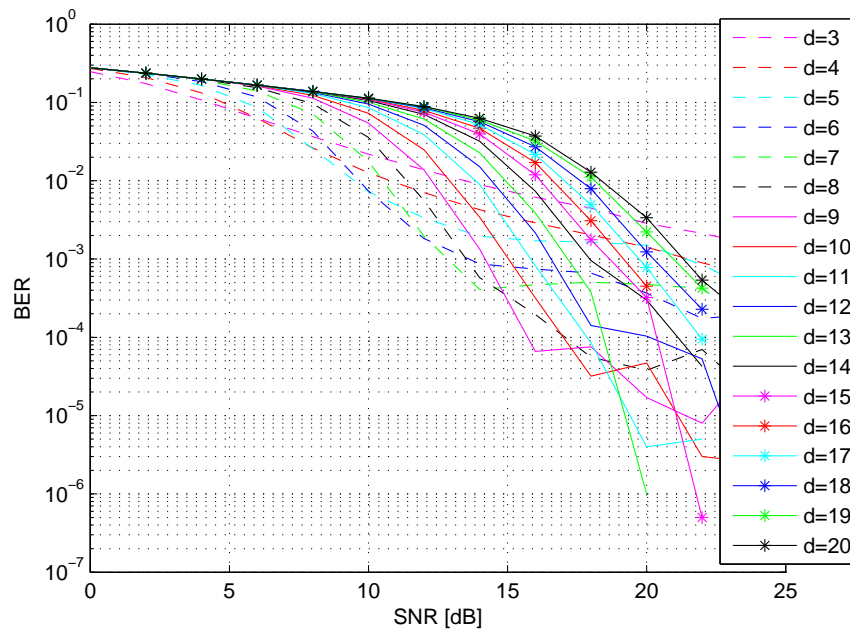


Figura 4.4: Curva BER-SNR obtenida para una codificación LDPC distribuida (primer nivel del algoritmo presentado) en una WSN de 200 nodos. El número máximo de iteraciones del algoritmo de decodificación es 10 y el valor del *parámetro* d varía entre 3 y 20.

abrupta se vuelve menos pronunciada que la que se da para los valores óptimos. Esto se debe a que para esos valores las matrices generadoras del código LDPC dejan de ser muy dispersas, mostrando la necesidad de encontrar el rango de valores para este *parámetro* d en que ofrece unas mejores prestaciones.

Como se explicó al principio del capítulo, se considerará que una curva es mejor que otra en función de un compromiso entre que el umbral $u1$ se sitúe a valores de SNR más bajas y que a valores de SNR altas se obtengan valores de BER menores. Para ello se comparan las mejores curvas y se eligen como mejores las que presenten un mayor compromiso. En este caso, no se puede asegurar que el rango que se eligió anteriormente para el *parámetro* d (entre 5 y 13) sea válido para agrupaciones compuestas por un número diferente de nodos, ya que el rango varía en función del número de nodos, al influir ambos parámetros en la dispersión de la matriz.

4.1.3. Influencia del número de nodos de la red

En el capítulo anterior, se expuso la necesidad de que los códigos LDPC sean generados por matrices muy dispersas, ya que esto favorece el proceso de decodificación. El número de nodos que intervienen en la codificación influye en el tamaño de la matriz generadora, ya que para un número N de nodos, la matriz generadora tendrá un tamaño de $N \times 2N$. Así, si se fija el valor del *parámetro* d , un número mayor de nodos provocaría

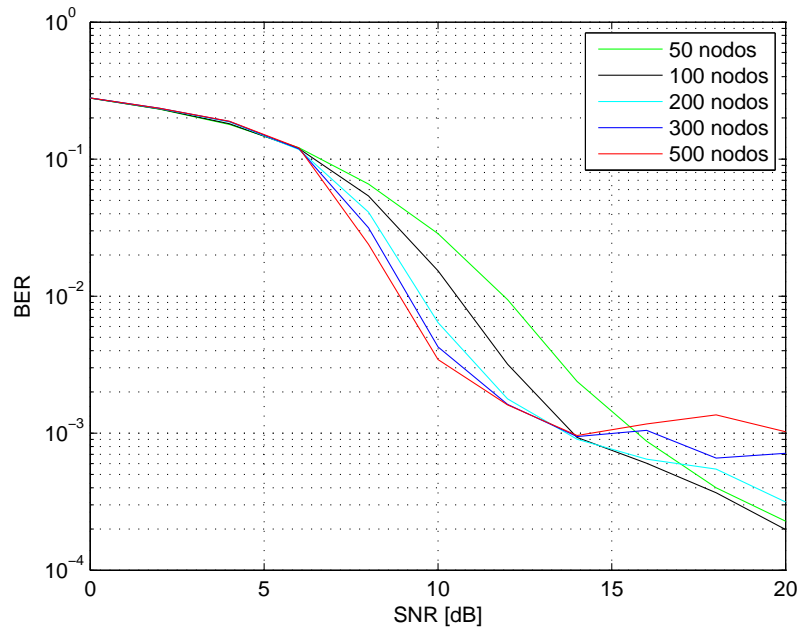


Figura 4.5: Curva BER-SNR obtenida para una codificación LDPC distribuida (primer nivel del algoritmo presentado) en una WSN para diferentes números de nodos. El número de iteraciones del algoritmo de decodificación es 10, el valor del *parámetro* d es 6 y el número de nodos varía tomando valores de 50, 100, 200, 300 y 500 nodos.

una mayor dispersión de la matriz generadora, ya que el ratio de valores no nulos respecto del total de elementos de la matriz será menor.

Se va a definir el grado de dispersión GD como la relación entre los símbolos que se combinan en la fase de reenvío y el número de nodos que pueden intervenir en dicha combinación. El grado de dispersión hace referencia a la dispersión que tiene la matriz generadora del código LDPC, al relacionar directamente los dos parámetros principales que afectan a dicha dispersión. Este grado de dispersión GD se va a calcular como:

$$GD = \frac{d}{N - 1} \quad (4.1)$$

Donde d es el valor del *parámetro* d y N el número total de nodos. Este parámetro GD corresponde al porcentaje de símbolos que intervienen en la respuesta en la fase de reenvío de entre los símbolos que podrían intervenir. El objetivo es ver si se puede obtener un rango de valores de GD entre los que el algoritmo se comporte de manera óptima, con el fin de poder establecer a priori los valores del *parámetro* d y del número de nodos que dan las mejores curvas BER-SNR.

En la Figura 4.5 se representan las curvas que se han obtenido en las simulaciones para el caso de un valor fijo del *parámetro* d de 6 y empleando conjuntos de nodos de

diferentes tamaños. Se ha elegido realizar las simulaciones con conjuntos de 50, 100, 200, 300 y 500 nodos por ser números redondos en los que se pueden dividir el conjunto de nodos totales y que provoca que haya entre 3 y 30 clusters de cara al segundo nivel.

Si se analiza la Figura 4.5, se ve cómo al aumentar el número de nodos se aumenta también la pendiente que aparece en la zona de valores de SNR medias. Esto provoca que se obtengan mejores valores de BER a valores de SNR más bajas. Los valores de BER obtenidos a valores de SNR altas para las curvas que corresponden a 300 y 500 nodos son, en este caso peores que los que se obtienen para un número menor de nodos. Este hecho se puede deber a que para esos conjuntos de nodos se esté utilizando unos valores del *parámetro d* que provoquen que las matrices generadoras sean menos dispersas y, por lo tanto, la pendiente en la región de *Waterfall* sea menor.

Número de nodos	GD
50 nodos	12,24 %
100 nodos	6,06 %
200 nodos	3,02 %
300 nodos	2,01 %
500 nodos	1,2 %

Cuadro 4.1: Grados de dispersión para $d=6$ y distinto número de nodos.

Los grados de dispersión que se dan en este ejemplo son los que se resumen en el Cuadro 4.1. A la vista de los resultados obtenidos en este Proyecto, se puede concluir que los valores óptimos del grado de dispersión dependen del número de nodos de la red, viéndose en dichos resultados que para valores del grado dispersión superiores a un 6 % la pendiente de las curvas BER-SNR se suavizan notablemente en su región de *Waterfall*, motivadas por la falta de dispersión de la matriz generadora.

4.2. Análisis del Primer Nivel

Como se explicó en el apartado anterior, el número de nodos es uno de los parámetros fundamentales a la hora de elaborar un algoritmo basado en Codificación de Red con códigos LDPC, ya que afecta directamente al tamaño de la matriz generadora del código y a su dispersión.

Si definimos el número de nodos en el primer nivel como N_{nivel1} , el tamaño de la matriz generadora será $N_{nivel1} \times 2 \cdot N_{nivel1}$. Esto lleva a que uno de los objetivos sea hacer agrupaciones para el primer nivel de gran tamaño, englobando al mayor número de nodos posible. Además, el tamaño de estas agrupaciones influye de cara al segundo nivel, ya que si se define el número total de nodos como N_{total} y se dividen en agrupaciones regulares de

N_{nivel1} nodos se tiene que el número de clusters y, por lo tanto, número de nodos cabeza de cluster para el segundo nivel es de:

$$N_{clusters} = \frac{N_{total}}{N_{nivel1}} \quad (4.2)$$

En esta sección se van a analizar los datos obtenidos de las simulaciones para el primer nivel del algoritmo, que corresponde al envío de datos de los nodos de cada agrupación a su nodo cabeza de cluster. Se empezará por dividir los distintos escenarios en función del número de nodos que englobe cada agrupación. Dentro de cada escenario también se estudiará el comportamiento en cada agrupación para distintos valores del *parámetro d*.

Para seguir una línea continuista con los trabajos presentados sobre el esquema ANCC en [Bao and Li, 2008] y [Morgado, 2009] donde utilizaban un número total de 1000 nodos, en este trabajo se ha elegido utilizar un número total de nodos que se encuentre en el mismo orden de magnitud. Para las simulaciones se ha utilizado un total de 1000 ó 1500 nodos totales, dependiendo del número de nodos por agrupación en el primer nivel y de manera que pueda dar un valor exacto de nodos cabeza de cluster en el segundo nivel.

4.2.1. Agrupaciones de 50 nodos

El primer caso que se va a analizar en esta sección es el que consiste en utilizar agrupaciones de 50 nodos en el primer nivel. Con 50 nodos, la matriz generadora tendrá un tamaño de 50x100, que podría considerarse un tamaño pequeño, debido a los altos grados de dispersión que origina. El rango en el que se va a estudiar el comportamiento del *parámetro d* es el que comprende entre 4 y 8, ya que es el que ofrece una curva BER-SNR mejor.

Parámetro d	GD
4	8,16 %
5	10,2 %
6	12,24 %
7	14,29 %
8	16,33 %

Cuadro 4.2: Grados de dispersión para 50 nodos y distinto valores del *parámetro d*.

La Figura 4.6 representa las distintas curvas obtenidas para una agrupación de 50 nodos, variando el valor del *parámetro d* y usando un máximo de 10 iteraciones del algoritmo de decodificación Suma-Producto. En dicha figura se ve cómo al aumentar el valor del *parámetro d* la pendiente de las curvas apenas se ve afectada. Sin embargo, el

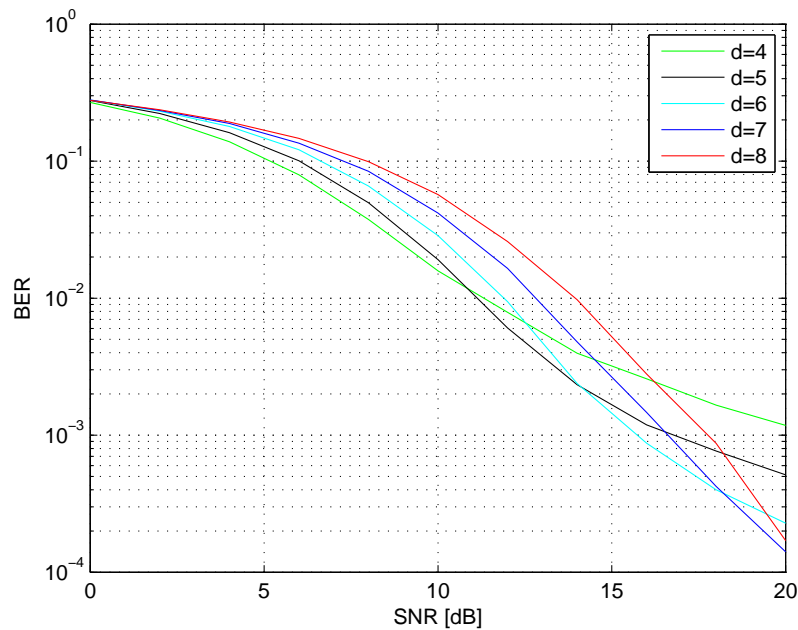


Figura 4.6: Curva BER-SNR obtenida para una codificación LDPC distribuida (primer nivel del algoritmo presentado) en una WSN de 50 nodos. El número máximo de iteraciones del algoritmo de decodificación es 10 y el valor del *parámetro* d varía entre 4 y 8.

umbral u_1 a partir del cual comienza la región de *Waterfall* se da cada vez a valores más altos de SNR. Además, la pendiente en la región de *Waterfall* es muy poco pronunciada, lo que provoca que los valores de BER obtenidos sean bastante altos. La curva que presenta un mejor compromiso entre la posición del umbral u_1 y los valores de BER obtenidos en la región de *Error Floor* es la curva correspondiente a $d=6$.

Los grados de dispersión que presentan estas curvas se resumen en el Cuadro 4.2. Estos valores del grado de dispersión son demasiado altos y se podrían asociar más a matrices dispersas que a muy dispersas, con lo que no parece buena opción la de utilizar agrupaciones de este tamaño.

4.2.2. Agrupaciones de 100 nodos

El segundo caso a analizar es el de tener agrupaciones de 100 nodos en el primer nivel. Con 100 nodos, la matriz generadora de la Codificación de Red con códigos LDPC va a tener un tamaño de 100×200 . Además, se variará el *parámetro* d de 5 a 11 para ver cuál de ellos es el que ofrece una curva BER contra SNR con un compromiso mejor.

La Figura 4.7 representa las distintas curvas obtenidas para una agrupación de 100 nodos variando el valor del *parámetro* d y usando un máximo de 10 iteraciones del algoritmo de decodificación Suma-Producto. En esta figura se aprecia que para valores del

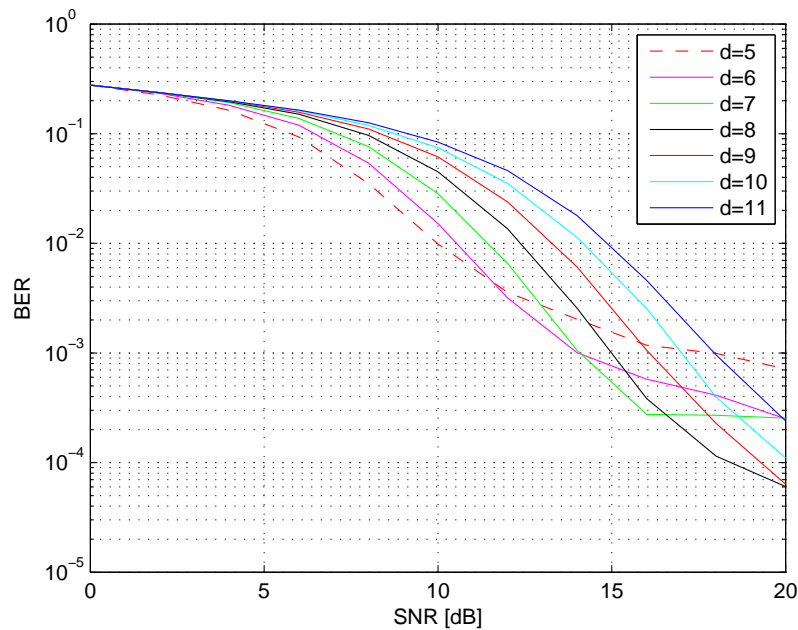


Figura 4.7: Curva BER-SNR obtenida para una codificación LDPC distribuida (primer nivel del algoritmo presentado) en una WSN de 100 nodos. El número máximo de iteraciones del algoritmo de decodificación es 10 y el valor del *parámetro d* varía entre 5 y 10.

Parámetro d	GD
6	6,06 %
7	7,07 %
8	8,08 %

Cuadro 4.3: Grados de dispersión para 100 nodos y distinto valores del *parámetro d*.

parámetro d superiores a 8 la región de *Error Floor* aparece para valores de SNR muy altos (por encima de los 18 dB), lo que provoca que se obtengan buenos valores de BER a valores de SNR muy altos, situado en muchos casos por encima del rango de SNR en el que se encuentra un sistema inalámbrico.

Analizando las gráficas, se puede ver que las curvas con un mayor compromiso se dan para los valores del *parámetro d* de 6, 7 y 8, cuyas matrices ofrecen los grados de dispersión que se muestran en el Cuadro 4.3.

4.2.3. Agrupaciones de 200 nodos

El tercer caso que se va a analizar es el de dividir los nodos en agrupaciones de 200 nodos en el primer nivel. Con 200 nodos, la matriz generadora de la Codificación de Red con códigos LDPC tiene un tamaño de 200x400. Además, se variará el *parámetro d* de 7 a 12 para ver cual de ellos es el que ofrece una curva BER contra SNR mejor.

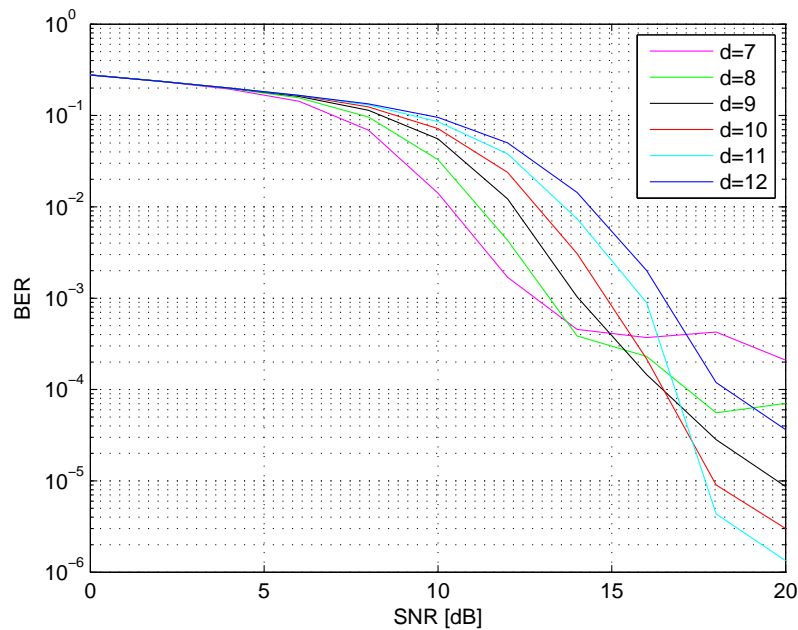


Figura 4.8: Curva BER-SNR obtenida para una codificación LDPC distribuida (primer nivel del algoritmo presentado) en una WSN de 200 nodos. El número máximo de iteraciones del algoritmo de decodificación es 10 y el valor del *parámetro d* varía entre 7 y 12.

Parámetro d	GD
8	4,02 %
9	4,52 %
10	5,03 %

Cuadro 4.4: Grados de dispersión para 200 nodos y distinto valores del *parámetro d*.

La Figura 4.8 representa las distintas curvas obtenidas para una agrupación de 200 nodos, variando el valor del *parámetro d* y usando un máximo de 10 iteraciones del algoritmo de decodificación Suma-Producto.

El análisis para este caso se realiza de manera similar que en los casos anteriores. En esta figura se ve que las curvas BER-SNR que obtienen un mayor compromiso son para valores del *parámetro d* de 8, 9 y 10, y cuyos grados de dispersión se muestran en el Cuadro 4.4.

4.2.4. Agrupaciones de 300 nodos

El cuarto caso a analizar es el que consiste en dividir los nodos en agrupaciones de 300 nodos en el primer nivel. Con 300 nodos, la matriz generadora del código LDPC tiene un tamaño de 300×600 . Además, como en los casos anteriores, se variará el *parámetro d* de 7 a 12 y se analizarán las curvas BER contra SNR resultantes.

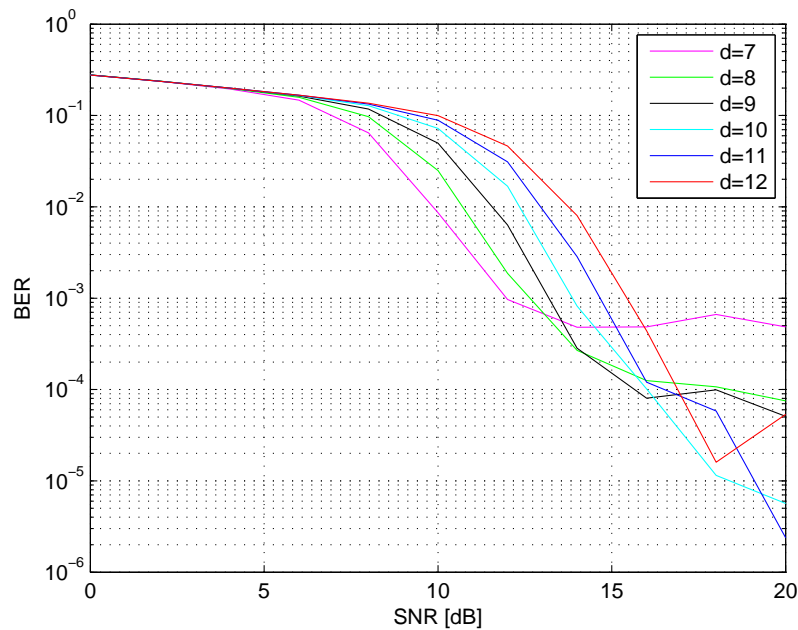


Figura 4.9: Curva BER-SNR obtenida para una codificación LDPC distribuida (primer nivel del algoritmo presentado) en una WSN de 300 nodos. El número máximo de iteraciones del algoritmo de decodificación es 10 y el valor del *parámetro d* varía entre 7 y 12.

La Figura 4.9 representa las distintas curvas obtenidas para una agrupación de 300 nodos, variando el valor del *parámetro d*. En total, las simulaciones se han realizado utilizando un máximo de 10 iteraciones del algoritmo de decodificación Suma-Producto.

Parámetro d	GD
8	2,68 %
9	3,01 %
10	3,34 %

Cuadro 4.5: Grados de dispersión para 300 nodos y distinto valores del *parámetro d*.

Observando las gráficas resultantes para estas agrupaciones, se puede apreciar que las mejores curvas obtenidas son para los valores del *parámetro d* comprendidos entre 8 y 10. Para estos valores del *parámetro d*, se obtienen los grados de dispersión que se marcan en el Cuadro 4.5.

4.2.5. Agrupaciones de 500 nodos

El quinto y último caso que se va a analizar en esta sección es el caso en el que se dividen los nodos en agrupaciones de manera que cada agrupación engloba un total de 500 nodos en el primer nivel. Con 500 nodos, la matriz generadora del código LDPC tiene un tamaño de 500x1000. Además, como en los casos anteriores, se variará el *parámetro d* de 8 a 13 y se analizarán las curvas BER contra SNR obtenidas en las simulaciones.

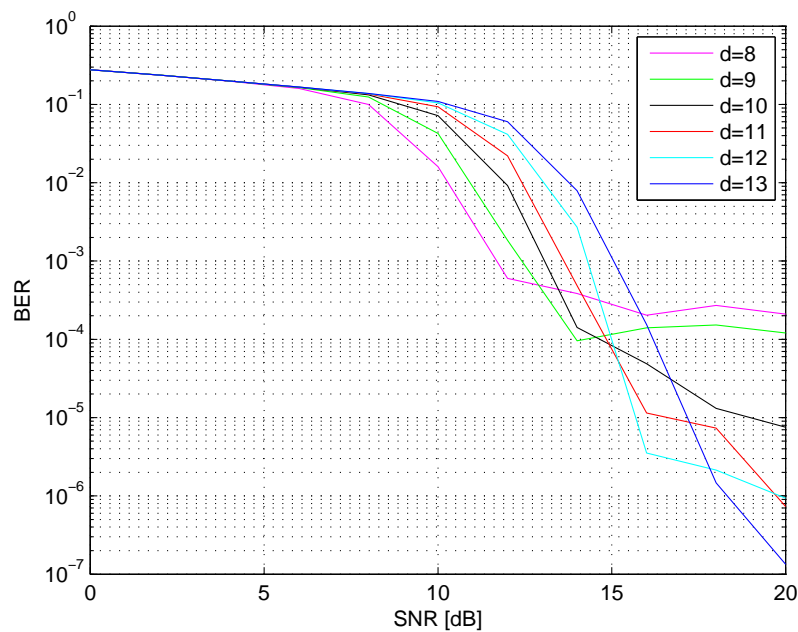


Figura 4.10: Curva BER-SNR obtenida para una codificación LDPC distribuida (primer nivel del algoritmo presentado) en una WSN de 500 nodos. El número máximo de iteraciones del algoritmo de decodificación es 10 y el valor del *parámetro d* varía entre 8 y 13.

Parámetro d	GD
9	1,8 %
10	2 %
11	2,2 %
12	2,4 %

Cuadro 4.6: Grados de dispersión para 500 nodos y distinto valores del *parámetro d*.

La Figura 4.10 representa las curvas obtenidas en las simulaciones utilizando agrupaciones de 500 nodos, variando el valor del *parámetro d* entre 8 y 13 y empleando un máximo de 10 iteraciones del algoritmo de decodificación Suma-Producto.

Observando las gráficas resultantes para estas agrupaciones, se puede apreciar que las curvas que ofrecen un mejor compromiso se dan para los valores del *parámetro d* comprendidos entre 9 y 12. Los grados de dispersión obtenidos son los que se detallan en el Cuadro 4.6.

4.2.6. Mejores opciones para el primer nivel

En esta sección se van a analizar conjuntamente las mejores curvas BER frente a SNR obtenidos para los escenarios anteriormente descritos. Se ha considerado para cada agrupación una de las curvas seleccionadas con mejor compromiso entre empezar a caer

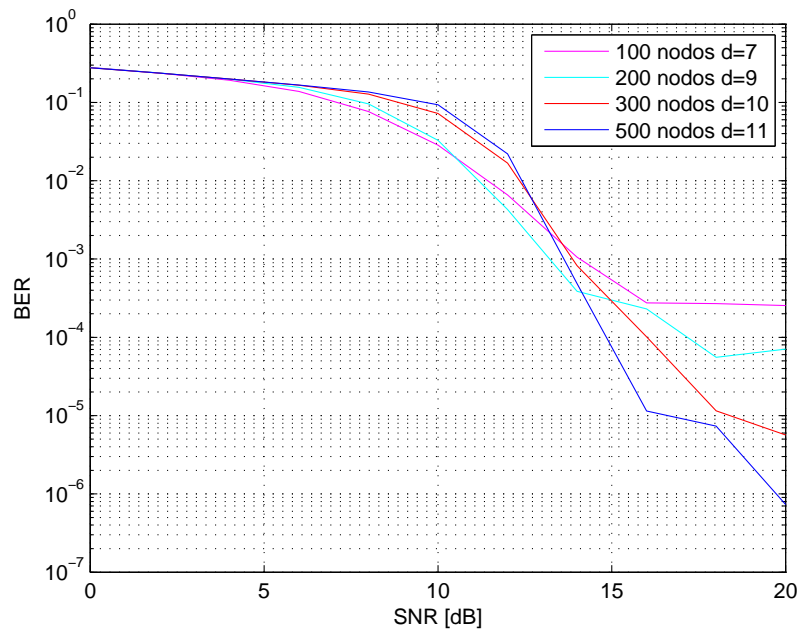


Figura 4.11: Comparación de las curvas BER-SNR obtenidas para una codificación LDPC distribuida (primer nivel del algoritmo presentado) y que ofrecen un mejor compromiso. El número máximo de iteraciones del algoritmo de decodificación en todas las curvas es 10 y se comparan agrupaciones de 100 nodos y *parámetro* $d = 7$, de 200 nodos y *parámetro* $d = 9$, de 300 nodos y *parámetro* $d = 10$ y de 500 nodos y *parámetro* $d = 11$.

a valores de SNR más bajas y obtener valores de BER mayores a valores de SNR altos.

Las curvas que se representan en la Figura 4.11 son las curvas que ofrecen un mejor compromiso para cada agrupación de las estudiadas para el primer nivel. En dicha figura no se ha incluido el caso de 50 nodos por agrupación al ofrecer unos valores de BER bastante superiores al resto de casos analizados. De esta manera, se ha elegido para agrupaciones de 100 nodos un valor del *parámetro* d de 7, para agrupaciones de 200 nodos un valor del *parámetro* d de 9, para agrupaciones de 300 nodos un valor del *parámetro* d de 10 y para agrupaciones de 500 nodos un valor del *parámetro* d de 11. Los grados de dispersión que se corresponden con esas curvas son los que se muestran en el Cuadro 4.7.

Número nodos	Parámetro d	GD
100 nodos	7	7,07 %
200 nodos	9	4,52 %
300 nodos	10	3,34 %
500 nodos	11	2,2 %

Cuadro 4.7: Grados de dispersión para las mejores curvas del primer nivel.

Al analizar las curvas obtenidas en la la Figura 4.11 se ve que la pendiente de la curva BER-SNR en la región de *Waterfall* aumenta cuando se utiliza un mayor número de nodos por agrupación. De la misma manera también se obtienen unos valores de BER menores para el caso de agrupaciones con mayor número de nodos. Esto se debe a que al

aumentar el número de nodos que intervienen en la codificación la matriz generadora del código se hace más dispersa y este hecho provoca que los códigos LDPC tengan mejores propiedades.

A la vista de los resultados obtenidos, se puede concluir que las mejores curvas BER frente a SNR obtenidas para el primer nivel corresponden a agrupaciones de 300 y 500 nodos. Como 500 nodos en el primer nivel implica que el número de nodos cabeza de cluster sea muy bajo, se va a probar con agrupaciones de 300 y 100 nodos para la segunda etapa de nuestro algoritmo.

En la segunda etapa del algoritmo, cada nodo cabeza de cluster envía todos los símbolos que ha recibido en su agrupación. La segunda etapa se va a realizar suponiendo que el conjunto total de nodos de la red es de 1500 nodos y, por lo tanto, al dividirlo en agrupaciones de 300 nodos se tiene un total de 5 nodos cabezas de cluster y si se divide en agrupaciones de 100 nodos supone un total de 15 nodos cabeza de cluster.

4.3. Análisis del Segundo Nivel

En la sección anterior se analizó cómo el número de nodos que tiene cada agrupación es uno de los parámetros fundamentales a la hora de diseñar la red que implemente el algoritmo que se está evaluando en este Proyecto. En la ecuación (4.2) se mostraba que el número de nodos de las agrupaciones determina el número de nodos cabeza de cluster que hay, al utilizar agrupaciones regulares con el mismo tamaño cada una.

El esquema de transmisión del primer nivel consiste en que un conjunto de N_{nivel1} nodos en la fase de difusión transmiten sus símbolos mientras el resto de nodos permanecen en silencio escuchando la transmisión. De esta manera, cada nodo escucha a cada uno de los nodos de la agrupación a través de un canal distinto y combina la información recibida de cara a la fase de reenvío. Esto permite que las columnas de la matriz generadora estén incorreladas, ofreciendo así unas buenas prestaciones de BER frente a SNR. A la vista de los resultados de dicha sección se puede concluir que al aumentar el tamaño de las agrupaciones se obtienen mejores resultados, como se observa en los casos de tener agrupaciones de 300 y 500 nodos.

En el segundo nivel, al elegir un número tan grande de nodos por agrupación, se tienen muy pocas agrupaciones y esto afecta negativamente a nuestro algoritmo, ya que habrá pocos nodos cabeza de agrupación que vayan a transmitir en el segundo nivel. Si se siguiera el mismo esquema que en el nivel anterior, habría un total de $N_{clusters}$ nodos cabeza de cluster que transmiten los N_{nivel1} símbolos que recibieron en su agrupación, y

por lo tanto cada nodo cabeza de cluster es responsable de N_{nivel1} columnas de la matriz generadora. Esto hace que las matrices generadoras que aparecen en este nivel sean más grandes que las del nivel anterior, pero en este nivel hay menos nodos que cooperen, con lo que si el canal que se da entre los nodos no varía a lo largo de la transmisión va a haber únicamente $N_{clusters}$ canales independientes. En estas condiciones, los símbolos recibidos procedentes del mismo nodo se van a encontrar correlados, y si en la fase de reenvío se eligieran únicamente los símbolos mejor recibidos provocaría que la mayor parte de las columnas de la matriz generadora se encuentren correladas, con lo que muchas columnas de esta matriz no ofrecen ninguna información adicional y por lo tanto se pierden las buenas propiedades que ofrece usar este tipo de codificación, a pesar del hecho de que las matrices generadoras sean más dispersas.

Para afrontar el problema de tener pocas fuentes en este segundo nivel, se ha elegido una alternativa que consiste en que cada nodo cabeza de cluster elija de cara a la fase de reenvío y de manera aleatoria un total de d símbolos de entre todos los símbolos que ese nodo ha recibido correctamente. Cada nodo cabeza de cluster repite este paso un total de N_{nivel1} veces, uno por cada columna que le corresponda de la matriz generadora en la fase de reenvío.

Cada nodo cabeza de cluster un número alto de símbolos en el segundo nivel, con lo que es necesario estudiar cómo afecta el tiempo de coherencia (T_c) a nuestro sistema. Un T_c elevado conlleva una variación lenta del canal y, por tanto, los símbolos transmitidos de forma consecutiva verían el mismo canal, apareciendo correlación en las columnas de la matriz LDPC generadora. Lo deseable para el buen funcionamiento de nuestro algoritmo sería que el T_c fuera igual al tiempo de símbolo (T_s), haciendo que el canal varíe cada símbolo de manera equivalente a como sucedía en el primer nivel al provenir cada símbolo de un nodo distinto.

En el algoritmo que se propone en este Proyecto no se puede variar sobre el T_c del canal, pero sí se puede hacer que los nodos transmitan por ráfagas de símbolos. De esta forma, si el tiempo que transcurre entre ráfagas de símbolos es mayor que el T_c , entonces el canal habrá variado respecto a la ráfaga anterior. Así, en los siguientes apartados, en lugar de sobre T_c se va a trabajar con el tiempo de ráfaga ($T_R = R \cdot T_s$), definido como el tiempo que se tarda en transmitir R símbolos seguidos, y se puede asumir que el canal varía entre ráfaga y ráfaga.

El análisis de este segundo nivel se va a dividir en tres partes: en la primera se analiza cuales son los mejores valores del parámetro d para un total de 1500 símbolos transmitidos. Esto supone un caso ideal para el segundo nivel, ya que se supone cada símbolo se recibe por un canal independiente, equiparando el valor de T_R a T_s . En la segunda y tercera parte, se analiza para los valores de d obtenidos en la parte anterior como se comportan un total de 5 y 15 nodos cabeza de cluster en el segundo nivel, respectivamente. Este

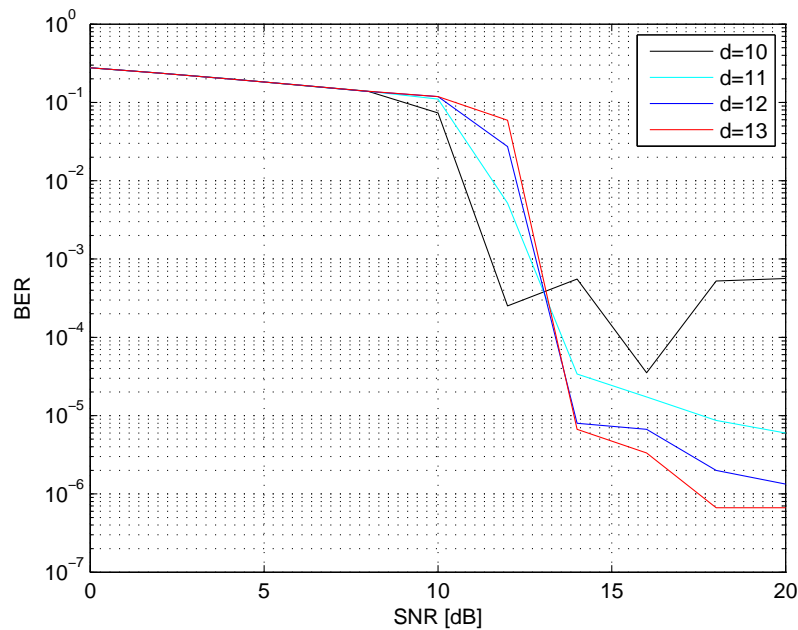


Figura 4.12: Curva BER-SNR obtenida para una codificación LDPC distribuida (segundo nivel del algoritmo presentado) en una WSN de 1500 nodos en condiciones ideales, con 1500 clusters y tamaño de ráfaga igual a un tiempo de símbolo. El número máximo de iteraciones del algoritmo de decodificación es 10 y el valor del *parámetro* d varía entre 10 y 13.

análisis se realiza para distintos tamaños de ráfaga.

4.3.1. Caso ideal en el Segundo Nivel

En esta sección se va a analizar el caso ideal de este segundo nivel para un total de 1500 nodos. Esta sección tiene un único objetivo: obtener los valores del *parámetro* d para los que mejor se comportaría el algoritmo en el segundo nivel, por lo que se puede suponer que es el límite al que se acercarían los resultados del segundo nivel en condiciones ideales. Este caso es inalcanzable por las condiciones en las que se da, ya que se ha supuesto que los clusters están compuestos por un único nodo y, por lo tanto, hay un total de 1500 clusters. Este caso es equivalente a estudiar un primer nivel en el que el tamaño de las agrupaciones sea de 1500 nodos por agrupación. De este caso se puede extraer cuáles son los valores adecuados del *parámetro* d para poder utilizarlo en la codificación en este segundo nivel.

En la Figura 4.12 se ve la curva obtenida para las condiciones ideales y aplicando las asunciones descritas para este caso. Se ha simulado la existencia de 1500 clusters compuestos cada uno por un único nodo, transmitiendo cada uno un único símbolo. Además, se ha equiparado el tamaño de la ráfaga a un tiempo de símbolo. El número máximo de iteraciones del algoritmo de decodificación se ha mantenido con 10 iteraciones, igual que

sucedía para el primer nivel. Los valores del *parámetro* d se han variado entre 10 y 13, siendo para $d=12$, $d=13$ las curvas que ofrecían un mejor compromiso. De esta manera, para los escenarios del segundo nivel se va a realizar un análisis para distintos tamaños de ráfaga con $d=12$ y $d=13$.

4.3.2. Segundo Nivel: 5 clusters

En esta sección se representan los datos obtenidos para el segundo nivel del algoritmo si el total de nodos de la red se dividiera en 5 agrupaciones regulares. Esta división provoca que haya 5 nodos cabeza de cluster que, de acuerdo a la ecuación (4.2), transmiten un total de 300 símbolos cada uno, uno por cada nodo de su agrupación.

En este nivel, se va a analizar cómo afectan los tamaños de las ráfagas que se transmiten a las curvas BER-SNR. En concreto se han elegido 6 tamaños distintos de ráfaga para este caso: 300 símbolos, 150 símbolos, 30 símbolos, 10 símbolos, 5 símbolos y 1 símbolo por ráfaga. Estos tamaños se han elegido de manera que se vean el caso mejor (cuando el canal varía a cada tiempo de símbolo) y los casos peores (cuando el canal se mantiene constante para la transmisión de todos los símbolos del nodo o para la mitad de símbolos) y un par de casos intermedios.

En este segundo nivel, el valor del grado de dispersión GD no se puede hallar de la misma manera que en el caso del primer nivel, ya que los símbolos no son independientes. Atendiendo a la definición, GD es el cociente entre los símbolos que intervienen en la respuesta de la fase de reenvío entre los símbolos posibles. En nuestro segundo nivel, se transmiten un total de 1500 símbolos de los que 300 son transmitidos por cada nodo cabeza de cluster. De esta manera, solo 1200 símbolos son los que pueden intervenir en la respuesta de la fase de reenvío, ya que los otros 300 corresponden al mismo nodo. De esta manera, el valor de GD para este caso es de un 1 % para el caso de $d=12$ y de 1,08 % en el caso de $d=13$.

En las Figuras 4.13 y 4.14 se ve el comportamiento para 5 nodos cabeza de cluster y valores del *parámetro* d de 12 y 13, respectivamente. En ambos casos, el número máximo de iteraciones del algoritmo de decodificación es de 10 iteraciones y se ha estudiado para los tamaños de ráfaga que se ven en dicha figura.

Se aprecia en ambas figuras que, para ráfagas muy pequeñas ($R=1$, $R=4$ y $R=10$) las curvas BER-SNR se asemejan a las mostradas para las condiciones ideales, siendo mejores cuanto menor es el tamaño de la ráfaga. Para tamaños de ráfaga muy grandes (de $R=300$ y $R=150$) se aprecian unos valores de BER muy malos, ya que nunca llega a aparecer la curva típica de los códigos LDPC, lo que provoca que los errores que sucedan en este salto

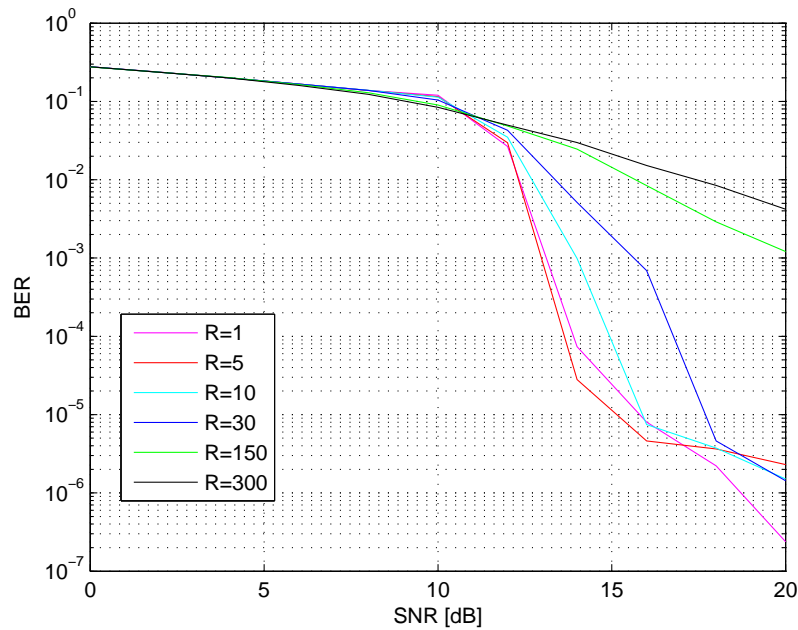


Figura 4.13: Curva BER-SNR obtenida para una codificación LDPC distribuida (segundo nivel del algoritmo presentado) en una WSN de 1500 nodos, con 5 nodos cabeza de clusters y valor del *parámetro* d de 12. El número máximo de iteraciones del algoritmo de decodificación es 10 y el valor de R es de 1, 5, 10, 30, 150 y 300 símbolos por ráfaga.

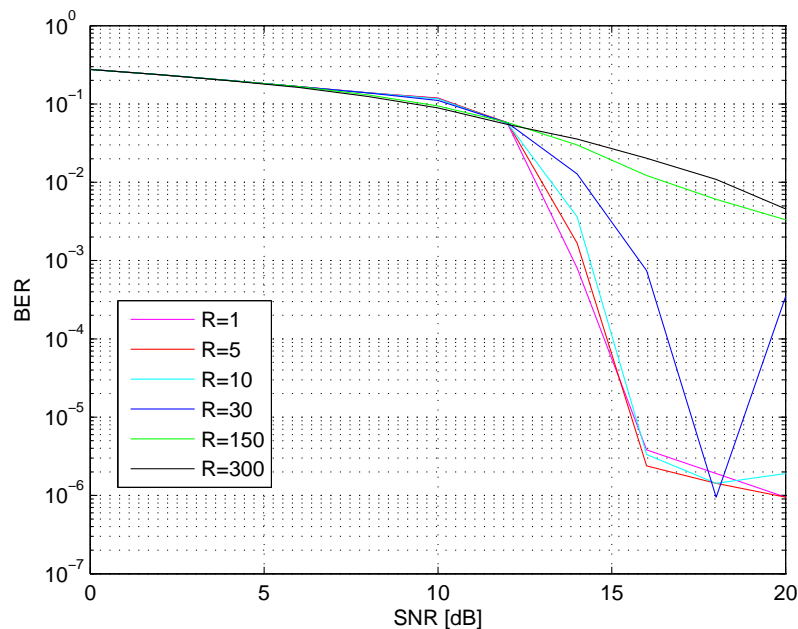


Figura 4.14: Curva BER-SNR obtenida para una codificación LDPC distribuida (segundo nivel del algoritmo presentado) en una WSN de 1500 nodos, con 5 nodos cabeza de clusters y valor del *parámetro* d de 13. El número máximo de iteraciones del algoritmo de decodificación es 10 y el valor de R es de 1, 5, 10, 30, 150 y 300 símbolos por ráfaga.

hagan que la codificación no produzca ninguna ventaja respecto al caso sin codificar. Esto se debe a que la información que llega al nodo central se encuentra demasiado correlada, y por lo tanto, las columnas de la matriz generadora del código no son independientes y el algoritmo de decodificación no puede funcionar de manera correcta. Para el caso intermedio, con ráfagas de 30 símbolos, se ve cómo la curva se ha desplazado hacia la derecha con respecto al mejor de los casos, en el que el tamaño de la ráfaga es equivalente a un tiempo de símbolo.

En este escenario se ha demostrado que el estado del canal es el factor más importante para modelar el segundo nivel, mostrando muy buenas prestaciones para canales que presentan tiempos de coherencia muy bajos y sin aportar ventajas con el caso de no codificar para tiempos de coherencia prolongados.

4.3.3. Segundo Nivel: 15 clusters

En el apartado anterior se analizaban las curvas BER-SNR obtenidas en el segundo nivel del algoritmo para el caso de tener 5 clusters. En esta sección se va a hacer un análisis muy similar, pero empleando 15 clusters y, por lo tanto, 15 nodos cabeza de cluster. Al utilizar 15 agrupaciones y de acuerdo a la ecuación (4.2), cada nodo transmite un total de 100 símbolos cada uno, uno por cada nodo proveniente de su agrupación.

En este apartado, se va a analizar cómo afectan los tamaños de las ráfagas que se transmiten a las curvas BER-SNR de manera similar al análisis del apartado anterior, pero con diferentes tamaños de ráfaga, ya que hay ráfagas en el caso anterior que englobarían a un número mayor de símbolos de los que va a transmitir cada nodo en esta fase. En este caso, se han elegido 5 tamaños distintos de ráfaga: 100 símbolos, 50 símbolos, 10 símbolos, 5 símbolos y 1 símbolo por ráfaga. Al igual que en el apartado anterior, estos tamaños se han elegido de manera que se vean el caso mejor (cuando el canal varía a cada tiempo de símbolo) y los casos peores (cuando el canal se mantiene constante para la transmisión de todos los símbolos del nodo o para la mitad de símbolos) y algunos casos intermedios.

El grado de dispersión GD para este apartado se calcula de la misma manera que en el caso anterior. En este caso, cada nodo cabeza de cluster participa en la codificación con 100 símbolos, con lo que hay 1400 símbolos que pueden intervenir en la respuesta. Esto da un valor de GD de 0,86 % para $d=12$ y de 0,93 % para $d=13$.

En las Figuras 4.15 y 4.16 se ve el comportamiento de las curvas BER frente a SNR para 15 nodos cabeza de cluster y valores del *parámetro* d de 12 y 13, respectivamente. En ambos casos, el número máximo de iteraciones del algoritmo de decodificación es de 10 iteraciones y se ha estudiado para los tamaños de ráfaga que se ven en dicha figura.

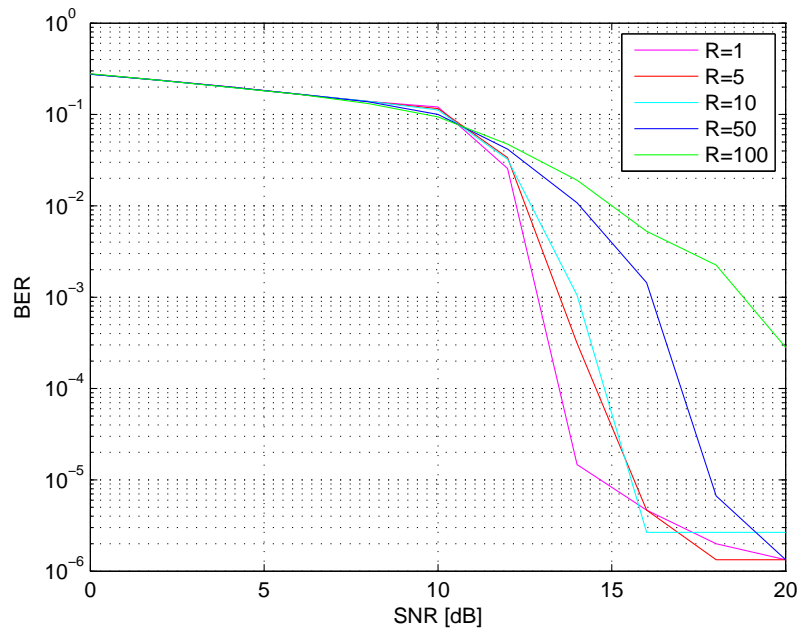


Figura 4.15: Curva BER-SNR obtenida para una codificación LDPC distribuida (segundo nivel del algoritmo presentado) en una WSN de 1500 nodos, con 15 nodos cabeza de clusters y valor del *parámetro* d de 12. El número máximo de iteraciones del algoritmo de decodificación es 10 y el valor de R es de 1, 5, 10, 50 y 100 símbolos por ráfaga.

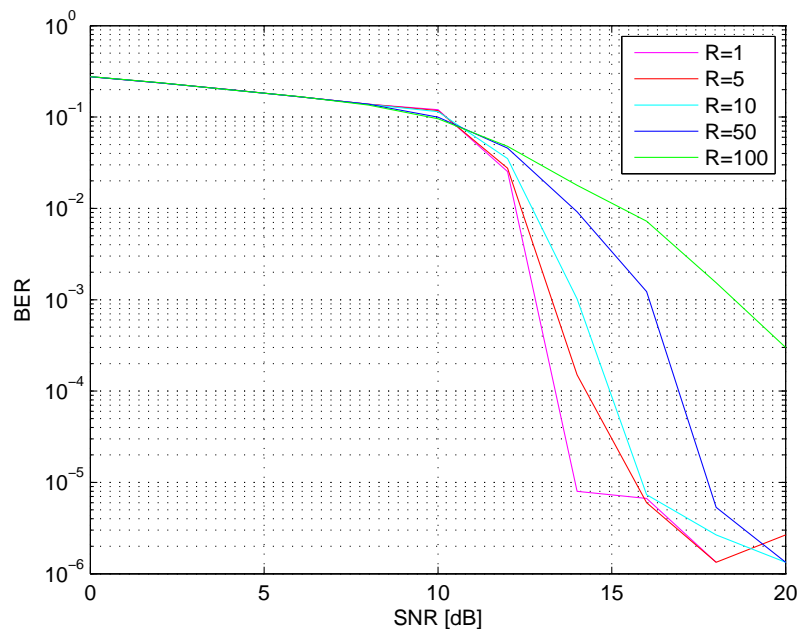


Figura 4.16: Curva BER-SNR obtenida para una codificación LDPC distribuida (segundo nivel del algoritmo presentado) en una WSN de 1500 nodos, con 15 nodos cabeza de clusters y valor del *parámetro* d de 13. El número máximo de iteraciones del algoritmo de decodificación es 10 y el valor de R es de 1, 5, 10, 50 y 100 símbolos por ráfaga.

Al igual que ocurría cuando había 5 nodos cabeza de cluster, para ráfagas muy pequeñas ($R=1$, $R=5$ y $R=10$) las curvas BER-SNR obtenidas ofrecen unos valores de BER en la región de *Error Floor* muy bajos. Sin embargo, cuando el tamaño de la ráfaga es proporcional al número de símbolos de la agrupación ($100 T_s$) la región de *Waterfall* aparece a valores de SNR muy altos y, por lo tanto la región de *Error Floor* se da para valores de SNR que están por encima de los típicos de una comunicación inalámbrica. Esto implica que, en el caso en que en las ráfagas se transmitan todos los símbolos procedentes de cada agrupación, los resultados de BER obtenidos para una comunicación inalámbrica sean mucho peores que en el caso en el que los canales varíen rápidamente, aunque los resultados de BER obtenidos se mantienen mejores que para el caso en el que había 5 clusters y el canal se mantenía constante para la transmisión de todos los símbolos. Que los resultados obtenidos en ese caso sean mejores que los obtenidos para 5 clusters se debe a que en el caso de 15 clusters hay menos columnas correladas de la matriz generadora del código LDPC, al haber un mayor número de nodos distintos que transmiten (15 en el caso de 100 nodos por agrupación, frente a los 5 del caso de 300 nodos por agrupación). Para el resto de tamaños de ráfaga, el algoritmo obtiene valores menores de BER en el caso de 300 nodos por agrupación.

4.4. Análisis del sistema completo

En este apartado se van a analizar los resultados del sistema completo a partir de los datos obtenidos para el primer y el segundo nivel del algoritmo. El objetivo es ver el comportamiento final de las curvas BER frente a SNR tras ambos saltos y compararlo con el caso de no usar codificación.

En la Sección 4.3 se analizaron dos escenarios posibles: uno en el que los nodos se dividían en 5 clusters compuesto cada uno por 300 nodos, y otro de 15 clusters con 100 nodos por cluster. En esta sección se van a ver los resultados finales tras estos dos niveles para estos los dos posibles escenarios.

De acuerdo con [Morgado et al., 2010], la BER en un escenario multisalto se calcula como la suma de probabilidades de que haya errores en alguno de los saltos habiendo recibido correctamente el dato en el resto de saltos:

$$BER_{tot} = \sum_{i=1}^H BER(i) \left(\prod_{j=i+1}^H (1 - 2 \cdot BER(j)) \right) \quad (4.3)$$

Donde BER_{tot} es la BER extremo a extremo, $BER(i)$ es la BER que se produce en

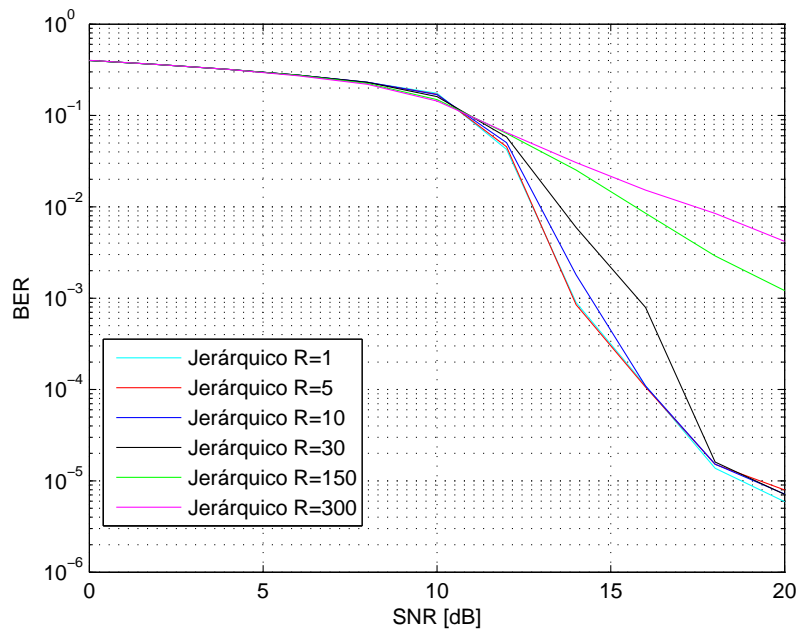


Figura 4.17: Curva BER-SNR obtenida para una codificación LDPC distribuida (a través de los dos niveles del algoritmo presentado) en una WSN de 1500 nodos, con 5 nodos cabeza de clusters y 300 nodos por cluster, con valor del *parámetro* d de 10 y 12 para el primer y el segundo nivel, respectivamente. El número máximo de iteraciones del algoritmo de decodificación es 10 y el valor de R es de 1, 5, 10, 30, 150 y 300 símbolos por ráfaga.

el salt número i y H es el número total de saltos para alcanzar el destino. Si se desarrolla la ecuación (4.3) para el caso de dos saltos se obtiene que:

$$BER_{tot} = BER_1 + BER_2 - 2BER_1 \cdot BER_2 \quad (4.4)$$

Donde BER_1 y BER_2 son la probabilidad de error del primer y del segundo nivel, respectivamente. De esta manera, se puede calcular la BER total extremo a extremo del sistema como la suma de las BER de los dos primeros niveles menos dos veces el producto entre ambas.

En el caso de utilizar 5 clusters se ha elegido como mejor curva la que equivalía a un *parámetro* d de 12 en el segundo nivel. Para el primer nivel se obtuvo que, con 300 nodos por cluster, la mejor curva se daba para $d=10$ combinaciones. Al realizar las simulaciones a través de los dos niveles, el resultado es el que se obtiene en la Figura 4.17.

En la Figura 4.17 se ve el comportamiento para 5 nodos cabeza de cluster a través de ambos niveles. El número máximo de iteraciones del algoritmo de decodificación es de 10 iteraciones y se ha estudiado para distintos tamaños de ráfaga. Se ve en esta figura cómo los valores de las curvas obtenidas se asemejan mucho a las obtenidas para el segundo

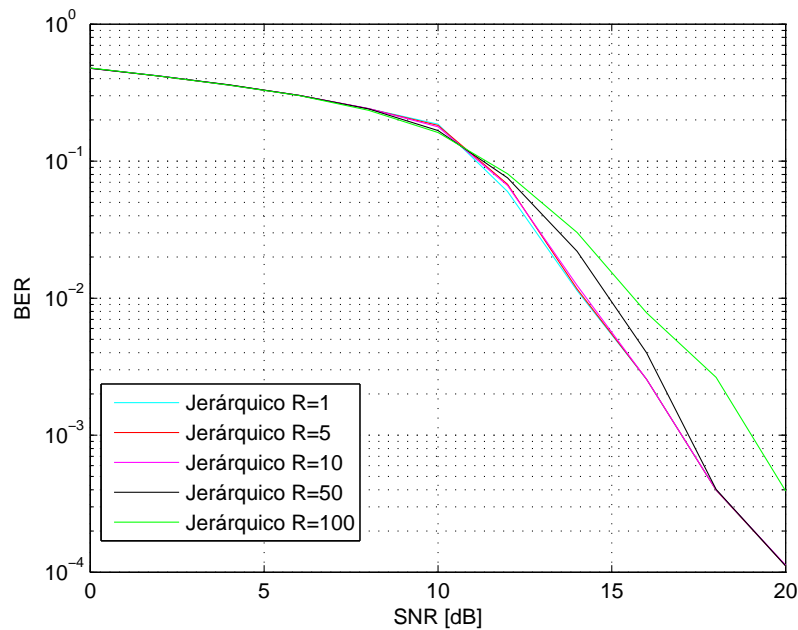


Figura 4.18: Curva BER-SNR obtenida para una codificación LDPC distribuida (a través de los dos niveles del algoritmo presentado) en una WSN de 1500 nodos, con 15 nodos cabeza de clusters y 100 nodos por cluster, con valor del *parámetro* d de 10 y 12 para el primer y el segundo nivel, respectivamente. El número máximo de iteraciones del algoritmo de decodificación es 10 y el valor de R es de 1, 5, 10, 50 y 100 símbolos por ráfaga.

nivel de simulaciones. Esto se debe a que el segundo nivel es el más restrictivo y por lo tanto el que más afecta a la transmisión, ya que valores de BER altos en uno de los niveles acarrearán que los valores de BER totales sean también altos.

La Figura 4.18 representa la curva BER-SNR total tras pasar por los dos niveles en el caso de tener 15 agrupaciones con 100 nodos cada una. Se ve en dicha figura que la región de *Waterfall* se prolonga hasta valores de SNR muy altos y que los valores de BER obtenidos en la región de *Error Floor* son mayores que los obtenidos para el caso de tener 5 agrupaciones. Esto se debe a que los resultados de BER obtenidos en el primer nivel eran muy superiores en el caso de utilizar 100 nodos a utilizar 300, con lo que en nuestro algoritmo es preferible que las agrupaciones engloben al mayor número de nodos posible.

A continuación se van a comparar los resultados obtenidos utilizando este algoritmo de jerarquización con los casos de un algoritmo multisalto que no utilice codificación y otro que utilice codificación LDPC con dos saltos [Morgado, 2009]. Los resultados son los que se ven en las Figuras 4.19 y 4.20 para los casos estudiados de 5 y 15 clusters, respectivamente.

En la Figura 4.19 se comparan los resultados obtenidos después de ambas fases para el caso de dividir los nodos en 5 clusters de 300 nodos cada uno con tamaños de ráfaga de 1, 30, 150 y 300 símbolos por ráfaga, con los casos de no usar codificación o usar un

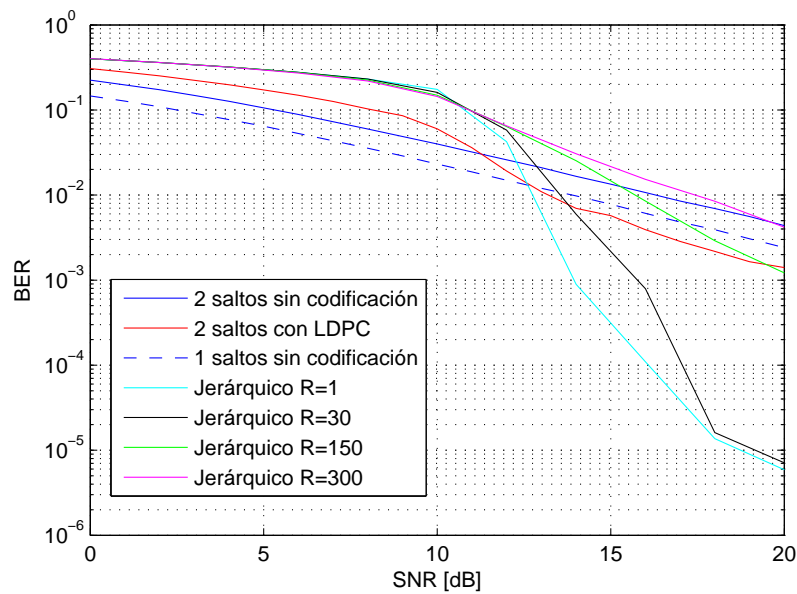


Figura 4.19: Curva BER-SNR obtenida para una codificación LDPC distribuida (a través de los dos niveles del algoritmo presentado) en una WSN de 1500 nodos, con 5 nodos cabeza de clusters y 300 nodos por cluster, comparado con el caso de dar 1 y 2 saltos sin codificación y 2 saltos con codificación LDPC. El valor del *parámetro* d para nuestro algoritmo es de 10 y 12 para el primer y el segundo nivel, respectivamente, el número máximo de iteraciones del algoritmo de decodificación es 10 y el valor de R es de 1, 30, 150 y 300 símbolos por ráfaga.

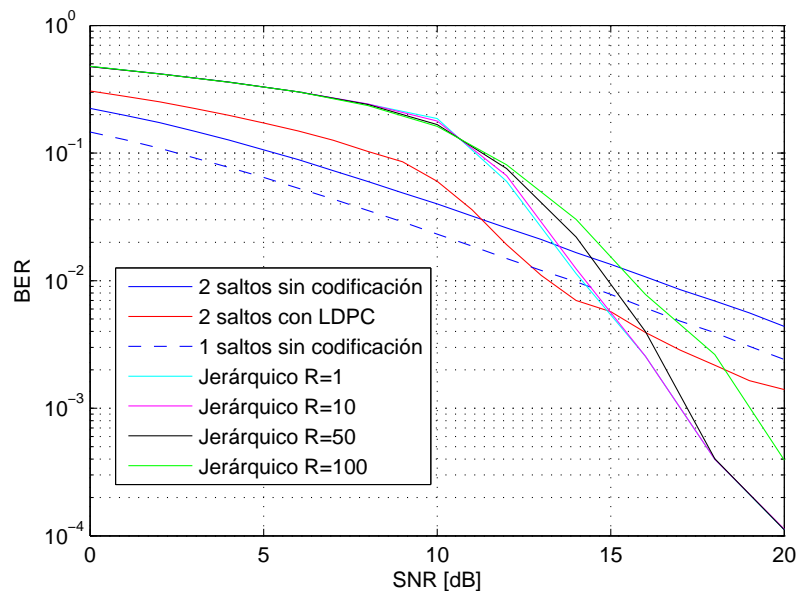


Figura 4.20: Curva BER-SNR obtenida para una codificación LDPC distribuida (a través de los dos niveles del algoritmo presentado) en una WSN de 1500 nodos, con 15 nodos cabeza de clusters y 100 nodos por cluster, comparado con el caso de dar 1 y 2 saltos sin codificación y 2 saltos con codificación LDPC. El valor del *parámetro* d para nuestro algoritmo es de 10 y 12 para el primer y el segundo nivel, respectivamente, el número máximo de iteraciones del algoritmo de decodificación es 10 y el valor de R es de 1, 10, 50 y 100 símbolos por ráfaga.

algoritmo multisalto con LDPC. Para este caso, al igual que para la Figura 4.18, se han utilizado los valores del *parámetro d* para nuestro algoritmo es de 10 y 12 para el primer y el segundo nivel, respectivamente y el número máximo de iteraciones del algoritmo de decodificación es 10.

Como se definió en la Sección 3.2, el Punto de Mejora es el valor de SNR a partir del cuál se obtienen valores menores de BER con el algoritmo presentado en este Proyecto que con los otros esquemas con los que se compara. De esta manera, en la Figura 4.19 se ve que el Punto de Mejora respecto al caso de no usar codificación se sitúa en torno a los 12 dB en el caso de que las ráfagas sean de $R=1$. De manera similar, en el caso de que las ráfagas sean de una duración un poco mayor, como es el caso de ráfagas de $R=30$, el Punto de Mejora se sitúa en los 13 dB. En esta misma figura se aprecia que en el caso en el que el canal en el segundo nivel varíe muy poco, con ráfagas que engloben un alto número de símbolos ($R=150$ y $R=300$), no se obtiene mejora si se compara con el caso de no usar codificación hasta valores de SNR de 20 dB. Sin embargo, en el caso de utilizar esta codificación con canales que varíen más rápidamente se obtienen grandes mejoras a valores altos de SNR, al compararlos con los de dos saltos sin codificación y con codificación LDPC, aunque los valores del *parámetro d* elegidos hacen que la región de *Waterfall* aparezca a valores de SNR altos, permitiendo así que los valores de BER obtenidos sean muy bajos en la región de *Error Floor*.

Para cuantificar la mejora que supone nuestro algoritmo, se va a hallar la Ganancia de Codificación para valores de SNR de 15 dB y 20 dB para el caso en que las ráfagas sean de tamaño $R=1$ y $R=30$. En estos casos, aunque las gráficas se encuentran en su región de *Waterfall*, se va a calcular para estos valores porque se encuentran por encima del Punto de Mejora y son razonables para una comunicación inalámbrica. Los resultados obtenidos son muy buenos, obteniéndose a valores de SNR de 15 dB Ganancias de Codificación al compararlo con el caso en que haya dos saltos sin codificación de 16 dB y 8 dB para los casos en que las ráfagas sean de tamaño $R=1$ y $R=30$, respectivamente, muy superiores a las que se usaban para el caso de LDPC con un algoritmo multisalto. Si se compara la Ganancia de Codificación a valores de SNR de 15 dB respecto al caso del algoritmo multisalto con LDPC se obtienen unas ganancias de 12 dB y 4 dB para ráfagas de tamaño $R=1$ y $R=30$, respectivamente, lo que demuestra que a esos valores de SNR ya es bastante ventajoso el algoritmo propuesto sobre los otros dos casos analizados. Si se hiciera el mismo cálculo a 20 dB, las Ganancias de Codificación que se obtienen son mucho mayores: comparando con el caso sin codificación se obtienen ganancias por encima de los 30 dB para ambos tamaños de ráfaga, y con el caso del LDPC multisalto, las ganancias obtenidas se encuentran por encima de los 25 dB.

Las mismas conclusiones se pueden sacar con el caso de 15 clusters y 100 nodos por cluster, como se puede apreciar en la Figura 4.20, donde se han comparado los casos de tener ráfagas de 1, 30, 150 y 300 símbolos por ráfaga, con los casos de no usar codificación

o usar un algoritmo multisalto con LDPC. Para este caso, al igual que para la Figura 4.18, se han utilizado los valores del *parámetro* d para nuestro algoritmo es de 10 y 12 para el primer y el segundo nivel, respectivamente y el número máximo de iteraciones del algoritmo de decodificación es 10.

En la Figura 4.20 se ve cómo se obtiene una mejora al utilizar el algoritmo propuesto a valores de SNR altos (cerca de los 15 dB) para todos los tamaños de ráfaga respecto al caso de no usar codificación, con lo que el Punto de Mejora se sitúa cerca de los 15 dB. Para estos valores, a excepción del caso en el que el tamaño de la ráfaga sea de $R=100$, los resultados obtenidos son peores que los del caso de usar 5 agrupaciones de 300 nodos. Es necesario ir a valores de SNR superiores a los 16 dB para obtener una mejora utilizando el algoritmo propuesto respecto al LDPC multisalto, obteniéndose para una SNR de 16 dB y utilizando tamaños de ráfaga equivalentes a $R=1$ Ganancias de Codificación cercanas a los 6,5 dB al compararla con el caso en el que no se use codificación y de 1,8 dB comparado con el caso del algoritmo LDPC multisalto. Si se calcula esta Ganancia de Codificación a 20 dB, las ganancias obtenidas para ráfagas de tamaño $R=1$ son de 17 dB al compararla con el caso sin codificar y de 13 dB comparada con la codificación LDPC multisalto. Además para una SNR de 20 dB la ganancia de codificación obtenida en el caso de que la ráfaga sea de tamaño $R=100$ es de 11 dB respecto al caso sin codificar y de 7 dB respecto a la codificación LDPC multisalto, lo que demuestra que a esos valores de SNR se obtiene una gran ventaja con nuestro algoritmo en los dos escenarios analizados (300 y 100 nodos por cluster).

Capítulo 5

Conclusiones y trabajos futuros

El contenido de este capítulo surge del análisis del trabajo realizado durante la realización del presente Proyecto. Este capítulo se va a dividir en dos partes: la primera parte se ocupa de exponer las conclusiones que se pueden sacar a partir de los resultados obtenidos en el desarrollo del algoritmo propuesto. En la segunda parte se proponen varias líneas futuras que pueden dar continuidad a este Proyecto, dentro de la investigación de prestaciones en WSN densas y extensas.

5.1. Conclusiones

A partir de los resultados obtenidos en el Capítulo 4 se pueden obtener una serie de conclusiones sobre el algoritmo propuesto:

- El número de iteraciones del algoritmo de decodificación Suma-Producto no es un factor que mejore significativamente las curvas de BER frente a SNR. Empleando 10 iteraciones es suficiente para obtener buenos resultados en la decodificación de los códigos LDPC. Aumentar el número de iteraciones a partir de ese valor mejora levemente los resultados obtenidos a consta de aumentar el tiempo que se tarda en la decodificación de una manera casi proporcional al número de iteraciones.
- El aumento del número de nodos que interviene en la codificación LDPC distribuida mejora sus prestaciones. Como la matriz generadora tiene un mayor tamaño, aumenta la dispersión de esta matriz, lo que lleva a un mayor número de columnas linealmente independientes y, de esta forma, los códigos tienen una mayor capacidad de corrección de errores.

- El valor del *parámetro d* tiene un rango en el que su funcionamiento es óptimo. Si se utiliza un valor muy bajo del *parámetro d*, en la fase de reenvío se envía poca información acerca de los símbolos enviados por el resto de nodos, lo que provoca que los valores de BER obtenidos sean muy altos. Por otro lado, si se utiliza un valor muy alto de este parámetro las matrices generadoras resultantes son poco dispersas y esto provoca que la pendiente en la zona de *Waterfall* de las curvas BER-SNR sea menor y se obtengan peores resultados.
- Dentro del rango óptimo para el *parámetro d*, si se emplea un valor elevado la región de *Waterfall* se sitúa a valores mayores de SNR, aumenta su pendiente y se obtienen valores mejores de BER en la región de *Error Floor*, aunque esta zona se alcanza en valores de SNR más elevados.
- El grado de dispersión *GD* da información acerca de lo dispersa que es la matriz generadora del código LDPC, pero no se puede concluir una serie de valores óptimos para el mismo, ya que las curvas obtenidas no responden a una relación sencilla entre los valores del *parámetro d* y el número de nodos de la red.
- Para el segundo nivel, lo deseable es que el tiempo de coherencia de los canales sea muy pequeño, puesto que así es el propio canal inalámbrico el que consigue crear la independencia entre las columnas de la matriz generadora LDPC distribuida. En caso de no tener un tiempo de coherencia suficientemente corto, se pueden alcanzar los mismos resultados mediante la transmisión por ráfagas cortas. Si el tiempo de coherencia del canal es elevado, al utilizar ráfagas de larga duración en el segundo nivel los resultados de BER obtenidos empeoran notablemente respecto a los obtenidos utilizando ráfagas que engloben muy pocos símbolos.
- El análisis del sistema completo muestra que es preferible utilizar agrupaciones que engloben un gran número de nodos, ya que las matrices utilizadas en el segundo nivel son tan grandes que el hecho de que haya pocos nodos cabeza de cluster no tiene gran relevancia en la independencia de las columnas de la matriz generadora. Sin embargo, según se reduce el número de nodos de las agrupaciones para el primer nivel se ve que los valores de BER obtenidos en la región de *Error Floor* en las curvas BER-SNR son menores, repercutiendo con mayor importancia en las curvas BER-SNR obtenidas para el sistema completo.
- Al comparar con los casos de no utilizar codificación se ve que se obtiene una gran mejora al utilizar el algoritmo presentado en este proyecto para valores altos de SNR, superiores a los 15 dBs. Utilizando el algoritmo propuesto para redes de 1500 nodos con agrupaciones de 300 nodos se pueden llegar a obtener ya a 15 dB ganancias de codificación de 16 dB respecto al caso de no utilizar codificación y de 12 dB respecto al algoritmo multisalto con LDPC.

A parte de las conclusiones obtenidas a partir de los resultados en el presente Proyecto, se ha conseguido instruir al proyectante en dos técnicas que no se ven a lo largo de la carrera como son la Codificación de Red, que permite aprovechar las características de una red para mejorar las prestaciones de manera inteligente, y los códigos LDPC, unos potentes códigos correctores de errores que emplean actualmente en la recuperación de paquetes perdidos en la distribución de datos masivos a través de Internet, en algunos estándares para la transmisión por satélite de televisión digital o en cables de categoría CAT6 para 10GBASE-T Ethernet. Además, el proyectante ha podido participar en un proyecto de investigación y familiarizarse con las técnicas que se emplean en los mismos y permitiéndole profundizar en las técnicas utilizadas en este Proyecto.

5.2. Lineas Futuras

El campo de las WSN es uno de los campos que se encuentra de mayor actualidad. Las propias limitaciones que presentan estas redes hacen que surjan numerosas oportunidades en el campo de la investigación. A partir del trabajo realizado en este Proyecto se proponen algunas líneas continuistas con el mismo:

- Probar diferentes decodificadores LDPC y comprobar su influencia en las curvas de BER frente a SNR, para el algoritmo presentado en este Proyecto.
- Realizar un estudio más detallado del grado de dispersión GD con el fin de analizar su influencia en la implementación de códigos LDPC. Lo óptimo sería encontrar expresiones analíticas que nos permitieran relacionar este parámetro con el comienzo y la pendiente de la zona de *Waterfall*.
- Estudiar, de entre todas las tecnologías inalámbricas disponibles, cuál sería la optima para implementar este algoritmo.
- Implementar el algoritmo presentado en este Proyecto en una maqueta de una WSN extensa y estudiar su viabilidad en sistema real.
- Estudiar las prestaciones obtenidas en WSN más pequeñas, bien con el algoritmo propuesto o bien explorando la codificación distribuida con Codificación de Red y códigos de canal distintos a los LDPCs.

Lista de Acrónimos

ANCC	<i>Adaptive Network Coded Cooperation</i>
BER	<i>Bit Error Rate</i>
CECI	Centro Experimental de Comunicaciones Inalámbricas
CDMA	<i>Code Division Multiple Access</i>
FDMA	<i>Frequency Division Multiple Access</i>
GD	Grado de Dispersión
IDE	<i>Integrated Development Environment</i>
LDGM	<i>Low Density Generator Matrix</i>
LDPC	<i>Low Density Parity Check</i>
MATLAB	<i>MATrix LABoratory</i>
SER	<i>Symbol Error Rate</i>
SNR	<i>Signal Noise Ratio</i>
TDMA	<i>Time Division Multiple Access</i>
WSN	<i>Wireless Sensor Network</i>

Bibliografía

- Rudolf Ahlswede, Ning Cai, Shuo yen Robert Li, and Raymond W. Yeung. Network information flow. *IEEE TRANSACTIONS ON INFORMATION THEORY*, 46(4):1204–1216, 2000.
- Leonardo José Arnone. Transmisión segura en comunicaciones inalámbricas de corto alcance, 2008.
- Xingkai Bao and Jing Li. Matching code-on-graph with network-on-graph: Adaptive network coding for wireless relay networks. In *Proc. Allerton Conf. on Commun., Control and Computing IL*, 2005.
- Xingkai Bao and Jing Li. Adaptive network coded cooperation (ancc) for wireless relay networks: matching code-on-graph with network-on-graph. *Wireless Communications, IEEE Transactions on*, 7(2):574 –583, february 2008. ISSN 1536-1276. doi: 10.1109/TWC.2008.060439.
- C. Berrou, A. Glavieux, and P. Thitimajshima. Near shannon limit error-correcting coding and decoding: Turbo-codes. 1. In *Communications, 1993. ICC 93. Geneva. Technical Program, Conference Record, IEEE International Conference on*, volume 2, pages 1064 –1070 vol.2, may 1993. doi: 10.1109/ICC.1993.397441.
- E. Biglieri, J. Proakis, and S. Shamai. Fading channels: information-theoretic and communications aspects. *Information Theory, IEEE Transactions on*, 44(6):2619 –2692, oct 1998. ISSN 0018-9448. doi: 10.1109/18.720551.
- M. Cheney. *Tesla: Man Out of Time*. Simon & Schuster, 2001. ISBN 9780743215367.
- Frank H. P. Fitzek and Marcos Katz. *Cooperation in Wireless Networks: Principles and Applications: Real Egoistic Behavior is to Cooperate!* Springer, July 2006. ISBN 140204710X.
- Garcia J. Frias and Wei Zhong. Approaching Shannon performance by iterative decoding of linear codes with low-density generator matrix. *IEEE Communications Letters*, 7(6), June 2003. doi: 10.1109/LCOMM.2003.813816.
- Robert G. Gallager. Low-density parity-check codes, 1963.

- F.R. Gfeller and U. Bapst. Wireless in-house data communication via diffuse infrared radiation. *Proceedings of the IEEE*, 67(11):1474 – 1486, nov. 1979. ISSN 0018-9219. doi: 10.1109/PROC.1979.11508.
- Zheng Guo, Bing Wang, and Jun hong Cui. Efficient error recovery with network coding in underwater sensor networks. In *In Proc. IFIP Networking*, 2007.
- National Instruments. Selecting the right wireless technology tutorial, 2009. URL <http://www.ni.com/white-paper/8939/en>.
- Sidharth Jaggi, Peter Sanders, Philip A. Chou, Michelle Effros, Sebastian Egner, Kamal Jain, and Ludo Tolhuizen. Polynomial time algorithms for multicast network code construction, 2003.
- David J.C. MacKay and Radford M. Neal. Near shannon limit performance of low density parity check codes. *Electronics Letters*, 32:1645–1646, 1996.
- D.J.C. MacKay. Good error-correcting codes based on very sparse matrices. *Information Theory, IEEE Transactions on*, 45(2):399 –431, mar 1999. ISSN 0018-9448. doi: 10.1109/18.748992.
- Eduardo Morgado. Prestaciones de las redes ad hoc inalámbricas: teoría a través de capas, 2009.
- Eduardo Morgado, Inmaculada Mora-Jimenez, Juan J. Vinagre, Javier Ramos, and Antonio J. Caamano. End-to-end average ber in multihop wireless networks over fading channels. *Trans. Wireless. Comm.*, 9(8):2478–2487, August 2010. ISSN 1536-1276. doi: 10.1109/TWC.2010.070710.090240.
- M. Panda, P.M. Khilar, T. Panigrahi, and G. Panda. An energy efficient search in dense wireless sensor network. In *Computational Intelligence and Communication Networks (CICN), 2010 International Conference on*, pages 234 –238, nov. 2010. doi: 10.1109/CICN.2010.56.
- W. Wesley Peterson. Error-correcting codes. *MIT Technology Press*, 1961.
- Robert Poor. Wireless mesh networks, 2003. URL <http://www.sensorsmag.com/networking-communications/>.
- John Proakis. *Digital Communications*. McGraw-Hill Science/Engineering/Math, 4 edition, August 2000. ISBN 0072321113.
- T.Q.S. Quek, D. Dardari, and M.Z. Win. Energy efficiency of dense wireless sensor networks: to cooperate or not to cooperate. *Selected Areas in Communications, IEEE Journal on*, 25(2):459 –470, february 2007. ISSN 0733-8716. doi: 10.1109/JSAC.2007.070220.

- K.K. Rachuri and C. Murthy. Energy efficient and scalable search in dense wireless sensor networks. *Computers, IEEE Transactions on*, 58(6):812–826, june 2009. ISSN 0018-9340. doi: 10.1109/TC.2009.29.
- Thomas J. Richardson and Rüdiger L. Urbanke. The capacity of low-density parity-check codes under message-passing decoding, 2001.
- Robert E. Shannon. Introduction to the art and science of simulation. In *Proceedings of the 30th conference on Winter simulation*, WSC '98, pages 7–14, Los Alamitos, CA, USA, 1998. IEEE Computer Society Press. ISBN 0-7803-5134-7. URL <http://dl.acm.org/citation.cfm?id=293172.293175>.
- R. Tanner. A recursive approach to low complexity codes. *Information Theory, IEEE Transactions on*, 27(5):533–547, 1981.
- Tao Tian, C. Jones, J.D. Villasenor, and R.D. Wesel. Construction of irregular ldpc codes with low error floors. In *Communications, 2003. ICC '03. IEEE International Conference on*, volume 5, may 2003.
- Terry J Van Der Werff. 10 emerging technologies that will change the world. *Technology Review*, 2(February):32–52, 2003. URL <http://www.technologyreview.com/infotech/13060/page2/>.
- Arthur Van Westerop. Radio leek 100 jaar geleden erg op internet, 2010. URL <http://www.arthurvanwesterop.nl/wordpress/2011/07/shaping-the-future-of-radio/>.