UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Mapeamento de Bits para Adaptação Rápida a Variações de Canal de Sistemas QAM Codificados com LDPC

Fernanda Regina Smith Neves Corrêa

TD:10/2017

UFPA / ITEC / PPGEE
Campus Universitário do Guamá
Belém-Pará-Brasil
2017

UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Fernanda Regina Smith Neves Corrêa

Mapeamento de Bits para Adaptação Rápida a Variações de Canal de Sistemas QAM Codificados com LDPC

TD:10/2017

UFPA / ITEC / PPGEE
Campus Universitário do Guamá
Belém-Pará-Brasil
2017

UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Fernanda Regina Smith Neves Corrêa

Mapeamento de Bits para Adaptação Rápida a Variações de Canal de Sistemas QAM Codificados com LDPC

Tese submetida à Banca Examinadora do Programa de Pós-graduação em Engenharia Elétrica da Universidade Federal do Pará, como parte dos requisitos para a obtenção do Grau de Doutor em Engenharia Elétrica, ênfase em Telecomunicações.

UFPA / ITEC / PPGEE
Campus Universitário do Guamá
Belém-Pará-Brasil
2017

Dados Internacionais de Catalogação - na - Publicação (CIP) Sistema de Bibliotecas da UFPA

Corrêa, Fernanda Regina Smith Neves, 1984 -

Mapeamento de Bits para Adaptação Rápida a Variações de Canal de Sistemas QAM Codificados com LDPC / Fernanda Regina Smith Neves Corrêa - 2017

Orientador: Evaldo Gonçalves Pelaes.

Tese (Doutorado) - Universidade Federal do Pará, Instituto de Tecnologia, Programa de Pós-Graduação em Engenharia Elétrica, Belém, 2017.

1. Códigos corretores de erros (Teoria da informação). 2. Teoria da codificação. 3. Modulação em amplitude. 4. Comunicações digitais. I. Título.

CDD 23. ed. 005.717

UNIVERSIDADE FEDERAL DO PARÁ

INSTITUTO DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Mapeamento de Bits para Adaptação Rápida a Variações de Canal de Sistemas QAM Codificados com LDPC

AUTOR: FERNANDA REGINA SMITH NEVES CORRÊA

TESE DE DOUTORADO SUBMETIDA À AVALIAÇÃO DA BANCA EXAMINADORA APROVADA PELO COLEGIADO DO PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA DA UNIVERSIDADE FEDERAL DO PARÁ A SER JULGADA PARA OBTENÇÃO DO GRAU DE DOUTORA EM ENGENHARIA ELÉTRICA NA ÁREA DE TELECOMUNICAÇÕES.

JULGADA EM 29/09/2017

BANCA EXAMINADORA:

.....

Prof. Dr. Evaldo Gonçalves Pelaes (UFPA) - Orientador

.....

Prof. Dr. Bartolomeu Ferreira Uchôa-Filho (UFSC) - Co-Orientador

.....

Prof. Dr. Cecílio Pimentel (UFPE) - Membro externo

.....

Prof. Dr. Francisco Marcos de Assis (UFCG) - Membro externo

.....

Prof. Dr. Johelden Campos Bezerra (IFPA) - Membro externo

.....

Prof. Dr. Aldebaro Barreto da Rocha Klautau Júnior (UFPA) - Membro

VISTO:

.....

Prof. Dr. Evaldo Gonçalves Pelaes

COORDENADOR DO PPGEE/ITEC/UFPA

Agradecimentos

Agradeço primeiramente a minha família, em especial a minha mãe Sandra e ao meu pai Marcos, por sempre me incentivarem a estudar e pelo amor e apoio que têm me dado durante a vida. Ao meu irmão Gustavo, minha irmã Hannah, minha cunhada Letícia, meus avós Ruy e Regina, tios e primos pelo apoio, e por sempre acreditarem em mim.

Agradeço a todos os amigos do Laboratório de Processamento de Sinais da UFPA, pela companhia e por fazerem o ambiente de trabalho um lugar agradável. Aos colegas do projeto Ericsson e ao Prof. Aldebaro Klautau, pela ajuda e confiança e, em especial ao amigo Patrício pela amizade desde a graduação.

Agradeço também aos meus novos colegas da UNIFAP pela paciência e incentivo.

Ao meu orientador, Prof. Evaldo Pelaes, por ser muito mais do que um orientador, sempre atencioso e disposto a me ajudar. Agradeço pela confiança em mim.

Um agradecimento todo especial ao meu coorientador, Prof. Bartolomeu Ferreira Uchôa-Filho, pela valiosa coorientação, por acreditar no meu trabalho, nas minha ideias, pelas incansáveis horas de dedicação, atenção e apoio, sempre disposto a tirar as minhas dúvidas e a me orientar.

Agradeço a todos aqueles que direta ou indiretamente contribuíram para a conclusão de mais essa fase na minha vida acadêmica.

Resumo

Os códigos com matriz de verificação de paridade de baixa densidade (LDPC) têm sido adotados como estratégia de correção de erros em diversos padrões de sistemas de comunicação, como nos sistemas G.hn (padrão que unifica as redes domésticas) e IEEE 802.11n (padrão para redes sem fio locais). Nestes sistemas com modulação de amplitude em quadratura (QAM) codificados com LDPC, mapear propriamente os bits codificados para os diferentes sub-canais, considerando o fato de os sub-canais terem diferentes qualidades, garante uma melhora no desempenho geral do sistema. Nesse sentido, esta Tese apresenta uma nova técnica de mapeamento de bits, baseada na suposição de que bits transmitidos em sub-canais "bons" ajudam bits transmitidos em sub-canais "ruins". Isto é possível através de algumas restrições impostas ao grafo de Tanner associado, semelhantes aos códigos *Root*-LDPC. A otimização deste mapeamento de bits utilizando curvas de transferência de informação extrínseca (EXIT *charts*) também é apresentada. Observa-se que esse mapeamento tem a vantagem de um espaço de busca de otimização reduzido quando aplicado ao sistema com modo de transmissão de portadora única. Além disso, em situações nas quais o espaço de busca não é tão reduzido, como em aplicações baseadas em multiplexação por divisão de frequência ortogonal (OFDM), chegou-se a uma simples regra prática associada às restrições do mapeamento de bits que praticamente elimina a necessidade de uma otimização. Por fim, um estudo do impacto do nível de desequilíbrio de confiabilidade através dos sub-canais sobre o desempenho do mapeamento de bits é apresentado. Os resultados das simulações mostram que a estratégia de mapeamento de bits melhora o desempenho do sistema, e que, na presença de variações do canal, o sistema pode, adaptativamente, aplicar um novo mapeamento de bits sem a necessidade de recorrer a uma otimização complexa, podendo ser muito útil em sistemas práticos.

PALAVRAS-CHAVES: LDPC, mapeamento de bits, EXIT *charts*, sistemas de comunicação.

Abstract

Low-Density parity-check (LDPC) codes are being adopted as the error correction strategy in different system standards, such as the G.hn (home networking standard) and the IEEE 802.11n (wireless local standard). In these LDPC-coded quadrature amplitude modulation (QAM) systems, mapping the LDPC coded bits properly to the different sub-channels considering the fact that sub-channels have different qualities ensures an improved overall system performance. Accordingly, this thesis presents a new bit mapping technique based on the assumption that bits transmitted in "good" sub-channels, help bits transmitted in "bad" sub-channels. This can be made possible through some restrictions to be imposed on the associated Tanner graph, akin to Root-LDPC codes. An optimization of the root-like bit mapping through extrinsic information transfer (EXIT) charts analysis is also presented. We show that this mapping has the advantage of a reduced optimization search space when applied to single-carrier based systems. Moreover, in situations where the search space is not so reduced, such as in orthogonal frequency division multiplexing (OFDM)-based applications, we arrive at a rule of thumb associated with the bit mapping constraints that practically eliminates the need for an optimization. Finally, a study of the impact of the level of reliability imbalance across the sub-channels on the performance of the root-like bit mapping is presented. Simulation results show that the new bit mapping strategy improves performance, and that in the presence of channel variations, the system can, adaptively, apply a new bit mapping without the need of a complex optimization, which can be very useful in practical systems.

KEYWORDS: LDPC codes, bit mapping, EXIT charts, communication systems.

Sumário

Lista de Figuras **iii**

Lista de Tabelas **vi**

Lista de Siglas **vii**

Lista de Símbolos **viii**

1 Introdução **1**

1.1 Motivação 3

1.2 Objetivos da Tese 4

1.3 Contribuições da Tese 4

1.4 Organização da Tese 5

2 Códigos LDPC **7**

2.1 Códigos de Blocos Lineares 7

2.2 Códigos LDPC 11

2.2.1 Códigos LDPC Regulares e Irregulares 12

2.3 Codificação 14

2.3.1 Codificação com Complexidade Quase Linear 15

2.4 Decodificação 18

2.4.1 Decodificação Soma-Produto 19

2.5 Códigos QC-LDPC 22

3 Mapeamento de Bits *Root-Like* **26**

3.1 Mapeamento de bits 27

3.2 O Mapeamento de Bits Baseado nos Códigos *Root*-LDPC 29

3.2.1 Códigos *Root*-LDPC 29
3.2.2 Mapeamento de Bits *Root-Like* 30

4 Otimização do Mapeamento de Bits *Root-Like* através da análise de EXIT *charts* 34

4.1 EXIT *charts* para códigos LDPC 34
4.1.1 EXIT *charts* com Mapeamento de Bits 38
4.2 Algoritmo de Otimização 39

5 Resultados Numéricos 42

5.1 Modelo do Canal 43
5.1.1 Ruído Aditivo 43
5.1.2 Desvanecimento *Rayleigh* 43
5.2 Mapeamento de Bits *Root-Like* em OFDM Aplicado ao G.hn 44
5.3 Mapeamento de Bits *Root-Like* para o Canal AWGN de Portadora Única Aplicado ao G.hn 49
5.4 Mapeamento de Bits *Root-Like* para o Canal AWGN de Portadora Única e com Desvanecimento *Rayleigh* Aplicado ao IEEE 802.11n 54
5.5 Regra Prática para o Mapeamento de Bits *Root-Like* 56
5.6 O Impacto do Nível de Desequilíbrio de Confiabilidade através dos Sub-canais na Performance do Mapeamento de Bits *Root-Like* 57

6 Conclusões 60

Bibliografia 62

A Matrizes de Verificação de Paridade 66

B Distribuições de graus dos nós de variável $\lambda(x)$ e dos nós de paridade $\rho(x)$ 68

C Distribuições do Mapeamento de bits *Root-like* 70

Lista de Figuras

2.1 Grafo de Tanner. 11

2.2 Matriz de verificação de paridade na forma triangular inferior equivalente. . . . 15

2.3 Matriz de verificação de paridade na forma triangular inferior aproximada. . . 16

2.4 Matriz **H** do padrão G.hn ($c = 12, t = 24, b = 80$) para $N = 1920$ bits, $K = 960$ bits, taxa 1/2 [8]. 25

3.1 Diagrama de blocos de um sistema QAM codificado com LDPC com mapeamento de bits. 26

3.2 Conexões entre nós de variável, nós de paridade e sub-canais. 27

3.3 Esquema do mapeamento de bits *root-like*. 31

4.1 Diagrama de blocos da decodificação iterativa. 35

4.2 EXIT *chart* de um código LDPC regular ($d_v = 3, d_c = 6$) de taxa 1/2. 36

5.1 Matriz de verificação de paridade do padrão G.hn ($c = 12, t = 24, b = 80$) para $N = 1920$ bits, $K = 960$, taxa 1/2 [8]. 43

5.2 SNRs limiares de diferentes deslocamentos de bits (em bits) obtidos através de EXIT *charts* para a matriz do G.hn com modo de transmissão OFDM, sobre o canal AWGN, com $N = 1920$ bits, taxa 1/2 e 256-QAM. 46

5.3 BER da matriz G.hn com transmissão OFDM, para o canal AWGN, considerando diferentes valores de deslocamento (s), com $N = 1920$ bits, taxa 1/2, e 256-QAM e 240 sub-portadoras. A SNR mostrada é a SNR média entre todos os sub-canais. 47

5.4 Modelo de Canal OFDM especificando a SNR média para cada sub-canal, considerando 240 sub-canais e SNRs variando entre 18 e 31dB. 48

5.5 BER do sistema original vs. sistema com o mapeamento de bits *root-like* para as matrizes do G.hn, considerando o modo de transmissão OFDM, através do canal AWGN, com $N = 1440$ bits, taxa 2/3, 64-QAM e 1024-QAM, $N = 1920$ bits, taxa 1/2 e 256-QAM, e $N = 1152$ bits, taxa 5/6 e 64-QAM. A SNR mostrada é a SNR média entre todos os sub-canais. 49
5.6 Informação mútua dos sub-canais do 256-QAM no canal AWGN, replicada de [21]. 50
5.7 EXIT *charts* para o código LDPC com $N = 6480$, taxa 2/3 e 256-QAM como definido no G.hn. As funções VND são mostradas para o caso G.hn original vs. o caso com mapeamento *root-like* considerando os bits ajudantes mapeados para diferentes pares de bits (deslocamentos diferentes), e os seus respectivos $\mathrm{SNR}_{\mathrm{TH}}$. 51
5.8 BER do sistema original vs. o sistema com mapeamento *root-like* para a matriz do G.hn com o canal de portadora única AWGN, com $N = 6480$, taxa 2/3 e 256-QAM, considerando os bits ajudantes mapeados para os diferentes pares de bits. 52
5.9 BER do sistema original vs. o sistema com o mapeamento de bits *root-like* para as matrizes do G.hn, considerando o canal de portadora única AWGN com $N = 8640$ bits, taxa 1/2, 64-QAM e 256-QAM, e com $N = 6480$ bits, taxa 2/3, 64-QAM e 256-QAM. 53
5.10 BER do sistema original vs. o sistema com mapeamento de bits *root-like* para a matriz do IEEE 802.11n com canais de portadora única AWGN e com desvanecimento *Rayleigh* vs. o sistema com o mapeamento de bits otimizado obitido em [22] para a matriz do IEEE 802.11n com canal de portadora única AWGN, com $N = 1944$, taxa 1/2 e 256-QAM. 55
5.11 Matriz de verificação de paridade do IEEE 802.11n ($c = 12, t = 24, b = 81$) para $N = 1944$ bits, $K = 972$, taxa 1/2 [6]. 56
5.12 SNRs médias dos sub-canais variando linearmente de valores definidos G_1 a G_2, de diferentes valores de gradiente $G_2 - G_1$. 58

5.13 BER do sistema original vs. o sistema com mapeamento de bits *root-like* para a matriz do G.hn com subcanais variando linearmente de G_1 a G_2, com diferentes valores de gradientes $G_2 - G_1$, através do canal AWGN, com $N = 1920$ bits, taxa 1/2 e 256-QAM. A SNR mostrada é a média de todas as sub-portadoras. 58

5.14 BER do sistema original vs. o sistema com mapeamento de bits *root-like* para a matriz do G.hn com sub-canais variando linearmente de G_1 a G_2 sobre o canal AWGN, com $N = 1920$ bits, taxa 1/2 e 256-QAM e com valor de gradiente $G_2 - G_1 = 25$ dB, para diferentes valores de deslocamento. A SNR mostrada é a média de todas as sub-portadoras. 59

A.1 Matriz de verificação de paridade do padrão G.hn ($c = 12, t = 24, b = 60$) para $N = 1440$ bits, $K = 960$, taxa 2/3 [8]. 66

A.2 Matriz de verificação de paridade do padrão G.hn ($c = 12, t = 24, b = 216$) para $N = 5184$ bits, $K = 4320$, taxa 5/6 [8]. 66

A.3 Matriz de verificação de paridade do padrão G.hn ($c = 12, t = 24, b = 270$) para $N = 6480$ bits, $K = 4320$, taxa 2/3 [8]. 66

A.4 Matriz de verificação de paridade do padrão G.hn ($c = 12, t = 24, b = 360$) para $N = 8640$ bits, $K = 4320$, taxa 1/2 [8]. 67

Lista de Tabelas

5.1 Parâmetros associados com os quatro cenários simulados na Figura 5.5. 48

5.2 Capacidade dos sub-canais do 256-QAM (Rotulamento *Gray* do G.hn) no canal AWGN. 50

5.3 Distribuição do mapeamento de bits *root-like* para a matriz do G.hn ($N = 6480$, $R = 2/3$) com canal de portadora única AWGN e 256-QAM [8]. 52

5.4 Capacidade dos sub-canais do 64-QAM (Rotulamento *Gray* do G.hn) no canal AWGN. 54

5.5 Capacidade dos sub-canais do 256-QAM (Rotulamento *Gray* do IEEE 802.11n). 55

5.6 Distribuição do Mapeamento *Root-like* com rotulamento *Gray* 256-QAM com transmissão através do canal AWGN e código LDPC com $N = 1944$ e $R = 1/2$ [6]. 56

C.1 Distribuição do mapeamento de bits *root-like* para a matriz do G.hn ($N = 6480$, $R = 2/3$) com canal de portadora única AWGN e 64-QAM [8]. 70

C.2 Distribuição do mapeamento de bits *root-like* para a matriz do G.hn ($N = 8640$, $R = 1/2$) com canal de portadora única AWGN e 256-QAM [8]. 70

C.3 Distribuição do mapeamento de bits *root-like* para a matriz do G.hn ($N = 8640$, $R = 1/2$) com canal de portadora única AWGN e 64-QAM [8]. 71

Lista de Siglas

AWGN	-	*Additive White Gaussian Noise*
BEC	-	*Binary Erasure Channel*
BICM	-	*Bit Interleaved Coded Modulation*
BPSK	-	*Binary Phase Shift Keying*
CND	-	*Check Node Decoder*
CSI	-	*Channel state information*
DE	-	*Density Evolution*
DTMB	-	*Digital Terrestrial Multimedia Broadcast*
DVB	-	*Digital Video Broadcasting*
ETSI	-	*European Telecommunications Standards Institute*
EXIT	-	*Extrinsic Information Transfer*
FEC	-	*Forward Error Correction*
G.hn	-	*Gigabit Home Networking*
IEEE	-	*Institute of Electric and Electronic Engineers*
ITU	-	*International Telecommunication Union*
LDPC	-	*Low-Density Parity-Check*
LLR	-	*Log-Likelihood Ratio*
OFDM	-	*Orthogonal frequency division multiplexing*
QAM	-	*Quadrature Amplitude Modulation*
QC-LDPC	-	*Quasi-Cyclic LDPC*
SNR	-	*Signal-to-Noise Ratio*
SPA	-	*Sum-Product Algorithm*
UEP	-	*Unequal Error Protection*
VND	-	*Variable Node Decoder*

Lista de Símbolos

m	Número de bits em um símbolo QAM
n	Tamanho da palavra-código de um código de blocos linear, número de nós de variável na matriz de verificação de paridade
k	Número de bits de mensagem de um código de blocos linear
$\mathbf{G}$	Matriz geradora de um código
C	Código de bloco linear
$\mathbf{u}$	Vetor contendo os bits de mensagem
$\mathbf{v}$	Vetor contendo os bits codificados, palavra-código
$\mathbf{H}$	Matriz de verificação de paridade de um código
$\mathbf{P}$	Sub-matriz de paridade, matriz de coeficientes de paridade
$\mathbf{I}_k$	Matriz identidade $k \times k$
$\mathbf{I}_{n-k}$	Matriz identidade $(n-k) \times (n-k)$
$\hat{\mathbf{v}}$	Vetor contendo a palavra-código recebida
R	Taxa de um código
p	número de nós de paridade, número de linhas na matriz de verificação de paridade
d_v	Número de 1s por coluna (ou grau do nó de variável) na matriz $\mathbf{H}$ de um código LPDC
d_c	Número de 1s por linha (ou grau do nó de paridade) na matriz $\mathbf{H}$ de um código LPDC
$\lambda(x)$	Distribuição de graus de nós de variável

λ_i	Fração de ramos conectados a nós de variável de grau i
d_v^{max}	Valor máximo do grau de variável
$\rho(x)$	Distribuição de graus de nós de paridade
ρ_q	Fração de ramos conectados a nós de paridade de grau q
d_c^{max}	Valor máximo do grau de paridade
z_i	Número de nós de variável com grau i
c_q	Número de nós de paridade com grau q
$\tilde{\lambda}_i$	Fração de nós de variável com grau i
$\tilde{\rho}_q$	Fração de nós de paridade com grau q
g	*gap* da representação aproximada q
$\mathbf{p}$	Vetor contendo os bits de paridade
$\mathbf{y}$	Vetor contendo os bits codificados após a decisão abrupta
LLR(x)	Razões de verossimilhança logarítmica (LLR) de x
P_i^{int}	Probabilidade *intrínseca* ou *a priori* do bit i
P_i^{ext}	Probabilidade *extrínseca* ou *a posteriori* do bit i
$M_{j,i}$	Mensagem inicial enviada do nó de bit i para o nó de paridade j
R_i	LLR intrínseca do sinal recebido
$E_{j,i}$	A LLR extrínseca do nó de paridade j para o nó de bit i
L_i	A LLR de cada bit i
$\mathbf{R}$	Vetor contendo as LLRs dos bits recebidos
ε	Probabilidade de um canal binário simétrico (BSC)
N	Número de bits codificados de um código QC-LDPC
K	Número de bits de informação de um código QC-LDPC
b	Tamanho das sub-matrizes da matriz de verificação de paridade de um código QC-LDPC e fator de expansão de $\mathbf{H}$
$P_{i,j}$	Sub-matriz da matriz de verificação de paridade de um código QC-LDPC

c	Número de linhas da matriz de verificação de paridade de um código QC-LDPC na forma compacta
t	Número de colunas da matriz de verificação de paridade de um código QC-LDPC na forma compacta
P	Matriz identidade rotativa
$\mathbf{H_c}$	Matriz de verificação de paridade de um código QC-LDPC na forma compacta
$p_{i,j}$	Sub-matriz da matriz de verificação de paridade de um código QC-LDPC na forma compacta
I	Matriz identidade
$P^{p_{i,j}}$	Matriz $b \times b$ identidade rotativa em que $p_{i,j}$ determina o número de deslocamento na matriz identidade
j	Nível de confiabilidade
b_κ	Bit de um símbolo QAM
e	Sub-canais com diferentes níveis de confiabilidade
$m(x)$	Distribuição de graus de nós associado aos sub-canais, distribuição de mapeamento de bits
$m_{j,i}$	Fração de ramos que mapeia os nós de variável de grau i para o nível de confiabilidade j
$d_v^{\min}$	Valor mínimo do grau de variável
$\mathcal{B}', \mathcal{H}'$	Conjuntos de nós de variável
$\mathcal{B}$	Subconjunto de nós de variável "ruins"
$\mathcal{H}$	Subconjunto de nós de variável "ajudantes (*helping*)"
$\mathcal{O}_v$	Subconjunto de "todos os outros" nós de variável
$\mathcal{R}$	Nós de paridade "*rootcheck*"
$\mathcal{O}_c$	"todos os outros" nós de paridade
s	Parâmetro de deslocamento
$\mathcal{O}_v$	Subconjunto de "todos os outros" nós de variável
I_{A}	Informação mútua entre o bit transmitido e a informação *a priori*

I_{E}	Informação mútua entre o bit transmitido e a informação extrínseca
$I_{\mathrm{E,V}}$	Informação mútua entre o bit transmitido e a informação extrínseca na saída de VND
$I_{\mathrm{A,V}}$	Informação mútua entre o bit transmitido e a informação *a priori* na entrada de VND
$I_{\mathrm{E,C}}$	Informação mútua entre o bit transmitido e a informação extrínseca na saída de CND
$I_{\mathrm{A,C}}$	Informação mútua entre o bit transmitido e a informação *a priori* na entrada de CND
$L_{\mathrm{A}j}$	LLR *a priori* na entrada do decodificador para o j-ésimo nó de paridade
$L_{\mathrm{CH}j}$	LLR do canal
$L_{\mathrm{E}i}$	A LLR extrínseca na saída do decodificador VND para o i-ésimo nó de variável
σ^2_{CH}	A variância de L_{CHi}
$\sigma^2_{\mathrm{ch},j}$	Variância do ruído para o nível de confiabilidade j
$\mathfrak{E}$	O conjunto de sub-canais
$\mathfrak{B}$	O conjunto dos piores sub-canais
$\mathfrak{H}$	O conjunto dos sub-canais ajudantes
$\mathrm{SNR}_{\mathrm{TH}}$	SNR limiar
$n(t)$	Ruído do canal
${\sigma_n}^2$	Variância do ruído
N_0	Densidade espectral de potência do ruído
h	Fator de atenuação
$G_2 - G_1$	Valores de gradientes

Capítulo 1

Introdução

Os códigos LDPC (*Low-Density Parity-Check*) pertencem a uma classe de poderosos códigos corretores de erro, com alto poder de correção, com os quais é possível se aproximar da capacidade de canal de Shannon. Desde o artigo de Shannon [1] em 1948, que definiu limites para a capacidade de transmissão em canais de comunicação, diversos esquemas de correção de erros foram propostos, tentando alcançar tal capacidade. O objetivo era desenvolver códigos que conseguissem corrigir os erros introduzidos pelo canal, mas que ao mesmo tempo fossem práticos, ou seja, a estrutura da decodificação não fosse tão complexa, ao ponto de inviabilizar a implementação. Nesse contexto, em 1962, surgiram os códigos LDPC, propostos inicialmente por Gallager em sua tese de doutorado [2], permanecendo esquecidos por 35 anos, devido ao poder de processamento limitado da época, sendo redescobertos por McKay e Neal [3] na década de 90. Junto com os códigos Turbo formam uma classe de códigos chamada *capacity-approaching*, ou seja, que se aproximam da capacidade de Shannon.

Devido ao seu excelente desempenho, os códigos LDPC são recomendados em diversos padrões de sistemas de comunicação, como o ETSI DVB-T2 [4], o *Digital Terrestrial Multimedia Broadcast* (DTMB) [5], IEEE 802.11n [6], IEEE 802.16e [7], e o G.hn [8]. Por exemplo, o recente padrão do ITU-T que unifica as redes domésticas, chamado G.hn [8], permite conexões de alta velocidade, de até 1 Gb/s, especificando um conjunto de matrizes de verificação de paridade de uma classe de códigos LDPC, conhecida como códigos LDPC quase-cíclicos (*Quasi-Cyclic* LDPC (QC-LDPC)), como o esquema de correção direta de erros (*forward error correction* (FEC)) obrigatório [9]. Esquemas de altas ordens de modulações de amplitude em quadratura (*quadrature amplitude modulation* (QAM)) [10] também são recomendados para alavancar altas taxas de dados.

Um código LDPC é definido pela sua matriz de verificação de paridade, e para se projetar um "bom" código LDPC, alguns parâmetros da matriz de verificação de paridade do

código precisam ser otimizados. Um *ensemble* é um conjunto de todos os possíveis códigos com certos parâmetros definidos. No caso dos códigos LDPC irregulares, o *ensemble* é especificado pelo par de distribuição de graus dos nós (um para o nó de variável e outro para o nó de paridade). Esse par otimizado pode ser obtido pelas técnicas de *density evolution* (DE) [11, 12] ou pelas curvas de transferência de informação extrínseca (*extrinsic information transfer* (EXIT *chart*)) [13, 14], através da procura no espaço de todos os possíveis pares de distribuição de graus, o que pode ser uma tarefa bastante complexa.

Gong *et al.* [15] observaram que a propriedade de proteção desigual de erro (*unequal error protection* (UEP)), intrínseca dos códigos LDPC irregulares, pode ser explorada sempre que os bits codificados forem enviados através de sub-canais de diferentes qualidades. Esse é o caso em inúmeras situações. Por exemplo, considere um sistema que utiliza 2^m símbolos QAM transmitidos em um canal com ruído gaussiano branco aditivo (*additive white Gaussian noise* (AWGN)). Esses símbolos podem ser decompostos em m sub-canais AWGN paralelos de entradas binárias com diferentes qualidades de canal. Quando a transmissão por multiplexação por divisão de frequências ortogonais (*orthogonal frequency division multiplexing* (OFDM)) [16, 17] é utilizada, os diferentes sub-canais geralmente têm diferentes relações sinal ruído (SNRs) médias. Assim, o desempenho de um sistema QAM codificado com LDPC pode ser melhorado através do mapeamento seletivo dos nós de variável da matriz de verificação de paridade para os sub-canais. Isso pode ser garantido colocando-se um mapeador de bits entre o codificador LDPC e o modulador. Baseado nessa informação, esquemas de mapeamento de bits que alteram a ordem na qual os bits codificados são mapeados para diferentes sub-canais foram propostos em [15, 18, 19, 20, 21], para equilibrar essas características de UEP.

Por sua vez, pode-se recorrer à técnica de *density evolution* ou a EXIT *charts* para procurar a distribuição de mapeamento de bits ótima. Por exemplo, em [19], o mapeamento dos bits é otimizado através de *density evolution* e uma melhoria no desempenho para constelações até 64-QAM é descrita. Um mapeamento baseado na confiabilidade do bit é proposto em [18], no qual os bits com menor confiabilidade (aqueles decodificados nos nós de variável com menor grau) são mapeados para os níveis de modulação com menor proteção e os bits com maior confiabilidade para os níveis de modulação com maior proteção. Em [22], uma discussão é feita sobre mapeamentos baseados somente na otimização do limiar, assim como os baseados na confiabilidade do bit tenderem a apresentar um comportamento de *error-floor*. Em [23], menciona-se que as desvantagens dessas análises do limiar usando DE ou EXIT *charts* é que elas referem-se a comprimentos de blocos infinitos e a um número alto de iterações, e propõe-se um método baseado em EXIT *charts* para otimizar o mapeamento, especialmente para blocos de comprimento curto.

Nesses trabalhos, as distribuições dos nós de variável e de paridade são consideradas

fixas. No entanto, um novo conjunto de distribuição de nós de variável é definido, agora associados aos sub-canais, chamado de distribuição de mapeamento de bits. Assim, o processo de otimização é baseado unicamente nessa nova distribuição de nós de variável.

1.1 Motivação

No caso dos padrões especificados, as matrizes de verificação de paridade para os códigos LDPC geralmente já são fornecidas. Assim, uma pergunta interessante é como o desempenho do sistema pode ser melhorado, uma vez que a matriz de verificação de paridade já foi escolhida. Observa-se que o mapeamento dos bits codificados para os sub-canais é normalmente realizado de maneira sequencial, ou seja, sem nenhuma seleção apropriada de quais bits serão alocados para quais sub-canais. No entanto, diversas técnicas de mapeamento de bits [15, 18, 19, 20, 21] que possibilitam mapear propriamente os bits codificados para os diferentes sub-canais, se aproveitando da propriedade de UEP dos códigos LDPC irregulares e/ou da existência de diferentes sub-canais com diferentes qualidades, se mostraram uma excelente alternativa para melhorar o desempenho dos sistemas QAM codificados com LDPC.

Além disso, mesmo em sistemas nos quais a matriz de verificação de paridade não é previamente fornecida, o mapeamento de bits pode também se tornar uma importante ferramenta para possibilitar o projeto de matrizes, levando em consideração o impacto do mapeamento de bits no desempenho do sistema.

No entanto, a busca por uma distribuição de mapeamento de bits ótima é ainda uma tarefa complexa. Apesar de tentativas terem sido feitas para simplificar o projeto desse mapeador, usando técnicas de otimização, como *density evolution* e EXIT *charts*, o espaço de busca geralmente é grande, sendo necessário procurar a distribuição ótima entre todas as possíveis distribuições de mapeamento.

Nesse sentido, um esquema de mapeamento de bits de baixa complexidade é aconselhável para sistemas práticos, como é o caso de sistemas com grandes variações de canal. Desse modo, este trabalho concentra-se na proposta de um mapeamento de bits de baixa complexidade que permite adaptações rápidas a essas variações de canal de sistemas práticos, como é o caso dos sistemas QAM codificados com LDPC dos diversos padrões de sistema de comunicação.

1.2 Objetivos da Tese

O objetivo desta Tese é, portanto, propor uma nova técnica de mapeamento de bits, baseada na observação de que altruísmo pode ser vantajoso ao nível de grupo. O mapeamento de bits proposto promove cooperação unilateral dos bits codificados enviados em sub-canais "bons" (ou seja, com melhor qualidade), para bits enviados em sub-canais "ruins" (ou seja, com pior qualidade). Mais especificamente, ordenam-se os nós de variável de acordo com as qualidades dos seus sub-canais associados, selecionando-se um grupo de nós de variável "ruins" junto com um grupo de nós de variável "bons" ou "ajudantes". A ideia é forçar conexões (no grafo de Tanner do código LDPC) entre os dois grupos de nós, através de um conjunto de nós de paridade *"root"*, permitindo, assim, uma cooperação unilateral. Além disso, com o design proposto o espaço de busca do mapeamento de bits é reduzido significativamente, e a otimização pode ser possível sem acrescentar muita complexidade. Através da análise de EXIT *charts*, essa otimização pode ser realizada, visando alcançar um desempenho ainda melhor para sistemas QAM codificados com LDPC.

O grafo de Tanner resultante é semelhante ao dos códigos *root*-LDPC, introduzidos em [24] para canais de desvanecimento de bloco não-ergódico. Por esta razão, chamou-se o mapeamento de bits proposto de *root-like*.

1.3 Contribuições da Tese

A contribuição principal desta Tese esta na proposta de um mapeamento de bits *root-like* para sistemas QAM codificados com LDPC. O mapeamento de bits é feito de tal forma que nós de variável bons cooperem com nós de variável ruins visando um melhor desempenho geral do sistema. Diferentemente dos mapeamentos de bits anteriores, o mapeamento de bits *root-like*, baseia-se unicamente no desequilíbrio da confiabilidade dos sub-canais, não levando em consideração explicitamente a característica de UEP dos códigos LDPC, embora algumas restrições no grafo de Tanner do código precisem ser impostas. Isso faz com que o mapeamento de bits *root-like* seja simples de implementar.

A proposta de mapeamento de bits foi apresentada em [25], com bons resultados sem otimização, para ambos os sistemas de transmissão por portadora única (*single-carrier*) e sistemas baseados em OFDM.

Visando alcançar um desempenho ainda melhor para sistemas QAM codificados com LDPC, esta Tese também introduz a otimização do mapeamento de bits *root-like* através da análise de EXIT *charts*. Observou-se que no caso do canal de portadora única o espaço de

busca de otimização é significativamente reduzido pelas condições do projeto do mapeamento de bits *root-like*, em comparação com os métodos de otimização propostos em [15, 21] e [22]. Além disso, será mostrado que é possível alcançar resultados bastante semelhantes aos obtidos com uma otimização do mapeamento de bits mais complexa, como a proposta em [22]. No caso do canal OFDM e em outros cenários em que o espaço de busca não é tão reduzido, os resultados mostram que a ideia principal por trás do mapeamento de bits *root-like* associada a uma simples regra prática é suficiente para garantir um bom desempenho, sem a necessidade de otimização. Portanto, para ambos os cenários de portadora única e de multi-portadoras, e para diferentes padrões de aplicação (G.hn e IEEE 802.11n), será mostrado nesta Tese, que o mapeamento de bits *root-like* é uma alternativa interessante para manter um bom desempenho através de canais com variações, aplicando uma otimização do mapeamento de bits de baixa complexidade quando necessário.

Esta Tese investiga também o impacto do nível de desequilíbrio de confiabilidade através de todos os sub-canais sobre o desempenho do mapeamento de bits *root-like*. Observou-se que, quanto maior o desequilíbrio entre os sub-canais "ruins" e os "bons", em termos de qualidade, mais benéfico é o efeito do mapeamento de bits *root-like* no desempenho do sistema, mostrando o seu potencial nesse cenário.

Os resultados da otimização do mapeamento de bits utilizando EXIT *charts* foram publicados recentemente em [26].

Além disso, esta Tese realiza uma revisão de códigos LDPC, do mapeamento de bits, da técnica de otimização através de EXIT *charts* e de algumas características envolvendo codificação de canal dos padrões de sistemas de comunicação G.hn e IEEE 802.11n.

1.4 Organização da Tese

O restante deste trabalho está estruturado da seguinte maneira:

- **Capítulo 2** - Apresenta uma introdução aos códigos de blocos lineares, seguida de uma descrição dos aspectos teóricos dos códigos LDPC, incluindo métodos para codificação e decodificação. Também descreve os códigos QC-LDPC, a classe dos códigos LDPC usadas nas simulações.

- **Capítulo 3** - Descreve o funcionamento geral da técnica de mapeamento de bits. Também apresenta a nova técnica de mapeamento de bits proposta, inspirada nos códigos *root*-LDPC, chamada mapeamento de bits *root-like*.

- **Capítulo 4** - Descreve a técnica de EXIT *charts* aplicada aos códigos LDPC e também utilizada na otimização do mapeamento de bits. Também descreve a otimização do mapeamento de bits *root-like* através da análise de EXIT *charts*.

- **Capítulo 5** - Apresenta os resultados numéricos para o mapeamento de bits *root-like* e para a sua otimização, aplicado às matrizes de verificação de paridade e aos esquemas QAM dos padrões G.hn e IEEE802.11n, considerando-se os modos de transmissão por portadora única e OFDM com AWGN e com canais de desvanecimento *Rayleigh*. Além disso, apresenta um estudo do impacto do nível de desequilíbrio de confiabilidade através dos sub-canais no desempenho do mapeamento de bits *root-like*.

- **Capítulo 6** - Apresenta as conclusões do trabalho e sugestões para trabalhos futuros.

Capítulo 2

Códigos LDPC

Os códigos LDPC foram inicialmente propostos por Gallager em sua tese de doutorado, em 1962 [2], permanecendo esquecidos por 35 anos até serem redescobertos por McKay e Neal [3], na década de 90. São uma classe de códigos de blocos lineares com decodificação iterativa de fácil implementação, conseguindo atingir uma baixa probabilidade de erro e se aproximar do limite de Shannon em aplicações que exigem alta confiabilidade.

Este capítulo apresenta uma breve introdução aos códigos de blocos lineares necessária ao entendimento dos códigos LDPC, seguida de uma descrição dos códigos LDPC, e dos códigos QC-LDPC, a classe de códigos utilizada no presente trabalho.

2.1 Códigos de Blocos Lineares

Códigos de blocos lineares são especificados como códigos (n, k), em que k é o número de bits de mensagem e n é o tamanho da palavra-código. Em um codificador de bloco linear, a mensagem é dividida em uma sequência de blocos de k bits de mensagem cada. Para gerar uma palavra-código n, o codificador de bloco adiciona, de acordo com uma regra de codificação predefinida, $n - k$ bits de paridade ou redundância aos bits de mensagem. A adição dessa paridade é que garante, no decodificador, que essa mensagem possa ser recuperada, mesmo quando alguns bits na palavra-código são corrompidos durante a transmissão pelo canal.

Cada código de bloco linear é gerado através de uma matriz geradora $\mathbf{G}$ da qual se obtém as várias palavras do código. A matriz geradora $\mathbf{G}$ de um código $C(n, k)$ é uma matriz de dimensão $k \times n$, caracterizada por um conjunto de k linhas linearmente independentes entre si, que formam uma base para C [27, 28]:

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k-1} \end{bmatrix} = \begin{bmatrix} g_{0,0} & g_{0,1} & g_{0,2} & \cdots & g_{0,n-1} \\ g_{1,0} & g_{1,1} & g_{1,2} & \cdots & g_{1,n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{k-1,0} & g_{k-1,1} & g_{k-1,2} & \cdots & g_{k-1,n-1} \end{bmatrix}, \tag{2.1}$$

em que $\mathbf{g}_i = (g_{i0}, g_{i1}, ..., g_{i,n-1})$, para $0 \leq i < k$, e as k linhas linearmente independentes $\mathbf{g}_0, \mathbf{g}_1, ..., \mathbf{g}_{k-1}$ são designadas de tal forma que cada palavra-código $\mathbf{v}$ pode ser representada como uma combinação linear desses vetores.

No geral, a matriz geradora $\mathbf{G}$ é utilizada na codificação dos códigos de blocos lineares. Considerando $\mathbf{u} = (u_0, u_1, ..., u_{k-1})$, uma mensagem a ser codificada, a palavra-código resultante $\mathbf{v}$ pode ser obtida da seguinte forma:

$$\mathbf{v} = \mathbf{u} \cdot \mathbf{G}, \tag{2.2}$$

$$\mathbf{v} = (u_0, u_1, ..., u_{k-1}) \cdot \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k-1} \end{bmatrix} = u_0\mathbf{g}_0 + u_1\mathbf{g}_1 + ... + u_{k-1}\mathbf{g}_{k-1}. \tag{2.3}$$

Para um código linear *sistemático* (n, k), no qual a mensagem é inalterada e a paridade adicionada no final (ou no início) da mensagem, e no qual uma matriz identidade possa ser identificada através das linhas de $\mathbf{G}$, a matriz geradora $\mathbf{G}$ é composta de duas sub-matrizes, uma matriz identidade $k \times k$, indicada por $\mathbf{I}_k$ e uma sub-matriz de paridade $k \times (n-k)$, que gera os bits de paridade, indicada por $\mathbf{P}$, tal que [27, 28],

$$\mathbf{G} = \begin{bmatrix} \mathbf{I}_k & \mathbf{P} \end{bmatrix}, \tag{2.4}$$

em que

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k-1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & ... & 0 & p_{0,0} & p_{0,1} & ... & p_{0,n-k-1} \\ 0 & 1 & 0 & ... & 0 & p_{1,0} & p_{1,1} & ... & p_{1,n-k-1} \\ 0 & 0 & 1 & ... & 0 & p_{2,0} & p_{2,1} & ... & p_{2,n-k-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & ... & 1 & p_{k-1,0} & p_{k-1,1} & ... & p_{k-1,n-k-1} \end{bmatrix}. \tag{2.5}$$

Um exemplo simples do processo de codificação utilizando a matriz geradora é mostrado no Exemplo 2.1.

Exemplo 2.1. Considere o código linear (7,4) com a seguinte matriz geradora:

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix},$$

e a mensagem a ser codificada como sendo $\mathbf{u} = (1\ 1\ 0\ 1)$. A palavra-código $\mathbf{v}$ resultaria em

$$\mathbf{v} = 1\cdot\mathbf{g}_0 + 1\cdot\mathbf{g}_1 + 0\cdot\mathbf{g}_2 + 1\cdot\mathbf{g}_3 = (1\ 0\ 0\ 0\ 1\ 1\ 0) + (0\ 1\ 0\ 0\ 0\ 1\ 1) + (0\ 0\ 0\ 1\ 1\ 0\ 1) = (1\ 1\ 0\ 1\ 0\ 0\ 0).$$

Para cada matriz $\mathbf{G}$ de $k \times n$, existe uma matriz $(n-k) \times k$ denominada de matriz $\mathbf{H}$ ou *matriz de verificação de paridade*. Um código de blocos linear é, portanto, unicamente especificado pela sua matriz geradora ou pela sua matriz de verificação de paridade. A matriz $\mathbf{H}$ de dimensão $(n-k) \times k$ é definida como

$$\mathbf{H} = \begin{bmatrix} \mathbf{h}_0 \\ \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_{n-k-1} \end{bmatrix} = \begin{bmatrix} h_{0,0} & h_{0,1} & h_{0,2} & \ldots & h_{0,n-1} \\ h_{1,0} & h_{1,1} & h_{1,2} & \ldots & h_{1,n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_{n-k-1,0} & h_{n-k-1,1} & h_{n-k-1,2} & \ldots & h_{n-k-1,n-1} \end{bmatrix}. \tag{2.6}$$

Se a matriz $\mathbf{G}$ for sistemática, então a matriz $\mathbf{H}$ é composta de duas sub-matrizes, uma matriz identidade $(n-k)\times(n-k)$, indicada por $\mathbf{I}_{n-k}$, e a transposta da matriz de coeficientes $\mathbf{P}$, indicada por $\mathbf{P}^T$, tal que

$$\mathbf{H} = [\mathbf{P}^T\ \mathbf{I}_{n-k}] = \begin{bmatrix} p_{0,0} & p_{1,0} & \ldots & p_{k-1,0} & 1 & 0 & 0 & \ldots & 0 \\ p_{0,1} & p_{1,1} & \ldots & p_{k-1,1} & 0 & 1 & 0 & \ldots & 0 \\ p_{0,2} & p_{1,2} & \ldots & p_{k-1,2} & 0 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ p_{0,n-k-1} & p_{1,n-k-1} & \ldots & p_{k-1,n-k-1} & 0 & 0 & 0 & \ldots & 1 \end{bmatrix}. \tag{2.7}$$

As $n-k$ linhas de $\mathbf{H}$ são linearmente independentes, e cada vetor do espaço das linhas da matriz $\mathbf{G}$ é ortogonal as linhas da matriz $\mathbf{H}$ e vice-versa, ou seja, $\mathbf{v}$ só é uma palavra-código de um código gerado por $\mathbf{G}$, se e somente se [28]

$$\mathbf{H}\cdot\mathbf{v}^T = 0. \tag{2.8}$$

No geral, a codificação de um código de bloco linear é baseada na matriz geradora e, a decodificação é baseada na matriz de verificação de paridade. No entanto, muitas classes de códigos de blocos lineares, como os códigos LDPC, por exemplo, são construídos em termos da matriz de verificação de paridade, como será visto a seguir. A matriz de verificação de paridade $\mathbf{H}$ pode usada para verificar se a palavra recebida é uma palavra-código válida de um código C, ou se erros foram introduzidos durante a transmissão, como pode ser visto no Exemplo 2.2.

Exemplo 2.2. Seja $\mathbf{v} = [v_1\ v_2\ v_3\ v_4\ v_5\ v_6]$ uma palavra-código de um código C, e $\mathbf{H}$ a sua matriz de verificação de paridade. Então, de acordo com a Eq. (2.8) [29]

$$\underbrace{\begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{H}} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}.$$

Cada linha de $\mathbf{H}$ corresponde a uma equação de verificação de paridade e cada coluna de $\mathbf{H}$ corresponde a um bit da palavra-código. A (j,i)ésima entrada de $\mathbf{H}$ é 1 se o i-ésimo bit da palavra-código é incluído na j-ésima equação de verificação de paridade. Assim, as equações de verificação de paridade são definidas como

$$\begin{aligned} v_1 \oplus v_2 \oplus v_4 &= 0, \\ v_2 \oplus v_3 \oplus v_5 &= 0, \\ v_1 \oplus v_2 \oplus v_3 \oplus v_6 &= 0. \end{aligned}$$

Considere que a palavra-código recebida seja $\hat{\mathbf{v}} = [1\ 1\ 0\ 0\ 0\ 0]$. Checando a palavra-código $\hat{\mathbf{v}}$, tem-se

$$\begin{aligned} 1 \oplus 1 \oplus 0 &= 0, \\ 1 \oplus 0 \oplus 0 &= 1, \\ 1 \oplus 1 \oplus 0 \oplus 0 &= 0, \end{aligned}$$

então $\hat{\mathbf{v}}$ não é uma palavra-código válida desse código, pois não satisfaz a Eq. (2.8).

2.2 Códigos LDPC

Os códigos LDPC são códigos de blocos lineares caracterizados pela sua matriz de verificação de paridade **H**, que deve ser esparsa, ou seja, possuir uma baixa densidade de 1's. Além dessa condição de esparsidade, um código LDPC por si só não é diferente de nenhum outro código de bloco linear. A grande diferença está no fato dos códigos LDPC utilizarem algoritmos de decodificação iterativa que funcionam bem quando a matriz de verificação de paridade é esparsa.

A matriz de verificação de paridade de um código LDPC pode ser representada na forma de um grafo bipartido, ou grafo de Tanner [30]. Tanner considerou que qualquer código de bloco linear poderia ser representado por uma grafo bipartido. Um grafo bipartido é formado por nós de dois tipos, que se conectam apenas aos nós de tipo diferente. Os nós de variável (ou nós de bit) são representados por círculos, e os nós de paridade, por quadrados, como pode ser visto na Figura 2.1.

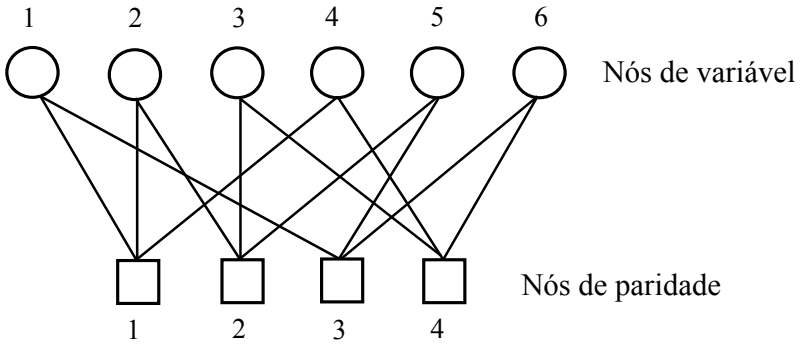

Figura 2.1: Grafo de Tanner.

O grafo possui ramos ligando os nós de variável aos nós de paridade apenas quando o valor do elemento $h_{i,j}$ na matriz de verificação de paridade for 1. A matriz de verificação de paridade **H**, de tamanho $p \times n$, correspondente ao grafo da Figura 2.1 é dada por:

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}, \tag{2.9}$$

na qual as colunas da matriz **H** correspondem aos nós de variável, e as linhas correspondem aos nós de paridade.

Um conceito importante relativo aos grafos de Tanner é o conceito de *girth* [29, 27]. O *girth* é o menor ciclo em um grafo de Tanner. E um ciclo é um caminho fechado em um grafo

que se inicia e termina no mesmo nó. Qualquer ciclo em um grafo de Tanner tem comprimento par e o menor comprimento de ciclo em um grafo bipartido é igual a 4. Para o algoritmo de decodificação Soma-Produto (explicado mais adiante), ciclos no grafo de Tanner levam a correlações nas probabilidades trocadas no decodificador, afetando a sua convergência. Quanto menor o ciclo, maior o efeito. Por isso, melhoras no desempenho do decodificador podem ser obtidas evitando-se ciclos de tamanho 4, logo *girth-4*.

Quanto a sua construção, os códigos LDPC podem ser classificados em duas categorias:

- Códigos aleatórios ou pseudo-aleatórios e
- Códigos algébricos.

Os códigos aleatórios ou pseudo-aleatórios são aqueles cujas matrizes de verificação de paridade são geradas aleatoriamente. Já os códigos algébricos são construídos usando-se corpos finitos ou geometrias finitas.

As matrizes de verificação de paridade dos códigos aleatórios geralmente não possuem uma estrutura. Essa falta de estrutura dificulta a codificação e a decodificação, deixando-as complexas. Recentemente, códigos com estrutura, como os códigos Cíclicos ou Quasi-Cíclicos, foram criados, simplificando a codificação e a decodificação. Os códigos Quasi-Cíclicos serão os usados neste trabalho e serão descritos mais adiante.

2.2.1 Códigos LDPC Regulares e Irregulares

Um código LDPC é dito *regular* se o número de 1's por coluna, d_v, e o número de 1's por linha, d_c, da matriz $\mathbf{H}$ forem constantes. A taxa R de um código regular é definida por [29]:

$$R = 1 - \frac{d_v}{d_c}. \tag{2.10}$$

Note que o grafo de Tanner da Figura 2.1 é regular, ou seja, cada nó de variável tem dois ramos conectados e cada nó de paridade tem três ramos conectados. Diz-se que o *grau* de cada nó de variável é igual a 2 (ou $d_v = 2$) e o *grau* de cada nó de paridade é igual a 3 (ou $d_c = 3$).

Um código LDPC é dito *irregular* quando possui os números de 1's por coluna e por linha diferentes. Nesse caso, a notação utilizada para os códigos LDPC regulares não é útil, já que o número de ramos no grafo de Tanner não é constante, ou seja, os nós de variável ou de paridade podem ter diferentes graus.

Assim, para os códigos LDPC irregulares, introduz-se o conceito de *distribuição de graus*. Cada tipo de nó (de variável e de paridade) possui uma distribuição de grau, sendo possível obtê-las através de uma expressão utilizando polinômios. A distribuição de graus de nós de variável, $\lambda(x)$, é dada por [11, 27]

$$\lambda(x) = \sum_{i=1}^{d_v^{max}} \lambda_i x^{i-1}, \tag{2.11}$$

em que λ_i denota a fração de ramos conectados a nós de variável de grau i e d_v^{max} denota o valor máximo do grau de variável. Similarmente, a distribuição de graus de nós de paridade, $\rho(x)$, é dada por

$$\rho(x) = \sum_{q=1}^{d_c^{max}} \rho_q x^{q-1}, \tag{2.12}$$

em que ρ_q denota a fração de ramos conectados a nós de paridade de grau q e d_c^{max} denota o valor máximo do grau de paridade.

Os coeficientes λ_i e ρ_q dos polinômios necessitam satisfazer as seguintes restrições [31, 32]:

$$\begin{aligned} 0 \leq \lambda_i \leq 1 \\ 0 \leq \rho_q \leq 1 \\ \sum_{i=1}^{d_v^{max}} \lambda_i = 1 \\ \sum_{q=1}^{d_c^{max}} \rho_q = 1. \end{aligned} \tag{2.13}$$

Se um grafo de Tanner tem E ramos, então o número de nós de variável com grau i é dado por

$$z_i = \frac{E\lambda_i}{i}, \tag{2.14}$$

e o número de nós de paridade com grau q é dado por

$$c_q = \frac{E\rho_q}{q}. \tag{2.15}$$

Assim, o número n de nós de variável é

$$n = \sum_i z_i = E \sum_i \frac{\lambda_i}{i} = E \int_0^1 \lambda(x) dx, \tag{2.16}$$

e o número p de nós de paridade é

$$p = \sum_q c_q = E \sum_q \frac{\rho_q}{q} = E \int_0^1 \rho(x) dx. \tag{2.17}$$

A taxa de código de um código LDPC irregular é calculada como

$$R = \frac{k}{n} = \frac{n-p}{n} = 1 - \frac{p}{n}. \tag{2.18}$$

Assim,

$$R = 1 - \frac{\sum_i \frac{\lambda_i}{i}}{\sum_q \frac{\rho_q}{q}} = 1 - \frac{\int_0^1 \rho(x) dx}{\int_0^1 \lambda(x) dx}. \tag{2.19}$$

Os polinômios $\lambda(x)$ e $\rho(x)$, representam a distribuição de graus de um grafo de Tanner a partir de um perspectiva de ramos. No entanto, as distribuições de grau também podem ser representadas por uma perspectiva de nós, em que $\tilde{\lambda}_i$ é a fração de nós de variável com grau i e $\tilde{\rho_q}$ é a fração de nós de paridade com grau q:

$$\tilde{\lambda}_i = \frac{\frac{\lambda_i}{i}}{\int_0^1 \lambda(x) dx}, \tag{2.20}$$

$$\tilde{\rho_q} = \frac{\frac{\rho_q}{q}}{\int_0^1 \rho(x) dx}. \tag{2.21}$$

Os códigos LDPC irregulares [31, 33] possuem um desempenho melhor se comparados aos códigos LDPC regulares com blocos longos e estão entre os códigos binários mais poderosos conhecidos hoje. Em [34], por exemplo, é mostrado que códigos irregulares podem chegar a 0,0045 dB do limite de Shannon em um canal AWGN, quando projetados corretamente.

2.3 Codificação

Uma das formas de se realizar a codificação de códigos LDPC é através da matriz geradora **G**, assim como na codificação dos códigos de blocos lineares convencionais, como visto na Sessão 2.1. Como os códigos LDPC são definidos pela sua matriz de verificação de paridade, a matriz geradora **G** pode ser encontrada através da matriz **H**, por eliminação gaussiana.

No entanto, as matrizes de verificação de paridade dos códigos LDPC geralmente não são sistemáticas. Assim, para se transformar a matriz **H** para a forma sistemática, aplicam-se diversas operações entre as linhas da matriz, colocando a matriz na forma escalonada e na forma escalonada reduzida e ainda realizando permutações entre as colunas. A partir da matriz **H** na forma sistemática, a matriz **G** pode ser encontrada.

Apesar de todo esse processamento poder ser feito *off-line*, esse método de codificação não é viável para os códigos LDPC, pois estes geralmente possuem comprimento de palavra-código longos, e pelo fato de **G** não ser esparsa, a complexidade da operação de multiplicação na codificação $\mathbf{v} = \mathbf{uG}$, acaba sendo da ordem de n^2, em que n é o tamanho da palavra-código. Outros métodos podem ser usados que não utilizam a matriz **G**, como por exemplo o método de codificação com complexidade quase linear proposto por Richardson e Urbanke [35], que será o utilizado no presente trabalho e descrito a seguir.

2.3.1 Codificação com Complexidade Quase Linear

De acordo com o método proposto por Richardson e Urbanke em [35], ao invés de se utilizar a matriz **G** na codificação, é possível utilizar diretamente a matriz de verificação de paridade, bastando transformá-la para a forma triangular inferior aproximada, através da permutação de linhas e colunas, mantendo assim a matriz esparsa.

Considere uma matriz $p \times n$ de verificação de paridade **H**. Através de eliminação gaussiana, pode-se transformar a matriz H para a forma triangular inferior equivalente, como mostra a Figura 2.2.

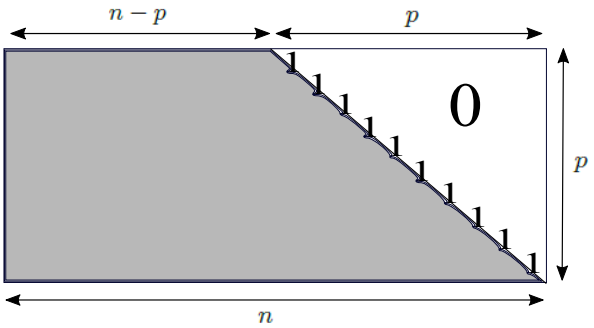

Figura 2.2: Matriz de verificação de paridade na forma triangular inferior equivalente.

Através somente da permutação de linhas e colunas, pode-se também trazer a matriz de verificação de paridade para a forma triangular inferior aproximada, como mostra a Figura 2.3. Nota-se que, como essa transformação foi possível apenas com permutações, a matriz continua sendo esparsa [35].

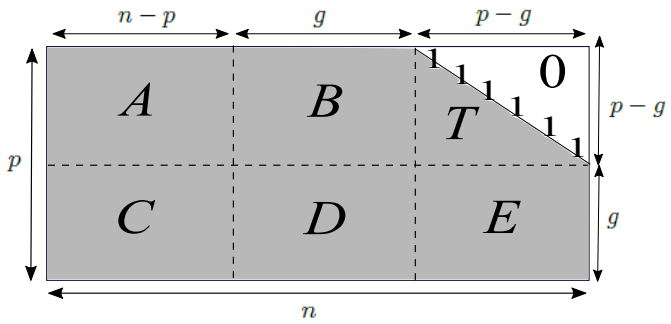

Figura 2.3: Matriz de verificação de paridade na forma triangular inferior aproximada.

Considere, então, a matriz $\mathbf{H}$ de tamanho $p \times n$ como tendo a forma:

$$\mathbf{H} = \begin{bmatrix} A & B & T \\ C & D & E \end{bmatrix}, \tag{2.22}$$

em que A é $(p-g) \times (n-p)$, B é $(p-g) \times g$, C é $g \times (n-p)$, D é $g \times g$, E é $g \times (p-g)$ e T é uma matriz triangular inferior de tamanho $(p-g) \times (p-g)$, com 1's ao longo da diagonal. As matrizes C, D e E têm g linhas cada, em que g representa o *gap* da representação aproximada. Quanto menor o *gap*, menor a complexidade da codificação.

Primeiramente, utilizando eliminação Gaussiana, multiplica-se a matriz $\mathbf{H}$ por:

$$\begin{bmatrix} I & 0 \\ -ET^{-1} & I \end{bmatrix} \begin{bmatrix} A & B & T \\ C & D & E \end{bmatrix} = \begin{bmatrix} A & B & T \\ -ET^{-1}A + C & -ET^{-1}B + D & 0 \end{bmatrix}, \tag{2.23}$$

em que I é uma matriz identidade.

Seja o vetor $\mathbf{v} = \begin{bmatrix} \mathbf{u} & \mathbf{p_1} & \mathbf{p_2} \end{bmatrix}$ uma palavra-código, em que $\mathbf{u}$ representa a mensagem e $\mathbf{p}$ a paridade, dividida em duas partes $\mathbf{p_1}$ e $\mathbf{p_2}$, e devendo satisfazer a equação de verificação de paridade $\mathbf{Hv^T} = 0$. Obtêm-se:

$$A\mathbf{u}^T + B\mathbf{p_1}^T + T\mathbf{p_2}^T = 0 \tag{2.24}$$

e

$$(-ET^{-1}A + C)\mathbf{u}^T + (-ET^{-1}B + D)\mathbf{p_1}^T = 0. \tag{2.25}$$

Determina-se $\phi = (-ET^{-1}B + D)$. Note que ϕ necessita ser não-singular. A não singularidade também pode ser encontrada através da permutação das linhas e colunas.

O objetivo da codificação é encontrar os vetores de paridade, portanto, isolando $\mathbf{p_1}$ na Eq. (2.25), obtêm-se:

$$\mathbf{p_1}^T = -\phi^{-1}(-ET^{-1}A + C)\mathbf{u}^T. \tag{2.26}$$

Nesse caso, a complexidade alta na resolução da equação está na multiplicação entre as matrizes. No entanto, mantendo g o menor possível esta complexidade pode ser diminuída. De posse de $\mathbf{p_1}$, é possível encontrar $\mathbf{p_2}$ através de:

$$\mathbf{p_2}^T = -T^{-1}[A\mathbf{u}^T + B\mathbf{p_1}^T]. \tag{2.27}$$

A complexidade dessa operação é mantida baixa, já que A e B são esparsas, e como T é triangular inferior, $\mathbf{p_2}$ pode ser encontrado por *back substitution* [29].

Esse método de codificação será demonstrado através do Exemplo 2.3.

Exemplo 2.3. Considere a matriz de verificação de paridade de um código ($d_v = 3, d_v = 6$)-regular com comprimento 12, já na forma triangular superior aproximada [35]:

$$\mathbf{H} = \left[\begin{array}{c|c|c} A & B & T \\ \hline C & D & E \end{array}\right] = \left[\begin{array}{cccccc|cc|cccc} 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ \hline 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{array}\right].$$

Considere a mensagem a ser codificada $\mathbf{u} = [1\ 0\ 0\ 0\ 0\ 0]$. Para determinar $\mathbf{p_1}$ utilizou-se a Eq. (2.26). Calculando cada parcela da Equação de forma separada, tem-se:

$$\begin{aligned} A\mathbf{u}^T &= (1\ 1\ 0\ 1)^T, \\ T^{-1}[A\mathbf{u}^T] &= (1\ 1\ 0\ 0)^T, \\ \left[-ET^{-1}A\mathbf{u}^T\right] &= (0\ 1)^T, \\ C\mathbf{u}^T &= (0\ 0)^T, \\ \left[-ET^{-1}A\mathbf{u}^T\right] + C\mathbf{u}^T &= (0\ 1)^T. \end{aligned}$$

Lembrando que $\phi = (-ET^{-1}B + D)$, resultando em:

$$\phi = (-ET^{-1}B + D) = \left[\begin{array}{cc} 1 & 0 \\ 1 & 1 \end{array}\right].$$

Logo,

$$\mathbf{p_1}^T = \phi^{-1}(-ET^{-1}A + C)\mathbf{u}^T = (0\ 1)^T.$$

De maneira semelhante, determinou-se $\mathbf{p_2}$, fazendo:

$$B\mathbf{p_1}^T = (0\ 1\ 0\ 0)^T,$$
$$\left[A\mathbf{u}^T\right] + \left[B\mathbf{p_1}^T\right] = (1\ 0\ 0\ 1)^T.$$

Logo, tem-se

$$\mathbf{p_2}^T = \left[A\mathbf{u}^T + B\mathbf{p_1}^T\right] T^{-1} = (1\ 0\ 1\ 0)^T.$$

Portanto, a palavra-código resultante é

$$\mathbf{v} = \left[\begin{array}{ccc} \mathbf{u} & \mathbf{p_1} & \mathbf{p_2} \end{array}\right] = [1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 0].$$

2.4 Decodificação

Uma das grandes vantagens dos códigos LDPC está na sua decodificação iterativa, que possibilita ao código apresentar um excelente desempenho mesmo operando-se muito próximo da capacidade de Shannon [1], para diferentes tipos de canais.

A decodificação iterativa é baseada na troca de mensagens no grafo de Tanner, onde as mensagens passam de trás pra frente entre os nós de variável e de paridade, iterativamente, até que um bom resultado seja alcançado. Os algoritmos de *message-passing* são um dos algoritmos usados na decodificação iterativa dos códigos LDPC, podendo ser classificados como [29]:

- Algoritmo *Bit-flipping* (Decisão abrupta (*Hard*)): utiliza mensagens binárias. Uma decisão binária sobre cada bit recebido é passada ao decodificador. O algoritmo pára quando uma palavra-código válida é encontrada, ou seja, quando as equações de verificação de paridade são satisfeitas.

- Algoritmo *Belief-Propagation* (Decisão Suave (*Soft*)): as mensagens são funções de probabilidades que representam o nível de confiança sobre o valor dos bits da palavra-código. Os valores probabilísticos podem ser representados como razões de verossimilhança logarítmica (*log-likelihood ratios* (LLRs)), permitindo que os cálculos sejam feitos através de soma e produto. Enquanto probabilidades precisam ser multiplicadas, LLRs só precisam ser adicionadas, reduzindo a complexidade. A essa decodificação dá-se o nome Soma-produto (SPA).

O algoritmo com decisão abrupta possui a vantagem da baixa complexidade, mas oferece o pior desempenho em termos de ganho de codificação. Por outro lado, os algoritmos com decisão suave são mais complexos, mas apresentam um melhor desempenho, sendo mais utilizados.

2.4.1 Decodificação Soma-Produto

O algoritmo soma-produto [2, 29] é um algoritmo *message-passing* com decisão suave. As mensagens trocadas entre os nós na decodificação são valores probabilísticos representados pela LLR, definida por:

$$\mathrm{LLR}(x) = \log\left(\frac{p(x=0)}{p(x=1)}\right), \tag{2.28}$$

em que x é um valor binário. A vantagem está no fato de as LLRs poderem ser adicionadas, reduzindo a complexidade.

Chama-se de probabilidade *intrínseca* ou *a priori*, P_i^{int}, a probabilidade original do bit, independente das restrições do código. E de probabilidade *extrínseca* ou *a posteriori*, $P_{j,i}^{\mathrm{ext}}$, a informação extra sobre o bit i vinda da verificação de paridade. O algoritmo soma-produto segue os seguintes passos [29]:

Inicialização Calcula-se as probabilidades intrínsecas vindas do canal. A mensagem inicial enviada do nó de bit i para o nó de paridade j, $M_{j,i}$, é a LLR R_i do sinal recebido y_i, dado o conhecimento das propriedades do canal:

$$M_{j,i} = \mathrm{LLR}(P_i^{\mathrm{int}}) = R_i = \log\left(\frac{p(v_i=0 \mid y_i)}{p(v_i=1 \mid y_i)}\right). \tag{2.29}$$

Por exemplo, para um canal AWGN:

$$M_{j,i} = R_i = 4y_i\frac{E_b}{N_0}. \tag{2.30}$$

Paridade-para-bit A LLR extrínseca $E_{j,i}$ do nó de paridade j para o nó de bit i é calculada como:

$$E_{j,i} = \text{LLR}(P_{j,i}^{\text{ext}}) = \log\left(\frac{1+\prod_{i'\in B_j, i'\neq i}\tanh(M_{j,i'})/2)}{1-\prod_{i'\in B_j, i'\neq i}\tanh(M_{j,i'})/2)}\right), \tag{2.31}$$

em que B_j representa o conjunto de bits na j-ésima equação da matriz de verificação de paridade $\mathbf{H}$.

Para reduzir a complexidade do decodificador, o algoritmo soma-produto pode ser modificado, reduzindo o termo de produto da LLR extrínseca por um somatório:

$$E_{j,i} = \left(\prod_{i'}\text{sign}(M_{j,i'})\right)\phi\left(\sum_{i'}\phi(\mid M_{j,i'}\mid)\right)$$

em que $\phi(x) = \log\frac{e^x+1}{e^x-1}$.

Teste da palavra-código A LLR L_i de cada bit i é a soma das LLRs extrínsecas e da LLR intrínseca calculada na "Inicialização".

$$L_i = R_i + \sum_{j\in A_i} E_{j,i}, \tag{2.32}$$

em que A_i representa o conjunto de nós de paridade conectados ao nó de variável i.

Para cada bit, uma decisão abrupta é feita:

$$\hat{c}_i = \begin{cases} 1, \text{se } L_i \leq 0 \\ 0, \text{se } L_i \geq 0 \end{cases}$$

Se $\hat{\mathbf{c}} = [c_1, ..., c_n]$ é uma palavra-código válida ($\mathbf{H}\hat{\mathbf{c}}^T = 0$), ou se o número máximo de interações for alcançado, o algoritmo termina.

Bit-para-Paridade Atualiza-se a mensagem enviada do nó de bit i para o nó de paridade j, sem a componente $E_{j,i}$ que foi recebida do nó j:

$$M_{j,i} = R_i + \sum_{j\in A_i, j'\neq j} E_{j,i}. \tag{2.33}$$

Se a palavra não for válida, volta-se para o passo "Paridade-para-bit".

Um exemplo simples da decodificação soma-produto é mostrado no Exemplo 2.4.

Exemplo 2.4. O codificador de um código LDPC representado pelo matriz **H** na Eq. (2.9) e pelo grafo de Tanner na Figura 2.1 é usado para gerar a palavra-código $\mathbf{v} = [0\ 0\ 1\ 0\ 1\ 1]$. A palavra-código **v** é transmitida através de um canal binário simétrico (BSC) com probabilidade $\varepsilon = 0.2$ e o sinal recebido é $\mathbf{y} = [1\ 0\ 1\ 0\ 1\ 1]$ [29].

As LLRs a *priori* para o canal BSC são:

$$R_i = \begin{cases} \log \frac{\varepsilon}{1-\varepsilon} = \log \frac{0.2}{0.8} = -1.3863, \text{ se } y_i = 1 \\ \log \frac{1-\varepsilon}{\varepsilon} = \log \frac{0.8}{0.2} = 1.3863, \text{ se } y_i = 0 \end{cases}$$

Logo, as LLRs do sinal recebido **y** são definidas como

$$\mathbf{R} = [-1.3836\ 1.3836\ -1.3836\ 1.3836\ -1.3836\ -1.3836].$$

Nota-se na Figura 2.1, que o nó de variável 1 é incluído nos nós de paridade 1 e 3. Logo, $M_{1,1}$ e $M_{3,1}$ são inicializados com R_1:

$$M_{1,1} = R_1 = -1.3863 \text{ e } M_{3,1} = R_1 = -1.3863.$$

O mesmo pode ser obtido para os demais nós de variável.

No passo **Paridade-para-bit**, calcula-se a LLR extrínseca dos nós de paridade para os nós de variável. Assim, o nó de paridade 1 na Figura 2.1 inclui os nós de variável 1, 2 e 4. Logo a LLR extrínseca do nó de paridade 1 para o nó de variável 1 depende das LLRs dos nós de variável 2 e 4, podendo ser calculada através da Eq. (2.31):

$$E_{1,1} = \log \left(\frac{1 + \tanh(M_{1,2}/2) \tanh(M_{1,4}/2))}{1 - \tanh(M_{1,2}/2) \tanh(M_{1,4}/2))} \right) = 0.7538.$$

O mesmo pode ser obtido para os demais nós de paridade.

No passo **Teste da palavra-código**, a LLR do nó de variável 1, por exemplo, é calculada através da soma das LLRs extrínsecas dos nós de paridade 1 e 3 e da LLR intrínseca do canal calculada na inicialização, logo:

$$L_1 = R_1 + E_{1,1} + E_{3,1} = -1.3863 + 0.7538 + 0.7538 = 0.1213.$$

Apesar de a LLR intrínseca do canal ser negativa, indicando que o primeiro nó de variável (bit) é igual a 1, ambas as LLRs extrínsecas são positivas, indicando que o bit é zero. As LLRs extrínsecas são grandes o suficiente para gerar um resultado positivo, trocando a decisão com relação ao primeiro bit. Assim, como $L_1 > 0$, logo o primeiro bit é igual a 0.

Repetindo os cálculo para os bits de 2 a 6 na palavra-código $\mathbf{v}$, obtêm-se:

$$\hat{\mathbf{v}} = [0\ 0\ 1\ 0\ 1\ 1].$$

2.5 Códigos QC-LDPC

Os códigos LDPC utilizados no presente trabalho são os chamados *quasi-cyclic LDPC block codes* (QC-LDPC-BC) [36]. Os códigos QC-LDPC-BC são encontrados em praticamente todas as implementações utilizando códigos LDPC, devido a sua codificação eficiente. Além disso, serão utilizados no trabalho devido à estrutura das suas matrizes de verificação de paridade se encaixar muito bem no mapeamento de bits proposto no Cap. 3.

Os códigos QC-LDPC pertencem a uma classe de códigos LDPC irregulares cuja matriz de verificação de paridade consiste em sub-matrizes rotativas de blocos. Um código é dito quasi-cíclico se cada deslocamento cíclico de uma palavra-código por c posições resulta em uma palavra-código. Uma das vantagens desses códigos é a possibilidade de se codificar com complexidade linear em relação ao comprimento do código, utilizando o método de Richardson e Urbanke modificado para matrizes quasi-cíclicas, descrito em [36].

Um código QC-LDPC com taxa $R = K/N$, em que N é o número de bits codificados e K o número de bits de informação, é definido por uma matriz de verificação de paridade $(N - K) \times N$ consistindo num conjunto $c \times t$ de $b \times b$ sub-matrizes:

$$\mathbf{H} = \begin{bmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,t} \\ P_{2,1} & P_{2,2} & \cdots & P_{2,t} \\ \vdots & \vdots & \ddots & \vdots \\ P_{c,1} & P_{c,2} & \cdots & P_{c,t} \end{bmatrix}. \tag{2.34}$$

Cada $b \times b$ sub-matriz $P_{i,j}$ ou é uma matriz com apenas zeros, ou é uma matriz identidade rotativa, definida por exemplo, como

$$P = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ 1 & 0 & 0 & \cdots & 0 \end{bmatrix}. \tag{2.35}$$

Uma matriz identidade rotativa é uma matriz identidade que pode ter suas colunas deslocadas ciclicamente para a direita de uma valor inteiro. Note que o parâmetro $b = N/t$, chamado de fator de expansão de $\mathbf{H}$, controla o tamanho da palavra-código N.

A matriz de verificação de paridade $\mathbf{H}$ pode ser também expressa na sua forma compacta, sendo descrita como:

$$\mathbf{H_c} = \begin{bmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,t} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,t} \\ \vdots & \vdots & \ddots & \vdots \\ p_{c,1} & p_{c,2} & \cdots & p_{c,t} \end{bmatrix}. \tag{2.36}$$

Na forma compacta, uma sub-matriz com apenas zeros é especifica com $p_{i,j} = -1$, e uma sub-matriz identidade rotativa é especificada por um inteiro $p_{i,j}$ entre 0 e $b-1$, o qual define o número de deslocamentos para direita das colunas da matriz identidade. Claramente, a própria matriz identidade é especificada por $p_{i,j} = 0$. Veja a Figura 2.4 para um exemplo.

Para facilitar a codificação, $\mathbf{H}$ é também particionada em $\mathbf{H} = [\mathbf{A}_{c\times(t-c)}|\mathbf{D}_{c\times c}]$, onde a matriz $\mathbf{D}$ necessita ser de posto completo, ou seja, as linhas de $\mathbf{D}$ necessitam ser linearmente independentes.

O código QC-LDPC pode ser codificado utilizando uma modificação do método de Richardson e Urbanke (descrito na Sessão 2.3.1) proposta em [36] e resumida a seguir.

Considere a matriz $\mathbf{H}$ dividida em:

$$\mathbf{H} = [\mathbf{A}_{c\times(t-c)}|\mathbf{D}_{c\times c}] = \begin{bmatrix} & P^{p_{1,(t-(c-2))}} & I & 0 & ... & 0 & 0 \\ & 0 & P^{p_{2,(t-(c-3))}} & I & ... & 0 & 0 \\ & \vdots & 0 & P^{p_{3,(t-(c-4))}} & ... & 0 & 0 \\ \mathbf{A} & P^{y} & \vdots & \vdots & ... & \vdots & \vdots \\ & \vdots & \vdots & \vdots & ... & I & 0 \\ & 0 & 0 & 0 & ... & P^{p_{(c-1),(t-1)}} & I \\ & P^{x} & 0 & 0 & ... & 0 & P^{p_{c,t}} \end{bmatrix}, \tag{2.37}$$

em que $P^{p_{i,j}}$ é uma matriz $b \times b$ identidade rotativa, em que $p_{i,j}$ determina o número de deslocamento na matriz identidade, e I e 0 indicam a matriz identidade e a matriz apenas de zeros, respectivamente. A matriz identidade rotativa P^x, em que x determina o número de deslocamentos na matriz identidade, localizada na última linha de $\mathbf{H}$ e a matriz P^y, em que

y determina o número de deslocamentos na matriz identidade, localizada na l-ésima linha da matriz $\mathbf{H}$, em que $l \neq 1, c$, têm ambas um papel importante na eficiência do método proposto em [36], como será visto a seguir. Nota-se que l é convencionalmente escolhido na metade de c, apesar de l poder ser escolhido arbitrariamente.

A complexidade do método de codificação proposto por Richardson e Urbanke está principalmente no cálculo de $\mathbf{p_1}$, na Eq. (2.26), devido à multiplicação de matrizes, dado que ϕ^{-1} geralmente não é esparsa. No entanto, se ϕ puder ser escolhida como uma matriz identidade, então a complexidade da codificação pode ser dimensionada linearmente. Assim, a ideia principal do método proposto em [36] para garantir uma codificação eficiente é escolher a matriz ϕ como uma matriz identidade (ou como uma matriz identidade rotativa) através de uma seleção adequada de P^x e P^y na Eq. (2.37). Assim, a complexidade geral do cálculo de $\mathbf{p_1}$ pode ser reduzida, independentemente do tamanho da matriz identidade rotativa.

Por exemplo, para calcular $\phi = (-ET^{-1}B + D)$, considere que a matriz $\mathbf{H}$ de tamanho $c \times t$ na Eq (2.37) seja dividida na forma

$$\mathbf{H} = \begin{bmatrix} A & B & T \\ C & D & E \end{bmatrix},$$

em que A é $(c-1)b \times (t-c)b$, B é $(c-1)b \times b$, T é $(c-1)b \times (c-1)b$, C é $b \times (t-c)b$, $D = P^y$ é $b \times b$, E é $b \times (c-1)b$. Note que o *gap* (g) é igual a b.

Portanto, como

$$B^T = \left[(P^{p_{1,(t-(c-2))}})^T \; 0 \; \ldots \; (P^y)^T \; 0 \right], \tag{2.38}$$

e

$$D = P^x, \tag{2.39}$$

são dependentes das matrizes P^x e P^y, então uma escolha adequada de x e y garantirá que a matriz $\phi = (-ET^{-1}B + D)$ torne-se uma matriz identidade, reduzindo, assim, a complexidade da codificação.

Muitas vezes, a matriz $\mathbf{D}$ também apresenta uma estrutura de dupla diagonal, como pode ser visto na Figura 2.4, a qual mostra um exemplo de matriz $\mathbf{H}$ na forma compacta, retirada do padrão G.hn [8].

As matrizes do padrão G.hn e do padrão IEEE802.11n serão utilizadas nos resultados do presente trabalho. As matrizes utilizadas neste trabalho podem ser encontradas no Apêndice A. O restante das matrizes para outras taxas e diferentes tamanhos de blocos podem ser encontradas especificadas nos respectivos padrões [8, 6].

```
27 -1 -1 -1 55 19 -1 30 -1 -1 -1 -1 -1  0 -1 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1  0 -1  1 -1 70 -1 47 -1 62 -1 -1  0  0 -1 -1 -1 -1 -1 -1 -1 -1
-1 -1 41 -1 -1 -1 44 -1 -1 59 60 25 -1 -1  0  0 -1 -1 -1 -1 -1 -1 -1
16 77 -1 -1 -1  5 -1 48 -1 -1 -1 -1 -1 -1 -1  0  0 -1 -1 -1 -1 -1 -1
-1 -1 -1 45 -1 27 -1 46 19 -1 -1 -1 -1 -1 -1 -1  0  0 -1 -1 -1 -1 -1
-1 -1 63 -1 -1 -1 55 -1 -1 -1 48 26 10 -1 -1 -1 -1  0  0 -1 -1 -1 -1
-1 -1 -1 42 -1 21 -1 58 -1 41 -1 -1 -1 -1 -1 -1 -1 -1  0  0 -1 -1 -1
-1 -1 -1 -1 78  0 -1  7 52 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  0  0 -1 -1
-1 29  9 -1 -1 -1 37 -1 -1 -1 35 21 -1 -1 -1 -1 -1 -1 -1 -1  0  0 -1 -1
-1 -1 22 72 -1 -1 47 -1 -1 -1  0 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  0  0 -1
35 -1 -1 -1 -1 13 -1 35 -1 70 -1 -1  0 -1 -1 -1 -1 -1 -1 -1 -1 -1  0  0
-1 46 28 -1 -1 -1 38 -1 -1 -1  8 -1 10 58 -1 -1 -1 -1 -1 -1 -1 -1 -1  0
```

Figura 2.4: Matriz **H** do padrão G.hn ($c = 12, t = 24, b = 80$) para $N = 1920$ bits, $K = 960$ bits, taxa 1/2 [8].

Capítulo 3

Mapeamento de Bits *Root-Like*

Um "mapeador de bits" ou embaralhador (*interleaver*) pode ser usado em um sistema QAM codificado com LDPC para balancear as características de proteção desigual de erros (UEP) inerentes aos códigos LDPC irregulares e/ou o fato de os sub-canais terem diferentes qualidades, mapeando propriamente os bits codificados protegidos desigualmente para os diferentes sub-canais do modulador QAM, e assim melhorando o desempenho do sistema. O diagrama de blocos de um sistema QAM codificado com LDPC com mapeamento de bits é mostrado na Figura 3.1.

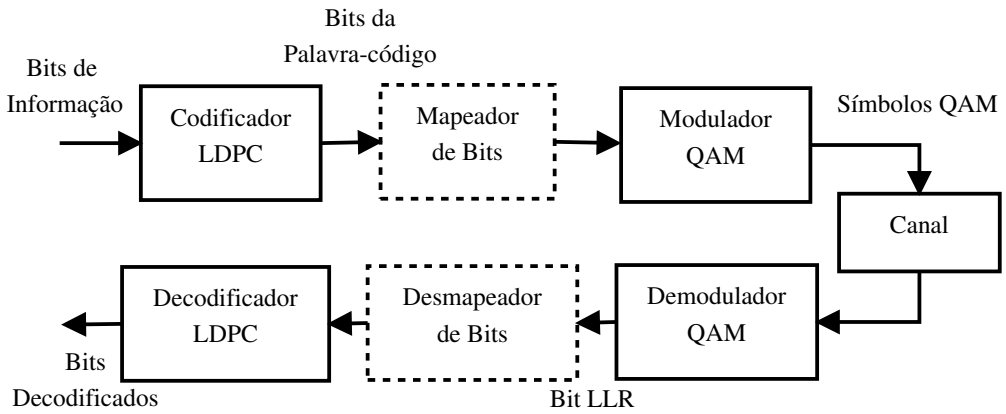

Figura 3.1: Diagrama de blocos de um sistema QAM codificado com LDPC com mapeamento de bits.

No transmissor, o codificador LDPC codifica os bits de informação binários gerando os bits codificados. Um mapeador de bits modifica a ordem em que os bits codificados são passados para o modulador QAM. No receptor, o "demodulador" produz LLRs que são colocadas na mesma ordem dos bits codificados originais por um desmapeador de bits. Finalmente, essas LLRs são usadas na decodificação *message-passing* iterativa do código LDPC.

3.1 Mapeamento de bits

O mapeamento de bits pode ser especificado pelas conexões entre nós de variável, nós de paridade e sub-canais, como mostrado na Figura 3.2.

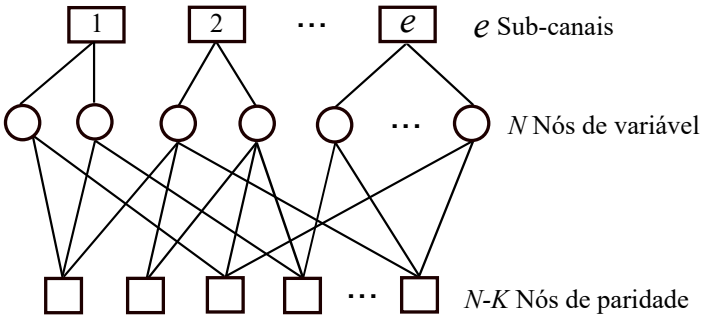

Figura 3.2: Conexões entre nós de variável, nós de paridade e sub-canais.

As conexões entre nós de variável e nós de paridade são as usuais (como especificadas no Cap. 2), e são representadas pelos ramos no grafo de Tanner de um código LDPC. Observa-se que códigos LDPC irregulares apresentam características de proteção desigual de erro (UEP), devido a irregularidade dos nós de variável, ou seja, diferentes graus de nós de variável implicam em diferentes níveis de proteção após a decodificação. Quanto maior o grau, maior é a proteção do nó de variável [18, 37]. Da mesma forma, sempre que bits codificados são enviados através de sub-canais com diferentes qualidades, eles também apresentam diferentes níveis de proteção. Nesse sentido, essa característica de UEP é, geralmente, balanceada com o uso do mapeamento de bits, selecionando cuidadosamente quais bits codificados são enviados em quais sub-canais.

Na Figura 3.2, se existe um ramo conectando um nó de variável a um nó de sub-canal, então o correspondente bit codificado é enviado através desse sub-canal. A confiabilidade (ou qualidade) dos sub-canais pode ser relacionada com parâmetros como a capacidade do canal ou a SNR média. Diz-se que um sub-canal j tem um nível de confiabilidade j.

Por exemplo, a transmissão de símbolos 2^m-QAM com rotulamento *Gray* através de um canal AWGN produz $e = \frac{m}{2}$ sub-canais de diferentes níveis de confiabilidade. Cada símbolo 2^m-QAM é identificado com um rótulo binário $(b_1, b_2, \ldots, b_{2e})$. O bit $b_\kappa, \kappa \in \{1, ..., 2e\}$ é enviado através do sub-canal j que tem nível de confiabilidade $j = [(\kappa - 1) \mod e] + 1$. Ou seja, considere, por exemplo, a transmissão de símbolos 256-QAM com rotulamento *Gray* através de um canal AWGN. Essa transmissão produzirá $e = \frac{8}{2} = 4$ sub-canais de diferentes níveis de confiabilidade. Cada símbolo 256-QAM é identificado com um rótulo binário de 8 bits $(b_1, b_2, \ldots, b_8)$. O bit b_1, de cada símbolo, é enviado através do sub-canal j com nível de confiabilidade $j = 1$, assim como o bit b_5. Os bits b_2 e b_6 são enviados através do sub-canal $j = 2$, os bits b_3 e b_7 são enviados através do sub-canal $j = 3$ e os bits b_4 e b_8 são enviados

através do sub-canal $j = 4$. Para um rotulamento *Gray* convencional, bits no nível $j = 1$ são os bits mais confiáveis (*most reliable bits* (MRBs)), e bits no nível $j = e$ são os bits menos confiáveis (*least reliable bits* (LRBs)) [1]. Essa confiabilidade está diretamente relacionada com a capacidade do canal ou informação mútua calculada para o nível de confiabilidade j associado (ver Figura 5.6 para um exemplo).

Como outro exemplo, quando um símbolo 2^m-QAM é transmitido em cada sub-portadora de um símbolo OFDM, os sub-canais na Figura 3.2 são as próprias sub-portadoras, e os seus níveis de confiabilidade são associados com diferentes SNRs médias das sub-portadoras. Nesse caso, o número de sub-canais e é igual ao número de sub-portadoras e m bits são enviados através de cada sub-canal.

No geral, o número de sub-canais, e, e as suas qualidades dependem do cenário de comunicação (tal como o tipo de modulação ou as características do canal). O número de bits (nós de variável) transmitidos em cada sub-canal é N/e.

Para os códigos irregulares, as distribuições de graus dos nós são definidas usando polinômios como mostrado no Cap. 2. Essas distribuições de graus dos nós em (2.11) e (2.12) são consideradas fixas.

No entanto, um outro conjunto de distribuição de graus de nós associado aos sub-canais pode ser definido como [19, 22]

$$m(x) = \sum_{j=1}^{e} \sum_{i=d_v^{\min}}^{d_v^{\max}} m_{j,i} \cdot x_j^i, \tag{3.1}$$

em que o coeficiente $m_{j,i}$ é a fração de ramos que mapeia os nós de variável de grau i para o nível de confiabilidade j, e $d_v^{\min}$ é o valor mínimo do grau de variável. Essas distribuições de graus dos nós podem variar, por exemplo, para fins de otimização, mas estão sujeitas a algumas restrições [19, 22]:

$$\begin{cases} 0 \leq m_{i,j} \leq 1 \\ \dfrac{\sum_{i=d_v^{\min}}^{d_v^{\max}} m_{j,i}.\lambda_i/i}{\sum_{i=d_v^{\min}}^{d_v^{\max}} \lambda_i/i} = 1/e \\ \sum_{j=1}^{e} m_{j,i} = 1 \end{cases} \tag{3.2}$$

[1] O padrão G.hn tem uma constelação baseada no rotulamento *Gray* com as posições dos bits na ordem inversa [8].

Uma observação importante é que, enquanto a permutação dos nós de variável não altera a distribuição de graus dos nós na Eq.(2.11), ela gera uma nova distribuição $\{m_{j,i}\}$ na Eq.(3.1). Isso é precisamente o que o mapeamento de bits faz no sistema codificado da Figura 3.1.

3.2 O Mapeamento de Bits Baseado nos Códigos *Root*-LDPC

O mapeamento de bits aqui proposto foi baseado na ideia de que altruísmo pode ser vantajoso a nível de grupo. Ou seja, um grupo de indivíduos pode ter mais chance de sobrevivência se alguns dos seus membros (altruístas) estiverem dispostos a subordinar os seus próprios recursos para ajudar os outros membros com necessidade, para o bem geral do grupo.

Nos termos do problema especificamente, uma vez que alguns bits LDPC codificados são transmitidos através de sub-canais de diferentes qualidades/capacidades, é possível projetar a matriz de verificação de paridade LDPC ou, talvez, se essa já for disponibilizada, mudar a ordem dos bits codificados, de modo que bits codificados enviados através de sub-canais "ruins" sejam ajudados por bits codificados enviados através de sub-canais "bons", melhorando assim o desempenho geral. Será mostrado a seguir, que isso pode ser possível forçando no grafo de Tanner do código LDPC algumas conexões entre nós de variável e de paridade, enquanto proibindo outras.

O mapeamento de bits proposto também foi inspirado nos chamados códigos *root*-LDPC, descritos a seguir.

3.2.1 Códigos *Root*-LDPC

Os códigos *Root*-LDPC são uma nova classe de códigos LDPC introduzidas em [24], projetados para alcançar máxima diversidade (*full diversity*) em canais com desvanecimento em blocos. A estrutura da matriz de verificação de paridade de um código *root*-LDPC (3,6) regular com taxa 1/2, é definida como [24]:

$$\mathbf{H} = \begin{array}{c} \\ 1c \\ 2c \end{array} \begin{array}{c} \begin{array}{cccc} 1i & 1p & 2i & 2p \end{array} \\ \left[\begin{array}{c|c|c|c} \begin{array}{c} 1 \\ \ddots \\ 1 \end{array} & 0 & H_{2i} & H_{2p} \\ \hline H_{1i} & H_{1p} & \begin{array}{c} 1 \\ \ddots \\ 1 \end{array} & 0 \end{array}\right] \end{array}. \tag{3.3}$$

Assumindo então 2 blocos de desvanecimento independentes, os bits de informação (i) e os bits de paridade (p) são divididos em dois grupos de nós de variável, $1i$ and $1p$, transmitidos no primeiro bloco de desvanecimento, e $2i$ e $2p$, transmitidos no segundo bloco de desvanecimento. Os nós de paridade também são divididos em dois grupos $1c$ e $2c$.

Para garantir máxima diversidade, todos os bits de informação são conectados a um tipo especial de nós de paridade chamados *rootchecks*, através das conexões *root* (matrizes identidade em **H**). Se um bit de informação V é transmitido em um bloco de desvanecimento, então um nó *rootcheck* para V conecta V a outros bits que são transmitidos no outro bloco de desvanecimento. Nota-se que bits de informação transmitidos em um mesmo bloco de desvanecimento não estão ligados entre si através de nós de paridade. Por outro lado, bits de informação transmitidos em blocos de desvanecimento distintos estão forçosamente ligados por nós de paridade (nós *rootcheck*). Desse modo, sob uma decodificação *message-passing* iterativa, os bits de informação podem ser recuperados também pela informação extrínseca dos bits transmitidos no outro bloco de desvanecimento, sendo essa a principal característica que serviu de inspiração para o mapeamento de bits "root-like".

Os bits de paridade não possuem essa proteção, representados pelas matrizes de zeros em **H**, e as outras partes de **H** são matrizes esparsas aleatórias.

3.2.2 Mapeamento de Bits *Root-Like*

A estratégia de mapeamento de bits aqui proposta pode ser explicada com ajuda do grafo na Figura 3.3 e do grafo mais geral na Figura 3.2, ambos aplicados ao sistema codificado embora desempenhando papeis diferentes ao longo da explicação.

Sejam $V_1, \ldots, V_N$ os nós de variável representando os bits codificados do código LDPC. Na sua ordem original, sem o mapeamento de bits, esses bits seriam transmitidos sequencialmente através dos e sub-canais distintos de acordo com as conexões na parte superior do grafo da Figura 3.2. Com o mapeamento de bits, os bits codificados são devidamente mapeados para os sub-canais, produzindo uma distribuição de mapeamento de bits diferente $m(x)$ em (3.1).

O mapeamento de bits *root-like* pode ser implementado a partir de uma matriz de

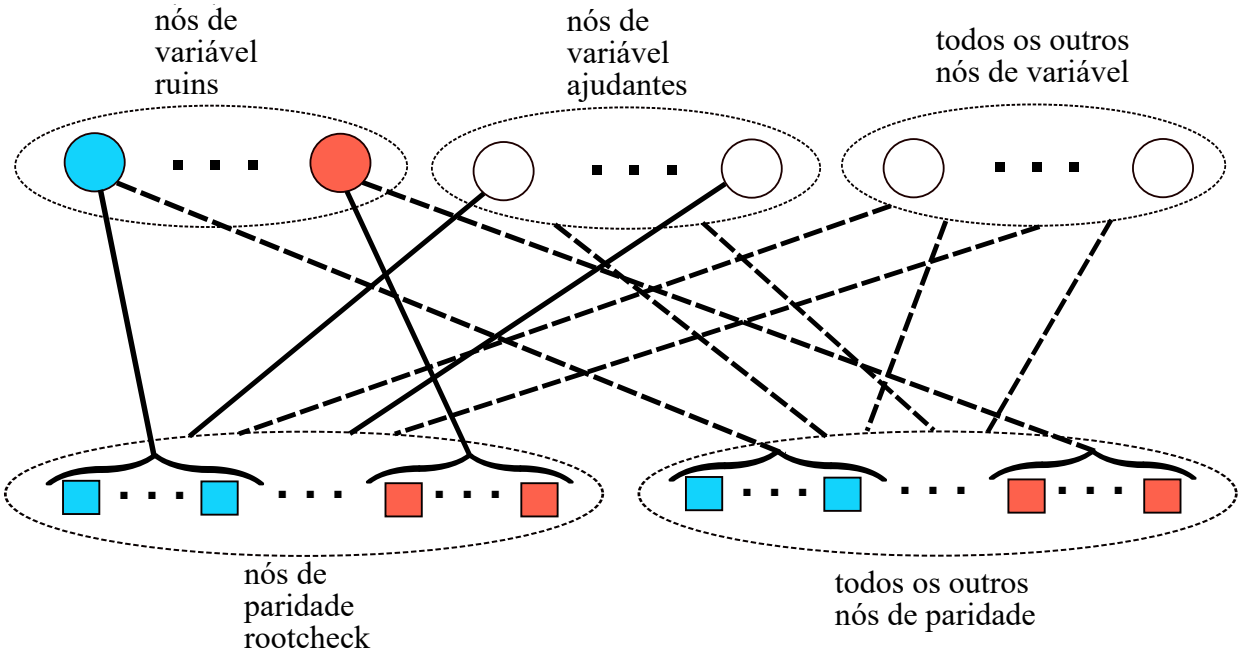

Figura 3.3: Esquema do mapeamento de bits *root-like*.

verificação de paridade já existente, satisfazendo certas condições ou então, especificado ao construir a matriz de verificação de paridade. Em ambos os casos, precisa-se primeiro identificar no grafo de Tanner dois conjuntos ($\mathcal{B}'$ and $\mathcal{H}'$) de nós de variável de acordo com as seguintes restrições:

1. O conjunto $\mathcal{B}'$ contém nós de variável satisfazendo a condição de que dois nós em $\mathcal{B}'$ não podem estar conectados, através de um nó de paridade.

2. O conjunto $\mathcal{H}'$ contém nós de variável que são conectados a nós em $\mathcal{B}'$ através de pelo menos um nó de paridade.

Os nós de variável $V_1, \ldots, V_N$ são então divididos em 3 sub-conjuntos:

- O sub-conjunto $\mathcal{B} \subseteq \mathcal{B}'$ de nós de variável "ruins"
- O sub-conjunto $\mathcal{H} \subseteq \mathcal{H}'$ de nós de variável "ajudantes" (ou "bons")
- O sub-conjunto $\mathcal{O}_v$ de "todos os outros" nós de variável.

Os nós de variável em $\mathcal{B}$ e $\mathcal{H}$ são nós especiais, cujas conexões precisam obedecer certas regras. Nós de variável em $\mathcal{O}_v$, por outro lado, não são restritos, e podem se conectar a qualquer nó de paridade[2].

[2]Se matriz de verificação de paridade já for disponibilizada, então algumas restrições também se aplicam aos nós em $\mathcal{O}_v$.

Os nós de paridade também são divididos em dois grupos:

- nós de paridade "*rootcheck*" ($\mathcal{R}$)
- "todos os outros" nós de paridade ($\mathcal{O}_c$)

Os nós "*rootcheck*" são nós de paridade que garantem as conexões entre nós "ruins" e nós "ajudantes". O grafo de Tanner com as conexões entre nós de variável e nós de paridade do mapeamento de bits *root-like* é mostrado na Figura 3.3.

As conexões entre nós de variável e nós de paridade também são reguladas. Algumas delas são *obrigatórias*, indicadas com uma linha sólida na Figura 3.3, e outras podem ou não existir (linhas tracejadas) sem afetar o projeto do mapeamento de bits proposto. Essas conexões podem ser selecionadas aleatoriamente ou seguindo alguma outra regra de projetos de códigos LDPC, e podem muito bem estar sujeitas a outras restrições, como *girth-4* ou as impostas por uma matriz de verificação de paridade já existente.

As linhas conectando nós de variável a nós de paridade são grossas, para indicar múltiplos ramos. Nota-se também que algumas dessas linhas (possivelmente de múltiplos ramos) estão direcionadas para nós específicos, enquanto outras conectam grupos de nós de variável a grupos de nós de paridade.

Considere que os sub-canais foram ordenados em ordem crescente de qualidade. Assim, os nós de variável $V_1, \ldots, V_N$ foram ordenados como segue:

$$\underbrace{V_1, \cdots, V_{|\mathcal{B}|}}_{\mathcal{B}}, V_{|\mathcal{B}|+1}, \cdots, \underbrace{V_{N-s-|\mathcal{H}|+1}, \cdots, V_{N-s}}_{\mathcal{H}}, \cdots, V_N, \tag{3.4}$$

onde s pode variar de 0 a $N - 2|\mathcal{H}| - 1$.

Deve-se notar que nós de variável "ruins" são transmitidos através de sub-canais com as piores qualidades, enquanto que os nós de variável "ajudantes" são transmitidos através dos sub-canais com as melhores qualidades. O parâmetro de deslocamento s representa um deslocamento para a esquerda (em bits) na seleção dos nós ajudantes. Isso permite que alguns dos melhores nós de variável possam ajudar outros nós intermediários.

Finalmente, deve-se notar que a restrição que proíbe dois nós de variável "ruins" de se conectarem através de um nó de paridade foi criada para evitar uma concentração de LLRs ruins no mesmo nó de paridade. Na Figura 3.3, recorreu-se a diferentes cores para mostrar essa restrição. Por outro lado, o fato de que cada nó de variável "ruim" é necessariamente conectado a pelo menos um nó de variável "ajudante" através de pelo menos um nó "*rootcheck*" implica que o nó de variável "ruim" é alimentado com boas LLRs na decodificação iterativa. Dessa

forma, os bits transmitidos através dos sub-canais com as piores qualidades são ajudados pelos bits transmitidos pelos sub-canais com as melhores qualidades.

Capítulo 4

Otimização do Mapeamento de Bits *Root-Like* através da análise de EXIT *charts*

Essencialmente, a otimização do mapeamento de bits *root-like* é baseada na otimização do parâmetro de deslocamento s. A escolha do parâmetro s tem um impacto direto na distribuição do mapeamento $m(x)$ em (3.1) no Cap. 3. Nesse sentido, um algoritmo de otimização do mapeamento de bits *root-like* através da análise de EXIT *charts* é proposto neste capítulo, para encontrar a distribuição de mapeamento ótima $m(x)$ relacionada ao parâmetro s.

4.1 EXIT *charts* para códigos LDPC

A técnica de EXIT (*extrinsic information transfer*) *charts* é uma ferramenta gráfica criada por Stephan ten Brink [13], para analisar o comportamento de convergência da decodificação iterativa de códigos concatenados. Ao tratar o decodificador LDPC como a concatenação de dois decodificadores em série, um decodificador para um código de repetição, também chamado decodificador de nós de variável (VND) e um decodificador para um código de verificação de paridade, também chamado decodificador de nó de paridade (CND), o comportamento de convergência do decodificador como um todo pode ser analisado. Nesse sentido, a técnica de EXIT *charts* é útil na avaliação de desempenho e projeto de um código LDPC, uma vez que é possível prever o desempenho do decodificador sem recorrer a longas simulações Monte Carlo.

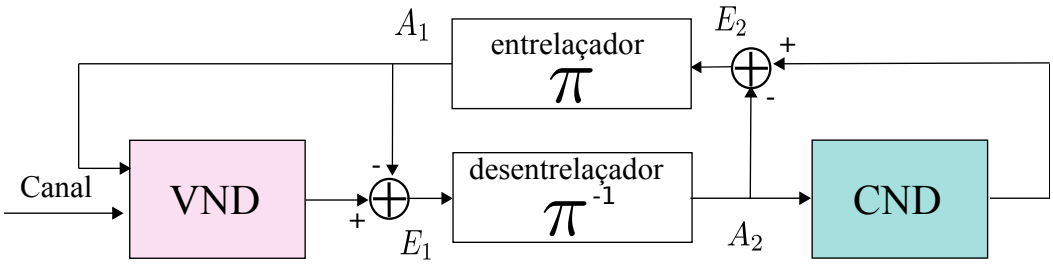

Figura 4.1: Diagrama de blocos da decodificação iterativa.

A ideia em que se baseiam os EXIT *charts* é rastrear a troca de informação mútua na decodificação iterativa. Na decodificação iterativa dos códigos LDPC, o decodificador interno VND e o decodificador externo CND, trocam informações entre si iterativamente até que se tomem decisões finais sobre os bits. A Figura 4.1 mostra o diagrama de blocos da decodificação iterativa de um código LDPC. As entradas dos decodificadores são as informações *a priori*, designadas por "A", e as saídas dos blocos são as informações extrínsecas, designadas por "E".

Uma "função de transferência" de informação pode ser gerada através da relação entre a entrada e a saída de cada decodificador VND e CND. Mais especificamente, calculam-se a informação mútua entre o bit transmitido e a informação *a priori*, a qual chama-se de I_{A}, e a informação mútua entre o bit transmitido e a informação extrínseca, a qual chama-se de I_{E}. Como a informação extrínseca de um decodificador é a informação *a priori* do outro decodificador, é possível plotar ambas as funções de transferência no mesmo gráfico, sendo a abscissa e a ordenada de um dos decodificadores invertida.

A Figura 4.2 mostra um exemplo de EXIT *chart* de um código LDPC regular ($d_v = 3, d_c = 6$) de taxa 1/2. Usando a notação em [27], a função EXIT para o decodificador VND é a curva gerada por $I_{\mathrm{E,V}} \times I_{\mathrm{A,V}}$ (curva sólida), em que $I_{\mathrm{E,V}}$ é a informação mútua entre o bit transmitido e a informação extrínseca na saída de VND, e $I_{\mathrm{A,V}}$ é a informação mútua entre o bit transmitido e a informação *a priori* na entrada de VND. A curva VND depende da SNR do canal. A função EXIT para o decodificador CND é a curva gerada por $I_{\mathrm{A,C}} \times I_{\mathrm{E,C}}$ (curva tracejada), que segue a mesma notação. Entre essas duas curvas está a trajetória de decodificação, que começa no ponto (0,0) e converge no ponto (1,1). A trajetória permite visualizar a quantidade de informação (em bits) que esta sendo trocada entre os decodificadores VND e CND.

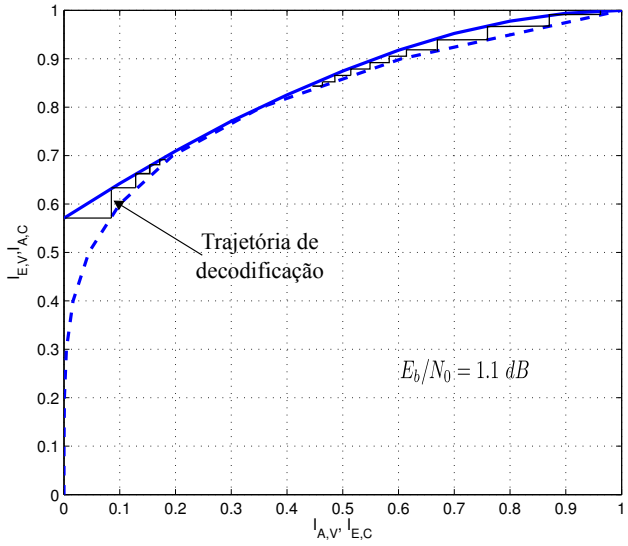

Figura 4.2: EXIT *chart* de um código LDPC regular ($d_v = 3, d_c = 6$) de taxa 1/2.

O limiar de decodificação é o valor de E_b/N_0 em que as duas curvas de transferência se tocam, impedindo a convergência. No caso da Figura 4.2, o limiar de decodificação é $E_b/N_{0_{limiar}} = 1.1\ dB$, portanto se E_b/N_0 for abaixo desse valor, as duas curvas se cruzam e não há a convergência do decodificador. No entanto, à medida que a E_b/N_0 aumenta, o "túnel" entre as duas curvas também aumenta, e quando as duas curvas estão muita afastadas significa que o valor de E_b/N_0 está longe do limite teórico (relacionado a capacidade de canal). Nesse caso, há a necessidade da otimização do par (d_v, d_c) para os códigos regulares ou da otimização das distribuições de graus ($\lambda(x)$ e $\rho(x)$) para os códigos irregulares, com o objetivo de aproximar as curvas e assim, se aproximar da capacidade do canal. Desse modo, EXIT *charts* podem ser usadas para avaliar a eficiência da decodificação e também no projeto de códigos LDPC.

As funções EXIT para os códigos LDPC regulares e irregulares são resumidamente definidas a seguir; mais detalhes podem ser encontrados em [13, 27], de onde foram retiradas as expressões que seguem.

Para os códigos LDPC regulares, considerando primeiramente a curva EXIT para o decodificador VND e seguindo o diagrama de blocos da Figura 4.1, a LLR *a priori* na entrada do decodificador, $L_{\mathrm{A}j}$, para o j-ésimo nó de paridade, pode ser modelada aplicando uma variável aleatória gaussiana n_j, com variância σ_A^2, e com média $\mu_A = \sigma_A^2/2$, em conjunto com os bits transmitidos $x_j \in \{\pm 1\}$, resultando em:

$$L_{\mathrm{A}j} = \mu_A . x_j + n_j. \tag{4.1}$$

A saída do decodificador VND para o i-ésimo nó de variável, de acordo com o algoritmo de

decodificação Soma-Produto, é definida por:

$$L_{\mathrm{E}i} = L_{\mathrm{CH}i} + \sum_{j \neq i} L_{\mathrm{A}j}, \tag{4.2}$$

em que $L_{\mathrm{CH}i}$ é a mensagem associada ao canal, a LLR do canal, e $L_{\mathrm{E}i}$ é gaussiana com variância $\sigma^2 = \sigma^2_{\mathrm{CH}} + (d_v - 1)\sigma^2_A$ (e portanto, média $\sigma^2/2$), sendo σ^2_{CH} a variância de $L_{\mathrm{CH}i}$. Para o caso BPSK (*Binary Phase Shift Keying*), tem-se $\sigma^2_{\mathrm{CH}} = 8.R.(E_b/N_0)$, em que R é a taxa do código. Para as demais modulações, esse valor pode ser obtido através de métodos numéricos, como simulações de Monte Carlo.

Para o cálculo da informação mútua entre o bit transmitido e a informação extrínseca na saída de VND, $I_{\mathrm{E,V}}$, usam-se a variável aleatória binária X e a LLR extrínseca $L_{\mathrm{E}i}$, chamada aqui de L, fazendo:

$$\begin{aligned} I(X;L) &= H(X) - H(X|L) \\ &= 1 - \int p(l \mid X = x) \log_2(1 + e^{-l}) dl. \end{aligned}$$

Como $L_{\mathrm{E}i}$ é gaussiana, substituindo $p(l \mid X = x)$ pela probabilidade condicional de l, obtém-se:

$$I(X;L) = I_{\mathrm{E,V}} = 1 - \int \frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{-1}{2\sigma^2}\left(l - \frac{\sigma^2}{2}\right)^2\right) \log_2(1 + e^{-l}) dl. \tag{4.3}$$

Considerando $I_{\mathrm{E,V}} = J(\sigma)$, a expressão acima pode ser resolvida integrando-se numericamente ou de forma aproximada usando-se as funções $J(.)$ e $J^{-1}(.)$ em [14]. Assim, $I_{\mathrm{E,V}}$ é dada por:

$$I_{\mathrm{E,V}} = J(\sigma) = J\left(\sqrt{(d_v - 1)\sigma^2_A + \sigma^2_{\mathrm{CH}}}\right). \tag{4.4}$$

Para expressar $I_{\mathrm{E,V}}$ em função de $I_{\mathrm{A,V}}$, considera-se $I_{\mathrm{A,V}} = J(\sigma_A)$, logo:

$$I_{\mathrm{E,V}} = J(\sigma) = J\left(\sqrt{(d_v - 1)[J^{-1}(I_{\mathrm{A,V}})]^2 + \sigma^2_{\mathrm{CH}}}\right). \tag{4.5}$$

A curva EXIT do decodificador CND segue o mesmo procedimento da curva do decodificador VND. A propriedade de dualidade dos códigos de paridade simples e de repetição, que se mostrou exata para o canal BEC (*binary erasure channel*) e bastante aproximada para o canal AWGN, como mostrado em [38], pode ser usada aqui para estabelecer a relação de dualidade entre $I_{\mathrm{E,V}}$ e $I_{\mathrm{E,C}}$. Assim, $I_{\mathrm{E,C}}$ é dada por:

$$\begin{aligned} I_{\mathrm{E,C}} &= 1 - I_{\mathrm{E,V}}(d_v \to d_c, I_{\mathrm{A,V}} \to 1 - I_{\mathrm{A,C}}, \sigma_{\mathrm{CH}} = 0) && (4.6) \\ &= 1 - J\left(\sqrt{(d_c - 1)[J^{-1}(1 - I_{\mathrm{A,C}})]^2}\right). && (4.7) \end{aligned}$$

Nesse caso não há informação do canal, logo $\sigma_{\mathrm{CH}} = 0$.

Para os códigos LDPC irregulares, as curvas EXIT para os decodificadores VND e CND são calculadas como médias ponderadas usando-se os pares de distribuição de graus $\lambda(x)$ e $\rho(x)$ em (2.11) e (2.12) respectivamente, resultando em:

$$I_{\mathrm{E,V}} = \sum_{i=1}^{d_v^{max}} \lambda_i I_{\mathrm{E,V}}(I_{\mathrm{A,V}}, i, \sigma^2_{\mathrm{CH}}) \tag{4.8}$$

$$= \sum_{i=1}^{d_v^{max}} \lambda_i J\left(\sqrt{(i-1)[J^{-1}(I_{\mathrm{A,V}})]^2 + \sigma^2_{\mathrm{CH}}}\right), \tag{4.9}$$

$$I_{\mathrm{E,C}} = \sum_{q=1}^{d_c^{max}} \rho_q I_{\mathrm{E,C}}(I_{\mathrm{A,C}}, q) \tag{4.10}$$

$$= \sum_{q=1}^{d_c^{max}} \rho_q \left[1 - J\left(\sqrt{(q-1)[J^{-1}(1-I_{\mathrm{A,C}})]^2}\right)\right]. \tag{4.11}$$

Apesar de a análise do limiar de decodificação utilizando EXIT *charts* não ser tão precisa quanto aquela usando *density evolution*, a abordagem do EXIT apresenta vantagens com a possibilidade de visualizar o comportamento da decodificação e conseguir bons resultados com baixa complexidade computacional [22].

4.1.1 EXIT *charts* com Mapeamento de Bits

EXIT *charts* também podem ser aplicadas com diferentes mapeamentos de bits usando a distribuição de mapeamento $m(x)$ em (3.1). Nesse caso, o comportamento da curva CND permanece inalterado devido a sua determinação pela estrutura do código. No entanto, a curva VND varia de acordo com os diferentes mapeamentos de bits, já que os nós de variável são atribuídos aos diferentes sub-canais. Portanto, tem-se uma curva VND para cada mapeamento de bits diferente e a curva geral VDN para um código irregular, em (4.8), é obtida somando-se todas as funções EXIT de acordo com $m(x)$ e ponderando as suas contribuições como

$$I_{\mathrm{E,V}}(I_{\mathrm{A,V}}, \sigma^2_{\mathrm{ch},j}, m(x)) = \sum_{j=1}^{e} \sum_{i=d_v^{\min}}^{d_v^{\max}} m_{j,i} \cdot \lambda_i \cdot I_{\mathrm{E,V}}(I_{\mathrm{A,V}}, i, \sigma^2_{\mathrm{ch},j}). \tag{4.12}$$

A equivalente variância do ruído para o nível de confiabilidade j é definida como $\sigma^2_{\mathrm{ch},j} = J^{-1}(C_{\mathrm{BICM}}(j, E_s/N_0))$, em que C_{BICM} é a capacidade ou informação mútua a nível

de bit da modulação codificada com entrelaçamento de bits (*bit interleaved coded modulation* (BICM)) [39], dado o nível de confiabilidade j, ou a variância do ruído de cada sub-canal (no caso OFDM).

EXIT *charts* podem ser usadas no projeto de códigos LDPC irregulares realizando uma otimização das distribuições de nós de variável e de paridade ($\lambda(x)$, $\rho(x)$), respectivamente, usando diferentes abordagens. Uma das abordagens é conhecida como ajuste de curva (*curve fitting*) [14, 40]. É mostrado em [40] que a capacidade de canais com apagamento pode ser abordada ao igualar as funções VND e CND, já que a área entre as curvas de transferência é proporcional a perda de taxa do sistema codificado. Então, igualar as curvas de transferência pode minimizar a perda de taxa. Outra abordagem é apenas monitorar o limiar de convergência do código. O limiar de convergência do código é definido como o desvio padrão do ruído máximo para o qual a probabilidade de erro tende a zero à medida que o número de iterações aumenta [41]. Alternativamente, o limiar de convergência é o valor da SNR, indicado aqui por SNR limiar (SNR *threshold* ($\mathrm{SNR_{TH}}$)), quando as curvas EXIT VND e CND são igualadas, com a curva EXIT VND acima da curva EXIT CND, sem intersecção. Pode-se dizer que a abordagem de ajuste de curva pode ser vista como um método indireto para otimizar para o melhor limiar de convergência.

EXIT *charts* também podem ser usadas para otimizar as distribuições de mapeamento, usando (4.12), como em [22, 23, 18]. Aqui, aplicou-se EXIT *charts* para prever a performance de um esquema de mapeamento de bits usando a abordagem do limiar de convergência, que provou ter melhores resultados de taxa de erro de bits (BER) em [41], quando usado no projeto de códigos LDPC. Prevê-se que esquemas de mapeamento de bits com SNRs limiares (SNRs *thresholds*) menores, tenham melhor capacidade de correção de erros.

4.2 Algoritmo de Otimização

A principal ideia do algoritmo de otimização do mapeamento de bits *root-like* através de EXIT *charts* é encontrar a distribuição do mapeamento de bits ótima $m(x)$ em (3.1), relacionada ao parâmetro s, considerando as restrições do mapeamento de bits *root-like* e a análise através de EXIT *charts* usando a abordagem do limiar de convergência.

O algoritmo de otimização é resumido no Algoritmo 1. Note que o Algoritmo 1 é essencialmente descrito considerando o modelo de sistema adotado. Para o algoritmo, define-se $\mathfrak{E} = \{1, \cdots e\}$ como o conjunto de sub-canais; define-se também $\mathfrak{B} \subset \mathfrak{E}$ e $\mathfrak{H} \subset \mathfrak{E}$ como o conjunto dos piores sub-canais e dos sub-canais ajudantes, respectivamente.

Algoritmo 1 Otimização do Mapeamento de Bits *Root-Like* através da análise do EXIT *charts*

Entrada: Matriz de verificação de paridade $\mathbf{H}$ do código QC-LDPC com N, K, e R associados, atendendo as restrições do mapeamento de bits *root-like*; ordem da modulação 2^m-QAM; número de sub-canais (níveis de confiabilidade) e.

Inicialização: Identificar o número de colunas na matrix $\mathbf{H}$ que satisfaçam as restrições do mapeamento de bits *root-like* para alocar nós de variável "ruins" ($\mathcal{B}$) e "ajudantes" ($\mathcal{H}$), aproveitando-se da estrutura de dupla diagonal, com grau 2, como na Figura 5.1. Determinar o número $|\mathcal{B}|$ de bits ruins e o número $|\mathcal{H}|$ de bits ajudantes como $|\mathcal{B}| = b$ (tamanho das sub-matrizes em $\mathbf{H}$) $\times$ "número de colunas ruins ($\mathcal{B}$) em $\mathbf{H}$" e como $|\mathcal{H}| = b$ (tamanho das sub-matrizes em $\mathbf{H}$) $\times$ "número de colunas ajudantes ($\mathcal{H}$) em $\mathbf{H}$".

para $j \in \mathfrak{B}$ **faça**

    Definir $m_{j,2} \leftarrow \frac{0.5}{|\mathfrak{B}|}$

    Definir $m_{j,i} \leftarrow$ Gerar aleatoriamente sujeito à restrição em (3.2).

fim para

Definir "tamanho do passo";

para $s \leftarrow 0$: tamanho do passo : $N - 2|\mathcal{H}| - 1$ **faça**

    para $j \in \mathfrak{H}$ **faça**

        Definir $m_{j,2} \leftarrow \frac{0.5}{|\mathfrak{H}|}$

        Definir $m_{j,i} \leftarrow$ Gerar aleatoriamente sujeito à restrição em (3.2).

    fim para

    para $j \notin \mathfrak{B}$ & $j \notin \mathfrak{H}$ **faça**

        Definir $m_{j,2} \leftarrow 0$

        Definir $m_{j,i} \leftarrow$ Gerar aleatoriamente sujeito à restrição em (3.2).

    fim para

    Executar EXIT *charts* (abordagem do limiar de convergência) usando o $m(x)$ gerado para encontrar o limiar de decodificação $\mathrm{SNR}_{\mathrm{TH}}$;

fim para

Saída: $m(x)$ ótimo com o menor $\mathrm{SNR}_{\mathrm{TH}}$

Seguindo o Algoritmo 1, após a inicialização, o primeiro passo é a determinação dos nós de variável "ruins" ($\mathcal{B}$) e "ajudantes" ($\mathcal{H}$) e as suas posições na matriz de verificação de paridade $\mathbf{H}$, obedecendo as restrições do mapeamento de bits *root-like* (ver Figura 5.1 para um exemplo). As matrizes de verificação de paridade QC-LDPC possuem a vantagem de uma estrutura de dupla diagonal, com grau 2, que satisfaz essas restrições. O próximo passo é selecionar os piores sub-canais (em termos da qualidade do sub-canal: capacidade do canal ou SNR) para mapear os nós de variável "ruins" ($\mathcal{B}$). Para simplificar, optou-se por definir $|\mathcal{B}|$ e $|\mathcal{H}|$ como tendo o mesmo tamanho, assim, metade das colunas de grau 2 são alocadas para os nós de variável "ruins" e a outra metade para os nós de variável "ajudantes". Nas equações dos Passos 4 e 10 do Algoritmo 1, utilizou-se da definição do termo "0.5" para evidenciar esta decisão. Para o mapeamento dos nós de variável "ajudantes" ($\mathcal{H}$), o primeiro passo é selecionar o parâmetro de deslocamento s entre as opções possíveis. Todas as opções possíveis de deslocamento podem ser avaliadas. No entanto, optou-se por variar o parâmetro de deslocamento em passos de uma quantidade determinável chamado "tamanho do passo", para fins de simulação. Assim, o mapeamento dos nós de variável ajudantes para os sub-canais ajudantes é possível de acordo com o s selecionado. O mapeamento do resto dos nós é gerado aleatoriamente. Após esses passos, é possível gerar um $m(x)$ diferente para cada opção de deslocamento s. Cada $m(x)$ é avaliado através de EXIT *charts*, usando a abordagem do limiar de convergência, para encontrar o limiar de decodificação $\mathrm{SNR}_{\mathrm{TH}}$. O $m(x)$ ótimo é determinado como aquele com o menor $\mathrm{SNR}_{\mathrm{TH}}$.

Capítulo 5

Resultados Numéricos

Este capítulo apresenta os resultados da aplicação do mapeamento de bits *root-like* e também da otimização do mapeamento através da técnica de EXIT *charts*. O G.hn e o IEEE 802.11n são os sistemas QAM codificados com LDPC utilizados para testar a performance do mapeamento de bits *root-like*. Ambos os sistemas tem características similares em termos de esquemas de correção de erros e de técnicas de transmissão. O esquema de correção de erros de ambos os sistemas especifica matrizes de verificação de paridade para códigos QC-LDPC com taxas de código 1/2, 2/3, 3/4, e 5/6 para o IEEE 802.11n, e taxas de código 1/2, 2/3, e 5/6, assim como taxas de código 16/18 e 20/21 obtidas através do puncionamento do código de taxa 5/6, para o sistema G.hn. Ambos os sistemas adotam um esquema multi-portadora baseado em OFDM como a técnica de transmissão, com um número flexível de bits em cada sub-portadora, entre 1 e 12, utilizando uma constelação QAM [6, 8]. O mapeamento de bits *root-like* é aplicado aos sistema G.hn e IEEE 802.11n considerando os modos OFDM e portadora única (*single-carrier*).

Os códigos utilizados nas simulações foram todos implementados em C/C++. As simulações foram realizadas de acordo com o diagrama de blocos da Figura 3.1. Cada bloco (codificador/decodificador QC-LDPC com as matrizes de verificação de paridade pré-definidas, mapeador/desmapeador de bits *root-like* e modulador/demodulador QAM) foi implementado em C/C++, assim como o fluxo de dados entre os blocos e a otimização do mapeamento com EXIT *charts*. O desempenho do mapeamento foi analisado através de simulações de BER.

A Figura 5.1 mostra a forma compacta de uma das matrizes de verificação de paridade do código QC-LDPC, definidas no padrão G.hn, como descrita na Seção 2.5. As matrizes de verificação de paridade definidas no padrão IEEE 802.11n tem um formato similar. A matriz da Figura 5.1 será utilizada para exemplificar o mapeamento de bits *root-like*, bem como a otimização através de EXIT *charts*.

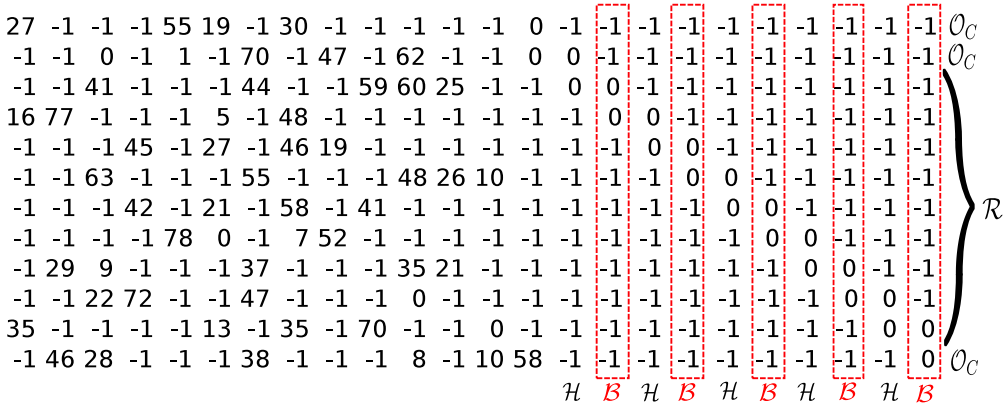

Figura 5.1: Matriz de verificação de paridade do padrão G.hn ($c = 12, t = 24, b = 80$) para $N = 1920$ bits, $K = 960$, taxa 1/2 [8].

5.1 Modelo do Canal

5.1.1 Ruído Aditivo

O modelo de canal utilizado nas simulações a seguir é o modelo de canal de ruído aditivo [10, 27]. Neste modelo, o sinal transmitido $x(t)$ é corrompido por um processo aleatório de ruído aditivo $n(t)$, resultando no sinal recebido $y(t)$:

$$y(t) = x(t) + n(t). \tag{5.1}$$

Quando $n(t)$ é um processo de ruído gaussiano branco chamamos esse modelo de canal, de canal de ruído gaussiano branco aditivo (AWGN). O sinal $n(t)$ é obtido a partir de um processo aleatório gaussiano com média zero e variância ${\sigma_n}^2 = N_0$. A quantidade N_0 indica a densidade espectral de potência do ruído unilateral. Este é o modelo de canal predominantemente usado na análise e no projeto de sistemas de comunicação e será, portanto, um dos modelos de canal utilizado para analisar o comportamento do mapeamento *root-like*.

5.1.2 Desvanecimento *Rayleigh*

A atenuação do canal pode ser incorporada ao modelo de canal de ruído aditivo descrito acima. Quando o sinal é submetido a atenuações na transmissão pelo canal, o sinal recebido resulta em [10]:

$$y(t) = hx(t) + n(t), \tag{5.2}$$

em que h é o fator de atenuação.

Quando modela-se um canal sem fio, como é o caso do sistema IEEE802.11n, a atenuação pode ser decorrente do desvanecimento por multi-percurso [42]. Em sistemas de comunicação sem fio, um sinal pode passar do transmissor para o receptor através de múltiplos percursos reflexivos, este fenômeno é definido como propagação multi-percurso. O seu efeito pode causar flutuações na amplitude, fase e ângulo de chegada do sinal recebido, dando origem a terminologia desvanecimento por multi-percurso. Desvanecimentos por multi-percurso em sistemas de comunicação sem fio são geralmente modelados por distribuições *Rayleigh* e *Rician* [42].

O canal utilizado neste trabalho é modelado como em [22], como um canal *fully-interleaved* com desvanecimento *Rayleigh* e com detecção coerente, assumindo informações perfeitas sobre o estado do canal no receptor. Em um canal *fully-interleaved* com desvanecimento *Rayleigh* assume-se que cada símbolo transmitido é afetado independentemente pelo desvanecimento *Rayleigh*, ou seja, os componentes da palavra-código transmitida são submetidos a desvanecimentos independentes. Este modelo de canal difere do canal com desvanecimento Rayleigh em blocos, no qual assume-se que o desvanecimento seja aproximadamente constante através do tamanho do bloco [42, 10]. A saída do canal para este modelo é definida como na Eq. 5.2, no qual o fator de atenuação é definido como o coeficiente de desvanecimento h, sendo obtido a partir de um processo aleatório gaussiano complexo com média zero e variância ${\sigma_h}^2 = 1$, fazendo com que a magnitude $|h|$ tenha distribuição *Rayleigh*. Este modelo de canal é definido pela função de densidade de probabilidade [10]

$$p(y/x) = \frac{1}{\pi N_0} e^{\frac{-|y-hx|^2}{N_0}}. \tag{5.3}$$

Observe que ao ajustar $h = 1$ para todos os símbolos, o canal se reduz a um canal de ruído gaussiano branco aditivos (AWGN) simples.

5.2 Mapeamento de Bits *Root-Like* em OFDM Aplicado ao G.hn

Em um sistema OFDM [17], um símbolo de modulação 2^m-QAM ($m > 2$), com m bits, é alocado para cada sub-portadora, cada uma tendo uma SNR média diferente. Ao associar cada sub-portadora a um sub-canal nesse esquema, tem-se m bits codificados (um símbolo QAM) transmitidos através do mesmo sub-canal.

Considerando a matriz de verificação de paridade **H** na Figura 5.1 com tamanho de palavra-código de $N = 1920$ bits e taxa de código de 1/2, se cada sub-canal for alocado com um símbolo 256-QAM ($m = 8$ bits), então tem-se $1920/8 = 240$ sub-canais. Uma vez que o número de colunas, t, em **H** é igual a 24, e cada sub-matriz de **H** tem tamanho 80×80, então 80 bits podem ser mapeados em dez símbolos 256-QAM em cada coluna de **H**.

As colunas ruins ($\mathcal{B}$) e as colunas ajudantes ($\mathcal{H}$) associadas com os bits ruins e os bits ajudantes são alocadas na matriz **H** na Figura 5.1 seguindo as restrições do mapeamento de bits *root-like* e aproveitando-se da dupla diagonal em **H**, parte do padrão do G.hn. Ao inspecionar a matriz **H** na Figura 5.1, pode-se facilmente descobrir que as 5 colunas indicadas em vermelho na parte direita da matriz, que apresenta a estrutura de dupla diagonal, satisfazem a Restrição 1) do mapeamento de bits *root-like*, descrita na Sessão 3.2. Ou seja, essas colunas (ruins) indicadas com $\mathcal{B}$ na Figura 5.1 não apresentam entradas diferentes de zero nas mesmas posições, garantindo que elas não se conectem. Uma vez que as sub-matrizes em **H** são matrizes 80 por 80, o número de bits (nós de variável) ruins pode ser definido até $80 \times 5 = 400$ bits. Optou-se por definir $|\mathcal{B}| = 400$ bits. Para simplificar, optou-se por definir $|\mathcal{B}|$ e $|\mathcal{H}|$ como tendo o mesmo tamanho. Agora, com 8 bits por sub-portadora, deve-se selecionar as $400/8 = 50$ piores sub-portadoras para alocar esses bits ruins.

Nota-se que as 5 colunas da matriz **H** na Figura 5.1 associadas com os nós ajudantes são indicadas com $\mathcal{H}$. Além disso, entre os $c.b = 12 \times 80 = 960$ nós de paridade, tem-se $9 \times 80 = 720$ nós *rootcheck*. Estes correspondem às 9 linhas de **H** marcadas com $\mathcal{R}$ na Figura 5.1. As restantes 3 linhas (marcadas com $\mathcal{O}_C$) são para os 240 nós de paridade classificados como "todos os outros".

Os primeiros resultados do mapeamento de bits *root-like* foram publicados em [25], assumindo a mesma configuração de nós ruins e nós ajudantes adotada aqui. No entanto, os primeiros resultados consideraram um mapeamento mais intuitivo, com a escolha do valor do parâmetro de deslocamento s, por exemplo, sendo de certa forma aleatória, mas já apresentando bons resultados e abrindo espaço para futuras otimizações. Em [25], optou-se por pular as melhores 70 sub-portadoras, e atribuir a partir da sub-portadora 71 até a 120 aos bits ajudantes. Isso implica que o parâmetro deslocamento foi definido para $s = 70 \times 8 = 560$ bits, ou seja, deixaram-se os 560 melhores nós de variável livres para ajudar outros nós intermediários, enquanto que os próximos 400 bits correspondem aos nós de variável ajudantes. Nota-se que aproveitou-se da dupla diagonal em **H**, já parte do padrão G.hn, para alocar nós de variável ruins e ajudantes. No entanto, uma vez que esses nós têm baixo grau 2, deslocou-se os nós ajudantes ligeiramente para longe dos melhores bits, de modo que a alta confiabilidade dos melhores nós de variável não seja desperdiçada, caracterizando a primeira escolha do s intuitivamente.

Assim, seguindo a definição do parâmetro de deslocamento s na Sessão 3.2.2, como um parâmetro a ser otimizado, executou-se o algoritmo de otimização com EXIT *charts* descrito no Algoritmo 1, para testar o efeito das diferentes opções de deslocamento. A Figura 5.2 mostra as SNRs limiares para diferentes valores de deslocamento em bits, obtidos com o algoritmo de otimização, com a linha em vermelho especificando a SNR limiar para o caso do G.hn original, onde a ordem original dos bits é mantida (ou seja, sem mapeamento de bits). Optou-se por variar o parâmetro de deslocamento, em passos com "tamanho de passo" igual a 10 sub-portadoras (80 bits).

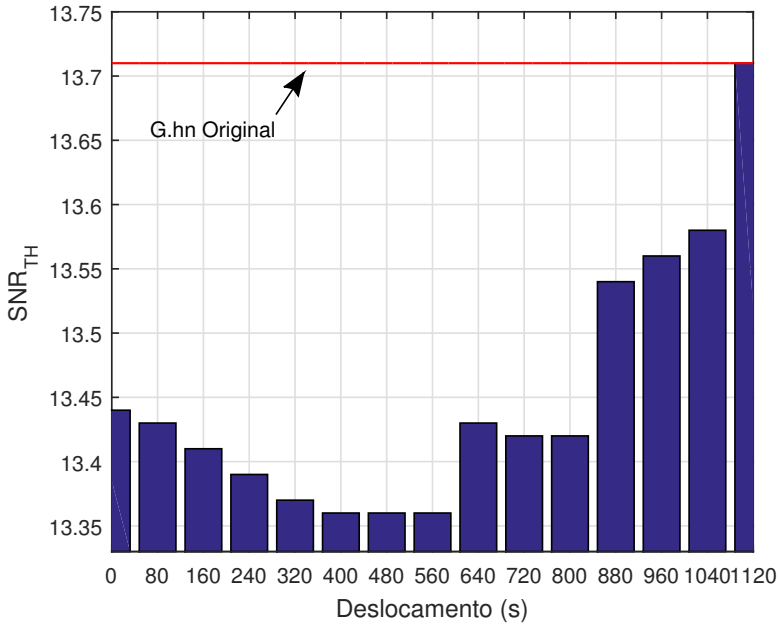

Figura 5.2: SNRs limiares de diferentes deslocamentos de bits (em bits) obtidos através de EXIT *charts* para a matriz do G.hn com modo de transmissão OFDM, sobre o canal AWGN, com $N = 1920$ bits, taxa 1/2 e 256-QAM.

Nota-se que o deslocamento $s = 1120$ bits é o último valor considerado na Figura 5.2, uma vez que deslocamentos maiores implicariam na atribuição de bits ruins como bits ajudantes, o que seria sem sentido. Deve-se notar que as melhores SNRs limiares são obtidas com s igual a 400, 480 ou 560 bits, o último sendo exatamente o usado em [25]. Portanto, uma otimização adicional não foi necessária nesse caso.

A Figura 5.3 mostra a comparação da BER para diferentes valores de deslocamento. Nota-se que os deslocamentos $s = 400$, 480, e 560 bits produzem resultados similares de BER, que são melhores que os obtidos com $s = 640$ bits e com o G.hn original, por exemplo. No entanto, também é importante notar que uma má escolha do valor de deslocamento pode produzir um desempenho pior, como no caso do $s = 1120$. Portanto, a abordagem do limiar

de convergência de EXIT *charts* considerado no algoritmo de otimização é consistente com os resultados da performance de BER. Concluiu-se também que a ideia inicial, introduzida em [25], de pular alguns dos melhores sub-canais ao selecionar os nós ajudantes provou ser eficiente.

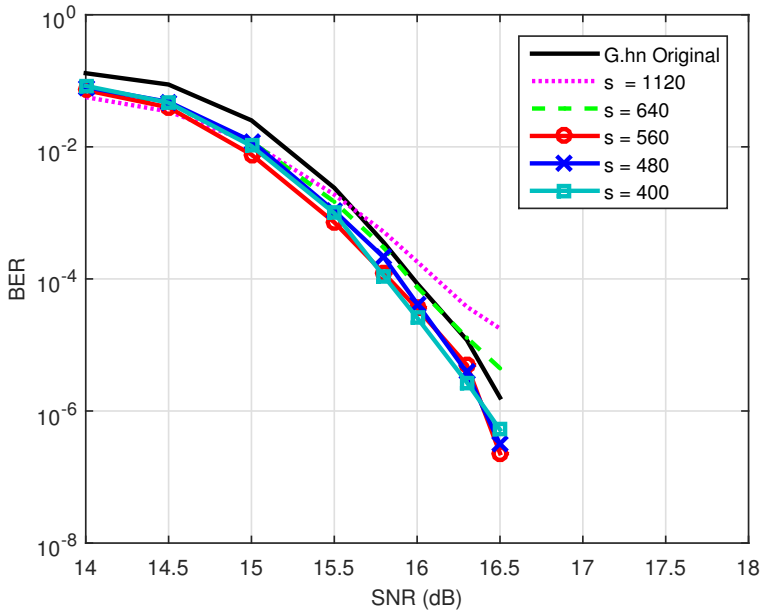

Figura 5.3: BER da matriz G.hn com transmissão OFDM, para o canal AWGN, considerando diferentes valores de deslocamento (s), com $N = 1920$ bits, taxa 1/2, e 256-QAM e 240 sub-portadoras. A SNR mostrada é a SNR média entre todos os sub-canais.

O canal OFDM utilizado na simulação é mostrado na Figura 5.4, contendo a SNR média por cada sub-canal, considerando um total de 240 sub-canais. O canal OFDM utilizado foi gerado considerando valores de SNR variando entre aproximadamente 18 e 31 dB, caracterizando um típico canal com fio (linha telefônica, por exemplo), sendo esse um dos tipos de canais para transmissão incluídos no padrão G.hn.

A Figura 5.5 mostra a comparação da BER entre o sistema original, sem mapeamento de bits, no qual os bits são transmitidos através dos sub-canais de forma sequencial, e o sistema com o mapeamento de bits *root-like* para quatro cenários diferentes para o caso OFDM. O algoritmo de decodificação soma-produto foi usado com número máximo de iterações igual a 50. Cada cenário é indicado por uma combinação de taxa de código, esquema de modulação, tamanho da palavra-código (N) em bits, e o número de sub-portadoras e é listado com uma linha na Tabela 5.1. O número correspondente de bits ruins ($|\mathcal{B}|$) e de bits ajudantes (lembre-se que $|\mathcal{B}| = |\mathcal{H}|$), o tamanho do passo e o parâmetro de deslocamento ótimo (s) associado com o mapeamento de bits *root-like* para cada cenário são listados nas últimas três colunas da tabela.

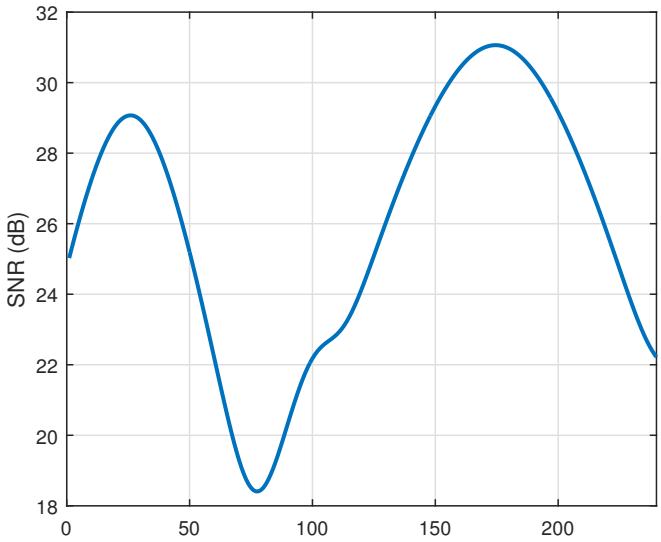

Figura 5.4: Modelo de Canal OFDM especificando a SNR média para cada sub-canal, considerando 240 sub-canais e SNRs variando entre 18 e 31dB.

Pode-se observar, na Figura 5.5, que ganhos de codificação foram obtidos em todos os quatro cenários. Mais especificamente, para o 64-QAM, o mapeamento de bits *root-like* pode fornecer um ganho de codificação de aproximadamente 0.2 dB para a taxa 2/3 e de 0.15 dB para a taxa 5/6, em uma BER de 10^{-5}. Para o 256-QAM, o ganho de codificação alcançado foi de aproximadamente 0.15 dB para a taxa 1/2 e para o 1024-QAM, o ganho de codificação foi de aproximadamente 0.18 dB.

Tabela 5.1: Parâmetros associados com os quatro cenários simulados na Figura 5.5.

Taxa	Modulação	N	# Sub-portadoras	$\lvert\mathcal{B}\rvert$	Tamanho do passo	s
1/2	256-QAM	1920	240	400	80	560
2/3	64-QAM	1440	240	180	60	540
2/3	1024-QAM	1440	144	180	60	600
5/6	64-QAM	5184	864	216	216	2160

As demais matrizes de verificação de paridade utilizadas nos cenários da Figura 5.5 podem ser encontradas no Apêndice A, e as distribuições de graus de nós de variável $\lambda(x)$ e de nós de paridade $\rho(x)$ para as matrizes de verificação de paridade utilizadas podem ser encontradas no Apêndice B.

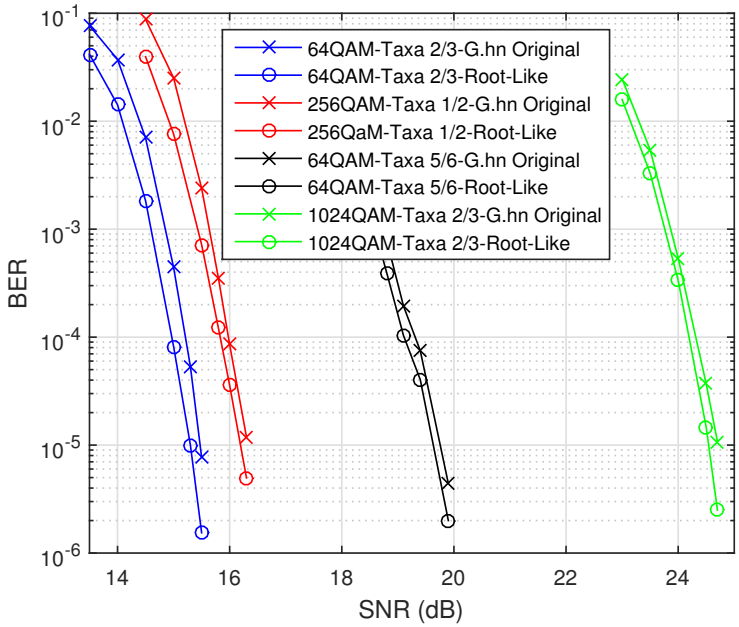

Figura 5.5: BER do sistema original vs. sistema com o mapeamento de bits *root-like* para as matrizes do G.hn, considerando o modo de transmissão OFDM, através do canal AWGN, com $N = 1440$ bits, taxa 2/3, 64-QAM e 1024-QAM, $N = 1920$ bits, taxa 1/2 e 256-QAM, e $N = 1152$ bits, taxa 5/6 e 64-QAM. A SNR mostrada é a SNR média entre todos os sub-canais.

5.3 Mapeamento de Bits *Root-Like* para o Canal AWGN de Portadora Única Aplicado ao G.hn

O esquema de mapeamento de bits *root-like* também foi aplicado ao sistema G.hn e avaliado através da análise de EXIT *charts* quando o modo OFDM é substituído pelo canal de portadora única AWGN. Nesse caso, todos os símbolos QAM são transmitidos e recebidos com a mesma SNR média do símbolo. No entanto, como podemos observar em [21], o canal AWGN com entrada sendo uma constelação QAM com 2^m símbolos com rotulamento *Gray* pode ser decomposto em m sub-canais AWGN simétricos com entrada binária e diferentes capacidades de canal. A informação mútua entre o κ-ésimo bit e o sinal recebido foi calculada em [21] para uma faixa de SNRs, em que $\kappa = 1, \ldots, m$. A Figura 5.6, replicada de [21], mostra a informação mútua considerando a modulação 256-QAM, para valores de SNR até 25 dB.

Nota-se pela Figura 5.6 que os bits 1 e 5 são os bits com menor capacidade, seguidos pelos bits 2 e 6, 3 e 7, e 4 e 8, o último par com a maior capacidade. Tem-se, assim, um total de 4 sub-canais nesse cenário, como mostra a Tabela 5.2.

Consequentemente, o mapeamento *root-like* considerou um total de 4 sub-canais, ou

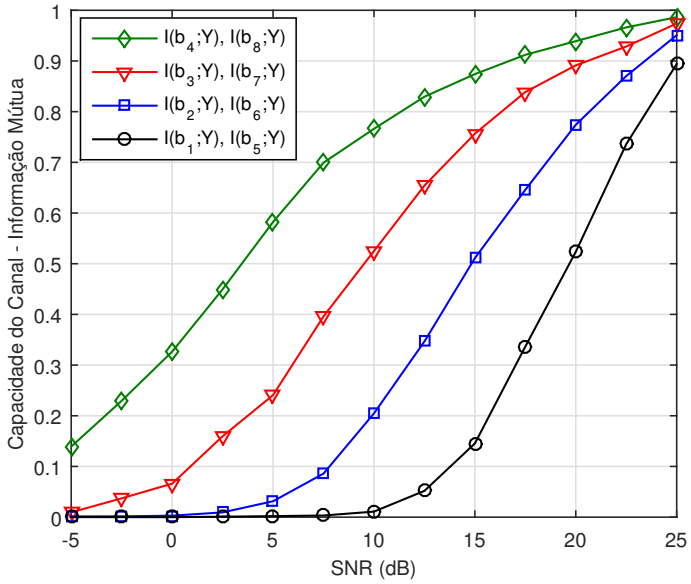

Figura 5.6: Informação mútua dos sub-canais do 256-QAM no canal AWGN, replicada de [21].

Tabela 5.2: Capacidade dos sub-canais do 256-QAM (Rotulamento *Gray* do G.hn) no canal AWGN.

j	**Par de bits**	**Capacidade**
1	1 e 5	menor capacidade
2	2 e 6	↓
3	3 e 7	↓
4	4 e 8	maior capacidade

seja, 4 níveis de confiabilidade (indexados pela variável j). Nos primeiros resultados publicados em [25], os nós de variável ruins foram associados com os bits 1 e 5 de menor capacidade dos símbolos 256-QAM (menos confiável, $j = 1$), e os nós de variável ajudantes foram associados com os bits 3 e 7 dos símbolos 256-QAM ($j = 3$), ou seja, o par com a segunda maior capacidade, deixando livres os bits com maior capacidade 4 e 8 para ajudar outros nós intermediários. Os nós de variável ruins e ajudantes são posicionados na matriz $\mathbf{H}$ da mesma forma como descrito no caso OFDM (ver Figura A.3 no Apêndice A para maiores detalhes). Como o comprimento da palavra-código LDPC nesse sistema é $N = 6480$ bits, cada sub-canal é alocado com $N/e = 6480/4 = 1620$ bits. Assim, os valores apropriados de deslocamento são 0, 1620, e 3240 bits. Considere $|\mathcal{B}| = |\mathcal{H}| = 270 \times 3 = 810$ bits (o tamanho das sub-matrizes em $\mathbf{H}$ $\times$ o número de colunas ajudantes ou ruins). Em [25], o deslocamento escolhido foi $s = 1620$ bits.

Nota-se que nesse caso, tem-se 3 opções disponíveis para mapear os nós ajudantes, considerando que optou-se por continuar associando os nós de variável ruins com os bits tendo a menor capacidade, ou seja, bits 1 e 5. Essas 3 opções podem ser avaliadas com a ajuda do algoritmo de otimização usando EXIT *charts*. A Figura 5.7 mostra a análise de EXIT *charts* utilizando a abordagem do limiar de convergência, com as curvas VND para o G.hn original, ou seja, sem mapeamento de bits e as curvas VND para os nós de variável ajudantes mapeados para os bits 4 e 8 (correspondendo ao $s = 0$), 3 e 7 (correspondendo ao $s = 1620$) e 2 e 6 (correspondendo ao $s = 3240$), com a as suas respectivas SNRs limiares, considerando os nós de variável ruins mapeados para os bits 1 e 5 (sub-canais menos confiáveis).

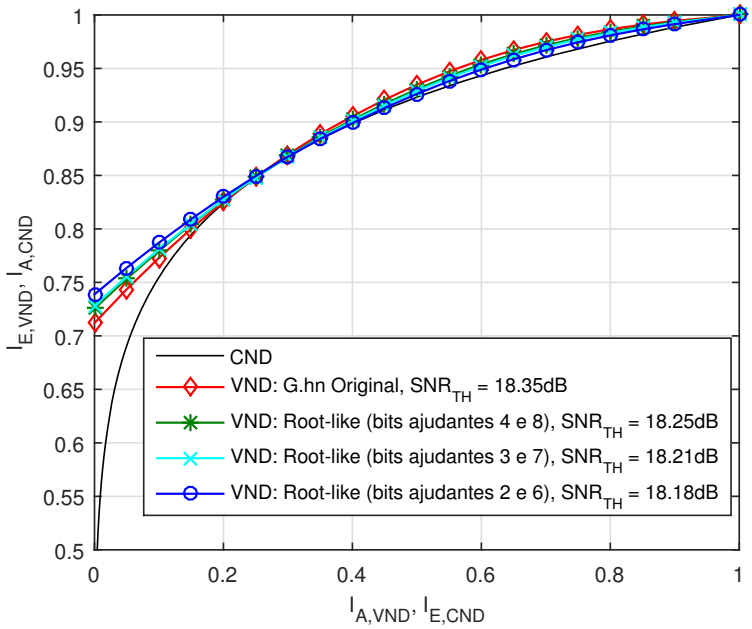

Figura 5.7: EXIT *charts* para o código LDPC com $N = 6480$, taxa 2/3 e 256-QAM como definido no G.hn. As funções VND são mostradas para o caso G.hn original vs. o caso com mapeamento *root-like* considerando os bits ajudantes mapeados para diferentes pares de bits (deslocamentos diferentes), e os seus respectivos SNR_{TH}.

Ao examinar os valores da SNR limiar, pode-se notar que o menor limiar é obtido ao mapear os nós de variável ajudantes para os bits 2 e 6 ($j = 2, s = 3240$ bits). Esse resultando é também verificado com as simulações de BER na Figura 5.8, mostrando que alguns ganhos são obtidos ao mapear os nós de variável ajudantes ao bits 2 e 6. Curiosamente, nesse caso, a otimização aqui produziu um desempenho melhorado.

A Tabela 5.3 mostra a distribuição do mapeamento de bits *root-like* $m(x)$ ótima para o canal de portadora única AWGN considerando 256-QAM com taxa 2/3 como resultado do algoritmo de otimização. Note que, na Tabela 5.3, seguindo os passos 4 e 10 do Algoritmo 1,

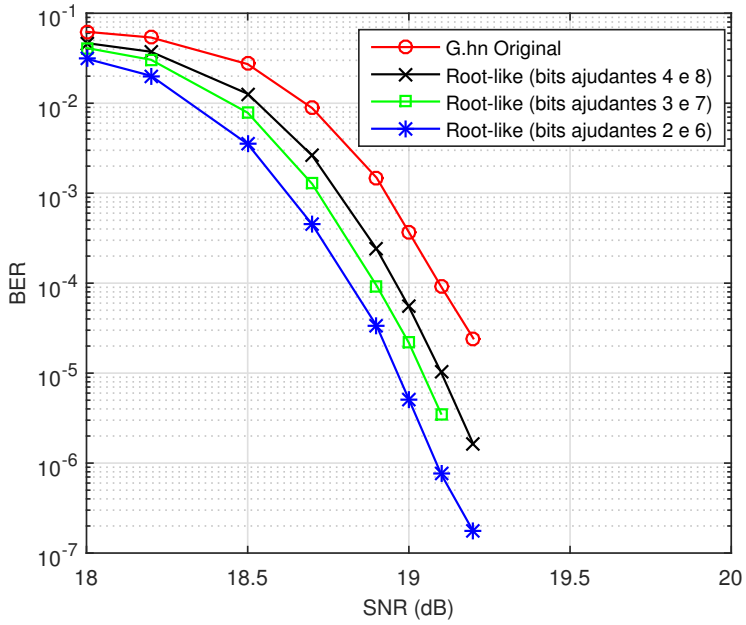

Figura 5.8: BER do sistema original vs. o sistema com mapeamento *root-like* para a matriz do G.hn com o canal de portadora única AWGN, com $N = 6480$, taxa 2/3 e 256-QAM, considerando os bits ajudantes mapeados para os diferentes pares de bits.

as restrições impostas pelo mapeamento *root-like* levaram a mapear 50% dos nós de variável com grau 2 (nós de variável ruins) ao nível de confiabilidade $j = 1$, e 50% dos nós de variável com grau 2 (nós de variável ajudantes) ao nível de confiabilidade $j = 2$, representado por $m_{1,i} = 0.5$ e $m_{2,i} = 0.5$, respectivamente. Além disso, seguindo o Algoritmo 1, o mapeamento do resto dos nós é de certa forma aleatório, não seguindo nenhuma regra específica.

Tabela 5.3: Distribuição do mapeamento de bits *root-like* para a matriz do G.hn ($N = 6480$, $R = 2/3$) com canal de portadora única AWGN e 256-QAM [8].

$i \rightarrow$	2	3	4
$m_{1,i}$	0.5000	1.0000	0.0625
$m_{2,i}$	0.5000	0.0000	0.1875
$m_{3,i}$	0.0000	0.0000	0.3750
$m_{4,i}$	0.0000	0.0000	0.3750

A Figura 5.9 mostra a comparação da BER entre o sistema original, sem mapeamento de bits, e o sistema com o mapeamento de bits *root-like* para o caso portadora única AWGN para diferentes taxas de código e ordens de modulação. As simulações foram realizadas para as taxas de código 1/2 e 2/3, com modulação 256-QAM e 64-QAM. O algoritmo de decodificação

soma-produto foi usado com número máximo de iterações igual a 50. Como pode ser visto, o mapeamento *root-like* melhorou o desempenho geral e ganhos de codificação foram obtidos em todos os casos. Mais especificamente, para o 64-QAM, o mapeamento de bits forneceu um ganho de codificação de aproximadamente 0.2 dB para a taxa de 1/2 e de 0.12 dB para a taxa de 2/3, em uma BER de 10^{-5}. Para o 256-QAM, o ganho de codificação obtido foi de 0.1 dB para a taxa de 1/2 e de 0.2 dB para a taxa de 2/3.

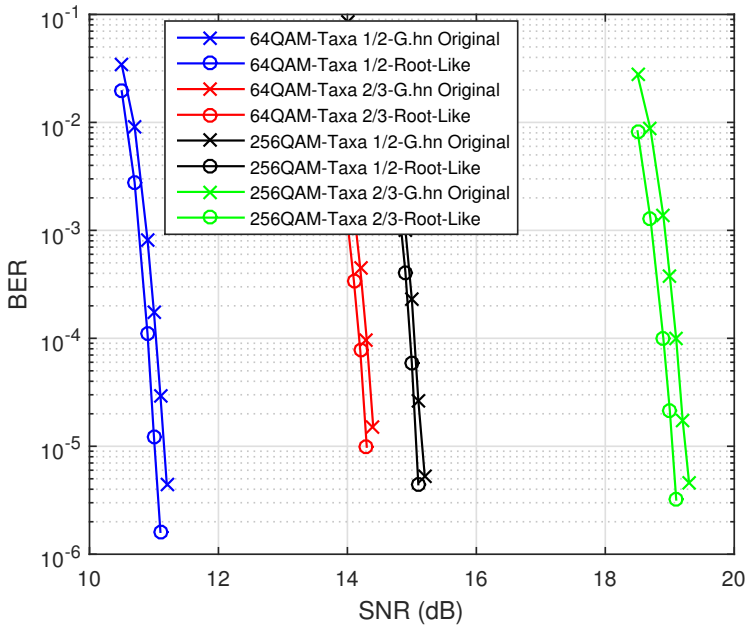

Figura 5.9: BER do sistema original vs. o sistema com o mapeamento de bits *root-like* para as matrizes do G.hn, considerando o canal de portadora única AWGN com $N = 8640$ bits, taxa 1/2, 64-QAM e 256-QAM, e com $N = 6480$ bits, taxa 2/3, 64-QAM e 256-QAM.

No caso dos resultados com o 64-QAM na Figura 5.9, o mapeamento dos bits segue a mesma regra do 256-QAM, no entanto contando com apenas 3 sub-canais, já que $m = 6$, como pode ser visto na Tabela 5.4. Os bits 1 e 4, de menor capacidade do símbolo 64-QAM, são associados aos nós de variável ruins. Restando duas opções de mapeamento dos bits ajudantes: os bits 3 e 6 de maior capacidade e os bits 2 e 5 com a segunda maior capacidade. Seguindo o mesmo resultado do caso do 256-QAM, e do algoritmo de otimização com EXIT *chart*, a melhor opção foi associar os bits 2 e 5, com a segunda maior capacidade, aos nós de variável ajudantes.

As demais matrizes de verificação de paridade, as distribuições de graus dos nós de variável $\lambda(x)$ e dos nós de paridade $\rho(x)$ e as distribuições de mapeamento de bits *root-like* $m(x)$ ótimas utilizadas nos resultados da Figura 5.9 podem ser encontradas nos Apêndices A, B e C, respectivamente.

Tabela 5.4: Capacidade dos sub-canais do 64-QAM (Rotulamento *Gray* do G.hn) no canal AWGN.

j	**Par de bits**	**Capacidade**
1	1 e 4	menor capacidade
2	2 e 5	$\downarrow$
3	3 e 6	maior capacidade

5.4 Mapeamento de Bits *Root-Like* para o Canal AWGN de Portadora Única e com Desvanecimento *Rayleigh* Aplicado ao IEEE 802.11n

O mapeamento de bits *root-like* também foi aplicado ao sistema IEEE 802.11 para o canal AWGN de portadora única e para o canal com desvanecimento *Rayleigh*.

A Figura 5.10 mostra a comparação da BER entre o sistema original, sem mapeamento, e o sistema com o mapeamento *root-like*, ambos através dos canais AWGN e com desvanecimento *Rayleigh*, e o mapeamento otimizado obtido em [22] apenas através do canal AWGN, uma vez que foi o único resultado possível de replicar. Nota-se que o mapeamento de bits *root-like* e o mapeamento otimizado em [22] produzem ganhos de codificação semelhantes através do canal AWGN. O mapeamento otimizado proposto em [22] é baseado em um algoritmo usando EXIT *charts* para obter uma boa distribuição do mapeamento $m(x)$ através da minimização da área entre as curvas VND e CND, que é obtida por uma busca exaustiva através de todas as possíveis distribuições de mapeamento.

A vantagem do mapeamento de bits *root-like*, em relação ao mapeamento proposto em [22], é que é possível obter os mesmo ganhos de código obtidos em [22], no entanto, com uma simples otimização, uma vez que é apenas necessário encontrar um deslocamento ótimo entre 3 possíveis valores, levando a um espaço de buscas reduzido. Além disso, como resultados já foram obtidos na Sessão 5.3 para o caso do G.hn, os resultados podem ser estendidos para o caso do IEEE 802.11n, e a opção de deslocamento s encontrada pelo algorítimo de otimização permanece a mesma.

Observe que, como já mencionado na Sessão 3.1, o rotulamento *Gray* no IEEE 802.11n é invertido. Dessa forma, o mapeamento de bits *root-like* é aplicado ao IEEE 8002.11n seguindo as restrições especificadas, no entanto, mapeando os nós de variável ruins para os bits no nível de confiabilidade $j = 4$, aqueles com a menor capacidade e mapeando os bits ajudantes

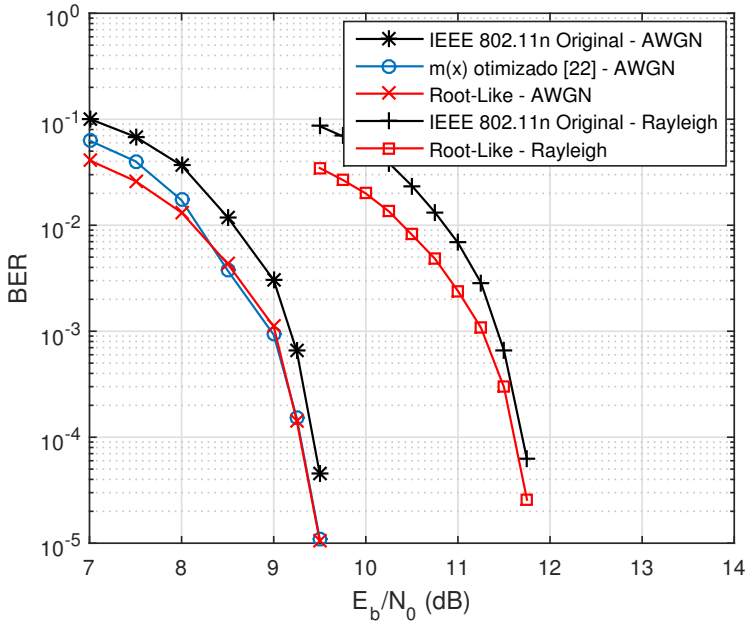

Figura 5.10: BER do sistema original vs. o sistema com mapeamento de bits *root-like* para a matriz do IEEE 802.11n com canais de portadora única AWGN e com desvanecimento *Rayleigh* vs. o sistema com o mapeamento de bits otimizado obitido em [22] para a matriz do IEEE 802.11n com canal de portadora única AWGN, com $N = 1944$, taxa 1/2 e 256-QAM.

para o nível de confiabilidade $j = 3$, aqueles com a segunda menor capacidade, como mostra a Tabela 5.5 e seguindo os resultados do algoritmo de otimização usando EXIT *charts* na Sessão 5.3.

Tabela 5.5: Capacidade dos sub-canais do 256-QAM (Rotulamento *Gray* do IEEE 802.11n).

j	**Par de bits**	**Capacidade**
1	1 e 5	maior capacidade
2	2 e 6	↑
3	3 e 7	↑
4	4 e 8	menor capacidade

A Tabela 5.6 mostra a distribuição $m(x)$ do mapeamento de bits *root-like* através do canal AWGN aplicado ao IEEE 802.11n. Note que, na Tabela 5.6, os nós de variável ruins e ajudantes não são mapeados com 0.5 na coluna de grau 2, como nos resultados da Tabela 5.3. Isto ocorre por que a matriz de verificação de paridade do IEEE 802.11n, nesse caso, tem uma coluna a mais com grau 2, como mostra a coluna em azul na Figura 5.11, não classificada como coluna ruim ou ajudante, mas com nós mapeados para o nível de confiabilidade $j = 4$,

seguindo o mapeamento aleatório do restante do nós, não seguindo nenhuma regra específica.

Tabela 5.6: Distribuição do Mapeamento *Root-like* com rotulamento *Gray* 256-QAM com transmissão através do canal AWGN e código LDPC com $N = 1944$ e $R = 1/2$ [6].

$i \rightarrow$	2	3	4	11
$m_{1,i}$	0.0000	0.3333	1.0000	0.6667
$m_{2,i}$	0.0000	0.5556	0.0000	0.3333
$m_{3,i}$	0.4545	0.1111	0.0000	0.0000
$m_{4,i}$	0.5455	0.0000	0.0000	0.0000

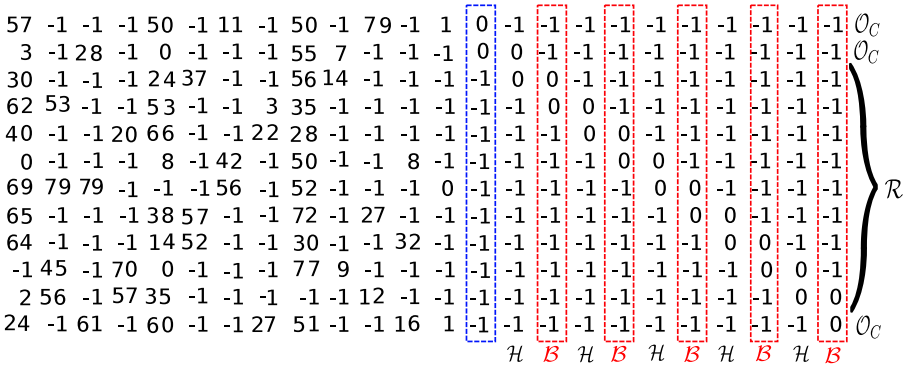

Figura 5.11: Matriz de verificação de paridade do IEEE 802.11n ($c = 12, t = 24, b = 81$) para $N = 1944$ bits, $K = 972$, taxa 1/2 [6].

5.5 Regra Prática para o Mapeamento de Bits *Root-Like*

É importante notar que encontrar a melhor distribuição do mapeamento de bits através da abordagem do *root-like* é muito menos complexa para o canal de portadora única AWGN do que para a transmissão OFDM, uma vez que, no primeiro, o número de níveis de confiabilidade é relativamente pequeno. Por exemplo, quando a constelação do 256-QAM é adotada, existem apenas 3 maneiras possíveis de selecionar nós ajudantes. Por outro lado, no caso OFDM, há um número maior de níveis de confiabilidade (da ordem do número de sub-portadoras) e vários valores de deslocamento possíveis.

No entanto, a partir da análise de EXIT *charts*, bem como das simulações de BER de todos os sistemas investigados, a seleção ótima de nós ruins e ajudantes no mapeamento

root-like quase seguiu um padrão que permite definir uma simples **regra prática**: atribua os $|\mathcal{B}|$ nós de variável ruins para os $|\mathcal{B}|$ bits menos significantes, e atribua os $|\mathcal{B}|$ nós de variável ajudantes de acordo com (3.4), com o parâmetro de deslocamento s definido em torno de $N/2 - |\mathcal{B}|$.

Baseado nesta observação, o Passo 7 ao Passo 18 no Algoritmo 1 descrito na Sessão 4.2 pode ser substituído pela regra prática, quando necessário.

5.6 O Impacto do Nível de Desequilíbrio de Confiabilidade através dos Sub-canais na Performance do Mapeamento de Bits *Root-Like*

Em sistemas com um número maior de sub-canais, como os sistemas OFDM, por exemplo, também é importante notar que o desequilíbrio entre as piores e as melhores qualidades dos sub-canais tem um impacto importante no desempenho do mapeamento de bits *root-like*. Se considerarmos e sub-canais organizados em ordem crescente com base nas qualidades do canal (SNRs) de cada sub-canal, então é possível obter uma curva que descreva a melhora gradual dessas qualidades do canal. Observou-se que o gradiente desta curva afeta significativamente o desempenho do mapeamento de bits *root-like*. Por exemplo, a Figura 5.12 mostra três curvas com as SNRs dos sub-canais variando linearmente de um valor G_1 para um valor G_2, com o gradiente da curva proporcional à diferença $G_2 - G_1$. À medida que $G_2 - G_1$ aumenta, observa-se um maior desequilíbrio entre os piores e melhores sub-canais. Optou-se por avaliar a performance do mapeamento de bits *root-like* quando três valores de gradiente ($G_2 - G_1 = 10$dB, 25dB e 35dB) são aplicados, considerando 240 sub-canais com SNRs geradas aleatoriamente, organizadas em ordem crescente.

O G.hn foi o sistema escolhido para avaliar este cenário, considerando o canal AWGN, com $N = 1920$ bits, taxa 1/2 e 256-QAM. A Figura 5.13 mostra os resultados de BER do mapeamento de bits *root-like* e do sistema G.hn original para diferentes valores de gradiente $G_2 - G_1$. A distribuição do mapeamento de bit *root-like* é aquela definida na Sessão 5.2, com $s = 560$ bits escolhidos como o deslocamento ótimo para o sistema G.hn com OFDM. Pode-se observar que, à medida que $G_2 - G_1$ aumenta, os benefícios em termos de melhorias no desempenho obtidos com o mapeamento de bits *root-like* sobre o o G.hn original tornam-se mais proeminentes, com ganhos de código alcançando quase 3 dB, no caso do $G_2 - G_1 = 35$ dB. Esse resultado enfatiza a importância da ajuda que os sub-canais "ruins" podem receber dos sub-canais "bons" para melhorar o desempenho do mapeamento de bits. Quanto maior o

desequilíbrio entre as SNRs dos piores sub-canais e dos sub-canais ajudantes, maior é a ajuda.

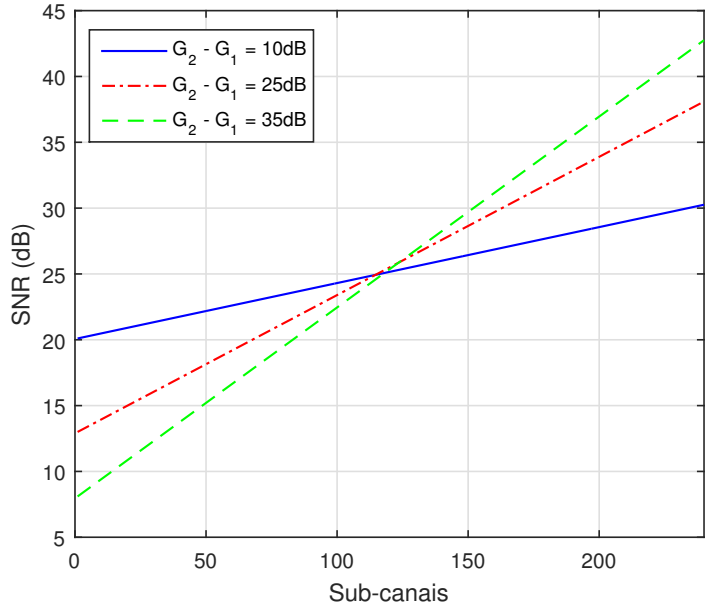

Figura 5.12: SNRs médias dos sub-canais variando linearmente de valores definidos G_1 a G_2, de diferentes valores de gradiente $G_2 - G_1$.

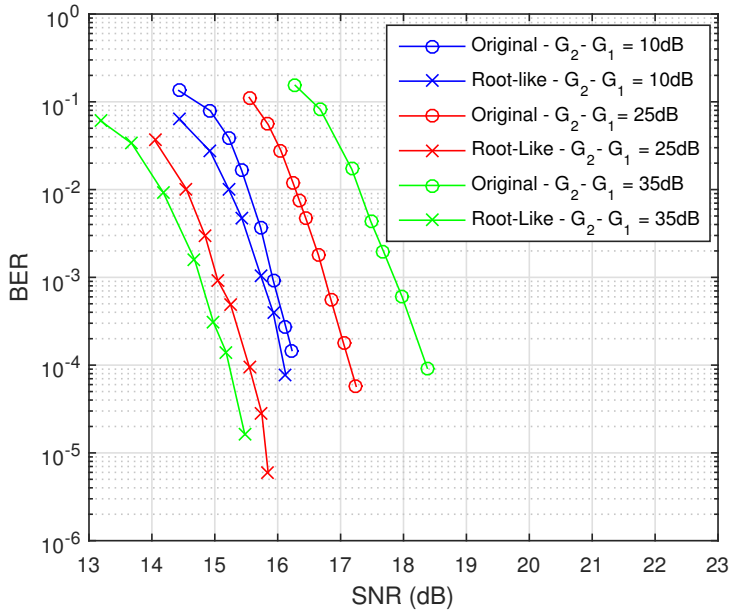

Figura 5.13: BER do sistema original vs. o sistema com mapeamento de bits *root-like* para a matriz do G.hn com subcanais variando linearmente de G_1 a G_2, com diferentes valores de gradientes $G_2 - G_1$, através do canal AWGN, com $N = 1920$ bits, taxa 1/2 e 256-QAM. A SNR mostrada é a média de todas as sub-portadoras.

Avaliou-se também o impacto do gradiente na regra prática descrita na Sessão 5.5. Como visto acima, a regra prática indica uma maneira fácil de definir o valor ótimo do parâmetro de deslocamento (s), que também pode ser validado quando a análise do gradiente é considerada. Por exemplo, a Figura 5.14 mostra a BER do mapeamento de bits *root-like* considerando o gradiente $G_2 - G_1 = 25$ dB, para diferentes deslocamentos. Nota-se que este resultado é similar aos resultados da Figura 5.3. Os deslocamentos $s = 400$, 480 e 560 bits produzem resultados de BER melhores que os deslocamentos $s = 640$ e 1120 bits, o último sendo o pior caso. Aqui, a diferença é que o deslocamento $s = 560$ bits apresenta um desempenho ainda melhor quando comparado aos deslocamentos $s = 400$ e 480 bits, que nesse caso apenas evidencia a precisão da regra prática.

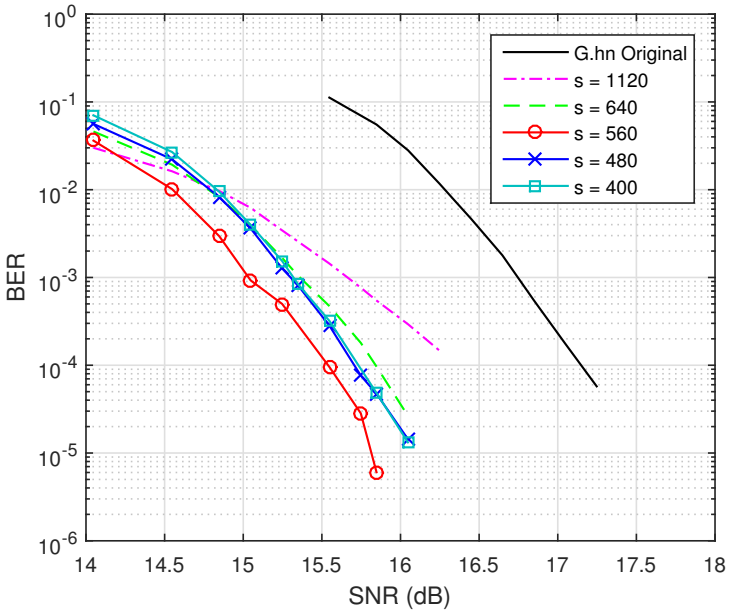

Figura 5.14: BER do sistema original vs. o sistema com mapeamento de bits *root-like* para a matriz do G.hn com sub-canais variando linearmente de G_1 a G_2 sobre o canal AWGN, com $N = 1920$ bits, taxa 1/2 e 256-QAM e com valor de gradiente $G_2 - G_1 = 25$ dB, para diferentes valores de deslocamento. A SNR mostrada é a média de todas as sub-portadoras.

Capítulo 6

Conclusões

Diversos padrões de sistemas de comunicação têm optado por usar códigos LDPC para melhorar o desempenho dos sistemas. Os códigos LDPC possuem a vantagem da alta correção de erros, se aproximando do limite de Shannon e da sua decodificação iterativa de fácil implementação. Uma forma de melhorar o desempenho do sistema QAM codificado com LDPC como os do sistema G.hn e IEEE 802.11n, por exemplo, é através da técnica de mapeamento de bits. Um mapeador de bits pode se aproveitar da propriedade de UEP dos códigos LDPC irregulares e/ou da existência de diferentes sub-canais com diferentes qualidades (ou em termo de SNR média ou de capacidade), para mapear propriamente os bits codificados nos sub-canais.

Nesse sentido, este Tese apresentou um novo mapeamento de bits chamado mapeamento de bits *root-like* para sistemas QAM codificados com LDPC. O mapeamento de bits é feito de tal forma que nós de variável bons cooperem com nós de variável ruins visando um melhor desempenho geral do sistema.

O método proposto foi primeiramente testado em dois cenários: modo de transmissão OFDM e de portadora única com AWGN, considerando o sistema G.hn, sem otimização da distribuição dos graus dos nós, no entanto, já mostrando bons resultados em termos de ganhos de codificação. Estes resultados foram publicados no Globecom'2013 [25].

Esta Tese também apresentou a otimização do mapeamento de bits através da técnica de EXIT *charts*, desenvolvido através de um algoritmo de otimização. A otimização foi testada nos dois cenários: modo de transmissão OFDM e de portadora única com AWGN, e com desvanecimento *Rayleigh*, para o caso do sistema IEEE 802.11n. Simulações foram realizadas e o desempenho da BER foi medido. Os resultados mostraram que, no modo de transmissão por portadora única, o espaço de busca para a otimização do mapeamento de bits *root-like* através da análise de EXIT *charts* é bastante reduzido, uma vez que os bits ruins são alocados

para um nível de confiabilidade fixo, restando apenas alguns níveis de confiabilidade para alocar os bits ajudantes.

No caso do modo de transmissão OFDM, a otimização do parâmetro de deslocamento s através de EXIT *charts*, revelou que bons resultados são obtidos simplesmente seguindo a ideia do mapeamento de bits *root-like* e de uma simples regra prática para especificar o parâmetro de deslocamento, sem precisar necessariamente de qualquer otimização.

Esta Tese foi concluída com um estudo do impacto do nível de desequilíbrio de confiabilidade através dos sub-canais sobre o desempenho do mapeamento de bits *root-like*. Observou-se que, quanto maior o desequilíbrio entre os sub-canais "ruins" e os "bons", maior será o efeito do mapeamento de bits no desempenho do sistema. Avaliou-se também o impacto do nível de desequilíbrio de confiabilidade na regra prática, ajudando a evidenciar a sua precisão.

Estes resultados de otimização do mapeamento de bits usando EXIT *charts* foram recentemente publicados em [26].

É importante ressaltar que o mapeamento de bits *root-like* com EXIT *charts*, pode também ser aplicado a outros códigos LDPC, desde que a matriz de verificação de paridade satisfaça as restrições do mapeamento de bits *root-like*.

Para aplicações práticas, como sistemas com variações de canal, o mapeamento de bits *root-like* provou ser um esquema de mapeamento de bits interessante e de baixa complexidade que pode rapidamente se adaptar em resposta as variações do canal, melhorando o desempenho geral do sistema. Nessa direção, um problema interessante para investigação seria a avaliação do impacto de uma informação do estado do canal (*channel state information* (CSI)) incorreta ou desatualizada, no desempenho do mapeamento de bits *root-like*. Devido a separação relativamente grande entre os nós mapeados "ruins" e os nós mapeados "ajudantes" na Eq.(3.4), em termos das qualidades dos sub-canais associadas, e devido a baixa sensibilidade de desempenho do erro em relação ao parâmetro s em torno do seu melhor valor (ver, por exemplo, Figura 5.2), acredita-se que o mapeamento de bits *root-like* é bastante robusto para pequenos desvios da CSI. No entanto, uma investigação adequada dessa questão é deixada como trabalho futuro.

Finalmente, deve-se mencionar que a relação de desempenho-complexidade superior do mapeamento de bits proposto nesta Tese foi mostrada apenas através de simulações computacionais, e foi atribuída ao simples conceito de altruísmo, que é bastante intuitivo. No entanto, uma análise rigorosa dessa superioridade esta programada e é deixada também como trabalho futuro.

Referências Bibliográficas

[1] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 28, 1948.

[2] R. G. Gallager, *Low-Density Parity-Check Codes*, Ph.D. thesis, Cambridge, MA: MIT Press, 1963.

[3] D. J. C MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Transactions on Information Theory*, vol. 45, pp. 399–431, Mar. 1999.

[4] *Frame Structure, Channel Coding and Modulation for a Second Generation Digital Terrestrial Television Broadcasting System (DVB-T2)*, ETSI Std. EN 302 755 V1.1.1, Set. 2009.

[5] *Framing Structure, Channel Coding and Modulation for Digital Television Terrestrial Broadcasting System*, Chinese National Standard GB 20600-2006, Ago. 2006.

[6] IEEE P802.11n^{TM}/D1.02, "Draft Amendment to STANDARD Information Technology Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Enhancements for Higher Throughput," *IEEE 802.11 document*, Out. 2009.

[7] IEEE P802.16e^{TM}, "Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems," *IEEE 802.16 document*, Fev. 2005.

[8] *ITU G.9960, Unified high-speed wire-line based home networking transceivers - System architecture and physical layer specification*, Geneva:ITU-T Std., Jun. 2010.

[9] V. Oksman and S. Galli, "G.hn: the new ITU-T home networking standard," *IEEE Commun. Mag.*, vol. 47, no. 10, pp. 138–145, Out. 2009.

[10] John G. Proakis, *Digital Communications*, McGraw-Hill, 4th edition, 2001.

[11] T. Richardson and R. L. Urbanke, *Modern Coding Theory*, Cambridge University Press, 2008.

[12] S.Y. Chung, T. Richardson, and R. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a gaussian approximation," *IEEE Trans. Inform. Theory*, vol. 47, pp. 657–670, Fev. 2001.

[13] Stephan ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes," *IEEE Trans. Commun.*, vol. 49, pp. 1727–1731, Out. 2001.

[14] S. ten Brink, G. Kramer, and A. Ashikhmin, "Design of low-density parity-check codes for modulation and detection," *IEEE Trans. Commun.*, vol. 52, pp. 670–678, Abr. 2004.

[15] Liang Gong, Lin Gui, Bo Liu, Bo Rong, Yin Xu, Yiyan Wu, and Wenju Zhang, "Improve the Performance of LDPC Coded QAM by Selective Bit Mapping in Terrestrial Broadcasting System," *IEEE Transactions on Broadcasting*, vol. 57, no. 2, pp. 263 – 269, Jun. 2011.

[16] Jr. L. J. Cimini and N. R. Sollenberger, "OFDM with diversity and coding for high-bit-rate mobile data applications," *Mobile Multimedia Commun.*, vol. 1, pp. 247–254, 1997.

[17] Ramjee Prasad, *OFDM for Wireless Communications Systems*, Artech House, Inc., 1a edition, 2004.

[18] Yan Li and William E. Ryan, "Bit-reliability mapping in ldpc-coded modulation systems," *IEEE Comm. Letters*, vol. 9, no. 1, pp. 1–3, Jan. 2005.

[19] Gerd Richter, Axel Hof, and Martin Bossert, "On the mapping of low-density parity-check codes for bit-interleaved coded modulation," *ISIT2007*, Jun. 2007.

[20] Jing Lei and Wen Gao, "Matching graph connectivity of ldpc codes to high-order modulation by bit interleaving," *46th Allerton Conference*, pp. 1059–1064, 2008.

[21] Keqian Yan, Kewu Peng, Fang Yang, Bingzhen Zhao, and Yirong Wang, "Performance improvement of G.hn system based on bit mapping technique," *IEEE International Symposium on Power Line Communications and Its Applications*, pp. 150–154, Mar. 2012.

[22] Stefan Nowak and Ruediger Kays, "Interleaver design for spectrally-efficient bit-interleaved ldpc-coded modulation," *International Symposium on Turbo Codes and Iterative Information Processing (ISTC)*, pp. 240–244, Ago. 2012.

[23] Stefan Nowak and Ruediger Kays, "On matching short ldpc codes with spectrally-efficient modulation," *IEEE International Symposium on Information Theory Proceedings*, 2012.

[24] J. J. Boutros, A. Fabregas, E. Biglieri, and G.Zemorn, "Low-density parity-check codes for nonergodic block-fading channels," *IEEE Trans. on Inf. Theory*, vol. 56, pp. 4286–4300, Out. 2007.

[25] Fernanda Smith, Evaldo Pelaes, and Bartolomeu F. Uchôa-Filho, "A simple root-like bit mapping to improve the performance of LDPC-coded QAM systems," *IEEE Global Communications Conference (GLOBECOM)*, pp. 1885–1890, Dez. 2013.

[26] Fernanda Smith, Evaldo Pelaes, and Bartolomeu F. Uchôa-Filho, "EXIT charts analysis of a root-like bit mapping for LDPC-coded QAM systems," *Digital Signal Processing*, vol. 70, pp. 39–48, Nov. 2017.

[27] Willian E. Ryan and Shu Lin, *Channel Codes: Classical and Modern*, Cambridge University Press, 2009.

[28] Shu Lin and Daniel Costello Jr., *Error Control Coding: Fundamentals and Applications*, Prentice-Hall, 1983.

[29] Sarah J. Johnson, *Iterative Error Correction Turbo, Low-Density Parity-Check and Repeat-Accumulate Codes*, Cambridge, 2009.

[30] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Transactions on Information Theory,*, vol. IT-27, no. 5, pp. 533–547, Set. 1981.

[31] T. J. Richardson, M. A. Shokrollahi, , and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inform. Theory,*, vol. 47, pp. 619–637, 2001.

[32] M. Franceschini, G. Ferrari, and R. Raheli, *LDPC Coded Modulations*, Springer Publishing Company, Incorporated, 2009.

[33] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Improved low-density parity-check codes using irregular graphs and belief propagation," *IEEE Trans. Inform Theory,*, vol. 47, pp. 585–598, Ago. 1998.

[34] S. young Chung, G. D. Forney, T. J. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 db of the shannon limit," *IEEE Communications Letters*, vol. 5, pp. 58–60, Mar. 2001.

[35] T. J. Richardson and R. Urbanke, "Efficient encoding of low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 638–656, Fev. 2001.

[36] S. Myung, K. Yang, and J. Kim, "Quasi-cyclic LDPC codes for fast encoding," *IEEE Trans. Inf. Theory*, vol. 51, pp. 2894–2900, Ago. 2005.

[37] Werner Henkel, Khaled Hassan, Neele von Deetzen, Sara Sandberg, Lucile Sassatelli, and David Declercq, "UEP concepts in modulation and coding," *Advances in Multimedia*, vol. 2010, 2010.

[38] E. Sharon, A. Ashikhminand, and S. Litsyn, "Exit functions for the gaussian channel," *40th Annu. Allerton Conf. on Communication, Control, Computers, Allerton, IL*, pp. 972–981, 2003.

[39] Giuseppe Caire, Giorgio Taricco, and Ezio Biglieri, "Bit-interleaved coded modulation," *IEEE Trans. Inf. Theory*, vol. 44, no. 3, pp. 927–946, Maio 1998.

[40] Alexei Ashikhmin, Gerhard Kramer, and Stephan ten Brink, "Extrinsic information transfer functions: Model and erasure channel properties," *IEEE Trans. on Inf. Theory*, vol. 50, pp. 2657–2673, 2004.

[41] Sreenivas Rao Kollu and Hamid Jafarkhani, "On the exit chart analysis of low-density parity-check codes," *IEEE Global Communications Conference (GLOBECOM)*, 2005.

[42] Jerry D. Gibson, *Mobile Communications Handbook*, CRC Press Book, 1999.

Apêndice A

Matrizes de Verificação de Paridade

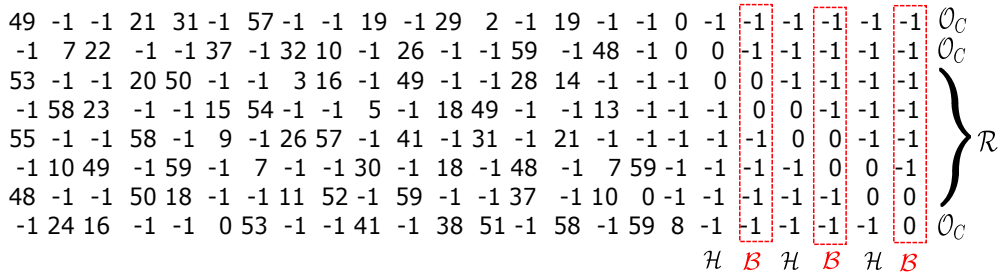

Figura A.1: Matriz de verificação de paridade do padrão G.hn ($c = 12, t = 24, b = 60$) para $N = 1440$ bits, $K = 960$, taxa 2/3 [8].

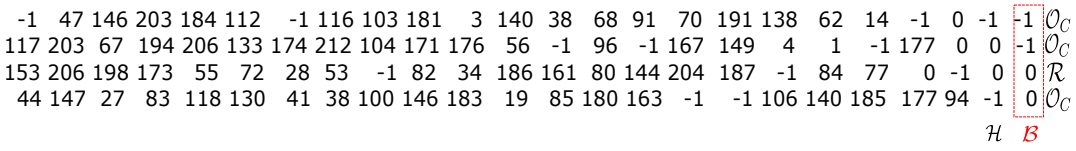

Figura A.2: Matriz de verificação de paridade do padrão G.hn ($c = 12, t = 24, b = 216$) para $N = 5184$ bits, $K = 4320$, taxa 5/6 [8].

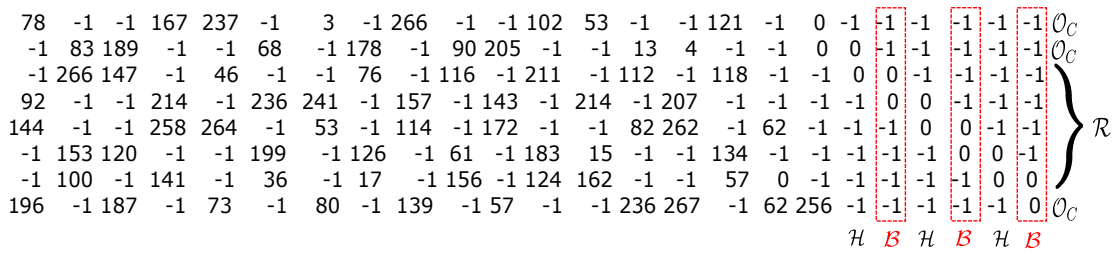

Figura A.3: Matriz de verificação de paridade do padrão G.hn ($c = 12, t = 24, b = 270$) para $N = 6480$ bits, $K = 4320$, taxa 2/3 [8].

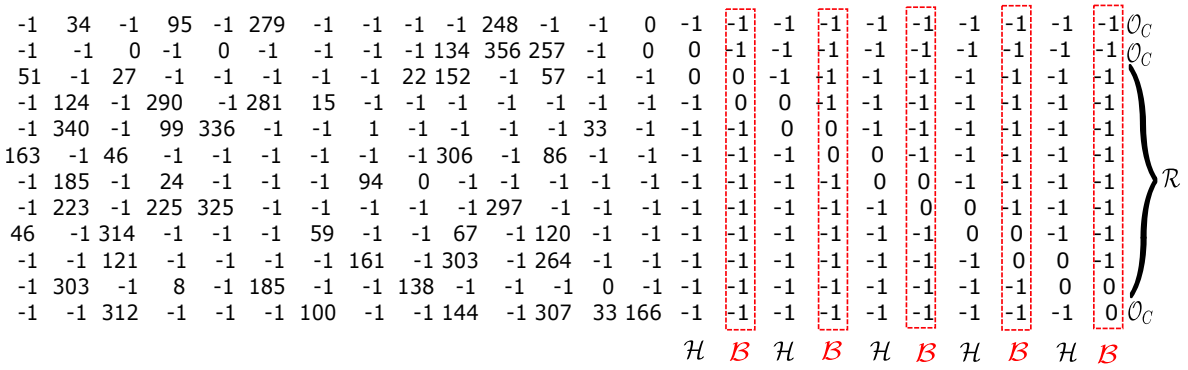

Figura A.4: Matriz de verificação de paridade do padrão G.hn ($c = 12, t = 24, b = 360$) para $N = 8640$ bits, $K = 4320$, taxa 1/2 [8].

Apêndice B

Distribuições de graus dos nós de variável $\lambda(x)$ e dos nós de paridade $\rho(x)$

Matrizes de verificação de paridade do Padrão G.hn:

- $\lambda(x)$ e $\rho(x)$ para as matrizes de verificação de paridade da Figura 5.1 ($N = 1920, K = 960, b = 80$) e da Figura A.4 ($N = 8640, K = 4320, b = 360$), ambas com taxa $R = 1/2$.

$$\lambda(x) = 0.2597x + 0.3506x^2 + 0.3896x^5 \tag{B.1}$$

$$\rho(x) = 0.0649x^4 + 0.3896x^5 + 0.5455x^6 \tag{B.2}$$

- $\lambda(x)$ e $\rho(x)$ para as matrizes de verificação de paridade da Figura A.1 ($N = 1440, K = 960, b = 60$) e da Figura A.3 ($N = 6480, K = 4320, b = 270$), ambas com taxa $R = 2/3$.

$$\lambda(x) = 0.1463x + 0.0732x^2 + 0.7805x^3 \tag{B.3}$$

$$\rho(x) = 0.1098x^8 + 0.4878x^9 + 0.4024x^{10} \tag{B.4}$$

- $\lambda(x)$ e $\rho(x)$ para a matrizes de verificação de paridade da Figura A.2 ($N = 5184, K = 4320, b = 216$), com taxa $R = 5/6$.

$$\lambda(x) = 0.0494x + 0.4074x^2 + 0.5432x^3 \tag{B.5}$$

$$\rho(x) = 0.2346x^{18} + 0.2469x^{19} + 0.5185x^{20} \tag{B.6}$$

Matrizes de verificação de paridade do Padrão IEEE 802.11n:

- $\lambda(x)$ e $\rho(x)$ para a matriz de verificação de paridade da Figura 5.11 ($N = 1944, K = 972, b = 81$), com taxa $R = 1/2$.

$$\lambda(x) = 0.2558x + 0.3140x^2 + 0.0465x^3 + 0.3837x^{10} \tag{B.7}$$

$$\rho(x) = 0.8140x^6 + 0.1860x^7 \tag{B.8}$$

Apêndice C

Distribuições do Mapeamento de bits *Root-like*

Tabela C.1: Distribuição do mapeamento de bits *root-like* para a matriz do G.hn ($N = 6480$, $R = 2/3$) com canal de portadora única AWGN e 64-QAM [8].

$i \rightarrow$	2	3	4
$m_{1,i}$	0.5000	1.0000	0.1875
$m_{2,i}$	0.5000	0.0000	0.3125
$m_{3,i}$	0.0000	0.0000	0.5000

Tabela C.2: Distribuição do mapeamento de bits *root-like* para a matriz do G.hn ($N = 8640$, $R = 1/2$) com canal de portadora única AWGN e 256-QAM [8].

$i \rightarrow$	2	3	6
$m_{1,i}$	0.5000	0.1111	0.0000
$m_{2,i}$	0.5000	0.1111	0.0000
$m_{3,i}$	0.0000	0.4444	0.4000
$m_{4,i}$	0.0000	0.3333	0.6000

Tabela C.3: Distribuição do mapeamento de bits *root-like* para a matriz do G.hn ($N = 8640$, $R = 1/2$) com canal de portadora única AWGN e 64-QAM [8].

$i \rightarrow$	2	3	6
$m_{1,i}$	0.5000	0.2222	0.2000
$m_{2,i}$	0.5000	0.2222	0.2000
$m_{3,i}$	0.0000	0.5555	0.6000