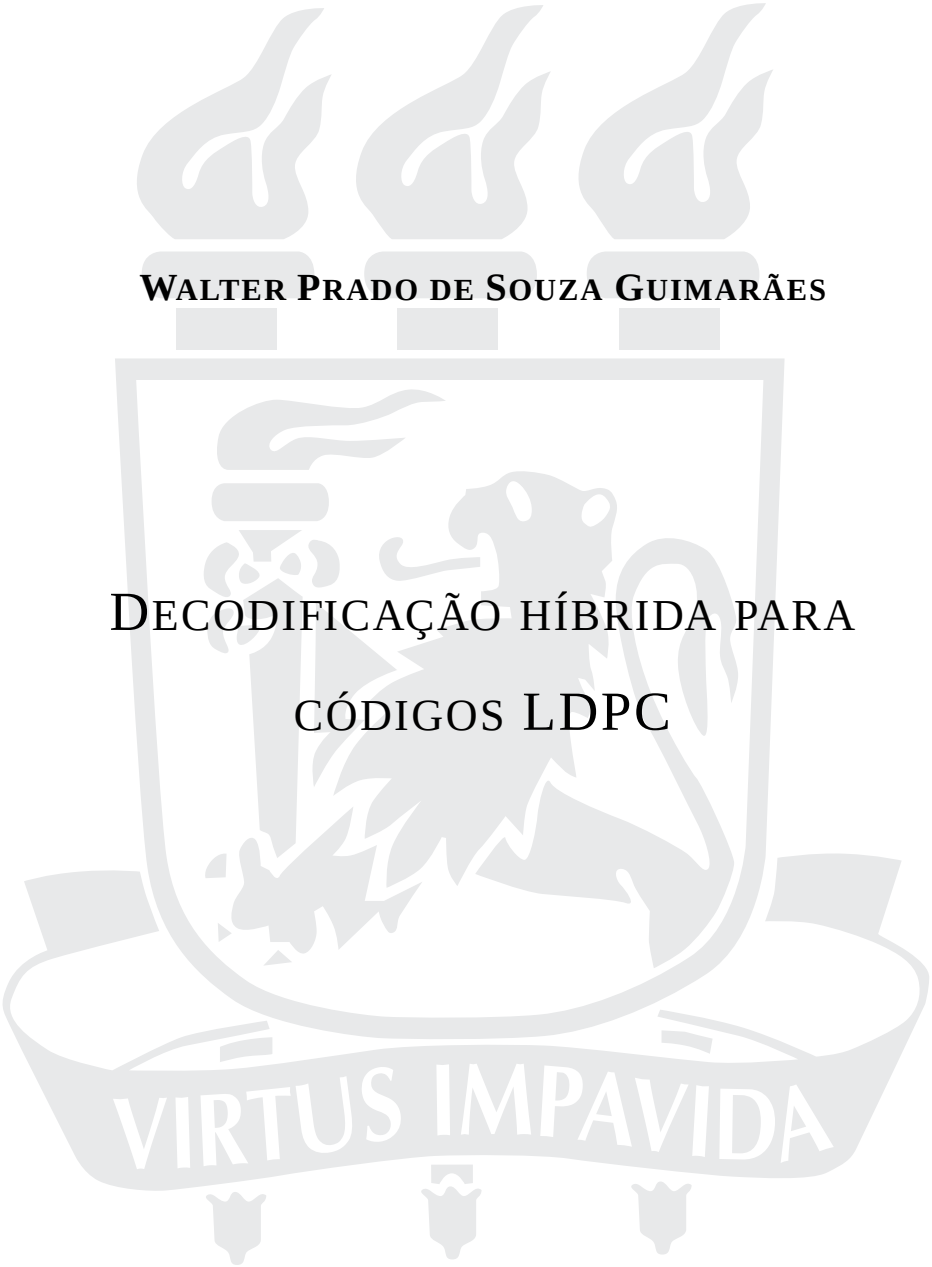
UNIVERSIDADE FEDERAL DE PERNAMBUCO

CENTRO DE TECNOLOGIA E GEOCIÊNCIAS

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA



WALTER PRADO DE SOUZA GUIMARÃES

DECODIFICAÇÃO HÍBRIDA PARA CÓDIGOS LDPC

RECIFE, FEVEREIRO DE 2013.

WALTER PRADO DE SOUZA GUIMARÃES

DECODIFICAÇÃO HÍBRIDA PARA CÓDIGOS LDPC

Tese submetida ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Pernambuco como parte dos requisitos para obtenção do grau de **Doutor em Engenharia Elétrica**

ORIENTADOR: PROF. VALDEMAR CARDOSO DA ROCHA JÚNIOR, PH.D.

Recife, Fevereiro de 2013.

Catalogação na fonte
Bibliotecário Marcos Aurélio Soares da Silva, CRB-4 / 1175

G963d Guimarães, Walter Prado de Souza.
Decodificação híbrida para códigos LDPC / Walter Prado de Souza Guimarães. - Recife: O Autor, 2013.
135 folhas, il., gráfs., tabs.

Orientador: Profº Drº. Valdemar Cardoso da Rocha Júnior.
Tese(Doutorado) – Universidade Federal de Pernambuco. CTG. Programa de Pós-Graduação em Engenharia Elétrica, 2013.
Inclui Referências.

1. Engenharia Elétrica 2. Decodificação de Apagamentos. 3.Códigos LDPC. 4.Decodificação Iterativa de Erros. I. Rocha Júnior, Valdemar Cardoso da (Orientador) II. Título.

621.3 CDD (22. ed.)

UFPE
BCTG/2013-115

Aos meus pais,

Antonio Silva Guimarães e

Neidson Souza Guimarães

AGRADECIMENTOS

Agradeço, primeiramente, a Deus por sempre iluminar meus pensamentos e, desta forma, permitir que eu sempre supere as dificuldades que encontro no decorrer da minha vida.

Agradeço aos meus pais, Antonio Silva Guimarães e Neidson Souza Guimarães, pois o amor e o apoio incondicional deles me motivam a sempre lutar para realizar meus sonhos. Agradeço a Elizabeth Souza Guimarães e também ao meu filho, Felipe Augusto Souza Guimarães, por estarem ao meu lado nos momentos mais difíceis.

Em especial, agradeço ao meu orientador, Prof. Valdemar C. da Rocha Jr., por sua dedicação, incentivo e amizade.

Agradeço aos professores Cecílio Pimentel, Eduardo Fontana, Hélio Magalhães, Rafael Dueire e Ricardo Campelo pelas disciplinas ministradas durante a pós-graduação.

Agradeço ao colega e doutorando José Sampaio pela parceria e apoio na elaboração de artigos. Aos colegas do curso do Doutorado DINTER UFPE-UEA. Aos Coordenadores Prof. Antenor e Prof. Francis Wagner. Ao núcleo de meteorologia da UEA por disponibilizar a sua infraestrutura de computadores.

Faço também um agradecimento póstumo ao ex-Chefe do Departamento de Eletrônica da UFAM, Prof Aldenir, por seu apoio incondicional. Agradeço ao Prof. Cícero Fernandes da UFAM pela série de convites para assistir a seminários voltados à defesa de dissertação de Mestrado.

Agradeço ao Ex-Diretor Técnico da FUCAPI, Dr. Evandro L. X. Vieiralves, e ao Diretor do Departamento Educacional da FUCAPI, Dr. Niomar L. Pimenta, pela amizade e pelo apoio incondicional.

Por fim, agradeço à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo apoio financeiro.

WALTER PRADO DE SOUZA GUIMARÃES

Universidade Federal de Pernambuco

22 de Fevereiro de 2013

Resumo da Tese apresentada à UFPE como parte dos requisitos necessários para a obtenção do grau de Doutor em Engenharia Elétrica

DECODIFICAÇÃO HÍBRIDA PARA CÓDIGOS LDPC

Walter Prado de Souza Guimarães

Fevereiro/2013

Orientador: Prof. Valdemar Cardoso da Rocha Júnior, Ph.D.
Área de Concentração: Comunicações
Palavras-chaves: Decodificação iterativa de erros, Códigos LDPC, Decodificação de apagamentos.
Número de páginas: 135

Os códigos *Low-Density Parity-Check* (LDPC) constituem uma família definida a partir de matrizes esparsas de verificação de paridade e que apresentam excelente desempenho no canal com ruído aditivo Gaussiano branco (RAGB). Devido às suas boas características, têm sido largamente empregados na codificação de canais em sistemas de transmissão via satélite, sistemas de telefonia móvel e sistemas de radiodifusão de TV digital. O sucesso desses códigos é devido à sua representação na forma de grafos, ao uso de métodos de construção mais simplificados e ao processo de decodificação iterativa. Esta tese introduz um método de decodificação iterativa híbrida que, diferentemente da maioria dos modelos existentes, associa a correção de erros à correção de apagamentos em canais com RAGB, como uma forma de melhorar o desempenho do código LDPC nestes canais. O alvo dessa abordagem é a região de patamar de erros dos códigos LDPC, em que os padrões de erros, em sua maioria, são de pequena cardinalidade e resultantes do que se conhece por conjunto de armadilhas. Alguns aspectos do funcionamento e da operação otimizada da decodificação iterativa híbrida são explorados e discutidos. Para confirmar a eficácia da técnica de decodificação introduzida, são apresentados resultados de simulação em computador para códigos LDPC empregados no padrão IEEE802.11n, acompanhados da respectiva análise.

Abstract of Thesis presented to UFPE as a partial fulfillment of the requirements for the degree of Doctor in Electrical Engineering

HYBRID DECODING FOR LDPC CODES

Walter Prado de Souza Guimarães

February/2013

Supervisor: Prof. Valdemar Cardoso da Rocha Júnior, Ph.D.
Area of Concentration: Communications
Keywords: Iterative error decoding, LDPC codes, Erasure decoding.
Number of pages: 135

The Low-Density Parity-Check (LDPC) codes are a family of codes defined by sparse parity check matrices that exhibit excellent performance over channels disturbed by additive white Gaussian noise (AWGN). Due to their good characteristics, LDPC codes have been widely used in channel coding of satellite broadcasting systems, mobile phone systems and digital TV broadcasting systems. The success of LDPC codes is due to their representation in the form of graphs, the availability of practical construction methods and iterative decoding process. This thesis introduces a hybrid iterative decoding method for LDPC codes that, unlike most existing techniques, adds erasure correction if the usual error correction procedure fails, as a means to improve the performance of LDPC codes on AWGN channels. The target of this approach is the error floor region of LDPC codes, in which most patterns of errors have small cardinality and result from what is known as trapping sets. Some aspects of its functioning and optimization are investigated and discussed. The results and computer simulation analysis for the LDPC codes adopted by IEEE802.11n standard are presented to confirm its effectiveness.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

BCH	Bose-Chaudhuri-Hocquenghem
BEC	Canal binário com apagamentos (*Binary Erasure Channel*)
BER	Taxa de erro de *bit* (*Bit Error Rate*)
BP	Propagação de estimativas de probabilidade (*Belief Propagation*)
dB	Decibel
DH	Decodificação Híbrida
DTMB	*Digital Terrestrial Television Multimedia Broadcasting*
DVB-S2	*Digital Video Broadcasting - Satellite - Second Generation*
FPGA	Arranjo programável de portas em campo (*Field Programmable Gate Array*)
LDPC	Código baseado em matriz esparsa de verificação de paridade (*Low-Density Parity-Check*)
LLR	Logaritmo da razão de verossimilhança (*Log-Likelihood Ratio*)
MAP	*Maximum a Posteriori*
ML	Máxima verossimilhança (*Maximum Likelihood*)
NASA	*National Aeronautics and Space Administration*
OSD	Decodificação estatisticamente ordenada (*Ordered Statistic Decoding*)
PP	Pós-processamento (*Post-Processing*)
QLLR	Logaritmo da razão de verossimilhança quantizado (*Quantized Log-Likelihood Ratio*)
RA	Repete-acumula (*Repeat-Accumulate*)
RAGB	Ruído Aditivo Gaussiano Branco
RAM	Memória volátil de acesso randômico *Random Access Memory*
RS	Reed-Solomon
SNR	Relação sinal-ruído (*Signal-to-Noise Ratio*)
SPA	Algoritmo Soma-Produto (*Sum-Product Algorithm*)
WER	Taxa de erro de palavra (*Word Error Rate*)
WIMAX	*Worldwide Interoperability for Microwave Access*

SUMÁRIO

CAPÍTULO 1

INTRODUÇÃO

A percepção da importância das técnicas de codificação de canal para a transmissão e o armazenamento robusto de dados tem suas origens no trabalho de Shannon [1], que lançou as bases da Teoria da Informação. A partir deste trabalho, colocou-se o desafio de se aproximar dos limites teóricos formulados para a taxa máxima de transmissão confiável de dados para um canal de comunicações, por meio do uso de técnicas de codificação de canal.

Quase meio século depois, Berrou et al. [2] apresentaram uma nova forma de codificação designada por códigos Turbo que, baseada em técnicas de decodificação iterativa, pela primeira vez aproximou-se do limite teórico de Shannon em canais afetados por ruído aditivo Gaussiano branco (RAGB).

Na realidade, os fundamentos para essa nova abordagem já haviam sido lançados na década de 1960 por Gallager [3], [4], que na sua tese de doutorado tinha proposto uma nova classe de códigos baseados em matrizes esparsas de verificação de paridade, conhecidos por *Low-Density Parity-Check* (LDPC), e um algoritmo de decodificação iterativa chamado Algoritmo Soma-Produto (*Sum-Product Algorithm* — SPA). Essas técnicas passaram despercebidas por quase duas décadas quando foram revistas por Tanner [5].

Em artigo publicado em 1997, Mackay e Neal [6] confirmaram as excelentes propriedades dos códigos LDPC para a correção de erros, tendo provado que, à semelhança dos códigos Turbo, os códigos LDPC conseguiam atingir uma probabilidade de erro muito próxima do limite de Shannon, especialmente para códigos de comprimento longo.

Após o trabalho de Mackay e Neal, muitas pesquisas têm sido realizadas no sentido de reduzir a

complexidade e aumentar a eficiência dos códigos LDPC, tomando como base os novos métodos de construção [7]–[11], de codificação e de decodificação [12]–[15].

Em termos de decodificação, os métodos empregados podem ser classificados em [16]:

- ▷ Decodificação por Decisão Suave — Nesse tipo, o algoritmo de decodificação leva em conta as informações de confiabilidade ou valores de probabilidade dos símbolos recebidos do canal;
- ▷ Decodificação por Decisão Abrupta — Nesse tipo, o algoritmo de decodificação processa apenas os valores quantizados dos dígitos recebidos do canal (elementos de um conjunto finito de símbolos, tipicamente '0' e '1'), desprezando as informações de confiabilidade ou valores de probabilidade em prol da simplificação.

O Algoritmo Soma-Produto, por exemplo, empregado na decodificação iterativa dos códigos LDPC, enquadra-se na categoria de decodificação por decisão suave. Este algoritmo pode tanto empregar o método invasivo (*flooding scheduling*) [17], [18], que atualiza simultaneamente todas as mensagens no grafo de Tanner [5], como também empregar o método serial (*serial scheduling*) [19]–[23], que, diferente do primeiro, procura atualizar as mensagens prioritárias, baseando-se em algum parâmetro de avaliação. Esse último método de atualização de mensagens possibilita aumentar a velocidade de decodificação, permitindo a aplicação dos códigos LDPC em sistemas de comunicação de altas taxas de dados e de baixa latência, como por exemplo, no padrão IEEE802.16e (WIMAX) [24].

A maioria das propriedades dos códigos LDPC são bem conhecidas, no entanto o seu comportamento na região de patamar de erros (*error floor*) da sua curva de desempenho continua sendo um problema aberto. Essa região é caracterizada por uma súbita saturação da taxa de erro de *bit* (*Bit Error Rate* — BER) ou da taxa de erro de palavra (*Word Error Rate* — WER) que ocorre para valores de relação sinal-ruído (*Signal-to-Noise Ratio* — SNR) elevados [25].

Há várias formas de melhorar o desempenho dos códigos LDPC na região de patamar de erros. Uma delas consiste na combinação dos códigos LDPC a outros códigos formando uma configuração conhecida por concatenação de códigos [26]. Atualmente, é possível encontrar códigos LDPC concatenados a códigos BCH, usados em padrões de transmissão de TV Digital [27], [28] e em padrões de comunicação de dados via satélite para transmissão de imagem e som [29]. Em ambas as situações, os códigos LDPC funcionam como código interno (*inner code*). Essa configuração de códigos concatenados substitui a configuração tradicional composta por códigos Reed-Solomon e códigos convolucionais [26].

Uma outra forma consiste na técnica de pós-processamento (*Post-processing* — PP). Na apli-

cação dessa técnica pode-se destacar o trabalho descrito em [30], em que é proposto um algoritmo de decodificação que reduz a frequência de eventos de erros causados por conjuntos de armadilhas (*trapping sets*) [25] pela extração das médias das mensagens em algumas iterações. Em [31], é empregada a decodificação baseada em programação linear. Em [32], é proposto um pós-processamento baseado em consulta à tabela (*lookup table*) de conjuntos de armadilhas conhecidos para um dado código LDPC. Em [33], é proposta uma concatenação serial de um decodificador *Belief Propagation* (BP) [34] com um reprocessamento por decodificação estatisticamente ordenada (*Ordered Statistic Decoding* — OSD) [35] empregando os valores de logaritmos da razão de verossimilhança (*Log-Likelihood Ratio* — LLR) [36] acumulados durante a decodificação iterativa como medidas de confiabilidade.

O artifício da correção de apagamentos, aplicado após uma falha na decodificação de erros como uma técnica de pós-processamento, tem recebido pouca atenção tanto em canais com RAGB como em outros modelos de canais. Em [37] é proposta uma técnica de pós-processamento baseada em correção de apagamentos que é ativada após a ocorrência de falha na decodificação SPA e sob certas condições de peso da síndrome [38], [39], [16].

Diferentemente, nesta tese é proposto um método de pós-processamento baseado em correção de apagamentos, que atua sempre que houver a falha da decodificação SPA. Os apagamentos são gerados por um canal binário com apagamentos (*Binary Erasure Channel* — BEC) [40] artificialmente criado e que se baseia nas informações de confiabilidade dos dígitos geradas ao fim da decodificação SPA do tipo *min-sum* BP [41], que emprega o algoritmo denotado por *min-sum* modificado que é descrito na Seção 3.6. O número de apagamentos gerados por esse canal é fixo e predeterminado, tornando menos complexa a ação da decodificação iterativa de apagamentos. Essa técnica proposta é designada como decodificação híbrida (DH).

1.1 MOTIVAÇÃO

A recente adoção dos códigos LDPC nos padrões de transmissão digital de vídeo via satélite (DVB-S2) [29], nos padrões de transmissão de dados por redes sem fio, a curta e a longa distância [42], [24], e a adoção no padrão de transmissão digital terrestre de TV [27], [28] demonstra bem a importância destes códigos no panorama atual como uma das classes de códigos de correção de erros que apresentam melhor desempenho e que, em futuro próximo, poderão integrar outros padrões. São lançados novos desafios na construção de sistemas codificadores e decodificadores capazes de cumprir os requisitos impostos pelas elevadas taxas de transmissão dos sistemas, aliados às restrições

de custo. Esses fatores motivam um estudo aprofundado das melhores técnicas de codificação e de decodificação dos códigos LDPC a serem implementadas, em *software* ou em *hardware*.

Além disso, há também a motivação para o uso da decodificação de apagamentos como um estágio complementar ou de pós-processamento para o estágio de decodificação de erros, em caso de falha desse último, em razão da seguinte premissa. É sabido que, para um determinado código de bloco linear [38], [39] com distância mínima de Hamming d_{min}, é possível a correção de qualquer padrão com até $d_{min} - 1$ apagamentos [38] e de uma grande quantidade de padrões contendo d_{min} ou mais apagamentos, desde que o total de apagamentos por palavra não exceda o número de dígitos de verificação de paridade do código [43].

1.2 OBJETIVOS

A pesquisa relacionada a esta tese concentra-se na questão da decodificação dos códigos LDPC. Os objetivos principais são:

- ▷ Propor uma nova abordagem de decodificação para códigos LDPC, designada como decodificação híbrida (DH), em canais com RAGB, que alie a decodificação iterativa de erros com a decodificação iterativa de apagamentos;
- ▷ Proporcionar a redução do número global de iterações da etapa de decodificação de erros *min-sum* BP do sistema DH, possibilitando, desse modo, obter complexidade e desempenho equivalentes ao que seria atingido por um decodificador *min-sum* BP convencional empregando um número bem superior de iterações;
- ▷ Executar o levantamento do desempenho dessa nova abordagem, por meio de simulação em computador, para canais com RAGB e, desse modo, possibilitar a análise comparativa com os métodos convencionais de decodificação de erros do tipo *min-sum* BP e com o tipo de decodificação concatenada serial formada pelos códigos LDPC e BCH.

1.3 ORGANIZAÇÃO DA TESE

Os demais capítulos estão organizados da seguinte forma. No Capítulo 2, são revisados os códigos LDPC, com ênfase nas suas características, formas de representação e classificação. No Capítulo 3, é feita uma descrição do SPA, mostrando alguns modelos simplificados, como é o caso do algoritmo *min-sum* BP. No Capítulo 4, é realizada uma descrição da decodificação iterativa de

apagamentos e a análise de seu comportamento para códigos LDPC, baseada na técnica de evolução de densidade de probabilidade (*density evolution*) [13]. O Capítulo 5 discorre sobre o modelo de decodificação híbrida proposto e os conceitos essenciais para o entendimento da técnica. O Capítulo 6 exibe os resultados experimentais obtidos por simulação em computador, procurando ao mesmo tempo realizar uma comparação crítica com os métodos convencionais, segundo o ponto de vista do desempenho. Por último, o Capítulo 7 apresenta as principais conclusões extraídas do trabalho realizado e contém algumas sugestões de trabalhos futuros.

CAPÍTULO 2

CÓDIGOS LDPC

Os códigos LDPC foram propostos por R. G. Gallager em 1962 [3] e ficaram quase duas décadas no esquecimento, até que em 1981 R. M. Tanner retomou e generalizou o trabalho de Gallager e introduziu a representação gráfica de códigos LDPC por meio de grafos biparticionados [5]. No entanto, uma vez mais, os códigos LDPC caíram no esquecimento até que, em meados da década de 1990, após o advento dos códigos Turbo [2], Mackay e Neal [6] verificaram que é possível atingir uma probabilidade de erro muito próxima do limite de Shannon [1] com códigos LDPC longos e decodificados usando o SPA [3], [4], para canais com RAGB. Adicionalmente, os avanços tecnológicos, o surgimento de simplificações do SPA e o aparecimento de formas alternativas de construção dos códigos LDPC, fazendo uso de métodos algébricos e geométricos [44]–[46], contribuiram para torná-los um dos principais alvos de investigação científica.

2.1 INTRODUÇÃO

Um código de bloco linear [38], [39], $\mathcal{C}(N, K)$, com alfabeto q-ário, possui palavras-código representadas por vetores de comprimento N com K dígitos de informação, que formam um subespaço K-dimensional do espaço vetorial $\mathbb{F}_q^N$ de todas as N-uplas q-árias.

Um código de bloco linear é caracterizado por uma matriz geradora $\mathbf{G}$ e uma matriz de verificação de paridade $\mathbf{H}$. O código $\mathcal{C}(N,K)$ é o espaço nulo de $\mathbf{H}$, i.e., uma N-upla binária $\mathbf{c} = (c_1, c_2, \ldots, c_N)$ é uma palavra-código de $\mathcal{C}(N,K)$ se, e somente se, $\mathbf{cH}^T = \mathbf{0}$, o que corresponde ao teste da síndrome [38], [39]. Códigos LDPC são códigos de bloco lineares, em geral binários, cuja matriz de verificação de paridade $\mathbf{H}$ é esparsa, ou seja, o número de elementos não-nulos

em $\mathbf{H}$ é muito menor do que o seu número total de elementos. Na matriz $\mathbf{H}$, as linhas representam as equações de verificação de paridade e cada coluna está associada à posição de um símbolo (ou variável) numa palavra-código.

Baseado na distribuição dos elementos não-nulos por linha e por coluna da matriz $\mathbf{H}$, pode-se classificar os códigos LDPC em dois grupos: os regulares e os irregulares.

2.2 CÓDIGOS LDPC REGULARES

Definição 2.1

Um código LDPC regular é definido como o espaço nulo de uma matriz de verificação de paridade $\mathbf{H}$ *esparsa de dimensões* $M \times N$ *que possui as seguintes propriedades: 1) Cada linha tem peso [38], [39], [16] constante* w_r*, tal que* $w_r \ll N$*; e 2) Cada coluna apresenta peso constante* w_c*, tal que* $w_c \ll M$ *[47], [13].* □

Os códigos LDPC regulares foram propostos pela primeira vez por R. Gallager em 1963 [4]. Em seu trabalho de tese, Gallager definiu um código LDPC (N, K) regular, com $w_c \ll M = N - K$, como um código de bloco linear cuja matriz de verificação de paridade $\mathbf{H}$ tem dimensões $(N-K) \times N$, contendo exatamente w_c posições não-nulas por coluna e w_r posições não-nulas por linha. Gallager provou ainda que fazendo $w_c \geq 3$, os códigos LDPC que podem ser obtidos possuem, na sua maioria, um alto valor para a distância mínima de Hamming [38], [39], d_{min}, bastando, para tal, seguir algumas regras simples de construção como, por exemplo, garantir que quaisquer duas colunas da matriz $\mathbf{H}$ possuam, quando muito, um só elemento não-nulo em comum.

Em um código LDPC regular, cada dígito pertencente a uma palavra-código é envolvido em w_c equações de verificação de paridade e cada equação de verificação de paridade envolve w_r dígitos de uma palavra-código. A densidade da matriz $\mathbf{H}$, denotada por r, é definida como a razão entre o número de elementos não-nulos e o número total de elementos na matriz $\mathbf{H}$. Para um código LDPC (N, K) regular, o número de elementos não-nulos em $\mathbf{H}$ é expresso por qualquer uma das duas expressões a seguir,

$$M \cdot w_r = N \cdot w_c.$$

Como o número total de elementos na matriz $\mathbf{H}$ é $M \cdot N$, a densidade de $\mathbf{H}$ é dada por

$$r = \frac{M \cdot w_r}{M \cdot N} = \frac{w_r}{N} = \frac{N \cdot w_c}{M \cdot N} = \frac{w_c}{M}.$$

Dado que $M = N - K$ e que portanto $K = N - M$, a taxa R do código é definida como

$$R = \frac{K}{N} = \frac{N - M}{N} = 1 - \frac{M}{N} = 1 - \frac{w_c}{w_r},$$

logo $w_c < w_r$, uma vez que $0 < R < 1$.

2.3 CÓDIGOS LDPC IRREGULARES

Definição 2.2

Um código LDPC irregular é definido como o espaço nulo de uma matriz de verificação de paridade $\mathbf{H}$ *esparsa de dimensões* $M \times N$ *em que o peso das linhas e o peso das colunas não são constantes e são caracterizados por funções de distribuição de grau [47], [13].* □

Na matriz $\mathbf{H}$ de um código LDPC (N, K) irregular de comprimento N, a fração de colunas com peso i é denotada por ν_i e a fração de linhas com peso j é denotada por h_j. O par de funções $(\boldsymbol{\nu}, \mathbf{h})$, em que $\boldsymbol{\nu} = \{\nu_1\ \nu_2\ \nu_3\ \ldots\}$ e $\mathbf{h} = \{h_1\ h_2\ h_3\ \ldots\}$, é definido como a distribuição de grau do código [13].

A taxa de um código LDPC (N, K) irregular é dada por [13]

$$R \approx 1 - \frac{\sum_i \nu_i \cdot i}{\sum_j h_j \cdot j}.$$

2.4 GRAFOS DE TANNER E GRAFOS DE FATORES

Diversas classes de códigos corretores de erros podem ser representadas graficamente de forma conveniente. Um exemplo são as treliças usadas na representação gráfica de códigos convolucionais e de códigos de bloco [38], [39]. Em se tratando de códigos LDPC, sejam regulares ou irregulares, pelo fato de sua matriz $\mathbf{H}$ ser do tipo esparsa, eles são melhor representados utilizando grafos de fatores [48], que possuem suas origens nos trabalhos de Tanner [5].

Definição 2.3

Um grafo $\mathcal{G}$ *é uma estrutura que consiste em um conjunto de vértices (ou nós), denotado por* $V = \{v_1, v_2, \ldots\}$*, e um conjunto de ramos, denotado por* $E = \{e_1, e_2, \ldots\}$*. Cada ramo* e_k *pertencente ao conjunto* E *está associado a um par (ordenado ou não-ordenado) (*v_i*,*v_j*) de vértices (não necessariamente distintos) do conjunto* V *[16], [47].* □

Em um grafo $\mathcal{G}$, os vértices v_i e v_j que são ligados via ramo e_k são chamados vértices terminais do ramo e_k e, ao mesmo tempo, também são considerados adjacentes. O conjunto de todos os vértices

que são adjacentes a um vértice v_i qualquer é chamado de vizinhança de v_i. Um ramo em um grafo $\mathcal{G}$ pode ser direcionado, quando o par de vértices que o define é ordenado, ou, em caso contrário, não-direcionado. Os grafos de Tanner se baseiam em classe de grafos biparticionados.

Definição 2.4

Um grafo biparticionado consiste em um conjunto de vértices (ou nós) V, dividido em dois subconjuntos disjuntos, denotados por V_c e V_b. Nenhum par de vértices pertencentes a um mesmo subconjunto é adjacente, isto é, não há ramos que os conecte [47], [49]. □

A Figura 2.1(a) é um exemplo de grafo biparticionado. Neste caso, os vértices identificados por quadrados cheios e círculos vazios constituem dois subconjuntos de vértices. Os quadrados cheios estão conectados apenas a círculos vazios e vice-versa. No caso da Figura 2.1(b), os quadrados cheios possuem conexões entre si e portanto o grafo não é biparticionado.

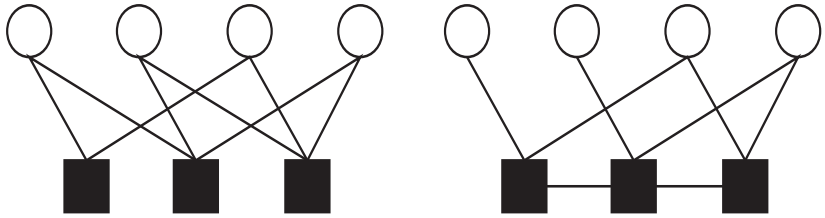


Figura 2.1: *(a) Grafo biparticionado; (b) Grafo não-biparticionado.*

Definição 2.5

Um grafo de Tanner de um código de bloco linear $\mathcal{C}(N, K)$ é um grafo biparticionado não-direcionado obtido a partir de uma matriz de verificação de paridade **H** *com N colunas e M linhas. O grafo de Tanner possui N nós de variável correspondentes às colunas de* **H**, *e M nós de verificação de paridade correspondentes às linhas de* **H**. *Um ramo conecta o i-ésimo nó de variável, x_i, ao j-ésimo nó de verificação de paridade, f_j, se, e somente se, $h_{j,i} = 1$, em que $h_{j,i}$ denota o elemento na j-ésima linha e i-ésima coluna de* **H** *[47], [49].* □

Para um grafo de Tanner $\mathcal{G}(V, E)$, o conjunto dos nós de variável é denotado por $V_b = \{x_1, x_2, \ldots, x_N\}$ e o conjunto dos nós de verificação de paridade é denotado por $V_c = \{f_1, f_2, \ldots, f_M\}$, de modo que $V = V_b \bigcup V_c$. Um elemento e_k do conjunto de ramos E é denotado por (f_j, x_i). O grau de um nó de variável (ou de verificação de paridade) x_i (ou f_j) em um grafo de Tanner, corresponde ao número de ramos que incidem neste nó e é denotado por $d(x_i)$ (ou $d(f_j)$).

Um grafo de Tanner é chamado de regular (representando, por exemplo, um código LDPC regular) quando $d(x_i) = w_c$, para todo i, em que $1 \leq i \leq N$ e $d(f_j) = w_r$, para todo j, em que $1 \leq j \leq M$, caso contrário, o grafo é chamado irregular.

Um dos mais importantes conceitos relativos aos grafos de Tanner é a definição de ciclos.

Definição 2.6

Um ciclo de comprimento l no grafo de Tanner é definido como um percurso fechado formado por l caminhos e que parte de um nó de variável x_i e retorna a este mesmo nó. Todos os ciclos no grafo de Tanner possuem comprimento l par. O ciclo no grafo de Tanner que possua o menor comprimento l é designado de ciclo mínimo ou girth [50], [51]. □

Por exemplo, tendo por base a Figura 2.2, pode-se observar um ciclo de comprimento seis que se encontra assinalado em negrito. Os círculos vazios indicam nós de variável e os quadrados cheios indicam nós de verificação de paridade.

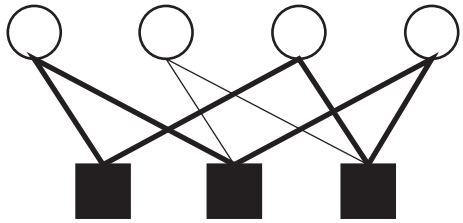


Figura 2.2: *Ciclo com comprimento seis (em negrito) em um grafo de Tanner. Os círculos vazios indicam nós de variável e os quadrados cheios indicam nós de verificação de paridade.*

Os grafos de fatores [48], [52], [53] são uma generalização dos grafos de Tanner. Com eles, é possível modelar não apenas códigos lineares como também funções densidade de probabilidade e equações de estado de sistemas dinâmicos.

Definição 2.7

Uma função de valores reais $g(x_1, x_2, \ldots, x_N)$ com N variáveis como argumento e que pode ser escrita na forma

$$g(x_1, x_2, \ldots, x_N) = \prod_{m=1}^{K_o} f_m(\mathbf{x}_m)$$

é denominada função global [48], na qual cada função $f_m(\mathbf{x}_m)$ é denominada uma função local que tem os elementos de $\mathbf{x}_m$ como argumentos e $\mathbf{x}_m$ é um subconjunto das variáveis $\{x_1, x_2, \ldots, x_N\}$ que participam da função local correspondente [47], [49]. □

Definição 2.8

Um grafo de fatores é um grafo biparticionado não-direcionado que expressa como uma função global de várias variáveis é fatorada como um produto de funções locais. Um grafo de fatores possui um nó para cada variável x_n, um nó para cada função local f_m, e um ramo e_k conectando o nó de variável x_n ao nó de função f_m, se, e somente se, x_n for um argumento de f_m [48], [47], [49]. □

Definição 2.9

A função característica de um código de bloco linear, denotada por $\lambda_c(x) = \lambda_c(x_1, x_2, ..., x_N)$, é uma função que mapeia em 0 um conjunto formado por todos os vetores binários $\mathbf{x} = (x_1, x_2, ..., x_N)$ de comprimento N, se pelo menos uma das equações de verificação de paridade não for satisfeita, ou mapeia em 1, se todas as equações de verificação de paridade forem satisfeitas simultaneamente [49]. □

Da Definição 2.9 é obtida a expressão

$$\lambda_c(\mathbf{x}) = \lambda_c(x_1, x_2, \ldots, x_N) = \begin{cases} 1, \text{ se } \mathbf{x}\mathbf{H}^T = \mathbf{0} \\ 0, \text{ caso contrário.} \end{cases}$$

Se P for uma proposição booleana envolvendo um conjunto de variáveis, a função indicadora $\delta[P]$, também chamada de *delta de Kronecker,* é definida da seguinte forma

$$\delta[P] = \begin{cases} 1, & \text{se } P \text{ for verdadeira} \\ 0, & \text{caso contrário.} \end{cases}$$

A função característica de um código linear é expressa usando a função *delta de Kronecker* para representar cada uma das equações de verificação de paridade. O Exemplo 2.1 a seguir ilustra este caso.

Exemplo 2.1

Supondo que uma dada equação de verificação de paridade seja constituída pelas variáveis (x_2, x_3, x_4), então se a adição em lógica ou-exclusivo ($\oplus$) destas três variáveis resultar em um valor igual a 0 (paridade par), a equação de verificação de paridade correspondente é dita satisfeita e, neste caso, a função característica local dever retornar o valor 1. Caso contrário, deve retornar o valor 0. Isto é representado da seguinte forma $\delta[x_2 \oplus x_3 \oplus x_4 = 0]$, em que $\delta[.]$ representa a função delta de Kronecker. Deste modo, se neste código de comprimento quatro existirem mais duas equações de verificação de paridade representadas por $\delta[x_1 \oplus x_3 = 0]$ e $\delta[x_1 \oplus x_4 = 0]$, a função característica fica assim representada

$\lambda_c(x_1, x_2, x_3, x_4) = \delta[x_1 \oplus x_3 = 0]\delta[x_1 \oplus x_4 = 0]\delta[x_2 \oplus x_3 \oplus x_4 = 0] =$

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

em que

$f_1(x_1, x_3) = \delta[x_1 \oplus x_3 = 0]$, $f_2(x_1, x_4) = \delta[x_1 \oplus x_4 = 0]$ *e* $f_3(x_2, x_3, x_4) = \delta[x_2 \oplus x_3 \oplus x_4 = 0]$.

Portanto, a sequência de bits (x_1, x_2, x_3, x_4) só constitui uma palavra-código se todas as funções locais assumirem valores não-nulos. O grafo de fatores para este exemplo está representado na Figura 2.3. □

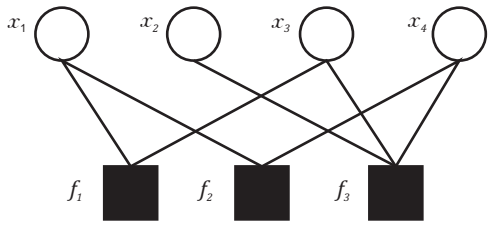


Figura 2.3: *Grafo de fatores usado na representação do código linear do Exemplo 2.1. Os círculos vazios indicam nós de variável e os quadrados cheios indicam nós de verificação de paridade.*

2.5 CÓDIGOS LDPC REPRESENTADOS POR GRAFO DE FATORES

Os códigos LDPC, em razão da baixa densidade de sua matriz de verificação de paridade, **H**, podem ser representados adequadamente usando grafos de fatores [16], [13], [54]. A seguir, são apresentados dois exemplos que demonstram este fato.

Exemplo 2.2

Seja um código LDPC (N, K) *regular de comprimento* $N = 12$ *com a seguinte matriz de verificação de paridade, na qual* $w_c = 3$ *e* $w_r = 4$*.*

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

Este código, com $(N, K) = (12, 3)$, com taxa $R = 1 - 3/4 = 1/4$ e densidade $r = 4/12 = 1/3$, pode ser representado pelo grafo de fatores que está mostrado na Figura 2.4.

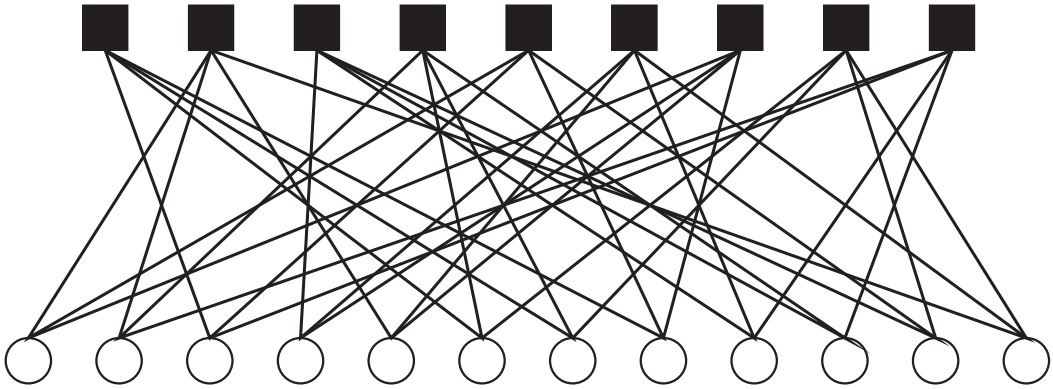


Figura 2.4: *Grafo de fatores para o código LDPC* $(12, 3)$ *regular do Exemplo 2.2. Os círculos vazios indicam nós de variável e os quadrados cheios indicam nós de verificação de paridade.*

Na Figura 2.4, os nós de verificação de paridade são representados por quadrados cheios e os nós de variável por círculos vazios.

Exemplo 2.3

Seja um código LDPC (N, K) *irregular de comprimento* $N = 12$*, definido pela matriz de verificação de paridade*

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

e pelo par de funções de distribuição de grau

$\boldsymbol{\nu} = \{\nu_2\ \nu_3\} = \{1/2\ 1/2\}$ *e*

$\mathbf{h} = \{h_3\ h_4\} = \{2/3\ 1/3\}$. □

Este código pode ser representado pelo grafo de fatores que está mostrado na Figura 2.5.

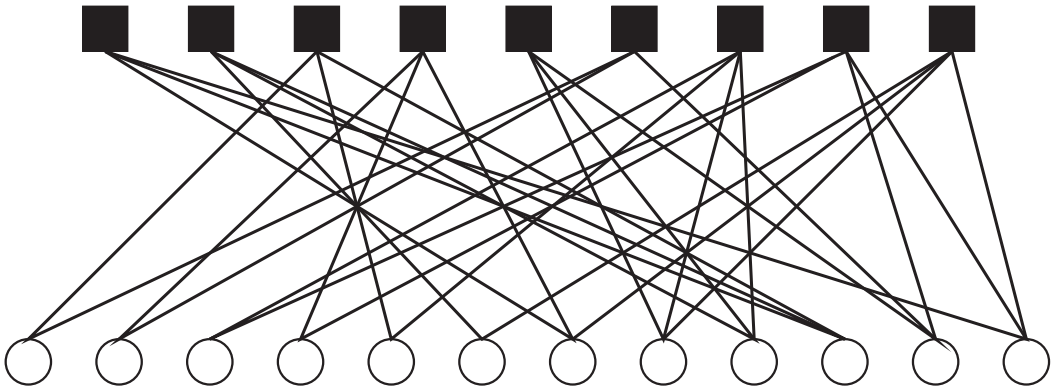


Figura 2.5: *Grafo de fatores para o código LDPC* $(12, 3)$ *irregular, do Exemplo 2.3.Os círculos vazios indicam nós de variável e os quadrados cheios indicam nós de verificação de paridade.*

Neste exemplo, a matriz $\mathbf{H}$ do código possui seis colunas de peso dois, seis colunas de peso três, seis linhas de peso três e três linhas de peso quatro, o que implica densidade $r = 30/108 \approx 0,2777$. Do mesmo modo, a taxa deste código pode ser calculada por

$$R \approx 1 - \frac{\sum_i \nu_i \cdot i}{\sum_j h_j \cdot j} = 1/4.$$

CAPÍTULO 3

DECODIFICAÇÃO ITERATIVA DE ERROS PARA CÓDIGOS LDPC

NA decodificação iterativa de um código linear existem duas possíveis abordagens: a decodificação por decisão abrupta e a decodificação por decisão suave. Na decodificação por decisão abrupta, primeiramente a saída do demodulador é quantizada em um número de valores igual à cardinalidade do alfabeto do código em uso, antes de alimentá-lo na entrada do decodificador, ou seja, no caso de um código binário haveria apenas os símbolos 0 e 1. Na decodificação por decisão suave não obstante os valores serem contínuos e reais, por razões práticas, a saída do demodulador é quantizada em um número de valores maior que a cardinalidade do alfabeto do código em uso, valores estes relacionados com as probabilidades dos símbolos transmitidos condicionadas ao sinal recebido. Os algoritmos usados em decodificação por decisão abrupta são, na maioria das vezes, menos complexos e, portanto, menos exigentes computacionalmente. Por outro lado, os algoritmos usados em decodificação por decisão suave apresentam um melhor desempenho em termos de menor taxa de erros, quer seja de BER, quer seja de WER.

O primeiro algoritmo de decodificação iterativa do tipo por decisão suave foi proposto por Gallager [3], [4]. Esse tipo é conhecido por Algorimo Soma-Produto (*Sum-Product Algorithm* — SPA) e o seu campo de aplicação vai além da decodificação de códigos, abrangendo as áreas de processamento de sinais, comunicações digitais e inteligência artificial, em que é conhecido por *Belief Propagation* (BP) [34].

O algoritmo BP opera sobre o grafo de Tanner de um código linear binário e funciona como um algoritmo de transferência de mensagens entre os nós, sendo considerado de máxima verossimilhança (*Maximum Likelihood* — ML) quando aplicado a códigos descritos por grafos de Tanner sem ciclos [5], [48]. No entanto, quando aplicado a códigos cujos grafos de Tanner possuem ciclos, o algoritmo deixa de ser ótimo, na medida em que estes ciclos introduzem uma realimentação positiva nas mensagens enviadas entre os nós. A menor dimensão destes ciclos, conforme a Definição 2.6, é designada ciclo mínimo ou *girth*, e é denotada por g. As mensagens enviadas ao longo das primeiras $g/2$ iterações são independentes [50], passando a haver correlação entre elas nas iterações seguintes. Este é o motivo pelo qual na construção dos códigos LDPC se procura maximizar o valor do *girth* [55]. Mesmo em algoritmos BP descritos a partir de grafos de Tanner com ciclos, a decodificação de códigos LDPC apresenta um excelente desempenho, tendo Chung et al. [9] demonstrado ser possível se aproximar (a menos de $0,0045$ dB) da capacidade de um canal do tipo RAGB.

3.1 INTRODUÇÃO

O algoritmo BP opera pela troca de mensagens nos ramos de um grafo de fatores, $\mathcal{G}$, o qual é constituído por dois grupos de nós distintos: nós de variável (ou de *bit*) e nós de verificação de paridade (ou de função). Os ramos do grafo conectam nós de variável a nós de verificação de paridade. Quando um ramo conecta um nó de variável, x_i, $1 \leq i \leq N$, a um nó de verificação de paridade f_j, $1 \leq j \leq M$, as mensagens (ou probabilidades) podem ser enviadas em ambas as direções ($x_i \rightarrow f_j$ ou $f_j \rightarrow x_i$). As mensagens de nós de variável para nós de verificação de paridade são denotadas por q_{ij} e as mensagens de nós de verificação de paridade para nós de variável são denotadas por r_{ji}.

A decodificação símbolo-a-símbolo realizada pelo algoritmo BP é baseada na estimativa do provável valor para o símbolo x_i (informação enviada), a partir de um conjunto de valores observados para $\mathbf{y} = (y_1, y_2, \ldots, y_N)$ (informação recebida), que maximiza a função distribuição de probabilidade *a posteriori* $P(x_i|\mathbf{y})$. Sabe-se que $P(x_i|\mathbf{y})$ é uma função marginal da função distribuição de probabilidade $P(\mathbf{x}|\mathbf{y})$ [47], [49]. O vetor $\mathbf{y}$ assume valores em $\mathbb{R}^n$ e é constante durante todo o processo de decodificação de uma N-upla.

Inicialmente, os nós de variável do grafo, x_i, recebem as mensagens do canal, referentes à probabilidade $p(y_i|x_i)$, que são representadas por $p_i(x_i)$, e as difundem igualmente para os nós de verificação de paridade adjacentes (mensagens de nós de variável para nós de verificação de paridade, q_{ij}). Em seguida, cada um dos nós de verificação de paridade adjacentes envia ao nó de variável x_i

uma nova mensagem que leva em conta exclusivamente as estimativas sobre x_i recebidas de outros nós de variável (mensagens de nós de verificação de paridade para nós de variável, r_{ji}). Com o recebimento das mensagens pelos nós de variável, termina uma iteração. Esse processo se repete com a transferência de mensagens dos nós de variável para os nós de verificação de paridade e vice-versa, até se atingir um determinado número preestabelecido de iterações ou até que a estimativa final, $\hat{\mathbf{x}}$, satisfaça o teste da síndrome $\hat{\mathbf{x}}\mathbf{H}^T = \mathbf{0}$.

3.2 MARGINALIZAÇÃO DE FUNÇÃO

O algoritmo BP, como mencionado, emprega a estrutura do grafo de fatores, $\mathcal{G}$, para fatorar uma determinada função global, gerando as funções marginais. Quando o grafo de fatores não possui ciclos, o algoritmo calcula funções marginais exatas [48].

A marginalização de funções é um processo que geralmente possui um custo computacional elevado e cuja complexidade cresce exponencialmente com o número de variáveis envolvidas, N [56]. A função global, no caso, é a função de distribuição de probabilidade *a posteriori* $P(\mathbf{x}|\mathbf{y})$ e as funções marginais correspondem às funções de distribuição de probabilidade *a posteriori* marginais $P(x_i|\mathbf{y})$, para $1 \leq i \leq N$, que são usadas para a estimativa dos valores dos *bits* que compõem a informação enviada $\mathbf{x}$, a partir de um conjunto de valores observados da informação recebida, $\mathbf{y}$ [47], [49].

Pode-se representar a função global $P(\mathbf{x}|\mathbf{y})$ por $g(x_1, \ldots, x_N|\mathbf{y})$ e a função distribuição de probabilidade *a posteriori* marginal $P(x_i|\mathbf{y})$ por $g_i(x_i|\mathbf{y})$. Desta forma, é possível expressar a função marginal $g_i(x_i|\mathbf{y})$ como

$$g_i(x_i|\mathbf{y}) \triangleq \sum_{x_1} \cdots \sum_{x_{i-1}} \sum_{x_{i+1}} \cdots \sum_{x_N} g(x_1, \ldots, x_N|\mathbf{y}). \tag{3.1}$$

A Expressão (3.1) corresponde à marginalização de $g(x_1, \ldots, x_N|\mathbf{y})$ em relação à variável x_i e pode ser simplificada levando em conta apenas as variáveis não empregadas na função global. Define-se $\sim x_i$ como a *não-soma* em relação a x_i. Desta forma, a função marginal $g_i(x_i|\mathbf{y})$ passa a ser descrita, de uma forma mais simplificada, por

$$g_i(x_i|\mathbf{y}) \triangleq \sum_{\sim x_i} g(x_1, \ldots, x_N|\mathbf{y}). \tag{3.2}$$

Teorema 3.1

Seja $g(x_1, x_2, \ldots, x_N|\mathbf{y})$ uma função global representada por um grafo de fatores finito e sem ciclos, $\mathcal{G}$. A função marginal para x_i, $g_i(x_i|\mathbf{y})$, calculada segundo o algoritmo BP, é dada por $g_i(x_i|\mathbf{y}) = \prod_{j\in\mathcal{M}(i)} r_{ji}(x_i)$ e é exatamente a função marginalizada expressa em (3.2). □

Por definição, $\mathcal{M}(i)$ representa o conjunto dos índices dos fatores (nós de verificação de paridade) dos quais a i-ésima variável é argumento. O Teorema 3.1 garante a convergência do algoritmo BP quando o grafo de fatores não possui ciclos [48], [57].

3.2.1 NOTAÇÃO

Seja a função global $g(x_1, \ldots, x_N|\mathbf{y}) = \prod_{j\in\mathcal{M}(i)} f_j(\mathbf{x}_j)$ cuja estrutura de fatoração é representada por um grafo de fatores $\mathcal{G}$. Cada um dos fatores $f_j(\mathbf{x}_j)$ é uma função de um subconjunto $\mathbf{x}_j$ das variáveis que compõem $\mathbf{x} = \{x_1, \ldots, x_N\}$.

O conjunto $\mathcal{N}(j)$ é composto pelos índices das variáveis em $\mathbf{x}_j$ e $\mathcal{M}(i)$ representa o conjunto dos índices dos fatores dos quais a i-ésima variável é argumento, como já definido anteriormente. O conjunto $\mathcal{N}(j)\backslash i$ denota o conjunto $\mathcal{N}(j)$ excluindo o elemento i e $\mathcal{M}(i)\backslash j$ denota o conjunto $\mathcal{M}(i)$ excluindo o elemento j.

Em um grafo de Tanner, os nós de variável, x_i, e os nós de verificação de paridade, f_j, trocam mensagens cuja notação adotada tem a seguinte forma [54]:

$q_{ij}(x_i)$– mensagem do nó de variável x_i para o nó de verificação de paridade f_j,

$r_{ji}(x_i)$– mensagem do nó de verificação de paridade f_j para o nó de variável x_i.

Essas mensagens, ou probabilidades, são funções de um único argumento (a variável x_i), quer circulem em um sentido quer no outro. A mensagem de x_i para f_j, i.e., $q_{ij}(x_i)$, representa a probabilidade de x_i ter um certo valor, dado o valor da estimativa de probabilidade recebida do canal (a quantidade $p_i(x_i)$) e dados os valores que recebeu dos outros nós de verificação de paridade, exceto f_j, que é denotado por $\sim \{f_j\}$. Se x_i for binária então

$$q_{ij}(x_i) = \begin{bmatrix} q_{ij}(x_i = 0) \\ q_{ij}(x_i = 1) \end{bmatrix} = \begin{bmatrix} p(x_i = 0| \sim \{f_j\}, \mathbf{y}) \\ p(x_i = 1| \sim \{f_j\}, \mathbf{y}) \end{bmatrix}.$$

Da mesma forma, a mensagem de f_j para x_i, i.e., $r_{ji}(x_i)$, é a probabilidade de x_i ter um certo valor e a equação de verificação de paridade f_j ser satisfeita, dado que se recebeu $\mathbf{y}$. Se x_i for binária então

$$r_{ji}(x_i) = \begin{bmatrix} r_{ji}(0) \\ r_{ji}(1) \end{bmatrix} = \begin{bmatrix} p(x_i = 0, f_j(\mathbf{x}_j) = 1|\mathbf{y}) \\ p(x_i = 1, f_j(\mathbf{x}_j) = 1|\mathbf{y}) \end{bmatrix}.$$

3.3 FLUXO DE MENSAGENS NO ALGORITMO BP

O algoritmo BP é definido em termos de uma regra de iniciação, duas regras de atualização (uma para cada tipo de nó) e uma regra de terminação. Essas regras de atualização são definidas de modo a satisfazer o princípio da informação extrínseca, introduzido junto com os códigos Turbo e que é descrito na Regra 3.1 [47], [49], [54].

Regra 3.1

Uma mensagem enviada de um nó (vértice) v_i por meio de um ramo e_k não pode depender de nenhuma mensagem previamente recebida pelo ramo e_k. □

No algoritmo BP, uma mensagem de saída de qualquer nó v_i só pode ser calculada e transmitida pelo ramo e_k assim que todas as mensagens, exceto a que chega pelo ramo e_k, chegarem ao nó v_i. Isto ocorre para que não seja violado o princípio da informação extrínseca. A passagem de mensagens ocorre em tempo discreto, de acordo com um cronograma que determina quais são os ramos que estão ativos e em que direções as mensagens são transmitidas. Uma forma tradicional é o cronograma invasivo (*flooding scheduling*) [17], [18], que ativa cada ramo nas duas direções a cada iteração do algoritmo. Esse é o cronograma padrão para decodificação de códigos LDPC e é definido como

Definição 3.1

No cronograma invasivo, cada iteração do algoritmo BP possui apenas dois passos. No primeiro passo, todos os nós de variável passam as mensagens para os nós de verificação de paridade. Em seguida, todos os nós de verificação de paridade passam as mensagens para os nós de variável.□

A terminação do algoritmo BP em um grafo de fatores subótimo (com ciclos) ocorre quando o número máximo de iterações é atingido ou quando o algoritmo resulta em decisões de símbolos que formam uma palavra-código.

As etapas do algoritmo BP, para um código binário, são as seguintes:

a. Iniciação - O algoritmo BP inicia pela passagem das mensagens $p_i(x_i)$ para seus respectivos nós de variável adjacentes, x_i. Então, todos os nós de variável passam para seus vizinhos a mensagem $q_{ij}(0) = p(y_i|x_i = 0)$ e $q_{ij}(1) = p(y_i|x_i = 1)$, que são probabilidades calculadas a partir do modelo matemático do canal. Assim, ao se enviar uma sequência binária $x_i = \pm 1$ ($+1$ equivalendo ao *bit* 0 e -1 ao *bit* 1) por um canal com RAGB, com ruído de média nula e

variância σ^2, e receber-se a sequência de valores reais y_i, estes valores de probabilidade valem

$$q_{ij}(0) = p(y_i|x_i = +1) = \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) \cdot \exp\left[-\frac{(y_i - 1)^2}{2\sigma^2}\right]$$

e

$$q_{ij}(1) = p(y_i|x_i = -1) = \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) \cdot \exp\left[-\frac{(y_i + 1)^2}{2\sigma^2}\right].$$

b. Atualização para nós de verificação de paridade - Cada nó de verificação de paridade, f_j, é representado por uma função indicadora que assume apenas dois valores, 0 (condição não-satisfeita) ou 1 (condição satisfeita), ou seja, $f_j(\mathbf{x}_j) = \delta\left[\sum^{\boxplus}_{i\in\mathcal{N}(j)} x_i\right]$, em que $\sum^{\boxplus}$ corresponde à adição ou-exclusivo. Desta forma, considerando o caso de variáveis binárias, ou seja, $x_i \in \{0, 1\}$, as mensagens $r_{ji}(x_i)$ são atualizadas do seguinte modo

$$r_{ji}(0) = \frac{1}{2}\left\{1 + \prod_{i'\neq i}[1 - 2q_{i'j}(1)]\right\}$$

e

$$r_{ji}(1) = 1 - r_{ji}(0).$$

A Figura 3.1 ilustra a passagem de mensagem entre o nó de verificação de paridade f_2 e o nó de variável x_6, r_{26}, na decodificação de um código de comprimento $N = 6$, com $M = N - K = 3$ equações de verificação de paridade.

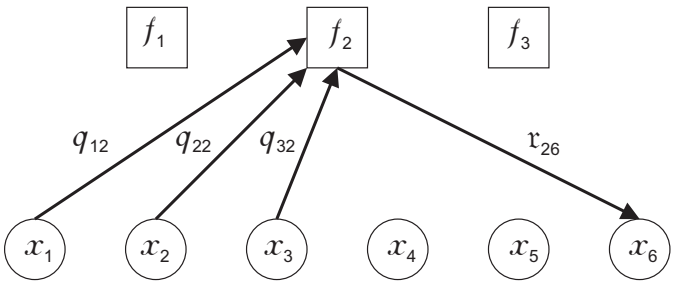


Figura 3.1: *Passagem de mensagem do nó de verificação de paridade f_2 para o nó de variável x_6.*

c. Atualização para nós de variável - Após a passagem das mensagens dos nós de verificação de paridade para os nós de variável, as mensagens $q_{ij}(x_i)$, considerando o caso de variáveis binárias, ou seja, $x_i \in \{0, 1\}$, são atualizadas do seguinte modo

$$q_{ij}(0) = \alpha_{ij} p(y_i|x_i = 0) \prod_{j'\neq j} r_{j'i}(0)$$

e

$$q_{ij}(1) = \alpha_{ij} p(y_i|x_i = 1) \prod_{j'\neq j} r_{j'i}(1),$$

em que α_{ij} é um fator de normalização com valor tal que $q_{ij}(0) + q_{ij}(1) = 1$.

A Figura 3.2 ilustra a passagem de mensagem entre o nó de variável x_3 e o nó de verificação de paridade f_3, q_{33}, na decodificação de um código de comprimento $N = 6$, com $M = N - K = 3$ equações de verificação de paridade.

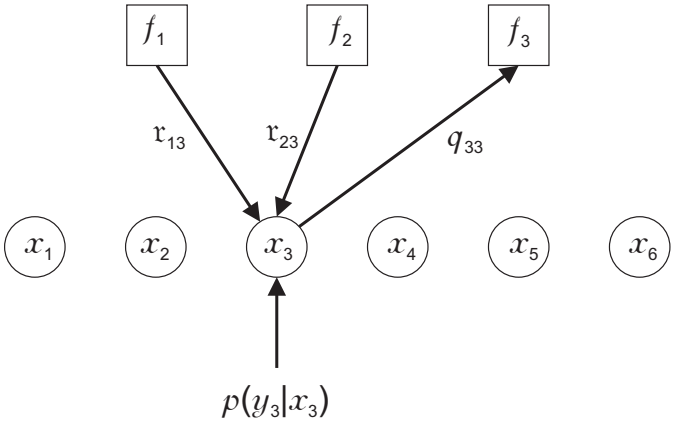


Figura 3.2: *Passagem de mensagem do nó de variável x_3 para o nó de verificação de paridade f_3.*

d. Terminação - A cada iteração são calculadas as probabilidades *a posteriori*, $p(x_i|\mathbf{y})$, para cada nó de variável x_i, com base no produto normalizado de todas as mensagens recebidas dos nós de verificação de paridade, incluindo a mensagem recebida do canal, na forma

$$q_i(x_i) = p(x_i|\mathbf{y}) = \alpha_i \prod_j r_{ji}(x_i).$$

O fator de normalização α_i, para o sistema binário, é arbitrado de modo que $q_i(0) + q_i(1) = 1$. A partir de $q_i(1)$, $1 \leq i \leq N$, o decodificador estima o valor mais provável para cada símbolo transmitido, $\hat{x}_i$, ou seja

$$\hat{x}_i \triangleq \begin{cases} 1, & \text{se } q_i(1) \geq 0,50; \\ 0, & \text{caso contrário.} \end{cases}$$

Se o vetor estimado $\hat{\mathbf{x}}$ satisfizer a condição $\hat{\mathbf{x}}\mathbf{H}^T = \mathbf{0}$, o algoritmo é terminado e retorna o vetor $\hat{\mathbf{x}}$ como a palavra-código decodificada. Caso contrário, o algoritmo repete os passos $\mathbf{b}$ e $\mathbf{c}$ até que o número máximo predefinido de iterações seja atingido. Neste último caso, ocorre uma falha de decodificação.

3.4 A REGRA DA TANGENTE HIPERBÓLICA

Quando as variáveis são binárias, em geral é mais prático e usual fazer circular mensagens que representem razões de probabilidade ou logaritmos da razão de verossimilhança.

Definição 3.2

Seja X uma variável aleatória binária e $p_X(x)$ a probabilidade de X assumir o valor x, sendo $x \in \{0,1\}$. Define-se o LLR como $L(X) \triangleq \ln\left(\frac{p_X(0)}{p_X(1)}\right)$ [36]. □

O sinal do logaritmo da razão de verossimilhança $L(X)$ denota o valor mais provável que X pode assumir e é a partir dele que é realizada a decisão abrupta. A magnitude $|L(X)|$ denota a confiabilidade desta decisão. Se a variável aleatória binária X for condicionada a uma variável aleatória discreta Y, então

$$L(X|Y) = \ln\left(\frac{p_{X|Y}(0|y)}{p_{X|Y}(1|y)}\right) = \ln\left(\frac{p_X(0)}{p_X(1)}\right) + \ln\left(\frac{p_{Y|X}(y|0)}{p_{Y|X}(y|1)}\right) = L(X) + L(Y|X).$$

Se os valores assumidos pela variável aleatória X forem equiprováveis, então $L(X) = 0$, o que implica $L(X|Y) = L(Y|X)$.

É possível expressar a distribuição de probabilidade de X em função de $L(X)$, como [13]

$$p_X(0) = \frac{\exp(L(X))}{1+\exp(L(X))}$$

e

$$p_X(1) = \frac{1}{1+\exp(L(X))},$$

o que implica

$$p_X(0) - p_X(1) = \frac{\exp(L(X))-1}{\exp(L(X))+1} = \tanh\left(\frac{L(X)}{2}\right). \tag{3.3}$$

Supondo que X e Y sejam variáveis aleatórias binárias estatisticamente independentes, o LLR da adição módulo 2 (ou-exclusivo) destas variáveis é calculado por

$$p(X \bigoplus Y = 0) = p_{X,Y}(0,0) + p_{X,Y}(1,1) = p_X(0)\cdot p_Y(0) + p_X(1)\cdot p_Y(1)$$

e

$$p(X \bigoplus Y = 1) = p_{X,Y}(0,1) + p_{X,Y}(1,0) = p_X(0)\cdot p_Y(1) + p_X(1)\cdot p_Y(0).$$

Usando o mesmo argumento de (3.3), obtém-se

$$\begin{aligned}\tanh\left(\frac{L(X\bigoplus Y)}{2}\right) = p(X\bigoplus Y = 0) - p(X\bigoplus Y = 1) = \\ (p_X(0)-p_X(1))\cdot(p_Y(0)-p_Y(1)) = \\ \tanh\left(\frac{L(X)}{2}\right)\cdot\tanh\left(\frac{L(Y)}{2}\right).\end{aligned} \tag{3.4}$$

Portanto, tendo por base a Expressão (3.4), obtém-se

$$L(X \bigoplus Y) = 2 \cdot \tanh^{-1}\left(\tanh\left(\frac{L(X)}{2}\right) \cdot \tanh\left(\frac{L(Y)}{2}\right)\right). \tag{3.5}$$

A Fórmula (3.5) é chamada de regra da tangente hiperbólica. Uma simplificação prática segue do fato de que as funções $\tanh(x)$ e $\tanh^{-1}(x)$ são monotonicamente crescentes e possuem simetria ímpar, implicando $\tanh(x) = \text{sign}(x)\tanh(|x|)$ e $\tanh^{-1}(x) = \text{sign}(x)\tanh^{-1}(|x|)$ [47], [54].

Portanto, o sinal e a magnitude de $L(Y \bigoplus X)$ são separáveis, no sentido de que o sinal de $L(X \bigoplus Y)$ depende apenas dos sinais de $L(X)$ e $L(Y)$ e a magnitude $|L(X \bigoplus Y)|$ depende apenas das magnitudes $|L(X)|$ e $|L(Y)|$. Então, a Expressão (3.5) pode ser reescrita como

$$L(X \bigoplus Y) = [\text{sign}(L(X)) \cdot \text{sign}(L(Y))] \cdot 2 \cdot \tanh^{-1}\left(\tanh\left(\frac{|L(X)|}{2}\right) \cdot \tanh\left(\frac{|L(Y)|}{2}\right)\right). \tag{3.6}$$

A regra da tangente hiperbólica é usada pelo algoritmo BP no domínio LLR para calcular as atualizações das mensagens dos nós de verificação de paridade. A mensagem enviada do nó x_i para o nó f_j é denotada por $L(q_{ij})$ e corresponde ao logaritmo da razão de verossimilhança da variável x_i. A mensagem enviada do nó f_j para o nó x_i é denotada por $L(r_{ji})$. Desta forma, as mensagens trocadas são as seguintes:

- Para a mensagem recebida do canal

$$\ln\left(\frac{p(y_i|x_i=0)}{p(y_i|x_i=1)}\right) = \ln\left(\frac{1-p_i}{p_i}\right) = L(p_i),$$

em que $p_i = p(y_i|x_i = 1)$.

- Para a mensagem do nó de verificação de paridade f_j para o nó de variável x_i

$$L(r_{ji}) = \ln\left(\frac{r_{ji}(0)}{r_{ji}(1)}\right) = \ln\left(\frac{1-r_{ji}}{r_{ji}}\right),$$

em que $r_{ji} = r_{ji}(1)$.

- Para a mensagem do nó de variável x_i para o nó de verificação de paridade f_j

$$L(q_{ij}) = \ln\left(\frac{q_{ij}(0)}{q_{ij}(1)}\right) = \ln\left(\frac{1-q_{ij}}{q_{ij}}\right),$$

em que $q_{ij} = q_{ij}(1)$.

O algoritmo BP no domínio LLR é descrito a seguir.

a. Iniciação – No grafo de Tanner, para cada nó de variável x_i, é calculado o valor inicial de LLR, $L(p_i) = \ln(p(y_i|x_i = 0)/p(y_i|x_i = 1))$. Em seguida, todos os nós de variável passam para seus vizinhos a mensagem $L(q_{ij}) = L(p_i)$. Assim, ao se enviar uma sequência binária $x_i = \pm 1$

($+1$ para *bits* 0 e -1 para *bits* 1) por um canal com RAGB de média nula e variância σ^2 e se receber a sequência de valores reais y_i, estes valores de LLR valem

$$L(p_i) = \ln \frac{p(y_i|x_i = +1)}{p(y_i|x_i = -1)} = \ln \frac{((1/\sqrt{2\pi\sigma^2}) \cdot \exp[-(y_i - 1)^2/2\sigma^2])}{((1/\sqrt{2\pi\sigma^2}) \cdot \exp[-(y_i + 1)^2/2\sigma^2])} = \frac{2y_i}{\sigma^2}.$$

Como a variância do ruído na saída de um filtro casado é expressa por $\sigma^2 = \frac{N_o}{2E_c}$, em que $N_o/2$ é a densidade espectral de potência do ruído e $E_c = RE_b$ é a energia de cada *bit* codificado (sendo E_b a energia de cada *bit* antes da codificação e $R = K/N$ a taxa do código), tem-se portanto

$$L(p_i) = 4\frac{E_c}{N_o}y_i = 4\frac{RE_b}{N_o}y_i.$$

b. Atualização para nós de verificação de paridade - O j-ésimo nó de verificação de paridade recebe as mensagens $L(q_{ij})$, em que $i \in \mathcal{N}(j)$, e atualiza as mensagens $L(r_{ji})$ segundo a expressão

$$L(r_{ji}) = 2 \cdot \tanh^{-1}\left(\prod_{i' \in \mathcal{N}(j)\setminus i} \tanh\left(\frac{L(q_{i'j})}{2}\right)\right),$$

ou alternativamente por

$$L(r_{ji}) = \left(\prod_{i' \in \mathcal{N}(j)\setminus i} \text{sign}(L(q_{i'j}))\right) \cdot 2 \cdot \tanh^{-1}\left(\prod_{i' \in \mathcal{N}(j)\setminus i} \tanh\left(\frac{L(|q_{i'j}|)}{2}\right)\right).$$

c. Atualização para nós de variável - O i-ésimo nó de variável recebe as mensagens $L(r_{ji})$, em que $j \in \mathcal{M}(i)$, e atualiza as mensagens $L(q_{ij})$ segundo a expressão

$$L(q_{ij}) = L(p_i) + \sum_{j' \in \mathcal{M}(i)\setminus j} L(r_{j'i}).$$

d. Terminação - O decodificador determina a estimativa de probabilidade *a posteriori* total sobre o símbolo i pela soma das mensagens transmitidas por todos os nós de verificação de paridade a ele conectados por meio da expressão

$$\Lambda_i = L(p_i) + \sum_{j \in \mathcal{M}(i)} L(r_{ji}).$$

Com base no vetor $\mathbf{\Lambda} = (\Lambda_1, \Lambda_2, \ldots, \Lambda_N)$ é gerada a estimativa $\hat{\mathbf{x}} = (\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_N)$ da seguinte forma

$$\hat{x}_i = \begin{cases} 0, & \text{se } \Lambda_i \geq 0; \\ 1, & \text{caso contrário.} \end{cases}$$

Os passos **b** e **c** do algoritmo são repetidos até que uma palavra-código válida seja encontrada, ou seja, até que a estimativa $\hat{\mathbf{x}}$ satisfaça a condição $\hat{\mathbf{x}}\mathbf{H}^{\mathbf{T}} = \mathbf{0}$, ou um número máximo de iterações predefinido seja alcançado. Neste último caso, ocorre uma falha de decodificação.

3.5 O ALGORITMO *min-sum*

Os valores de LLR são usados para simplificar cálculos, especificamente para substituir multiplicações por adições. No entanto, as multiplicações subsistem no cálculo das mensagens dos nós de verificação de paridade. O algoritmo *min-sum* as substitui por multiplicações de sinais, ou seja, por multiplicações com $+1$ ou -1 [41]. Para obter a aproximação, é preciso manipular a regra da tangente hiperbólica escrevendo a Expressão (3.6) na forma

$$L(X \bigoplus Y) = \max(0, L(X) + L(Y)) + \ln(1 + \exp(-|L(X) + L(Y)|)) - \max(L(X), L(Y)) - \ln(1 + \exp(-|L(X) - L(Y)|)). \quad (3.7)$$

É possível mostrar que a Expressão (3.7) é igual a

$$L(X \bigoplus Y) = \text{sign}(L(X)) \cdot \text{sign}(L(Y)) \cdot \min(|L(X)|, |L(Y)|) + \ln(1 + \exp(-|L(X) + L(Y)|)) - \ln(1 + \exp(-|L(X) - L(Y)|)). \quad (3.8)$$

Desprezando os logaritmos, a Expressão (3.8) é aproximada por

$$L(X \bigoplus Y) \approx \text{sign}(L(X)) \cdot \text{sign}(L(Y)) \cdot \min(|L(X)|, |L(Y)|).$$

Se esta aproximação conhecida por *max-log* for utilizada, o algoritmo passa a se chamar *min-sum*. A regra de atualização para os nós de verificação de paridade, neste caso, é dada por

$$L(r_{ji}) \approx \left(\prod_{i' \in \mathcal{N}(j) \setminus i} \text{sign}[L(q_{i'j})] \right) \min_{i' \in \mathcal{N}(j) \setminus i} (|L(q_{i'j})|).$$

A aproximação de uma das equações de atualização resulta em uma degradação de aproximadamente $0,5$ dB no desempenho do algoritmo de decodificação [47]. Apesar disto, o algoritmo *min-sum* é muito utilizado por apresentar um custo computacional abaixo do algoritmo BP baseado na regra da tangente hiperbólica, resultando em uma menor latência do decodificador. O Algoritmo 3.1 sintetiza as etapas de decodificação do algoritmo *min-sum*.

Algoritmo 3.1 – *min-sum*

Passo 1*: Iniciação*

$$L(p_i) = \ln(p(y_i|x_i = 0)/p(y_i|x_i = 1)) \ \triangleleft \quad \textit{Estimativa de LLR inicial}$$

$L(q_{ij}) = L(p_i)$ $\triangleleft$ *Inicia as mensagens de nós de variável para nós de verificação de paridade*

Passo 2*: Atualização dos nós de verificação de paridade*

$$L(r_{ji}) \approx \left(\prod_{i' \in \mathcal{N}(j) \setminus i} sign[L(q_{i'j})] \right) \cdot \min_{i' \in \mathcal{N}(j) \setminus i} (|L(q_{i'j})|)$$

Passo 3*: Atualização para os nós de variável*

$$L(q_{ij}) = L(p_i) + \sum_{j' \in \mathcal{M}(i) \setminus j} L(r_{j'i})$$

Passo 4*: Terminação*

$$\Lambda_i = L(p_i) + \sum_{j \in \mathcal{M}(i)} L(r_{ji}) \ \triangleleft \textit{ Estimação dos LLRs dos bits}$$

Passo 5*: Estimativa dos bits*

$$\hat{x}_i = \begin{cases} 0, & \textit{se } \Lambda_i \geq 0; \\ 1, & \textit{caso contrário.} \end{cases}$$

□

Passo 6*: Teste da síndrome e do número máximo de iterações*

Se $\hat{\mathbf{x}}\mathbf{H}^{\mathbf{T}} = \mathbf{0}$ *declara N-upla válida e vai para* **FIM**.

Se não, se $Iter \geq Iter_{max}$ *declara N-upla não-válida e vai para* **FIM**.

Ou se $Iter < Iter_{max}$ *vai para o Passo* 2.

FIM.

3.6 ALGORITMO *min-sum* MODIFICADO

Os valores de LLR, em geral, são números reais de ponto flutuante. Uma possível estratégia para reduzir o esforço computacional é transformar esses valores em números inteiros de ponto fixo, chamados LLR quantizados (*Quantized Log-Likelihood Ratio* — QLLR) [58], o que torna a computação mais eficiente.

Para a conversão de valores de LLR em QLLR é usada a expressão a seguir:

$$QLLR = \lfloor 2^q \cdot LLR \rfloor.$$

A notação $\lfloor x \rfloor$ indica o maior número inteiro menor que x. O valor q é arbitrado, mas deve ser adotado um que permita uma boa precisão nos cálculos, de forma a ficarem próximos àqueles realizados em ponto flutuante e que não comprometa o esforço computacional.

Além disso, uma outra estratégia é o uso de tabelas de dados pré-construídas (*lookup tables*) [17], que guardam valores inteiros constantes. Uma aplicação dessas tabelas pode ser vista no caso da Expressão (3.9) que é conhecida por Boxplus ($\boxplus$) de Hagenauer [59], que pode ser usada no cálculo das mensagens transferidas dos nós de verificação de paridade para os nós de variável, $L(r_{ji})$, em substituição à Expressão (3.8), tal que

$$\begin{aligned} L(X \bigoplus Y) = \text{sign}(L(X)) \cdot \text{sign}(L(Y)) \cdot \min(|L(X)|, |L(Y)|) + \\ \ln(1 + \exp(-|L(X) + L(Y)|)) - \ln(1 + \exp(-|L(X) - L(Y)|)) = \\ L(X \boxplus Y). \end{aligned} \tag{3.9}$$

O inconveniente da Expressão (3.9) é que ela apresenta uma complexidade computacional maior devido ao cálculo das funções logaritmicas. Neste caso, para simplificar o cálculo, pode-se usar a consulta a tabelas de tamanho predefinido, contendo alguns valores para os logaritmos. A Expressão (3.10) mostra esta nova forma de cálculo

$$L(X \boxplus Y) \approx \text{sign}(a) \cdot \text{sign}(b) \cdot \min(|a|, |b|) + f(|a + b|) - f(|a - b|), \tag{3.10}$$

em que $f(x) = \ln(1 + \exp(-x))$. Os valores de $f(x)$ são previamente calculados (antes da decodificação) e guardados na tabela (não precisam passar por nenhuma atualização), o que possibilita um menor esforço computacional durante a decodificação. Com base na Expressão (3.10) pode-se calcular as mensagens dos nós de verificação de paridade para os nós de variável a partir da expressão

$$L(r_{ji}) = \sum_{i' \in \mathcal{N}(j) \setminus i}^{\boxplus} L(q_{i'j}).$$

A precisão nos cálculos dos valores de $L(r_{ji})$ está relacionada à dimensão e à resolução da tabela, que corresponde ao intervalo entre dois valores sucessivos listados nela. Uma boa prática é considerar uma dimensão igual a 300 e uma resolução na ordem de $1/32$.

O emprego dos recursos de quantização aliado ao uso de consulta a tabelas tornam esse algoritmo rápido, eficiente e com uma boa precisão. O Algoritmo 3.2 apresenta um resumo das etapas de decodificação do algoritmo *min-sum* modificado.

Algoritmo 3.2 – *min-sum* modificado

Passo 1*: Iniciação*

$$L(p_i) = \textit{QLLR}(\ln(p(y_i|x_i = 0)/p(y_i|x_i = 1))) \triangleleft \textit{ Estimativa do LLR inicial}$$

$L(q_{ij}) = L(p_i) \triangleleft$ *Inicia as mensagens de nós de variável para nós de verificação de paridade*

Passo 2*: Atualização dos nós de verificação de paridade*

$$L(r_{ji}) = \boxplus_{i' \in \mathcal{N}(j) \setminus i} L(q_{i'j}) \triangleleft \textit{ Operação BOXPLUS}$$

Passo 3*: Atualização para os nós de variável*

$$L(q_{ij}) = L(p_i) + \sum_{j' \in \mathcal{M}(i) \setminus j} L(r_{j'i})$$

Passo 4*: Terminação*

$$\Lambda_i = L(p_i) + \sum_{j \in \mathcal{M}(i)} L(r_{ji}) \triangleleft \textit{ Estimação dos LLRs dos bits}$$

Passo 5*: Estimativa dos bits*

$$\hat{x}_i = \begin{cases} 0, & \textit{se } \Lambda_i \geq 0; \\ 1, & \textit{caso contrário.} \end{cases}$$

□

Passo 6*: Teste da síndrome e do número máximo de iterações*

Se $\hat{\mathbf{x}}\mathbf{H}^{\mathbf{T}} = \mathbf{0}$ *declara N-upla válida e vai para* **FIM**.

Se não, se $Iter \geq Iter_{max}$ *declara N-upla não-válida e vai para* **FIM**.

Ou se $Iter < Iter_{max}$ *vai para o Passo* 2.

FIM.

3.7 DESEMPENHO DE CÓDIGOS LDPC COM DECODIFICAÇÃO ITERATIVA

As curvas típicas de desempenho para esquemas de decodificação iterativa aplicados a códigos LDPC e a códigos Turbo indicam, a princípio, uma queda acentuada da taxa BER ou WER à medida que se aumenta o valor de SNR em um canal com RAGB [25], [16]. No entanto, a partir de uma certa região da curva, conhecida como patamar de erros e que ocorre para valores maiores de SNR, essa queda não se torna tão acentuada. Essa súbita saturação da curva de desempenho é devido

principalmente ao aparecimento de conjuntos de armadilhas (*trapping sets*) [25]. A quantidade e a estrutura destes conjuntos de armadilhas são fatores que regem o desempenho do algoritmo de decodificação iterativa de erros nessa região.

3.7.1 CONJUNTOS DE ARMADILHAS EM CÓDIGOS LDPC

A relação de conjuntos de armadilhas e a região de patamar de erros de códigos LDPC foi primeiramente observada em [60] e foi amplamente estudada em [25]. O seu aparecimento e suas relações com as propriedades do grafo de Tanner do código e com o algoritmo de decodificação estão detalhadas em [30].

Formalmente, pode-se analisar o aparecimento do conjunto de armadilhas da seguinte maneira:

Seja $\mathcal{Y}$ o conjunto de todas as possíveis entradas do decodificador. Assume-se que o código $\mathcal{C}$ é binário e é representado por um grafo de Tanner com N nós de variável [25]. Um decodificador iterativo é definido como uma sequência de mapas $D^l(\mathbf{y}) : \mathcal{Y} \rightarrow \{0,1\}^N,\ l = 0,1,2,3,4,\ldots,$ em que o índice l corresponde ao número da iteração. Assumindo que o par canal e decodificador é simétrico, pode-se então supor que uma palavra-código toda-nula foi transmitida, sem perda de generalidade, e sabendo que $D^l(\mathbf{y})[i]$ corresponde ao i-ésimo *bit* de saída, diz-se que o i-ésimo *bit* foi corrigido se existe I tal que, para todo $l \geq I$ tem-se $D^l(\mathbf{y})[i] = 0$. Uma decodificação ocorrerá com sucesso se todos os N *bits* estiverem corretos.

Definição 3.3

Para uma dada entrada $\mathbf{y}$*, define-se o conjunto de falhas (failure set)* $T(\mathbf{y})$*, como aquele conjunto de bits que não foram corrigidos ao final da decodificação. Portanto, a decodificação de* $\mathbf{y}$ *terá sido bem sucedida se, e somente se,* $T(\mathbf{y}) = \phi$*. Se* $T(\mathbf{y}) \neq \phi$*, então diz-se que* $T(\mathbf{y})$ *é um conjunto de armadilhas [25].* □

Portanto da Definição 3.3 tem-se

Definição 3.4

Um conjunto de armadilhas $TS(a,b)$ *é um conjunto de falhas* T *formado por uma configuração de* a *nós de variável para os quais o subgrafo induzido em* $\mathcal{G}$ *contém* $b \geq 0$ *nós de verificação de paridade de grau ímpar [25].* □

Um dos principais tipos de conjunto de armadilhas é conhecido por elementar e definido como

Definição 3.5

Um conjunto de armadilhas $TS(a,b)$ é dito elementar se, no conjunto com todos os nós de verificação de paridade no subgrafo induzido tendo grau um ou grau dois, há exatamente b nós de verificação de paridade de grau um [30]. □

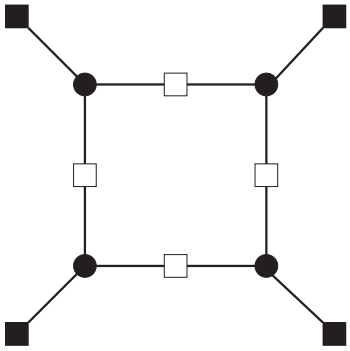


Figura 3.3: *Conjunto de armadilhas $TS(4,4)$ para o código Margulis (2640,1320) do Exemplo 3.1. Os círculos escuros indicam nós de variável em erro, os quadrados claros indicam nós de verificação de paridade satisfeitos e os quadrados escuros indicam nós de verificação de paridade não satisfeitos.*

Vale destacar que, no trabalho inicial sobre conjunto de armadilhas [25], foi observado que somente os conjuntos de pequena dimensão, em geral elementares, contribuem para o surgimento do patamar de erros.

Definição 3.6

Um conjunto de armadilhas $TS(a,b)$ em um código linear $\mathcal{C}$ é dito ser de pequena dimensão se $a \leq \sqrt{N}$ e $b \leq 4a$ [30]. □

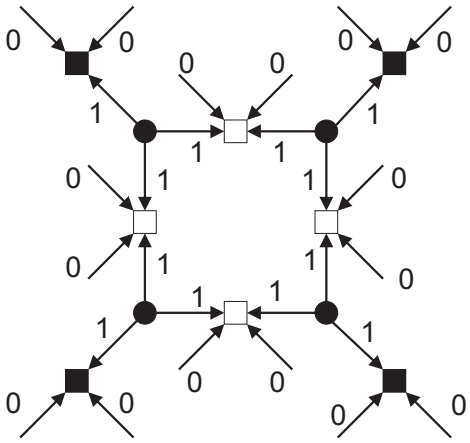
Caso todos os padrões de conjuntos de armadilhas de pequena dimensão sejam conhecidos, é possível estimar o desempenho do código na região de patamar de erros, em termos de WER, pela expressão [25]

$$WER = \sum_{T} \Pr\{\xi_T\},$$

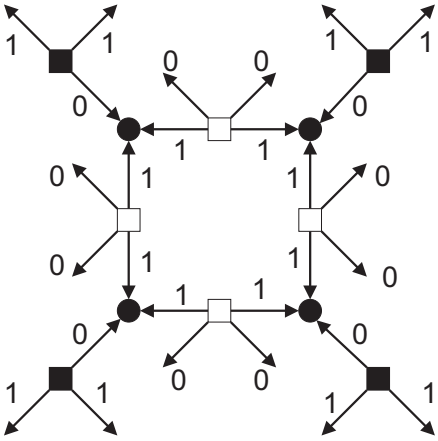
em que $\xi_T \subset \mathcal{Y}$ é o conjunto de todas as possíveis entradas que provocam uma falha de decodificação devido a conjuntos de armadilhas e $\Pr\{.\}$ corresponde à função probabilidade.

A identificação destes padrões de conjuntos de armadilhas, mesmo de pequena dimensão, não é uma tarefa trivial. Em [25], sugere-se, para a sua identificação, a seguinte sequência de simulação: Executa-se um grande número de iterações no decodificador BP (em torno de 200 iterações, aproximadamente), a menos que o decodificador convirja mais cedo para uma N-upla válida. Em seguida, se não convergiu, indicando falha de decodificação, são executadas mais algumas iterações (em torno

de 20 iterações) e identifica-se o conjunto de armadilhas como aquele formado pela união de todos os *bits* que não são decodificados corretamente após as iterações adicionais. A seguir são mostrados dois exemplos de conjuntos de armadilhas constituídos por padrões de erros de pequena dimensão.



(a) Mensagens dos nós de variável para nós de verificação de paridade.



(b) Mensagens dos nós de verificação de paridade para os nós de variável.

Figura 3.4: *Troca de mensagens no conjunto de armadilhas $TS(4,4)$ para o código Margulis (*2640,1320*). Os círculos escuros indicam nós de variável em erro, os quadrados claros indicam nós de verificação de paridade satisfeitos e os quadrados escuros indicam nós de verificação de paridade não satisfeitos.*

Exemplo 3.1

Seja um código Margulis (2640, 1320*) que possui ciclo mínimo igual a oito e grau de nós de variável (peso da coluna da matriz de verificação de paridade) igual a três [61]. Supõe-se que uma palavra-código toda-nula foi transmitida e que é recebido um vetor binário, cujo suporte (número de bits iguais a* 1*) é o conjunto de quatro nós de variável, que formam um ciclo de*

comprimento oito. A Figura 3.3 ilustra o subgrafo induzido por estes quatro nós de variável. Os círculos escuros indicam nós de variável em erro (bits 1*), os quadrados claros indicam nós de verificação de paridade satisfeitos e os quadrados escuros indicam nós de verificação de paridade não satisfeitos.*

Na primeira iteração, os nós de variável passam os valores recebidos do canal aos nós de verificação de paridade vizinhos, o que é mostrado na Figura 3.4(a). Em seguida, as mensagens são passadas pelos nós de verificação de paridade aos nós de variável, o que está mostrado na Figura 3.4(b). A estrutura do código de Margulis é tal que nenhum dos nós de verificação de paridade de grau ímpar (nós de verificação de paridade não satisfeitos, neste exemplo) são conectados a um nó de variável comum contendo erro. Devido a esta propriedade, as mensagens passadas pelos nós de variável na próxima iteração serão as mesmas que foram passadas na iteração anterior. Portanto, as mesmas mensagens serão transferidas em cada iteração. Após o algoritmo ser executado por um número predeterminado de iterações, o vetor decodificado será o mesmo que o vetor recebido. Portanto, um ciclo de comprimento oito para esse código de Margulis é um conjunto de armadilhas $TS(4,4)$*.* □

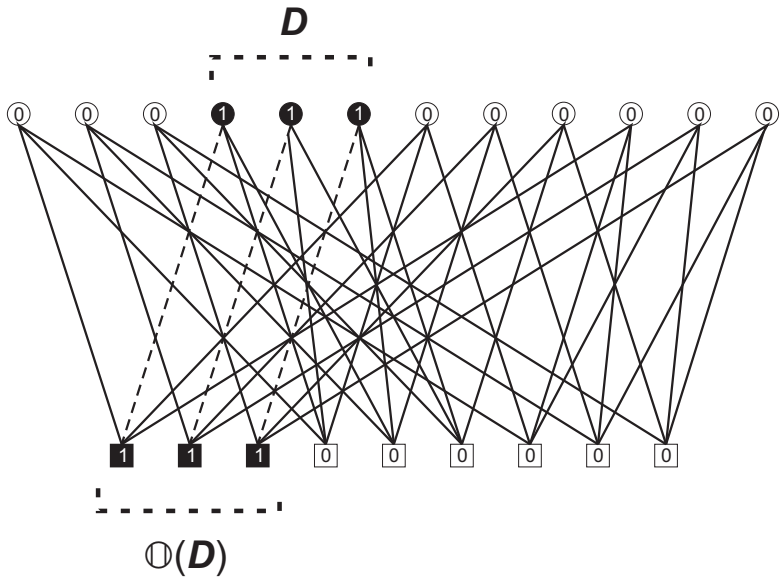


Figura 3.5: *Conjunto de armadilhas* $TS(3,3)$ *do tipo conjunto de absorção do Exemplo 3.2. Os círculos escuros indicam nós de variável em erro, os círculos claros indicam nós de variável sem erro, os quadrados claros indicam nós de verificação de paridade satisfeitos e os quadrados escuros indicam nós de verificação de paridade não satisfeitos. [62]*

Exemplo 3.2

O tipo de conjunto de armadilhas destacado neste exemplo está mostrado na Figura 3.5 e é definido em [62] como um conjunto de absorção (absorbing set).

Definição 3.7

Um conjunto de absorção é um subconjunto $D \in V_b$, sendo D um conjunto de armadilhas e V_b o subconjunto de todos os nós de variável, em que cada um dos seus elementos tem uma quantidade bem menor de vizinhos no subconjunto $\mathbb{O}(D) \in V_c$ do que em $V_c \backslash \mathbb{O}(D)$, sendo que $\mathbb{O}(D)$ é o subconjunto de nós de verificação de paridade vizinhos de D que não são satisfeitos, $V_c \backslash \mathbb{O}(D)$ corresponde aos outros nós de verificação de paridade vizinhos de D e que não pertencem a $\mathbb{O}(D)$ e V_c é o subconjunto de todos os nós de verificação de paridade. □

Para finalizar as considerações sobre conjuntos de armadilhas, é importante também destacar que todo conjunto de armadilhas *TS* é associado a um valor crítico m, que é definido como o número mínimo de nós de variável em *TS* que tem que estar inicialmente em erro para que o resultado da decodificação seja uma falha por conjunto de armadilhas. Quanto menores os valores para m, maiores são as possibilidades de que resulte em falha de decodificação [61], mesmo havendo um número pequeno de erros pertencentes a um específico conjunto de armadilhas.

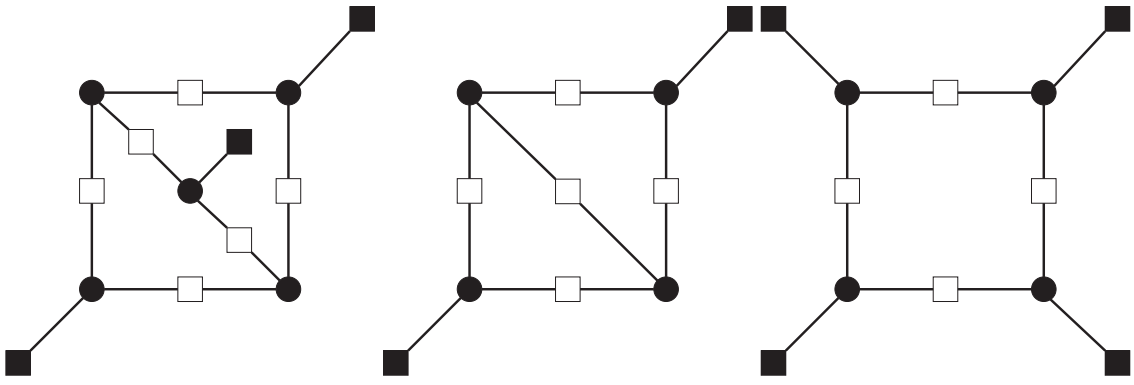


Figura 3.6: *(a) Conjunto de armadilhas $TS(5,3)$; (b) Conjunto de armadilhas $TS(4,2)$; (c) Conjunto de armadilhas $TS(4,4)$. Os círculos escuros indicam nós de variável em erro, os quadrados claros indicam nós de verificação de paridade satisfeitos e os quadrados escuros indicam nós de verificação de paridade não satisfeitos.*

Contudo, nem todas as configurações de m erros em um conjunto de armadilhas resultam em falha de decodificação. Por exemplo, para o conjunto de armadilhas $TS(5,3)$, mostrado na

Figura 3.6(a), somente uma das configurações contendo três erros leva a uma falha de decodificação. Já para o conjunto de armadilhas $TS(4,2)$ que é mostrado na Figura 3.6(b), tendo valor de $m = 3$, todas as quatro possíveis combinações de três erros levam a um conjunto de armadilhas.

Um subconjunto de m nós $\in TS$, que levam a uma falha de decodificação, que redundem em um conjunto de armadilhas, pode também ser classificado como *conjunto de falhas*. O número de *conjuntos de falhas* que contenham apenas m nós de variável em erro é chamado de robustez (*strength*) de *TS* e é denotado por s. Por exemplo, para o conjunto de armadilhas $TS(5,3)$ mostrado na Figura 3.6(a) tem-se $m = 3$ e $s = 1$, para o conjunto de armadilhas $TS(4,2)$ da Figura 3.6(b) tem-se $m = 3$ e $s = 4$ e para o conjunto de armadilhas $TS(4,4)$ mostrado na Figura 3.6(c) tem-se $m = 4$ e $s = 1$.

CAPÍTULO 4

DECODIFICAÇÃO ITERATIVA DE APAGAMENTOS PARA CÓDIGOS LDPC

Os códigos usados em correção de apagamentos e os seus respectivos decodificadores têm recebido renovado interesse em diversas áreas, como, por exemplo, na codificação de pacotes em redes [63]–[65] e em sistemas distribuídos de computação [66]. Para esses casos, o BEC é o modelo de canal mais apropriado, porquanto os *bits* transmitidos podem ser perdidos devido a falhas na rede causadas, entre outras, por atrasos excessivos nos nós intermediários. Um bom exemplo disto são as perdas de pacotes de dados que ocorrem na Internet [67] [68].

4.1 INTRODUÇÃO

O BEC, ilustrado na Figura 4.1, foi introduzido em 1955 por Elias [40]. Esse canal é discreto sem memória tendo dois símbolos de entrada $x_i \in \mathbf{X}$, tal que $\mathbf{X} = \{0, 1\}$, e três símbolos de saída $y_j \in \mathbf{Y}$, tal que $\mathbf{Y} = \{0, 1, \Delta\}$, em que o símbolo Δ representa um apagamento [43]. Os *bits* transmitidos por um canal BEC são recebidos corretamente com probabilidade $P_{\mathbf{Y}|\mathbf{X}}(0|0) = P_{\mathbf{Y}|\mathbf{X}}(1|1) = 1 - \varepsilon$, ou são recebidos como símbolos apagados com probabilidade $P_{\mathbf{Y}|\mathbf{X}}(\Delta|0) = P_{\mathbf{Y}|\mathbf{X}}(\Delta|1) = \varepsilon$, em que ε é a probabilidade de apagamento do BEC e seu valor, portanto, está na faixa $0 \leq \varepsilon \leq 1$. O BEC possui uma capacidade expressa por $\mathbb{C} = 1 - \varepsilon$ [13].

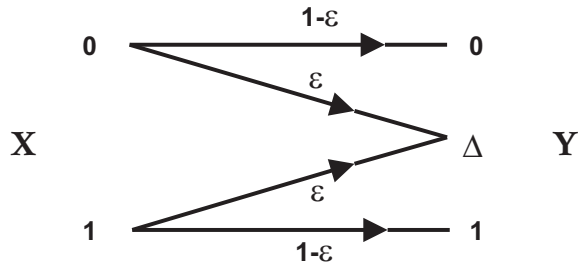


Figura 4.1: *Canal binário com apagamentos (BEC).*

4.2 DESCRIÇÃO DA DECODIFICAÇÃO ITERATIVA DE APAGAMENTOS

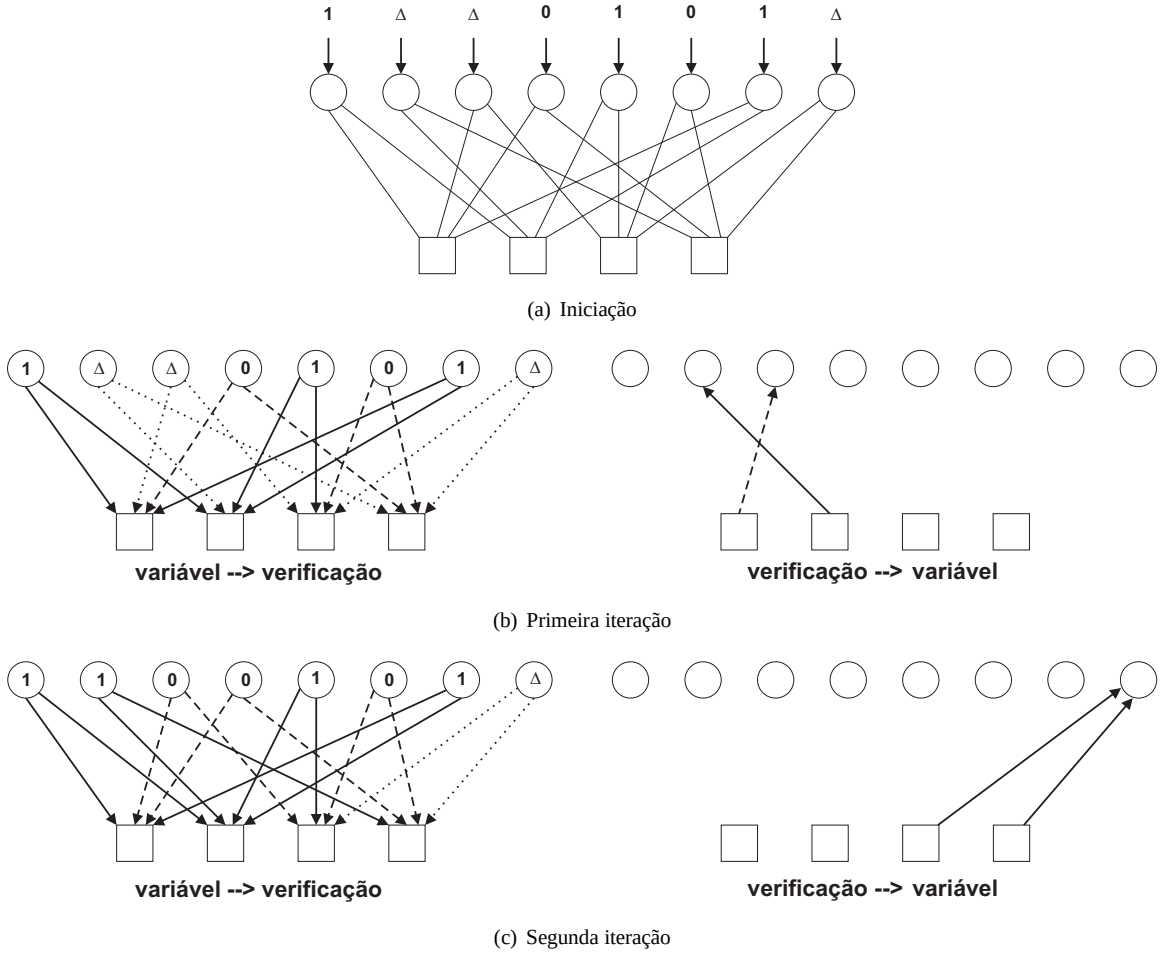
A decodificação iterativa de apagamentos aplicada a um *bit* recebido do BEC é relativamente simples, uma vez que o *bit* ou está correto ou está apagado. A cada iteração, a decodificação iterativa de apagamentos procura por equações de verificação de paridade que tenham apenas um único *bit* apagado para solucioná-las e, deste modo, corrigir o referido *bit*. Caso todos os *bits* apagados sejam corrigidos nessa iteração, o processo é encerrado. Caso contrário, o processo é repetido até que não haja *bits* apagados ou, em caso de falha, que só haja equações de verificação de paridade contendo no mínimo dois *bits* apagados [13] [69].

Em um decodificador iterativo por transferência de mensagens usado para a correção de apagamentos, o primeiro passo consiste na transferência das mensagens dos nós de variável, Ψ, que foram recebidas do canal, para cada um dos nós de verificação de paridade a eles conectados. Cada uma dessas mensagens é designada por Ψ_i, em que i denota o i-ésimo nó de variável, $0 \leq i \leq N$. A mensagem Ψ_i declara o valor do i-ésimo *bit* como 1 ou 0, se for conhecido, ou Δ se for apagado. Caso um nó de verificação de paridade receba somente uma única mensagem Δ, ele calcula o valor desse *bit* desconhecido de modo que a paridade seja satisfeita. Em uma segunda etapa, os nós de verificação de paridade enviam de volta diferentes mensagens a cada um dos nós de variável a eles conectados. Esta mensagem, designada por $r_{j,i}$, correspondendo à mensagem do j-ésimo nó de verificação de paridade, $0 \leq j \leq M$, para o i-ésimo nó de variável, declara o valor desse *bit* como 1, 0 ou Δ. Caso um nó de variável, originalmente correspondendo a um apagamento, receba uma mensagem igual a 1 ou a 0, automaticamente o seu valor será modificado para um desses valores recebidos. Esse processo de transferência de mensagens é repetido até que todos os *bits* apagados sejam decodificados com sucesso ou até que não haja equações de verificação de paridade que contenham apenas um único apagamento e, nesse caso, é declarada uma falha de decodificação.

Esse procedimento usado na decodificação iterativa de apagamentos está resumido no

Algoritmo 4.1 [13]. A notação empregada no Algoritmo 4.1 é a seguinte:

- B_j é o subconjunto dos índices dos nós de variável que são vizinhos ao j-ésimo nó de verificação de paridade.
- A_i é o subconjunto dos índices dos nós de verificação de paridade que são vizinhos ao i-ésimo nó de variável.



(a) Iniciação

(b) Primeira iteração

(c) Segunda iteração

Figura 4.2: *Exemplo de decodificação iterativa de apagamentos para um código de bloco linear. Os círculos representam nós de variável e os quadrados representam nós de verificação de paridade.*

A Figura 4.2 ilustra um exemplo dos passos da decodificação iterativa de apagamentos para um código de bloco linear de comprimento $N = 8$ e com $M = 4$ equações de verificação de paridade, representadas no grafo por nós de variável e nós de verificação de paridade, respectivamente. Nesse exemplo, o código gerou uma palavra-código $\mathbf{c} = (1\ 1\ 0\ 0\ 1\ 0\ 1\ 1)$ e a enviou por um canal BEC, com probabilidade de apagamentos ε. No destino, é recebida a sequência $\mathbf{y} = (1\ \Delta\ \Delta\ 0\ 1\ 0\ 1\ \Delta)$, contendo três posições de *bits* apagadas. A Figura 4.2(a) mostra a iniciação, a Figura 4.2(b) a primeira

iteração e a Figura 4.2(c) a segunda e última iteração. As linhas tracejadas indicam mensagem 0, as linhas sólidas indicam mensagem 1 e as linhas pontilhadas representam mensagem Δ (apagamento).

Algoritmo 4.1 – Decodificação Iterativa de Apagamentos [13]

Passo 1*: Iniciação das mensagens dos nós de variável.*

Executa para $i = 1 : N$

$\Psi_i = y_i$*. Sendo que* $y_i = 0,\ 1$ *ou* Δ*.*

Passo 2*: Testa todas as equações de verificação de paridade.*

flag $= 0$*.* $\triangleleft$ *Indica que nenhum bit foi corrigido.*

Executa para $j = 1 : M$

Executa para todo $i \in B_j$

Se todas as mensagens recebidas pelo nó de verificação paridade j *exceto* Ψ_i *são conhecidas,*

então executa

$$r_{j,i} = \sum_{i' \in B_j, i' \neq i} (\Psi'_i \mod 2).$$

Se não

$$r_{j,i} = \Delta. \qquad \square$$

Passo 3*: Atualiza os bits*

Executa para $i = 1 : N$

Se $\Psi_i = \Delta$ *então*

Se $r_{j,i} \neq \Delta$*, em que* $j \in A_i$*, então*

$\Psi_i = r_{j,i}$ *e*

flag $= 1$*.*

Passo 4*: Terminação*

Se todos Ψ_i *são conhecidos* ***OU*** *flag* $= 0$ *então*

vai para ***FIM****.*

Se não

vai para ***Passo 2****.*

FIM.

O primeiro passo da decodificação iterativa é mostrado na Figura 4.2(a) e corresponde à iniciação dos nós de variável com os mesmos valores recebidos do canal, ou seja, $\Psi_i = y_i$. Desta forma,

$\Psi = (1\ \Delta\ \Delta\ 0\ 1\ 0\ 1\ \Delta)$. Ao final da primeira iteração, mostrada na Figura 4.2(b), são corrigidos os *bits* nas posições 2 e 3 gerando o resultado parcial $\Psi = (1\ 1\ 0\ 0\ 1\ 0\ 1\ \Delta)$. Como há um apagamento restante no oitavo *bit*, o algoritmo continua e, na segunda iteração, mostrada na Figura 4.2(c), esse último *bit* é corrigido resultando na N-upla recuperada $\Psi = (1\ 1\ 0\ 0\ 1\ 0\ 1\ 1)$ como a palavra-código decodificada. Nesse exemplo, observa-se que, apesar da N-upla ter sido sido recebida com quase a metade dos seus *bits* apagados, o decodificador iterativo de apagamentos, em apenas duas iterações, consegue recuperar a palavra-código enviada.

4.3 DESEMPENHO DA DECODIFICAÇÃO DE APAGAMENTOS POR MÁXIMA VEROSSIMILHANÇA

Seja $C(N, K)$ um código binário corretor de erros de comprimento N, com dimensão K e distância mínima de Hamming [38], [39], d_{min}, que possui taxa igual a K/N e é definido por uma matriz geradora binária $G = [g_{ij}]$, $1 \leq i \leq K$ e $1 \leq j \leq N$. O código C é capaz de corrigir todos os padrões de apagamentos contendo não mais do que δ apagamentos, se $\delta \leq d_{\min} - 1$ [70]–[73]. Seja $\mathbf{u} = (u_i)$, $1 \leq i \leq K$, uma sequência de informação binária e seja $\mathbf{c} = (c_j)$, $1 \leq j \leq N$, a palavra-código gerada a partir da equação $\mathbf{c} = \mathbf{u}G$. A palavra-código $\mathbf{c} = (c_1, c_2, \ldots, c_N)$ é transmitida por um canal binário com apagamento. É recebida a N-upla $\mathbf{y} = (y_1, y_2, \ldots, y_N)$ contendo um padrão com δ apagamentos. Seja $I = \{i_1, i_2, \ldots, i_\delta\}$, o conjunto das δ posições apagadas. A decodificação por máxima verossimilhança (*Maximum Likelihood* – ML) é realizada mediante um teste para cada um dos 2^δ diferentes substitutos para o padrão de δ apagamentos em $\mathbf{y}$ e a verificação, após cada substituição, se a N-upla resultante é uma palavra-código. A decodificação é considerada um sucesso se somente um único substituto, dentre os 2^δ substitutos, produzir uma palavra-código. Se os substitutos binários, para o padrão de δ-apagamentos, produzirem mais do que uma palavra-código, então a decodificação é considerada falha devido à ambiguidade.

Definição 4.1

O conjunto I é definido como um conjunto de cobertura (covering set) se, e somente se, $\sup(\mathbf{c}) \subseteq I$ *para pelo menos uma palavra-código não-nula* $\mathbf{c} \in C$*, em que* $\sup(\mathbf{c}) = \{j | c_j = 1\}$ *é o suporte da palavra-código [43].* □

Teorema 4.1

Um dado código de bloco linear $C(N, K)$ corrige um padrão de δ apagamentos com coordenadas mapeadas em I se, e somente se, I não for um conjunto de cobertura em C [43]. □

Teorema 4.2

Qualquer padrão de δ-apagamentos para $\delta \geq N-K+1$, cobre pelo menos uma palavra-código de um código linear $C(N,K)$. □

O Teorema 4.2 deriva do Teorema 4.1 e do fato da capacidade do BEC, $\mathbb{C}$, ser maior ou no mínimo igual à taxa do código, R, i.e, $\mathbb{C} = 1-\varepsilon \geq R = K/N$, em que K representa a dimensão do código e ε a probabilidade de apagamento do canal. Desse modo, resulta que

$$\varepsilon \leq \frac{N-K}{N}. \tag{4.1}$$

A Expressão (4.1) implica que padrões corrigíveis de δ-apagamentos deverão conter no máximo $N-K$ apagamentos [67]. Portanto, mesmo para padrões de apagamentos com um número acima de $d_{min}-1$ e inferior a $N-K$ apagamentos, o código $C(N,K)$ pode corrigi-los usando a técnica de decodificação ML, desde que não constituam *conjuntos de cobertura*. Para ilustrar, a seguir é apresentado o Exemplo 4.1 extraído de [43] em que é analisado um caso de conjunto de padrões de apagamentos com cardinalidade maior do que $d_{min}-1$.

Exemplo 4.1

Seja o código binário $C(127,120)$ com $d_{min}=3$. Sabe-se que o número de palavras-código de peso 3 deste código é igual a 2667 e que o número total de padrões de apagamentos contendo 3 apagamentos, em um bloco de comprimento igual a 127 dígitos, é igual a 333375. Consequentemente, pela definição de conjunto de cobertura, o número de padrões com três apagamentos que podem ser corrigidos é $333375-2667=330708$. Desta forma, conclui-se que, para este código, $99,2\%$ dos padrões com três apagamentos são corrigíveis. □

Com base em [43], é possível também estimar a probabilidade de decodificação ambígua, Π, que ocorre quando a decodificação ML gera mais do que um substituto para o padrão de δ apagamentos, em função da distribuição de pesos de Hamming de suas palavras-código, como

$$\Pi = \sum_{t=d_{min}}^{N} \alpha_t \beta(t), \tag{4.2}$$

em que $\beta(t) = \begin{pmatrix} N \\ t \end{pmatrix} \varepsilon^t (1-\varepsilon)^{(N-t)}$ e α_t é a fração de padrões de apagamento de peso t que produzem decodificação ambígua. A probabilidade Π depende essencialmente do conjunto de valores $\{\alpha_{d_{min}}, \alpha_{d_{min}+1}, \ldots, \alpha_{(N-K)}\}$, uma vez que $\alpha_t = 0$ para $t \leq d_{min}-1$ e $\alpha_t = 1$ para $t > N-K$. Os valores de α_t e Π podem ser estimados para um dado código $C(N,K)$ em

função da distribuição de pesos das suas palavras-código $W = \{w_0, w_{d_{min}}, \ldots, w_N\}$, em que w_i, $d_{min} \leq i \leq N$, representa o número de palavras-código de peso i e $w_0 = 1$.

É possível demonstrar também que para um dado código linear binário, o desempenho no BEC depende essencialmente da quantidade de palavras-código de menor peso de Hamming [74]. A análise é baseada no polinômio enumerador de pesos do código [71], [13], [39], [16], que é dado por

$$W(D) = \sum_{j=0}^{N} A_j D^j,$$

em que A_j representa o número de palavras-código com peso igual a j e D é um parâmetro indeterminado que é usado apenas para indicar o peso da palavra-código na fórmula. A probabilidade de correção de apagamentos com sucesso, em um padrão com δ ou mais apagamentos, é igual à probabilidade de que nenhum subconjunto das δ coordenadas apagadas corresponda ao suporte de qualquer palavra-código, como já mencionado anteriormente no Teorema 4.1.

O número de possíveis padrões de apagamento com δ apagamentos em um código de comprimento N é $\begin{pmatrix} N \\ \delta \end{pmatrix}$. Para uma única palavra-código de peso w, o número de padrões de apagamento com δ coordenadas, I, que inclua o suporte desta palavra-código é $\begin{pmatrix} N-w \\ \delta-w \end{pmatrix}$. Portanto, a probabilidade de um subconjunto das δ coordenadas coincidir com o suporte de uma única palavra-código de peso w, $\Pr(\mathrm{sup}(\mathbf{c}) \subseteq I)$ [74], é dado por

$$\Pr(\mathrm{sup}(\mathbf{c}) \subseteq I) = \frac{\begin{pmatrix} N-w \\ \delta-w \end{pmatrix}}{\begin{pmatrix} N \\ \delta \end{pmatrix}} = \frac{(N-w)!\delta!(N-\delta)!}{N!(\delta-w)!(N-\delta)!)} = \frac{(N-w)!\delta!}{N!(\delta-w)!}.$$

Assumindo que nenhum padrão de apagamentos inclua mais do que uma única palavra-código, um limitante inferior da probabilidade de sucesso na correção de padrões com δ apagamentos pode ser derivado da probabilidade de que esses padrões contenham uma palavra-código de peso $w \leq \delta$ [74], ou seja,

$$\Pr(\delta) = \sum_{j=d_{min}}^{\delta} A_j \frac{(N-j)!\delta!}{N!(\delta-j)!}. \tag{4.3}$$

Disso decorre que o número médio de apagamentos corrigidos pelo código, N_{erase} [74], é calculado por

$$N_{erase} = (N-K) - \sum_{\delta=d_{min}}^{N-K} \Pr(\delta), \tag{4.4}$$

dado que há uma probabilidade igual a 1 que $(N-K+1)$ apagamentos não possam ser corrigidos e que $\Pr(\delta)=0$ para $\delta < d_{min}$. Substituindo a Expressão (4.3) em (4.4) [74] tem-se

$$N_{erase} = (N-K) - \sum_{\delta=d_{min}}^{N-K} \sum_{j=d_{min}}^{\delta} A_j \frac{(N-j)!\delta!}{N!(\delta-j)!}. \tag{4.5}$$

Os termos responsáveis pela queda em desempenho comparada a um código MDS (*Maximum Distance Separable*), $MDS_{shortfall}$, estão presentes na Expressão (4.5) [74], ou seja,

$$MDS_{shortfall} = \sum_{\delta=d_{min}}^{N-K} \sum_{j=d_{min}}^{\delta} A_j \frac{(N-j)!\delta!}{N!(\delta-j)!}. \tag{4.6}$$

A Expressão (4.6) destaca a importância das palavras-código de menor peso de Hamming, uma vez que, quanto menor for o fator A_j, $d_{min} \leq j \leq N-K$, o desempenho do código se aproxima mais da capacidade teórica do BEC. De certo modo, a Expressão (4.6) corrobora a Expressão (4.2), visto que, se não houver ambiguidade de decodificação para valores de $d_{min} \leq j \leq N-K$, o desempenho do código se aproxima do valor ótimo.

4.4 DESEMPENHO DA DECODIFICAÇÃO ITERATIVA DE APAGAMENTOS

A complexidade de decodificação de apagamentos por máxima verossimilhança cresce segundo $O(N^3)$ [43] e a torna pouco efetiva para códigos LDPC. Neste caso, é preferível usar a decodificação iterativa de apagamentos, aproveitando a representação do código LDPC na forma de grafo de Tanner e o método de transferência de mensagens do tipo BP. Para melhor compreensão do desempenho da decodificação iterativa de apagamentos para códigos LDPC, emprega-se o método conhecido por *density evolution* [13].

Quem primeiramente formulou o método *density evolution* para códigos LDPC regulares foi Gallager [4]. Já em [75], o método *density evolution* foi proposto para códigos LDPC irregulares, considerando um BEC, e foi generalizado para o SPA em canais sem memória em [8], [76].

4.4.1 *Density Evolution* PARA CÓDIGOS LDPC REGULARES

Seja uma configuração $T(w_c, w_r)$, que corresponde a todos os grafos de Tanner de códigos LDPC regulares com nós de variável com grau w_c e nós de verificação de paridade com grau w_r. Define-se q_l como a probabilidade que, em uma dada iteração l, uma mensagem de nó de verificação de paridade para nó de variável (mensagem *check-to-variable*) seja um *bit* apagado, Δ, e p_l como a

probabilidade que, em uma dada iteração l, uma mensagem de nó de variável para nó de verificação de paridade (mensagem *variable-to-check*) seja um *bit* apagado, Δ, i.e, p_l é a probabilidade que um *bit* da N-upla ainda esteja apagado na l-ésima iteração.

Seja ε a probabilidade de apagamento em um BEC. Considerando que não há ciclos no grafo de Tanner de comprimento inferior ou igual a $2l$, porquanto os ciclos tornariam as mensagens correlacionadas, de [13] extrai-se a seguinte fórmula

$$p_l = \varepsilon(1 - (1 - p_{l-1})^{w_r - 1})^{w_c - 1},\ l = 1, 2, 3, \ldots. \tag{4.7}$$

Na Fórmula (4.7), para $l = 1$, $p_{l-1} = p_0 = \varepsilon$. Este valor corresponde à probabilidade *a priori*. A Fórmula (4.7) permite calcular a probabilidade que um *bit* da N-upla ainda esteja apagado na l-ésima iteração, p_l, em função da probabilidade obtida na iteração anterior, p_{l-1}. Esta recorrência mostra portanto a evolução da probabilidade p_l em função do número de iterações l para códigos LDPC (N,K) regulares. Aplicando esta recorrência, pode-se determinar para quais valores de probabilidade de apagamento do BEC, ε, o decodificador por transferência de mensagens é capaz de corrigir os apagamentos.

Exemplo 4.2

Um código LDPC (N,K) regular com $w_c = 5$ e $w_r = 10$ é usado na transmissão de dados por um BEC que tem $\varepsilon = 0,30$ e é decodificado usando o algoritmo de transferência de mensagens. A probabilidade que um dado bit apagado, presente na N-upla recebida, permaneça apagado após l iterações do decodificador por transferência de mensagens (supondo que o grafo de Tanner do código não tenha ciclos) é dada pela expressão $p_l = \varepsilon(1 - (1 - p_{l-1})^9)^4$. Aplicando esta recorrência por sete iterações, gera-se a seguinte sequência de probabilidades de apagamento de bit: $p_0 = 0,3000$, $p_1 = 0,2544$, $p_2 = 0,2232$, $p_3 = 0,1943$, $p_4 = 0,1617$, $p_5 = 0,1202$, $p_6 = 0,0658$, $p_7 = 0,0132$ e $p_8 = 0,000048$. Com base neste resultado, conclui-se que a probabilidade que este respectivo bit permaneça apagado tende para um valor próximo a zero após oito iterações do decodificador por transferência de mensagens. Esse procedimento é repetido para os seguintes valores de probabilidade de apagamento do BEC: $\varepsilon = 0,32$, $\varepsilon = 0,33$, $\varepsilon = 0,34$ e $\varepsilon = 0,35$. Os resultados da avaliação são mostrados na Figura 4.3.

Há um valor de probabilidade de apagamento do BEC, ε_{th}, que é designado por limiar, acima do qual a probabilidade p_l não tende a zero ainda que l seja muito grande. Esse valor para o código LDPC regular do Exemplo 4.2 está representado na Figura 4.4 e é aproximadamente $\varepsilon_{th} = 0,341$. Nessa figura, observa-se que para $\varepsilon = 0,342 > \varepsilon_{th}$, mesmo para um número grande de iterações l, o

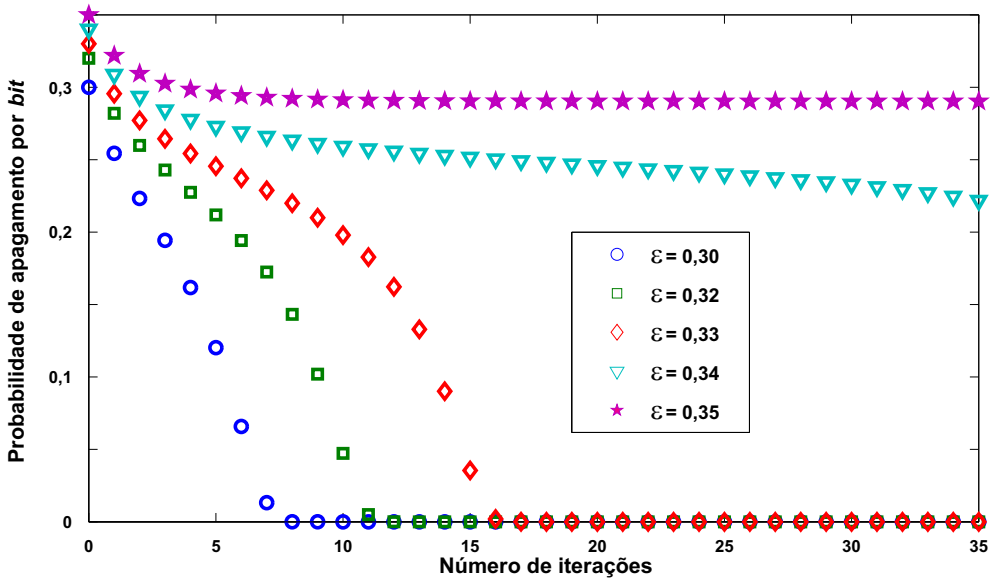


Figura 4.3: *Density evolution para o código LDPC regular do Exemplo 4.2.*

valor de p_l não tende a zero.

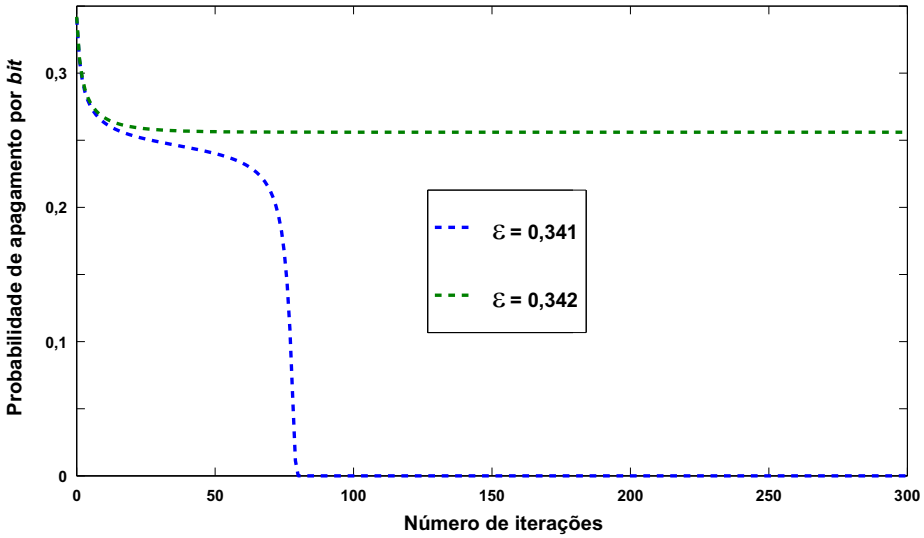


Figura 4.4: *Limiar do código LDPC regular do Exemplo 4.2.*

4.4.2 *Density Evolution* PARA CÓDIGOS LDPC IRREGULARES

Uma matriz de verificação de paridade para código LDPC irregular tem linhas e colunas com pesos variados (respectivamente, nós de variável e nós de verificação de paridade com diversos graus). Para gerar a evolução da função de densidade de probabilidade (*density evolution*) para um código LDPC irregular, é usada uma caracterização alternativa da distribuição de graus da perspectiva dos

ramos do grafo de Tanner. A fração de ramos que é conectada a nós de variável de grau i é representada por λ_i e a fração de ramos que é conectada aos nós de verificação de paridade de grau j é representada por ρ_j. Por definição $\sum_i \lambda_i = 1$ e $\sum_j \rho_j = 1$.

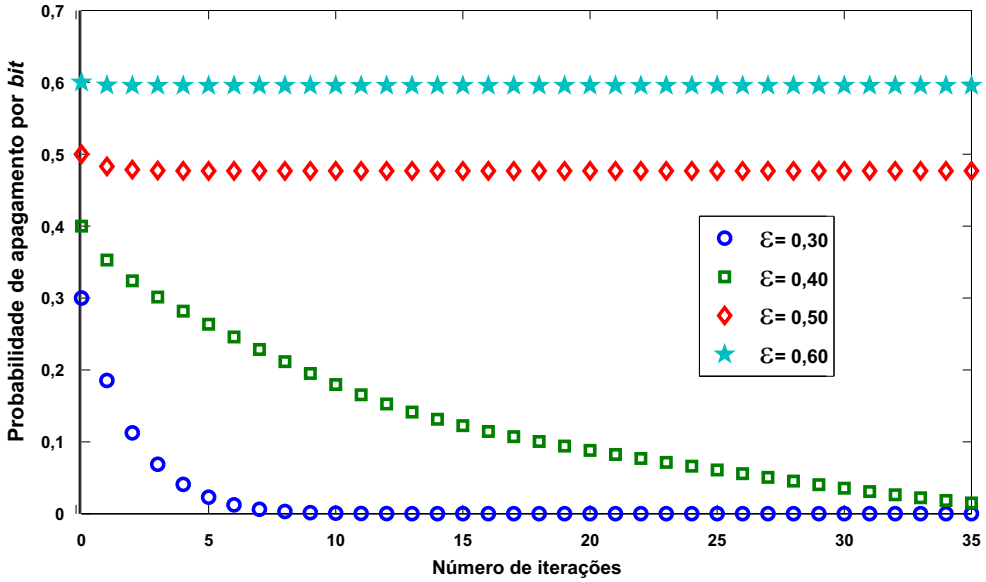


Figura 4.5: *Density evolution para o código LDPC (*648,324*) irregular do Exemplo 4.3.*

As funções

$$\lambda(x) = \lambda_2 x + \lambda_3 x^2 + \lambda_4 x^3 + \cdots + \lambda_i x^{i-1} + \cdots$$

e

$$\rho(x) = \rho_2 x + \rho_3 x^2 + \rho_4 x^3 + \cdots + \rho_i x^{i-1} + \cdots$$

descrevem as distribuições de graus.

De [13] extrai-se a seguinte fórmula

$$p_l = \varepsilon\lambda(1 - \rho(1 - p_{l-1})),\ l = 1, 2, 3, \ldots. \tag{4.8}$$

Na Fórmula (4.8), para $l = 1$, $p_{l-1} = p_0 = \varepsilon$. Este valor corresponde à probabilidade *a priori*.

Exemplo 4.3

Seja o código LDPC (648,324*) irregular que é empregado no padrão IEEE*$802.11n$ *[42] e que possui taxa* $R = 0,50$ *e distribuição de graus dada por* $\lambda(x) = 0,25x + 0,341x^2 + 0,409x^{11}$ *e* $\rho(x) = 0,637x^6 + 0,363x^7$*. Realizou-se a análise de desempenho de correção de apagamentos para este código, para os seguintes valores de probabilidade de apagamento do BEC:* $\varepsilon = 0,60$*,* $\varepsilon = 0,50$*,* $\varepsilon = 0,40$ *e* $\varepsilon = 0,30$*. Os resultados obtidos da evolução da probabilidade* p_l

em função do número de iterações l do decodificador por transferência de mensagens estão mostrados nas curvas da Figura 4.5. □

As curvas traçadas na Figura 4.6, por seu turno, mostram o resultado da análise para o valor de limiar para esse mesmo código LDPC irregular. Observa-se que para $\varepsilon \leq 0,42$ a probabilidade p_l tende a zero à medida que $l \to \infty$, ao passo que para valores de $\varepsilon \geq 0,43$, a probabilidade não cai a zero, ainda que l seja muito grande. Portanto, o valor de limiar para o código LDPC (648,324) irregular é de aproximadamente $0,43$.

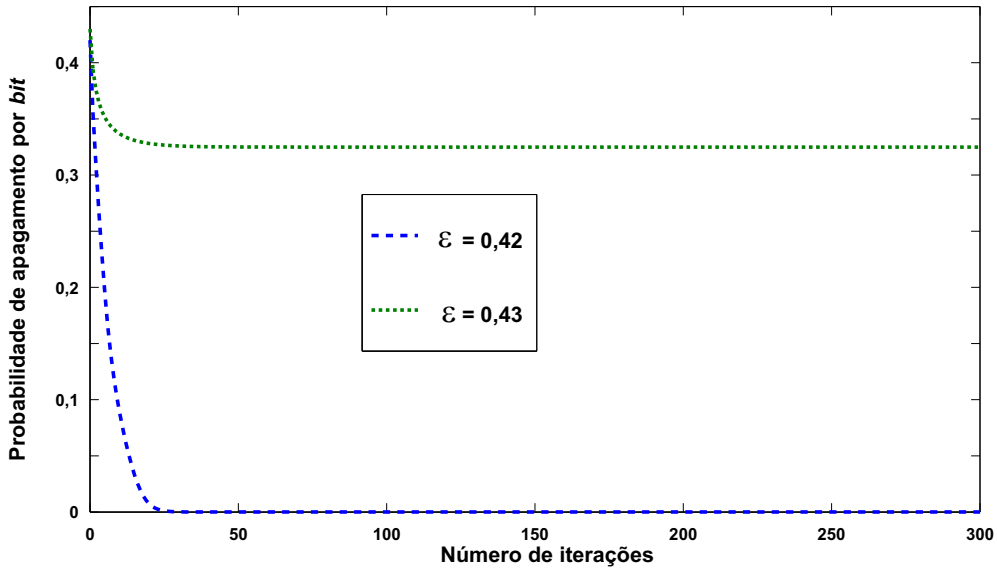


Figura 4.6: *Limiar para o código LDPC (*648,324*) empregado no padrão IEEE*802.11*n.*

4.4.3 CONJUNTOS DE PARADA (*Stopping Sets*)

A decodificação iterativa de apagamentos dos códigos LDPC regulares e irregulares, além da simplicidade, caracteriza-se pela eficiência e rapidez. No entanto, há situações em que a decodificação iterativa de apagamentos se depara com falhas decorrentes principalmente do fato de não haver equações de verificação de paridade que tenham apenas um único *bit* apagado e que, portanto, possam ser resolvidas. As estruturas combinacionais formadas por *bits* apagados e que não podem ser decodificadas são chamadas de conjuntos de parada ou *stopping sets* [77].

Definição 4.2

Um conjunto de parada $\mathcal{S}$ em um código linear $\mathcal{C}$ é um subconjunto dos nós de variável do grafo de Tanner deste código, tal que todos os vizinhos de $\mathcal{S}$ (nós de verificação de paridade) são conectados a $\mathcal{S}$ pelo menos duas vezes [77]. □

Definição 4.3

A menor cardinalidade dentre todos os conjuntos de parada, denotada por $s(\mathbf{H})$, é definida como a distância de parada de $\mathcal{C}$ [78] e [79]. □

A título de ilustração, a Figura 4.7 mostra o grafo de Tanner de um código com comprimento, N, igual a cinco e com número de equações de verificação de paridade, $N - K$, igual a três, em que os nós de variável representados por círculos escuros (círculos cheios), estão apagados e formam um conjunto de parada de cardinalidade igual a três.

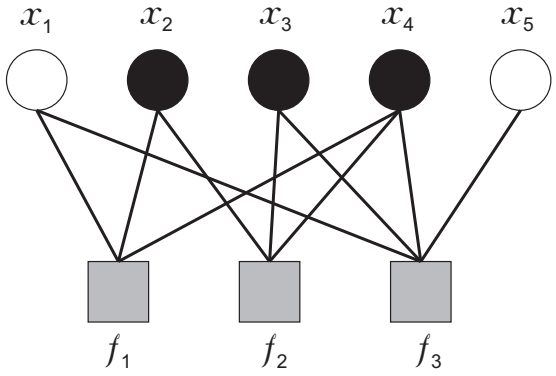
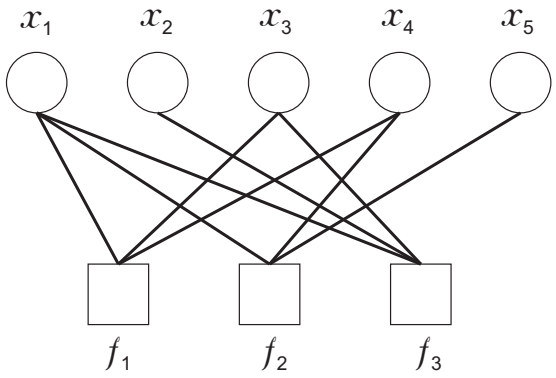


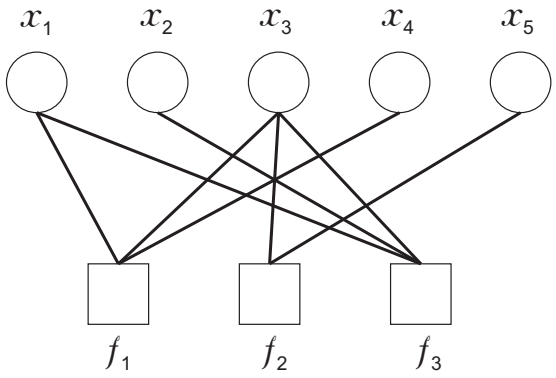
Figura 4.7: *Grafo de Tanner para um código de comprimento cinco com um conjunto de parada de cardinalidade três. Os círculos escuros representam nós de variável com apagamento, os círculos claros representam nós de variável sem apagamento e os quadrados representam nós de verificação de paridade.*

A distribuição dos conjuntos de parada depende da matriz de verificação de paridade $\mathbf{H}$ de um código LDPC e define para quais padrões de apagamento ocorre falha no decodificador de transferência de mensagens. Portanto, caso se conheça a distribuição de conjuntos de parada, é possível predizer o desempenho da decodificação iterativa de apagamentos de um código LDPC [77].

Vale ressaltar que nem todas as matrizes de verificação de paridade para o mesmo código têm a mesma distribuição de conjuntos de parada ou a mesma distância de parada, $s(\mathbf{H})$ [13]. A Figura 4.8, extraída de [13], mostra, por exemplo, os grafos de Tanner de duas matrizes de verificação de paridade para o mesmo código. A Figura 4.8(a) mostra o grafo de Tanner da matriz de verificação de paridade que apresenta três conjuntos de parada com cardinalidade três enquanto na Figura 4.8(b) é mostrado o grafo de Tanner da matriz de verificação de paridade que apresenta dois conjuntos de parada com cardinalidade três. Os dois conjuntos de parada comuns entre estas configurações incluem os *bits* $\{1, 2, 4\}$ e $\{1, 3, 5\}$. Todas as possíveis matrizes de verificação de paridade para este código também conterão conjuntos de parada nestas posições de *bits*. A matriz de verificação de paridade correspondente ao grafo de Tanner da Figura 4.8(a), com os respectivos padrões de



(a)



(b)

Figura 4.8: *Grafo de Tanner para duas matrizes de verificação de paridade diferentes que representam o mesmo código. Os círculos representam nós de variável e os quadrados representam nós de verificação de paridade. [13]*

conjuntos de parada que incluem os *bits* $\{1, 2, 4\}$ e $\{1, 3, 5\}$, é assim representada

	x_1	x_2	x_3	x_4	x_5
f_1	1	0	1	1	0
f_2	1	0	0	1	1
f_3	1	1	1	0	0
$\{1,2,4\}$	Δ	Δ	$-$	Δ	$-$
$\{1,3,5\}$	Δ	$-$	Δ	$-$	Δ

Do mesmo modo, a matriz de verificação de paridade correspondente ao grafo de Tanner da Figura 4.8(b), com os respectivos padrões de conjuntos de parada que incluem os *bits* $\{1, 2, 4\}$ e $\{1, 3, 5\}$, é assim representada

	x_1	x_2	x_3	x_4	x_5
f_1	1	0	1	1	0
f_2	0	0	1	0	1
f_3	1	1	1	0	0
$\{1,2,4\}$	Δ	Δ	$-$	Δ	$-$
$\{1,3,5\}$	Δ	$-$	Δ	$-$	Δ

Como pode ser observado nas duas representações, nenhuma das equações de verificação de paridade (f_1, f_2 e f_3) possui apenas um único apagamento e, portanto, estes padrões constituem conjuntos de parada.

O desempenho da decodificação iterativa de apagamentos, em função de uma dada matriz de verificação de paridade $\mathbf{H}$ de um código com comprimento de bloco N e em função da probabilidade de apagamento ε do BEC, pode ser medido pela probabilidade $P_{\mathbf{H}}(\varepsilon)$ expressa como [13]

$$P_{\mathbf{H}}(\varepsilon) = \sum_{\delta=0}^{N} \begin{pmatrix} N \\ \delta \end{pmatrix} \varepsilon^{\delta}(1-\varepsilon)^{N-\delta} \left(\frac{S(\delta)}{T(\delta)}\right), \tag{4.9}$$

na qual o parâmetro $T(\delta)$ denota o número total de combinações distintas de δ posições com apagamentos e $S(\delta)$ denota o número de combinações, dentre as possíveis $T(\delta)$ combinações, que resultam em um conjunto de parada. Portanto, a razão $S(\delta)/T(\delta)$ pode ser interpretada como a probabilidade de um dado conjunto de δ dígitos apagados ser um conjunto de parada.

Com base na Expressão (4.9), depreende-se que o desempenho da decodificação iterativa de apagamentos não é afetado para padrões de apagamentos com cardinalidade inferior à distância de parada, $s(\mathbf{H})$, do código linear. Este parâmetro, portanto, exerce um papel fundamental na decodificação iterativa de apagamentos, porquanto pode limitar o desempenho deste tipo de decodificador, mesmo para padrões de apagamentos com cardinalidade inferior à distância mínima do código, d_{min}. Da mesma forma que se tenta maximizar a distância mínima, d_{min}, caso a decodificação ML ou algébrica seja usada, se tenta também maximizar a distância de parada, no caso de decodificação iterativa. Há, contudo, uma importante diferença entre a distância mínima, d_{min}, e o parâmetro distância de parada, $s(\mathbf{H})$. Enquanto a primeira é uma propriedade do código $\mathcal{C}$, a segunda depende do grafo de Tanner ou, equivalentemente, da escolha de uma matriz de verificação de paridade, $\mathbf{H}$, específica para $\mathcal{C}$ [79].

Uma matriz de verificação de paridade, $\mathbf{H}$, para um código linear $\mathcal{C}$, possui $N - \dim(\mathcal{C}) = N - K$ linhas que são linearmente independentes. Contudo, no contexto de decodificação iterativa, adicionar

linhas linearmente dependentes a $\mathbf{H}$ pode ser vantajoso para o incremento da distância de parada $s(\mathbf{H})$, não obstante o aumento da complexidade [79].

Dada a dependência dos parâmetros conjunto de parada e distância de parada das características da matriz de verificação de paridade, $\mathbf{H}$, é possível reescrever as suas definições da seguinte forma [79]:

Definição 4.4

Um conjunto de parada é um conjunto de colunas de $\mathbf{H}$ *que, separadas, formam uma submatriz cujas linhas possuem peso maior do que um.* □

Definição 4.5

Seja $\mathcal{C}$ *um código linear (não necessariamente binário) e seja* $\mathbf{H}$ *uma matriz de verificação de paridade para* $\mathcal{C}$*; então a distância de parada de* $\mathbf{H}$ *é definida como o maior inteiro* $s(\mathbf{H})$ *tal que todo conjunto de no máximo* $s(\mathbf{H}) - 1$ *colunas de* $\mathbf{H}$ *contenha ao menos uma linha de peso um.* □

A Definição 4.5, referente à distância de parada, guarda uma grande semelhança com a definição de distância mínima de Hamming de um código linear. A distância minima de um código linear $\mathcal{C}$ pode ser definida como o maior inteiro d_{min} tal que quaisquer $d_{min}-1$ colunas de $\mathbf{H}$ são linearmente independentes [39]. Para códigos binários, isto é equivalente a dizer que d_{min} é o maior inteiro, tal que todo conjunto, de, no máximo, $d_{min} - 1$ colunas de $\mathbf{H}$, contenha pelo menos uma linha de peso ímpar. O seguinte corolário é uma consequencia imediata da justaposição das definições de $s(\mathbf{H})$ e d_{min}.

Corolário 4.1

Seja $\mathcal{C}$ *um código linear e seja* $\mathbf{H}$ *uma matriz de verificação de paridade arbitrária para* $\mathcal{C}$*; então* $s(\mathbf{H}) \leq d_{min}$. □

Portanto, dado um código linear $\mathcal{C}$, o maior valor de distância de parada que se poderia esperar seria d_{min}, não importando o quão elaborado é o seu grafo de Tanner. A igualdade só é alcançada pela adição a $\mathbf{H}$ de linhas linearmente dependentes.

4.4.4 Desempenho na Correção de Apagamentos dos Códigos LDPC do Padrão IEEE802.11n

A Figura 4.9 mostra as curvas de desempenho para a decodificação iterativa de apagamentos para os códigos LDPC usados no padrão IEEE802.11n, a saber: LDPC (1944,972) com comprimento

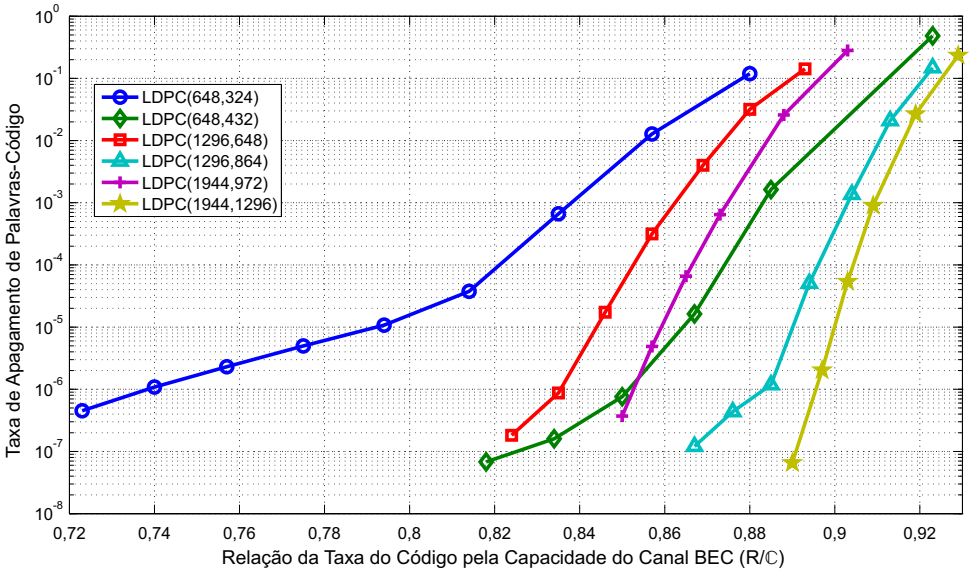


Figura 4.9: *Desempenho da decodificação iterativa para correção de apagamentos em códigos LDPC usados no Padrão IEEE*802.11*n.*

$N = 1944$ e com taxa igual a $1/2$, LDPC (1944,1296) com comprimento $N = 1944$ e com taxa igual a $2/3$, LDPC (1296,648) com comprimento $N = 1296$ e com taxa igual a $1/2$, LDPC (1296,864) com comprimento $N = 1296$ e com taxa igual a $2/3$, LDPC (648,324) com comprimento $N = 648$ e com taxa igual a $1/2$ e LDPC (648,432) com comprimento $N = 648$ e com taxa igual a $2/3$. Todos esses códigos LDPC são irregulares e quasi-cíclicos [13].

No eixo horizontal da Figura 4.9 estão representados os valores da relação entre a taxa do código e a capacidade do BEC, que são calculados como

$$\frac{R}{\mathbb{C}} = \frac{K}{N(1-\varepsilon)},$$

em que K corresponde à dimensão do código, N corresponde ao comprimento do código e ε corresponde à probabilidade de apagamentos do BEC. No eixo vertical da Figura 4.9 estão representados os valores das taxas de apagamentos de palavras-código (*Word-erasure rate*), que foram obtidos por simulação.

No método de simulação empregado, primeiramente define-se a probabilidade de apagamentos do BEC, ε, e, em seguida, executa-se a iteração constituída pelos passos mostrados no Algoritmo 4.2.

No Algoritmo 4.2, o valor de β define o critério de parada das iterações e corresponde ao número mínimo de N-uplas decodificadas erroneamente, ou seja, ainda com apagamentos remanescentes. Em geral, um valor típico que pode ser adotado para β é de 500 N-uplas. Após o final das iterações, calcula-se a taxa de apagamento de palavras-código pela relação entre o número de N-uplas

decodificadas erroneamente (β) e o número total de N-uplas geradas durante a simulação.

Algoritmo 4.2 – Obtenção de Desempenho da Decod. Iter. de Apagamentos para Códigos LDPC

1*: Gera-se uma quantidade K de valores binários aleatórios.*

2*: Codifica-se os K valores, gerando uma N-upla.*

3*: Submete-se cada bit da N-upla a um canal BEC com probabilidade de apagamento igual a ε.*

4*: Executa-se a decodificação iterativa de apagamentos sobre a N-upla resultante.*

5*: Os apagamentos foram corrigidos ?*

6*: Se sim, então vai para o passo descrito na linha* 1.

7*: Se não, então*

8*: Computa como N-upla decodificada erroneamente.*

9*: Número de N-uplas decodificadas erroneamente > β ?*

10*: Se não, então vai para o passo descrito na linha* 1.

11*: Se sim, então*

12*: Calcula a taxa de apagamentos por palavras-código.*

13*: **FIM**.*

CAPÍTULO 5

DECODIFICAÇÃO HÍBRIDA PARA CÓDIGOS LDPC

Ao longo dos anos, os códigos LDPC [3], [4] vêm empregando o algoritmo de decodificação iterativa de baixa complexidade, do tipo de transferência de mensagens (BP), para a correção de erros, aproveitando o fato de sua matriz de verificação de paridade, **H**, ser do tipo esparsa, o que a torna adequada à representação na forma de grafo de Tanner mesmo para códigos longos.

5.1 INTRODUÇÃO

Há duas formas de decodificação iterativa, a que utiliza o método de atualização simultânea de todas as mensagens a cada iteração, que é designada por método invasivo [17], [18] e a do tipo de atualização sequencial [19]–[23] que prioriza as mensagens a serem atualizadas, em conformidade com algum parâmetro de avaliação. As duas formas realizam a decodificação de erros da N-upla recebida baseando-se em estimativas de probabilidade, por isto são conhecidas como formas de decodificação com decisão suave.

Também é possível a forma de decodificação baseada em decisões abruptas [4], [15], [45], que é mais rápida mas apresenta desempenho inferior se comparada ao método de decodificação com decisão suave.

As formas de decodificação que aliam decodificação com decisão suave à decodificação com decisão abrupta são conhecidas por decodificação híbrida [80]–[82]. Esse método procura aliar as boas características de desempenho de decodificação do método com decisão suave com a rapidez de

decodificação do método com decisão abrupta.

Uma variante do método de decodificação híbrida que emprega o artifício de pós-processamento por decodificação de apagamentos, aplicado após uma falha na decodificação de erros, tem recebido pouca ou nenhuma atenção tanto em canais com RAGB como em outros modelos de canal. Em [37], é estudada essa técnica de pós-processamento, mas que atua somente sob certas condições de peso da síndrome.

Diferentemente, nesta tese, é proposto um novo método de decodificação híbrida (DH) para códigos LDPC, baseado em pós-processamento por decodificação de apagamentos, que atua sempre que houver a falha da decodificação iterativa de erros. Nesse sistema, os apagamentos são definidos com base nas informações de confiabilidade dos dígitos, geradas ao fim da decodificação de erros do tipo *min-sum* BP, que utiliza o algoritmo *min-sum* modificado que é descrito na Seção 3.6. Adicionalmente, emprega-se um número fixo e predeterminado de apagamentos a serem processados pela decodificação iterativa de apagamentos.

Essa abordagem proposta, visa reduzir o número de iterações e manter um desempenho equivalente ao alcançado com sistemas de decodificação iterativa do tipo soma-produto [17] e consiste em dois estágios, sendo o primeiro deles de decodificação iterativa de erros do tipo *min-sum* BP em canal com RAGB e o segundo estágio consistindo da decodificação iterativa em um canal BEC artificialmente criado, interagindo com o decodificador *min-sum* BP do primeiro estágio. Uma iteração no sistema DH inclui necessariamente uma passagem pelo algoritmo *min-sum* BP e, caso necessário, inclui também uma passagem pelo algoritmo de decodificação de apagamentos. Doravante, uma iteração do sistema DH será referida como *ciclo de operação* ou simplesmente *ciclo* para evitar confusão com o emprego da palavra iteração no estágio de decodificação de erros e no estágio de decodificação de apagamentos, que fazem parte do sistema DH.

5.2 DESCRIÇÃO DO CICLO DE OPERAÇÃO DO SISTEMA DH

O diagrama em blocos do sistema básico de comunicações adotado para a análise do sistema DH é mostrado na Figura 5.1 e pode ser descrito sucintamente da seguinte forma:

No transmissor, a informação binária $\mathbf{u} = (u_1, u_2, \ldots, u_K)$ é usada para gerar a palavra-código $\mathbf{c} = (c_1, c_2, \ldots, c_N)$ pertencente a $\mathcal{C}$ que denota um código binário LDPC.

Cada palavra-código binária, $\mathbf{c}$, antes de ser transmitida pelo canal, é entregue a um mapeador que a converte para a sequência $\mathbf{x} = (x_1, x_2, \ldots, x_N)$ de números inteiros correspondentes, sendo que cada *bit* 0 é convertido para o número inteiro $+1$ e cada *bit* 1 para o inteiro -1.

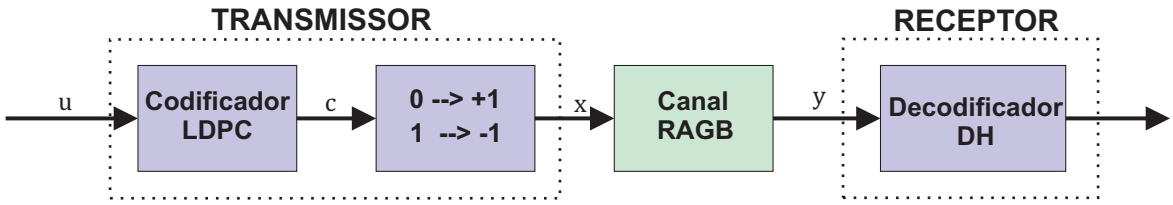


Figura 5.1: *Diagrama em blocos do sistema básico de comunicações adotado.*

A N-upla $\mathbf{x}$ é enviada por um canal com RAGB cuja a saída $\mathbf{y}$ é expressa como $\mathbf{y} = \mathbf{x} + \eta$, em que $\eta = (\eta_1, \eta_2, \ldots, \eta_N)$ denota um vetor cujas componentes são amostras de ruído branco Gaussiano introduzido pelo canal, com média nula e variância σ^2. A decodificação consiste em realizar o processamento de $\mathbf{y}$ para recuperar $\mathbf{u}$.

No receptor, o decodificador recebe do canal a N-upla $\mathbf{y}$ e a converte em uma sequência de valores de QLLR [58], representada por $\xi = (\xi_1, \xi_2, \ldots, \xi_N)$.

Para o primeiro estágio de decodificação, há a necessidade de se especificar o número I_{BP} de iterações do algoritmo *min-sum* BP. A escolha desse valor deve ser tal que, caso a palavra recebida não seja decodificada em até I_{BP} iterações, provavelmente o decodificador *min-sum* BP encontrou um conjunto de armadilhas [25]. Ao atingir um conjunto de armadilhas, não é possível por meio de mais iterações do algoritmo *min-sum* BP sair desta situação e conseguir decodificar com sucesso a palavra recebida.

Após cada iteração do algoritmo *min-sum* BP, é gerada uma sequência de valores de confiabilidade $\hat{\xi} = (\hat{\xi}_1, \hat{\xi}_2, \ldots, \hat{\xi}_N)$, a qual é convertida para a forma binária $\hat{\mathbf{c}} = (\hat{c}_1, \hat{c}_2, \ldots, \hat{c}_N)$ por decisão abrupta.

A N-upla binária $\hat{\mathbf{c}}$ é submetida a um teste de síndrome por meio da operação $\hat{\mathbf{c}}\mathbf{H}^T$, em que $\mathbf{H}^T$ denota a matriz de verificação de paridade transposta. Se $\hat{\mathbf{c}}$ for uma palavra código, então $\hat{\mathbf{c}}\mathbf{H}^T = \mathbf{0}$ e o ciclo de DH é interrompido com sucesso. Caso contrário, uma nova iteração do algoritmo *min-sum* BP é realizada e o teste sobre $\hat{\mathbf{c}}$ é novamente aplicado, até verificar-se $\hat{\mathbf{c}}\mathbf{H}^T = \mathbf{0}$ ou serem atingidas I_{BP} iterações.

Quando o número de iterações atinge o valor I_{BP}, sem que haja sucesso na decodificação, em vez de ser declarada a falha de decodificação, $\hat{\mathbf{c}}$ é convenientemente processada por um canal BEC artificialmente criado, como explicado a seguir, e convertida numa N-upla $\mathbf{z} = (z_1, z_2, \ldots, z_N)$ com dígitos binários e apagamentos. O segundo estágio de decodificação opera sobre a saída desse canal BEC somente quando há falha na decodificação no primeiro estágio.

A motivação para o emprego da decodificação de apagamentos no segundo estágio de decodifi-

cação é a seguinte. É sabido que, para um determinado código de bloco linear com distância mínima d_{min}, é possível a correção de qualquer padrão com até $d_{min} - 1$ apagamentos [38] e de uma grande quantidade de padrões contendo d_{min} ou mais apagamentos, desde que o total de apagamentos por palavra não exceda o número de dígitos de verificação de paridade do código [43].

Uma iteração da decodificação de apagamentos consiste em examinar cada equação de verificação de paridade, com relação ao número de apagamentos. Caso uma equação tenha um único apagamento entre as posições verificadas, este é corrigido. Caso uma equação não contenha apagamentos ou contenha dois ou mais apagamentos, o decodificador a ignora. Caso um ou mais apagamentos tenham sido corrigidos, é então repetido o processo de examinar todas as equações de verificação de paridade. Se ao completar a verificação de todas as equações e nenhum apagamento tiver sido corrigido, a decodificação de apagamentos é encerrada [13].

A sequência binária na entrada do BEC artificial consiste em $0'$s e $1'$s, obtidos por decisão abrupta tomada sobre as respectivas medidas de confiabilidade, obtidas a partir das probabilidades *a posteriori* na última iteração da decodificação *min-sum* BP, utilizando como medida os valores de QLLR. Em caso de falha na decodificação *min-sum* BP, o maior percentual dos erros remanescentes está concentrado em valores de QLLR próximos a zero [83].

Definição 5.1

$\mathbb{S}_{\mathcal{X}}$ denota o conjunto de cardinalidade $\mathcal{X}$, um número inteiro positivo, que contém as coordenadas dos $\mathcal{X}$ dígitos de menor confiabilidade da entrada do BEC, os quais são escolhidos para apagamentos. □

Desta forma, os dígitos associados às posições em $\mathbb{S}_{\mathcal{X}}$ correspondem àqueles com os menores valores de QLLR em módulo.

A Expressão (5.1) é usada para a obtenção das componentes da N-upla $\mathbf{z}$ na saída do BEC, e é descrita como

$$z_i = \begin{cases} \Delta, \text{ para } i \in \mathbb{S}_{\mathcal{X}} \\ \hat{c}_i, \text{ para } i \notin \mathbb{S}_{\mathcal{X}}, \end{cases} \tag{5.1}$$

em que Δ representa um apagamento.

Essa N-upla $\mathbf{z}$ é entregue ao decodificador iterativo de apagamentos. Caso nenhum apagamento tenha sido corrigido na primeira iteração do decodificador de apagamentos, o algoritmo DH é encerrado, com falha na decodificação. Entretanto, se pelo menos um apagamento tiver sido corrigido, ao ser encerrada a etapa de decodificação de apagamentos, a N-upla $\mathbf{z}$ é transformada na N-upla $\mathbf{w} = (w_1, w_2, \ldots, w_N)$, conforme explicado a seguir. As i-ésimas coordenadas binárias em $\mathbf{z}$, em

que $i \notin \mathbb{S}_{\mathcal{X}}$, terão os seus valores mantidos em $\mathbf{w}$. Para as demais coordenadas, a N-upla $\mathbf{w}$ assume valores binários estimados pela decodificação de apagamentos nas posições em que os apagamentos forem corrigidos ou, em caso contrário, assume os valores binários de $\hat{\mathbf{c}}$ estimados ao fim da decodificação *min-sum* BP.

A Expressão (5.2) é usada para gerar a N-upla $\boldsymbol{\xi} = (\xi_1, \xi_2, \ldots, \xi_N)$ de valores de QLLR que é associada a $\mathbf{w}$, e é descrita como

$$\xi_j = \begin{cases} \hat{\xi}_j, & w_j = \Delta \text{ ou } w_j = \hat{c}_j, \\ -\hat{\xi}_j & w_j \neq \hat{c}_j, \end{cases} \tag{5.2}$$

para $j \in \mathbb{S}_{\mathcal{X}}$. Para as demais coordenadas, $\xi_j = \hat{\xi}_j$. Observa-se na N-upla $\boldsymbol{\xi}$ que os valores de QLLR resultantes da decodificação *min-sum* BP anterior serão mantidos se, ao final da decodificação de apagamentos, os dígitos permanecerem apagados ou se os seus valores corresponderem aos mesmos que foram estimados pela decodificação *min-sum* BP anterior. Caso contrário, apenas o sinal do valor de QLLR do dígito correspondente é invertido.

Caso o número máximo de ciclos de operação I_{DH} do algoritmo DH não tenha sido atingido, a N-upla $\boldsymbol{\xi}$ é usada como entrada no algoritmo *min-sum* BP para dar início a um novo ciclo do algoritmo DH. Antes disso, é necessário testar a N-upla $\boldsymbol{\xi}$. Para tal, é gerada uma N-upla binária correspondente $\mathbf{b} = (b_1, b_2, \ldots, b_N)$ por decisão abrupta sobre os valores de $\boldsymbol{\xi}$. A ocorrência de $\mathbf{b} \in \mathcal{C}$, indica que o ciclo de decodificação híbrida é finalizado com sucesso. Caso contrário, um novo ciclo de decodificação híbrida é executado.

Para o caso do número máximo de ciclos I_{DH} ter sido atingido, a ocorrência de $\mathbf{b} \in \mathcal{C}$ indica que o ciclo de decodificação híbrida é finalizado com sucesso, enquanto a condição $\mathbf{b} \notin \mathcal{C}$ indica que a decodificação é encerrada sem sucesso. Uma descrição resumida do sistema DH é apresentada no Algoritmo 5.1.

Esse sistema de decodificação híbrida iterativa só se encerra quando:

- ▷ Ocorrer sucesso na etapa de decodificação iterativa de erros;
- ▷ Ocorrer correção de todos os apagamentos na etapa de decodificação iterativa de apagamentos, podendo a N-upla resultante ser válida (sucesso) ou não (falha);
- ▷ Ocorrer falha na decodificação iterativa de apagamentos e que se observe que nenhum *bit* originalmente apagado tenha sido corrigido; ou
- ▷ For atingido o número máximo de ciclos do decodificador DH, I_{DH}. Nesse caso, é declarada falha de decodificação.

Algoritmo 5.1 – Decodificação Híbrida Iterativa

1: *Conversão da N-upla recebida do canal com RAGB para valores de QLLR .*
2: $Iter = 1$.
3: *Fixa o valor para* $\mathcal{X}$.
4: *Executa a decodificação iterativa min-sum BP para erros (min-sum modificado).*
5: *Sucesso na decodificação de erros?*
6: *Se sim, então vai para* ***FIM****.*
7: *Se não, então*
8: *Seleciona e apaga as* $\mathcal{X}$ *coordenadas com os menores valores de QLLR (em módulo).*
9: *Executa o algoritmo de decodificação iterativa de apagamentos.*
10: *Número de apagamentos igual a zero?*
11: *Se sim, então*
12: *A* N*-upla é válida?*
13: *Se sim, então* ***SUCESSO*** *e vai para* ***FIM****.*
14: *Se não, então* ***FALHA*** *e vai para* ***FIM****.*
15: *Se não, então*
16: $Iter < I_{DH}$*?*
17: *Se sim, então*
18: *Foi corrigido algum apagamento (bit)?*
19: *Se sim, então*
20: *Atualiza apenas os valores de QLLR estimados para*
21: *os bits corrigidos.*
22: $Iter = Iter + 1$.
23: *Retorna ao passo descrito na linha 4.*
24: *Se não, então* ***FALHA*** *e vai para* ***FIM****.*
25: *Se não, então* ***FALHA*** *e vai para* ***FIM****.*
26: ***FIM****.*

Na Figura 5.2 é mostrado o fluxograma do sistema de decodificação híbrida iterativa, contendo os passos descritos anteriormente e que foram resumidos no Algoritmo 5.1.

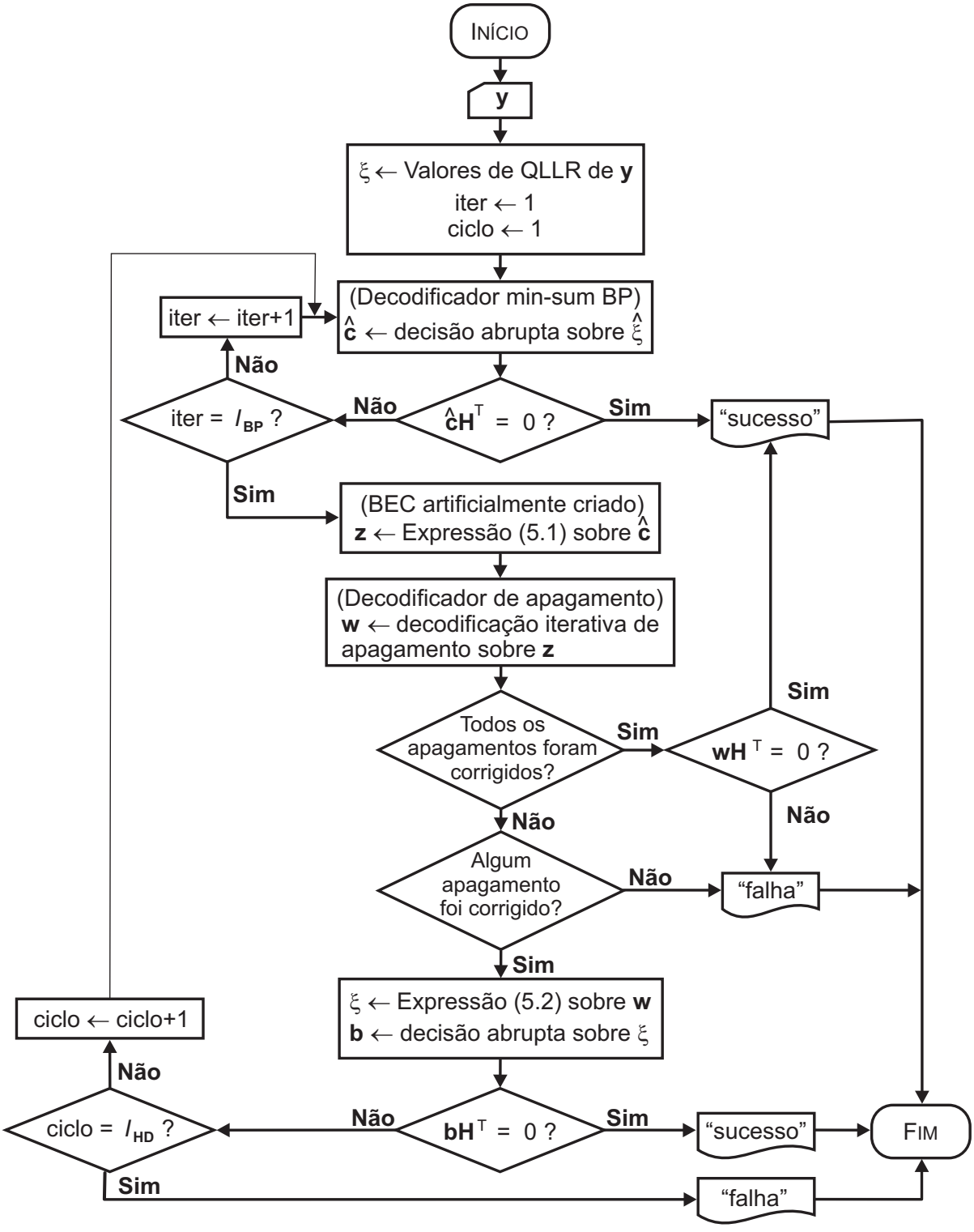
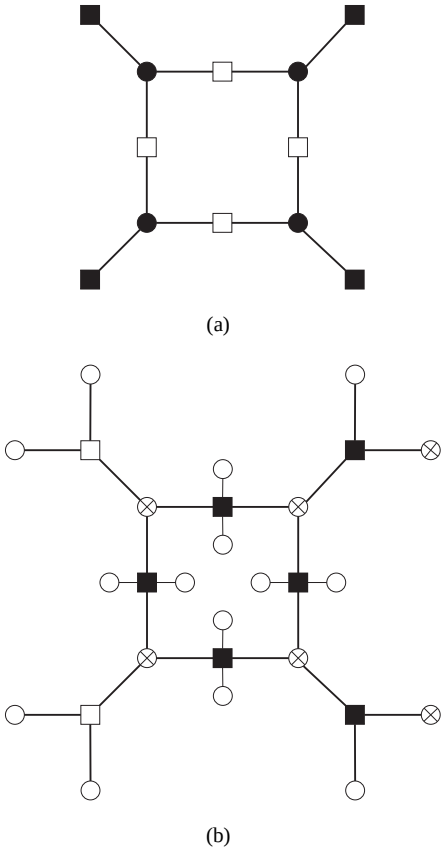


Figura 5.2: *Fluxograma do sistema de decodificação híbrida iterativa.*

5.3 ATUAÇÃO DO SISTEMA DH SOBRE CONJUNTOS DE ARMADILHAS

Os padrões de erros gerados após a decodificação *min-sum* BP, de uma maneira geral, são formados por *bits* presentes em conjuntos de armadilhas de pequena dimensão [25] que, ao serem apagados, poderão, em alguns casos, ser corrigidos pela decodificação de apagamentos do sistema DH proposto. A Figura 5.3(a) mostra um conjunto de armadilhas $TS(4,4)$ que pode ser gerado ao final



(a)

(b)

Figura 5.3: *(a) Conjunto de armadilha TS(4,4) ao final da decodificação iterativa de erros min-sum BP. Os círculos escuros representam nós de variável com erro, os quadrados escuros representam nós de verificação de paridade não satisfeitos e os quadrados claros indicam nós de verificação de paridade satisfeitos; (b) Conjunto de armadilha TS(4,4) corrigível pela decodificação iterativa de apagamentos. Os círculos vazios indicam nós de variável sem erro e sem apagamento, os círculos marcados com* **x** *representam nós de variável com apagamento, os quadrados claros representam nós de verificação de paridade sem conjunto de parada e os quadrados escuros representam nós de verificação de paridade com conjunto de parada.*

da decodificação iterativa de erros *min-sum* BP. Nessa figura, os quadrados claros representam nós de verificação de paridade satisfeitos, os quadrados escuros (cheios) representam nós de verificação de paridade não satisfeitos e os círculos escuros (cheios) representam nós de variável contendo erros.

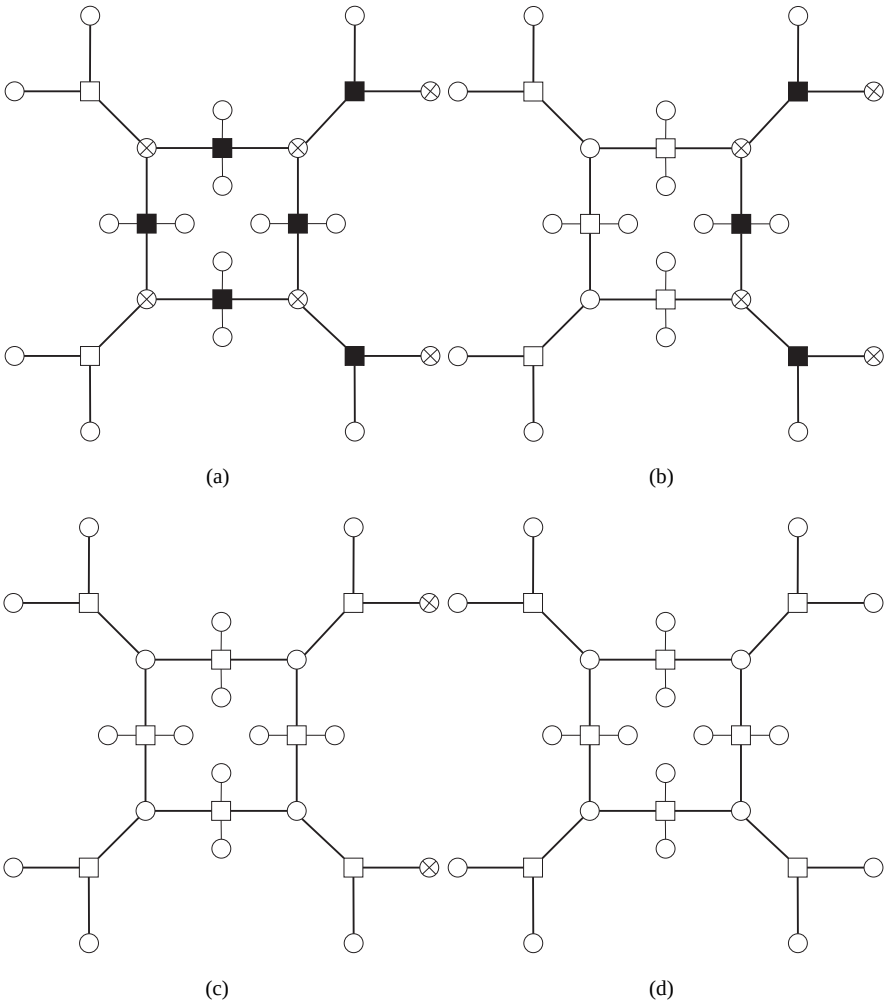


Figura 5.4: *(a) Seleção dos apagamentos; (b) Primeira iteração da decodificação de apagamentos; (c) Segunda Iteração da decodificação de apagamentos; (d) Terceira Iteração e fim da decodificação de apagamentos, com sucesso. Os círculos vazios indicam nós de variável sem erro e sem apagamento, os círculos marcados com* **x** *representam nós de variável com apagamento, os quadrados claros representam nós de verificação de paridade sem conjunto de parada (ou satisfeitos) e os quadrados escuros representam nós de verificação de paridade com conjunto de parada.*

Ao realizar a escolha dos $\mathcal{X}$ *bits* a serem apagados, usando o critério dos menores valores absolutos de QLLR, eventualmente pode-se escolher corretamente os *bits* que de fato estavam em erro, assim como escolher erradamente *bits* que estavam originalmente corretos. A Figura 5.3(b) mostra o resultado de uma possível escolha dos *bits* a serem apagados, que estão representados por círculos marcados por "**x**". Esta figura ilustra um caso em que é possível decodificar todos os apagamentos com sucesso, porquanto há a presença de alguns nós de verificação de paridade contendo apenas um único apagamento.

A Figura 5.4 mostra os passos da decodificação iterativa de apagamentos para o conjunto de

armadilhas $TS(4,4)$ resultante da decodificação iterativa de erros sem sucesso, que é mostrado na Figura 5.3(a). Na Figura 5.4(a), estão mostrados os apagamentos escolhidos, englobando *bits* corretos e os *bits* em erro, pertencentes ao conjunto de armadilhas $TS(4,4)$. Os *bits* apagados estão representados por círculos marcados com "**x**". Na Figura 5.4(b), é mostrado o resultado ao final da primeira iteração, em que são decodificados os primeiros *bits* apagados, que são os únicos *bits* apagados presentes em seus respectivos nós de verificação de paridade. Este procedimento prossegue na segunda e terceira iterações representadas nas Figuras 5.4(c) e 5.4(d), respectivamente, redundando na correção total de todos os apagamentos. Ao final do processo de correção iterativa de apagamentos, os erros originais do conjunto de armadilhas, convertidos em apagamentos, estarão totalmente corrigidos.

Por outro lado, a Figura 5.5 ilustra um caso em que, após o apagamento dos $\mathcal{X}$ *bits*, não há possibilidade de correção, mesmo considerando os nós vizinhos aos *bits* periféricos (externos) do conjunto de armadilhas. Nesta figura, os ramos do conjunto de armadilhas são representados por linhas cheias, ao passo que os ramos dos nós vizinhos são representados por linhas tracejadas. Observa-se que, para os *bits* apagados, não existe equação de verificação de paridade que possa corrigi-los. Nesse caso, o conjunto de armadilhas comporta-se para a decodificação de apagamentos como um conjunto de parada.

5.4 CONSIDERAÇÕES SOBRE O BEC ARTIFICIAL

Um dos aspectos críticos do sistema de decodificação DH proposto é a regra empregada para a atribuição de apagamentos na N-upla $\mathbf{z}$, gerada na saída do BEC artificial quando o primeiro estágio de decodificação *min-sum* BP atinge I_{BP} iterações sem sucesso de decodificação. Tal atribuição de apagamentos é baseada em estimativas de probabilidade geradas pela decodificação iterativa *min-sum* BP.

Em um canal BEC, os símbolos na saída são corretos ($0'$s e $1'$s) ou desconhecidos (Δ's). No caso do BEC artificial aqui introduzido, as saídas geradas continuam sendo $0'$s, $1'$s e Δ's, porém a regra usada para gerá-las deve ser tal que a confiabilidade dos $0'$s e dos $1'$s seja a maior possível e que o número $\mathcal{X}$ de apagamentos seja o maior possível, desde que não inviabilize a operação de correção de apagamentos do decodificador. O que se deseja com esse canal BEC artificial é que o conjunto $\mathbb{S}_{\mathcal{X}}$ contenha todas as posições que foram responsáveis pela falha do algoritmo *min-sum* BP. Isso só é verdadeiro se forem acertadas as alocações feitas para $0'$s e $1'$s da saída deste canal BEC. No processo de decodificação iterativa de erros, usando o algoritmo *min-sum* BP, à medida em que são

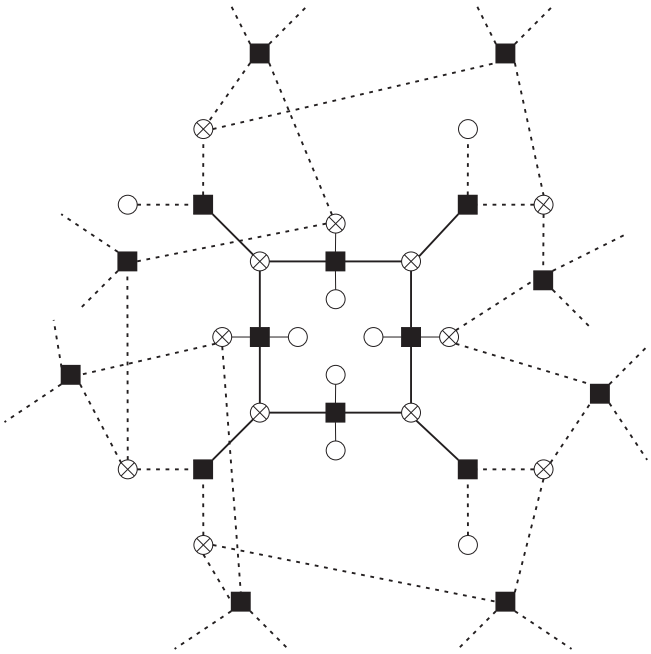


Figura 5.5: *Escolha de apagamentos para um conjunto de armadilhas TS(4, 4), após o final da decodificação iterativa de erros min-sum BP, que resulta em falha de correção de apagamentos. Os círculos vazios indicam nós de variável sem erro e sem apagamento, os círculos marcados com* **x** *representam nós de variável com apagamento e os quadrados escuros representam nós de verificação de paridade com conjunto de parada.*

realizadas mais iterações, em geral uma quantidade maior de valores de QLLR correspondentes aos dígitos da N-upla recebida se distancia mais do valor nulo, especialmente para os dígitos em que a decodificação tem um maior grau de certeza [83]. Tais conclusões foram verificadas nos resultados de simulação de valores de QLLR que estão mostrados na Figura 5.6 e na Figura 5.7. Tais valores de QLLR foram obtidos usando um algoritmo *min-sum* BP para um código LDPC com comprimento de bloco $N = 648$ e taxa igual a $1/2$, denotado por LDPC $(648, 324)$, do padrão IEEE802.11n [42], em canal com RAGB e uma SNR de $3,3$ dB, transmitindo uma palavra-código toda nula, sem que haja perda de generalidade devido à linearidade do código e simetria do canal.

A representação usada nas Figuras 5.6 e 5.7 mostra as posições dos dígitos do código no eixo horizontal (valores de 1 a N) e os valores de QLLR estimados no eixo vertical. A Figura 5.6 mostra os valores de QLLR recebidos antes da decodificação ter início e a Figura 5.7 mostra os valores estimados de QLLR após cinco iterações. Vale ressaltar que um valor de QLLR positivo representa uma decisão correta (*bit* 0), enquanto um valor de QLLR negativo representa um erro de *bit*.

A premissa de concentração de erros nos menores valores de QLLR é confirmada. No entanto, em razão da natureza do canal (RAGB) e da presença de ciclos no grafo de Tanner [5] do referido código LDPC, algumas posições contendo erro podem assumir valores elevados de QLLR. Pode-se

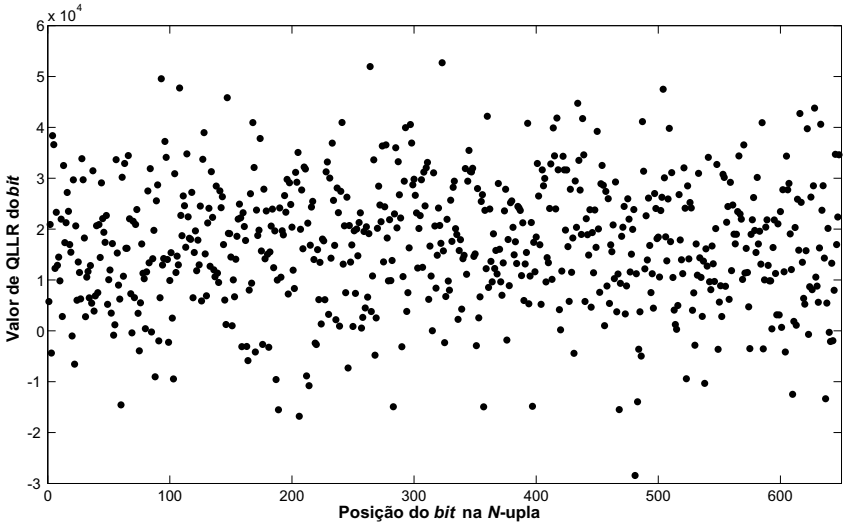


Figura 5.6: *Valores iniciais de QLLR da N-upla para o código LDPC (*648,324*).*

extrair as seguintes observações dos resultados dessas simulações:

- ▷ Após as primeiras iterações, os valores de QLLR dos dígitos da N-upla apresentam tendências. A maioria se distancia do valor zero enquanto uma quantidade pequena permanece oscilando em torno deste valor;
- ▷ O sinal do valor de QLLR dos *bits* permanence inalterado quando excede um certo valor, ou seja, estes *bits* foram confiavelmente decodificados e os seus valores de QLLR tendem a infinito (ou a um valor máximo estipulado) com o andamento das iterações.

Essa análise foi repetida para um número maior de N-uplas decodificadas como não-válidas, mantendo-se as condições de ensaio, ou seja, código LDPC (648,324), canal com RAGB, SNR de $3,3$ dB e transmissão de palavra-código toda nula. Nessa nova simulação, foi analisada a distribuição média dos erros considerando um número superior a 500 N-uplas decodificadas como não-válidas. O resultado, mostrado na Figura 5.8, é um gráfico do tipo "pizza" com os percentuais de concentração de erros dentre as N posições da sequência binária decodificada. Dessa figura, pode-se inferir, por exemplo, que, ao final da decodificação de erros sem sucesso, para um número máximo de 100 iterações deste decodificador *min-sum* BP, o percentual médio dos *bits* em erro que fazem parte dos 100 menores valores de QLLR (em módulo) é maior do que 60% (sessenta porcento), esse percentual passa a ser maior do que 15% (quinze porcento) para a faixa dos 100 a 200 menores valores de QLLR e assim sucessivamente. Portanto, da mesma forma que os resultados mostrados nas Figuras 5.6 e 5.7, obtidos da análise de distribuição de valores de QLLR dos *bits* em apenas uma N-upla, esses resultados exibidos na Figura 5.8 confirmam a premissa de concentração dos erros em

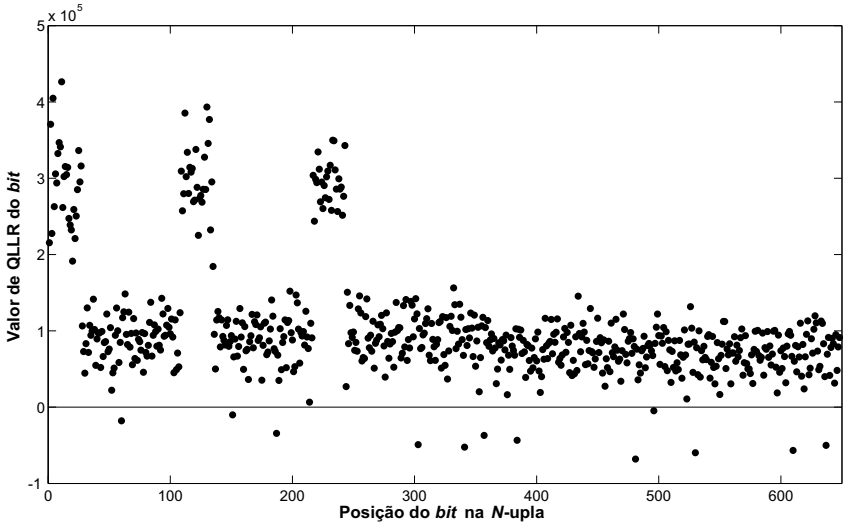


Figura 5.7: *Valores de QLLR da N-upla para o código LDPC (*648,324*) após cinco iterações do algoritmo min-sum BP.*

valores de estimativas de probabilidade próximos a zero.

5.5 ESCOLHA DE UM VALOR PARA $\mathcal{X}$

O parâmetro $\mathcal{X}$ denota a cardinalidade do conjunto $\mathbb{S}_{\mathcal{X}}$ que contém as coordenadas dos dígitos de menor confiabilidade da entrada do BEC, os quais são escolhidos para serem apagados ao fim de uma decodificação *min-sum* BP sem sucesso. A determinação de $\mathcal{X}$ é essencial para que seja obtido um desempenho satisfatório com esta abordagem, uma vez que no processo de decodificação iterativa de apagamentos, à medida em que se aumenta o número de apagamentos, se torna pouco provável encontrar equações de verificação de paridade que tenham apenas um único *bit* apagado e que, portanto, possam ser resolvidas. Do Corolário 4.1, sabe-se que, diferentemente da decodificação por máxima verossimilhança, a decodificação iterativa de apagamentos pode falhar mesmo para padrões de apagamento com cardinalidade $\delta \leq d_{min} - 1$, por conta da característica de distância de parada $s(\mathbf{H})$ do código.

Portanto, a determinação do valor de $\mathcal{X}$, em um primeiro momento, passa necessariamente pela estimação da distância de parada do código. Esse parâmetro, $s(\mathbf{H})$, serve como ponto de partida para a determinação do valor de $\mathcal{X}$, porquanto se sabe que um código linear é capaz de corrigir muitos padrões de apagamentos com cardinalidade superior a $s(\mathbf{H})$ [43].

O procedimento adotado para a determinação do valor de $\mathcal{X}$ prevê duas etapas. A primeira etapa refere-se à estimação da distância de parada $s(\mathbf{H})$ e a segunda etapa refere-se à análise de

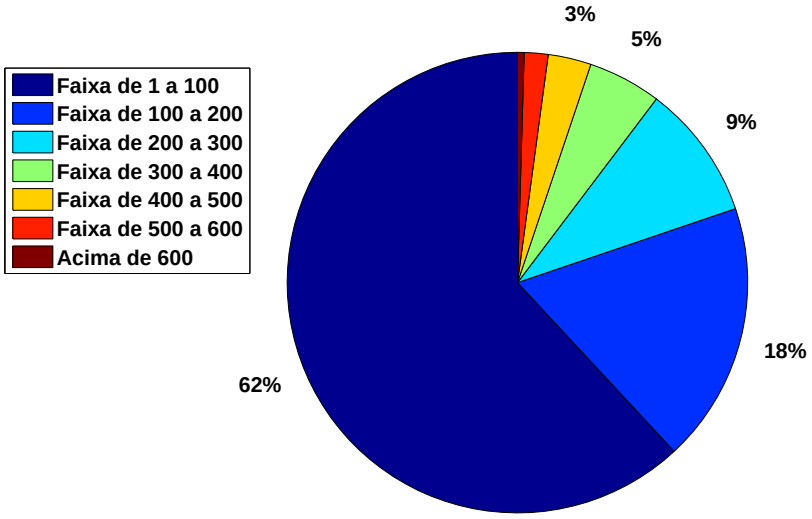


Figura 5.8: *Gráfico pizza da distribuição média de erros ao longo de N-uplas não-válidas, geradas após a decodificação min-sum BP do código LDPC (*648,324*).*

alguns parâmetros de desempenho do sistema DH em função de valores adotados para $\mathcal{X}$, tais que $\mathcal{X} > s(\mathbf{H})$.

5.5.1 Estimação da Distância de Parada $s(\mathbf{H})$

Há alguns métodos que foram desenvolvidos para a identificação e enumeração dos conjuntos de parada e para a determinação do valor da distância de parada [84]–[86]. Nesta tese, é usada uma forma alternativa da estimação do valor de $s(\mathbf{H})$, que é baseada no algoritmo descrito em [86] e que emprega alguns conceitos como, por exemplo, concentração de erros, conjuntos de parada e conjuntos de armadilhas. Esse método para a estimação do valor de $s(\mathbf{H})$ é descrito no Algoritmo 5.2.

Algoritmo 5.2 – Método para Estimativa da Distância de Parada, $s(\mathbf{H})$, para Códigos LDPC

1*: Inicializa os valores de* δ_{start} *,* δ_{step} *e* θ*.*

2*:* $\delta = \delta_{start}$ *e zera contador de* N*-uplas não-válidas.*

3*: Gera-se uma quantidade* K *de valores binários aleatórios.*

4*: Codifica-se os* K *valores, gerando uma* N*-upla.*

5*: Gera-se um sinal por mapeamento desta* N*-upla (*$0 \rightarrow +1$ *e* $1 \rightarrow -1$*).*

6*: Envia-se este sinal por um canal com RAGB.*

7*: Sobre a* N*-upla recebida, contendo valores reais, executa-se a decodificação min-sum BP para erros (min-sum modificado).*

8*: A N-upla decodificada é válida ?*

9*: Se sim, então vai para a linha* 3*.*

10*: Se não, então*

11*: Incrementa contador de* N*-uplas não-válidas na decodificação de erros.*

12*: Valor do contador de* N*-uplas não-válidas* $> \theta$ *?*

13*: Se não, então*

14*: Apaga-se* δ *posições aleatórias da* N*-upla.*

15*: Executa-se a decodificação iterativa de apagamentos sobre a* N*-upla.*

16*: Os apagamentos foram corrigidos ?*

17*: Se sim, então vai para a linha* 3*.*

18*: Se não, então*

19*:* $\delta = \delta - \delta_{step}$*.*

20*: Zera contador de* N*-uplas não-válidas.*

21*: Vai para a linha* 3*.*

22*: Se sim, então* $s(\mathbf{H}) \approx \delta$ *e vai para* ***FIM****.*

23*:* ***FIM****.*

Há algumas considerações a serem feitas em relação a esse algoritmo, que são as seguintes:

- Primeiramente, define-se um número inicial de apagamentos , $\delta = \delta_{start}$, o decremento de número de apagamentos, δ_{step}, e o critério de parada do algoritmo, θ, que representa o número mínimo de sucessos na correção de apagamentos que torne aceitável a estimativa para o valor de $s(\mathbf{H})$. Este valor de θ é arbitrado. Nas simulações realizadas, foi adotado $\theta = 1000$;
- O número de apagamentos, δ, é decrementado do valor δ_{step} tão logo ocorra uma falha de decodi-

Tabela 5.1: *Valores Estimados para* $s(\mathbf{H})$.

Código	$\delta_{\mathbf{start}}$	$\delta_{\mathbf{step}}$	$\mathbf{s(H)}$ [1]	$\mathbf{d_{min}}$ [2]
LDPC (1944,972)	90	5	45	27
LDPC (1944,1296)	70	5	30	[3]
LDPC (1296,648)	90	5	30	23
LDPC (1296,864)	90	5	20	[3]
LDPC (648,324)	60	5	20	15
LDPC (648,432)	40	5	15	12

[1]Valores de $s(\mathbf{H})$ obtidos por simulação. Foi adotado $\theta = 1000$ N-uplas.

[2]Valores de distância mínima, d_{min}, obtidos de [87].

[3]Valores não encontrados na literatura.

ficação de apagamentos, ou seja, $\delta = \delta - \delta_{step}$. Neste caso, um novo ciclo de testes será iniciado para este valor recalculado de δ;

- ▷ Adota-se para o canal com RAGB uma razão sinal-ruído que faça com que o código LDPC atue na região próxima ao patamar de erros. Isto garante que haja a geração de padrões de erros (conjuntos de armadilhas) de baixa cardinalidade e, ao mesmo tempo, possibilita uma maior eficiência do algoritmo, em termos de custo computacional;
- ▷ Na decodificação de erros do tipo *min-sum* BP, adota-se um número grande de iterações (em torno de 200 iterações) que assegure que, em caso de falha, haja a presença de conjuntos de armadilhas;
- ▷ Os *bits* a serem apagados, antes que ocorra a decodificação de apagamentos, são escolhidos dentre os δ menores valores absolutos de estimativas de QLLR gerados ao fim da decodificação *min-sum* BP de erros;
- ▷ O valor de $s(\mathbf{H})$ estimado será, portanto, correspondente ao último valor para δ (cardinalidade do padrão de apagamentos), para o qual tiverem ocorridos θ sucessos na decodificação iterativa de apagamentos.

Esse método foi adotado para as estimativas dos valores de $s(\mathbf{H})$ de alguns dos códigos LDPC usados no Padrão IEEE802.11n. Os resultados obtidos e os valores adotados para os parâmetros de inicialização, tais como δ_{start} e δ_{step}, estão listados na Tabela 5.1.

Nota-se que o método desenvolvido para a obtenção dos valores estimados para $s(\mathbf{H})$ resulta em valores próximos aos especificados na literatura.

5.5.2 AVALIAÇÃO DO SISTEMA DH EM FUNÇÃO DE $\mathcal{X}$

A segunda etapa consiste em simulação computacional do algoritmo DH para diversos valores de $\mathcal{X}$ maiores do que $s(\mathbf{H})$ avaliando o percentual de detecção de padrões de erros, o percentual de correção de apagamentos e o percentual de correção dos erros na N-upla recebida.

Definição 5.2

*Uma N-upla contendo erros, resultante da decodificação min-sum BP, é definida como **detectada** se as posições de todos os seus erros estiverem contidas em $\mathcal{S}_{\mathcal{X}}$.* □

Definição 5.3

*Uma N-upla contendo erros, resultante da decodificação min-sum BP, é definida como **corrigida** caso possa ser **detectada** e a palavra binária $\hat{\mathbf{c}}$ resultante da decodificação iterativa de apagamentos pertencer ao código $\mathcal{C}$, i.e., se $\hat{\mathbf{c}} \in \mathcal{C}$.* □

No Algoritmo 5.3 estão descritos os passos adotados para a obtenção dos parâmetros mencionados.

Preliminarmente, é importante fazer algumas considerações sobre o cálculo desses parâmetros:

- O percentual de detecção dos padrões de erros é dado pela relação do número de N-uplas **detectadas** pelo número total de N-uplas não-válidas, resultantes da decodificação de erros *min-sum* BP;
- O percentual de correção de padrões com $\mathcal{X}$ apagamentos é dado pela relação do número de N-uplas apagadas (contendo padrões com $\mathcal{X}$ apagamentos), que tenham sido corrigidas pela decodificação iterativa de apagamentos, pelo número total de N-uplas não-válidas, resultantes da decodificação de erros *min-sum* BP;
- O percentual de correção de padrões de erros nas N-uplas é dado pela relação do número de N-uplas com $\mathcal{X}$ apagamentos, que tenham sido **corrigidas**, pelo número total de N-uplas não-válidas, resultantes da decodificação de erros *min-sum* BP.

Algoritmo 5.3 – Levantamento de parâmetros para a estimação do valor de $\mathcal{X}$

1*: Fixa-se valores para $\mathcal{X}$ e para θ.*

2*: Gera-se uma quantidade K de valores binários aleatórios.*

3*: Codifica-se os K valores, gerando uma N-upla.*

4*: Gera-se um sinal por mapeamento desta N-upla ($0 \rightarrow +1$ e $1 \rightarrow -1$).*

5*: Envia-se este sinal por um canal com RAGB.*

6*: Sobre a N-upla recebida, contendo valores reais, executa-se a decodificação min-sum BP (min-sum modificado).*

7*: A N-upla decodificada é válida ?*

8*: Se sim, então vai para a linha 2.* ◁ *Não executa a decod.de apagamentos*

9*: Se não, então*

10*: Computa como N-upla não-válida gerada na decodificação de erros.*

11*: Apaga-se as $\mathcal{X}$ posições da N-upla.*

12*: Os erros estão inclusos nas $\mathcal{X}$ posições ?*

13*: Se sim, então computa como N-upla **detectada**.*

14*: Executa-se a decodificação iterativa de apagamentos sobre a N-upla.*

15*: Os apagamentos foram corrigidos ?*

16*: Se sim, então*

17*: Computa como N-upla com correção total de apagamentos.*

18*: A N-upla decodificada foi **corrigida**?*

19*: Se sim, então computa como N-upla **corrigida**.*

20*: Número de N-uplas não-válidas $\geq \theta$?* ◁ *Critério de Parada*

21*: Se não, então vai para a linha 2.*

22*: Se sim, então*

23*: Computa $\%$ de detecção de padrões de erros nas N-uplas.*

24*: Computa $\%$ de correção de N-uplas com $\mathcal{X}$-apagamentos.*

25*: Computa $\%$ de correção de padrões de erros nas N-uplas.*

26*: **FIM**.*

Vale salientar que o parâmetro relativo ao percentual de correção de padrões com $\mathcal{X}$ apagamentos leva em conta apenas a correção total de apagamentos, não importando se a N-upla resultante tenha sido **corrigida**.

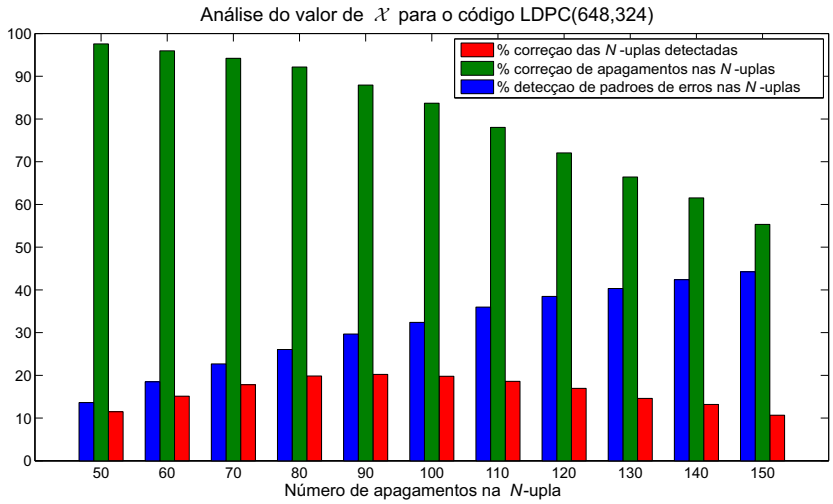


Figura 5.9: *Percentuais de detecção, correção de apagamentos e correção de erros em função do valor de* $\mathcal{X}$ *para o código LDPC* $(648, 324)$.

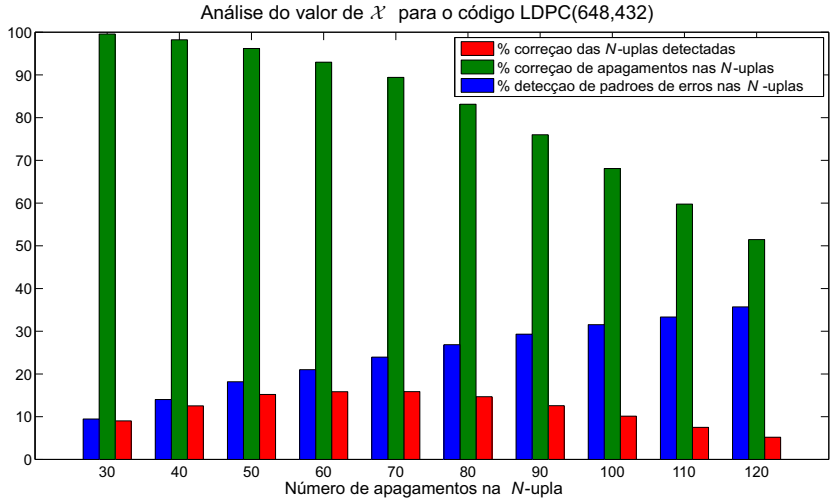


Figura 5.10: *Percentuais de detecção, correção de apagamentos e correção de erros em função do valor de* $\mathcal{X}$ *para o código LDPC* $(648, 432)$.

As Figuras 5.9, 5.10, 5.11, 5.12, 5.13 e 5.14 exibem os três percentuais para os códigos LDPC do padrão IEEE$802.11n$. Para esta análise foi adotado um valor de $\theta = 1000$ como critério de parada. Com relação aos resultados mostrados nas Figuras 5.9 a 5.14, podem ser feitas as seguintes considerações:

- Para todos os casos analisados, o percentual máximo de correção de apagamentos da N-upla é próximo a 100%, o que já era esperado dado que a cardinalidade $\mathcal{X}$ dos padrões de apagamentos é superior aos respectivos valores de $s(\mathbf{H})$ dos códigos LDPC sob análise;
- Para todos os casos analisados, também se observa que o percentual de correção de apagamentos

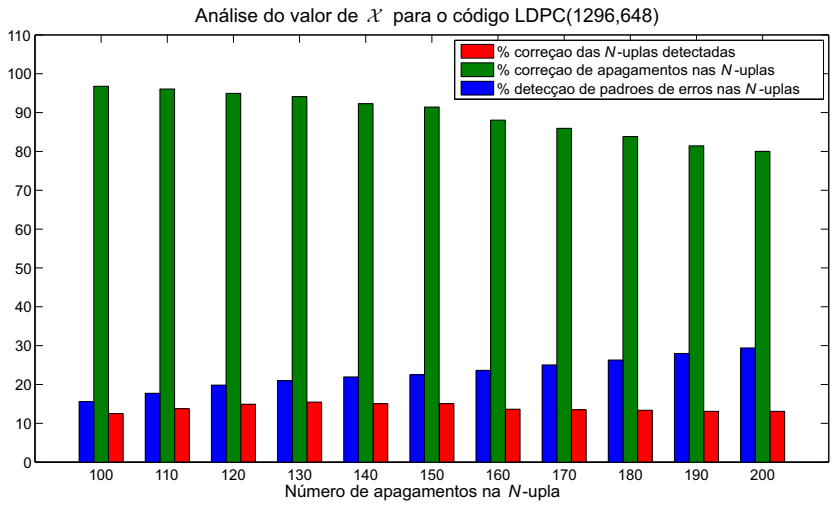


Figura 5.11: *Percentuais de detecção, correção de apagamentos e correção de erros em função do valor de* $\mathcal{X}$ *para o código LDPC* $(1296, 648)$.

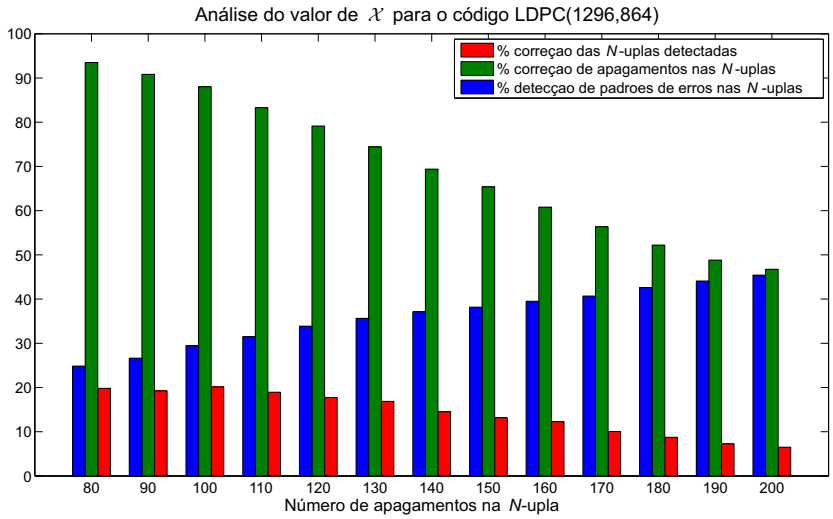


Figura 5.12: *Percentuais de detecção, correção de apagamentos e correção de erros em função do valor de* $\mathcal{X}$ *para o código LDPC* $(1296, 864)$.

da N-upla decai à medida que se aumenta a cardinalidade do padrão de apagamentos (aumenta o valor de $\mathcal{X}$), em função do aparecimento de conjuntos de parada e, para um número menor de casos, devido à ambiguidade de decodificação de apagamentos;

- Para todos os casos analisados, o percentual de detecção de N-uplas cresce à medida que aumenta o valor de $\mathcal{X}$, ou seja, aumenta a probabilidade de que, dentre estes $\mathcal{X}$ *bits* escolhidos para apagamentos, sejam selecionados os *bits* em erro. Mesmo assim, estes percentuais, ainda que para valores elevados de $\mathcal{X}$, são relativamente baixos, por conta da existência de padrões de erros em que alguns dos *bits* assumem valores elevados de QLLR;

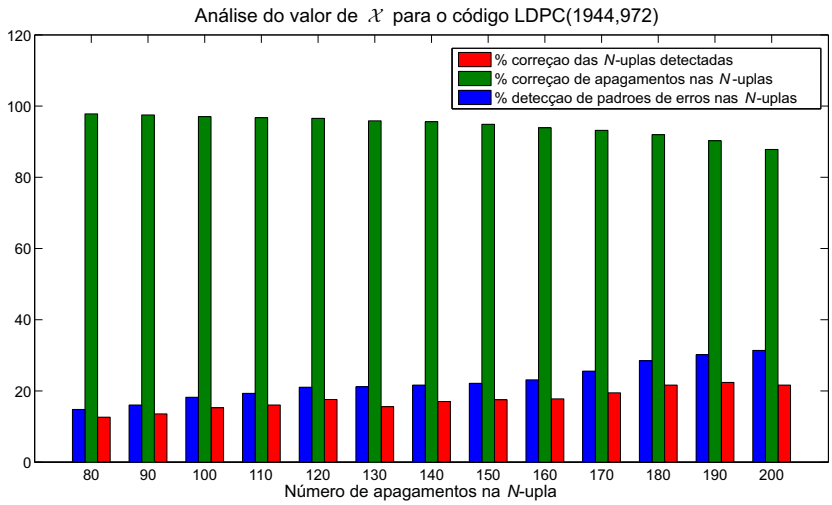


Figura 5.13: *Percentuais de detecção, correção de apagamentos e correção de erros em função do valor de* $\mathcal{X}$ *para o código LDPC* $(1944, 972)$.

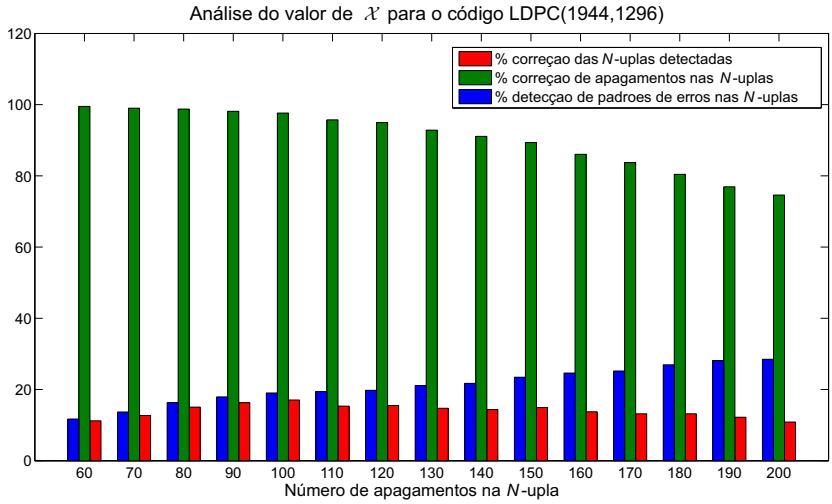


Figura 5.14: *Percentuais de detecção, correção de apagamentos e correção de erros em função do valor de* $\mathcal{X}$ *para o código LDPC* $(1944, 1296)$.

- Para todos os casos analisados, existe um valor de $\mathcal{X}$ para o qual se atinge o máximo percentual de correção de erros nas N-uplas. Para valores menores de $\mathcal{X}$, apesar de haver boa correção de apagamentos, há baixa detecção de erros, assim como para valores maiores de $\mathcal{X}$ ocorre o oposto, ou seja, ótima detecção de erros mas com baixa capacidade de correção de apagamentos.

A título de ilustração, considerando o caso do código LDPC $(648{,}324)$, observa-se na Figura 5.9 que o percentual de correção de erros atinge um valor máximo de aproximadamente 20% para $\mathcal{X} = 90$, que está associado a um percentual aproximado de 90% para a correção de apagamentos e a um percentual aproximado de 30% para a detecção de erros. Esta simulação, em particular, foi

Tabela 5.2: *Valores de $\mathcal{X}$ para os Códigos LDPC do Padrão IEEE*$802.11n$*.*

Código	$\mathcal{X}$	$\mathcal{X}/\mathbf{s}(\mathbf{H})$
LDPC (1944,972)	190	$4,2$
LDPC (1944,1296)	100	$3,3$
LDPC (1296,648)	130	$4,3$
LDPC (1296,864)	100	$5,0$
LDPC (648,324)	90	$4,5$
LDPC (648,432)	70	$4,7$

realizada para uma relação sinal-ruído de $3,2$dB em canal com RAGB. O valor $\mathcal{X} = 90$ corresponde a $4,5 \times s(\mathbf{H})$.

Os valores do parâmetro $\mathcal{X}$ que maximizam a correção de padrões de erros para os códigos LDPC em estudo, foram extraídos dessas simulações e estão listados na Tabela 5.2. Nessa tabela, a coluna $\mathcal{X}/s(\mathbf{H})$ relaciona o valor encontrado para $\mathcal{X}$ com o valor estimado para a distância de parada $s(\mathbf{H})$ e confirma que, para esses códigos LDPC do padrão IEEE802.11n, $\mathcal{X} > s(\mathbf{H})$.

CAPÍTULO 6

ANÁLISE, RESULTADOS E COMENTÁRIOS

Com o intuito de avaliar o modelo de decodificação híbrida (DH) iterativa proposto nesta tese, são realizadas simulações em computador para alguns dos códigos LDPC empregados no padrão IEEE802.11n [42]. Tais códigos são irregulares do tipo RA (*Repeat-Accumulate*) [88], [13] de comprimento, N, igual a 1944, 1296 ou 648 e taxas de $1/2$, $2/3$, $3/4$ ou $5/6$.

As principais simulações realizadas estão divididas em duas fases. Na primeira fase, faz-se a análise comparativa do desempenho do sistema DH com o do sistema convencional de decodificação iterativa do tipo *min-sum* BP e, na segunda fase, realiza-se análise comparativa do desempenho do sistema DH com o do sistema de decodificação concatenada em série. Em ambos os casos, é empregado um canal com RAGB.

São realizadas também análises adicionais com respeito ao tempo de execução do sistema DH, comparando-o com o que é obtido para o decodificador convencional do tipo *min-sum* BP, com respeito ao número máximo de ciclos de operação do sistema DH, com respeito ao impacto dos eventos de erros resultantes de conjuntos de armadilha sobre o desempenho do sistema DH e, por último, referente à atuação do sistema DH na região de patamar de erros dos códigos LDPC.

Em todas as referidas análises, são empregados para o sistema DH os valores encontrados para $\mathcal{X}$ que estão listados na Tabela 5.2. Do mesmo modo, para os decodificadores *min-sum* BP, seja do sistema DH ou dos outros sistemas de decodificação para códigos LDPC usados na comparação, é empregado o algorimo *min-sum* modificado que é descrito na Seção 3.6.

6.1 ANÁLISE COMPARATIVA COM SISTEMAS DE DECODIFICAÇÃO ITERATIVA *min-sum* BP CONVENCIONAL

6.1.1 INTRODUÇÃO

O sistema de decodificação DH, empregando um número bastante reduzido de iterações no estágio de decodificação *min-sum* BP e operando na região de patamar de erros dos códigos LDPC, pode ter desempenho comparável ao de sistemas convencionais de decodificação *min-sum* BP que empregam um número maior de iterações. Com o objetivo de corroborar essa afirmação são realizadas simulações em computador utilizando alguns códigos do padrão IEEE802.11n.

6.1.2 MODELO DE SIMULAÇÃO

Para a análise comparativa desses sistemas de decodificação, são empregados os modelos mostrados nas Figuras 6.1(a) e 6.1(b), que, a menos do decodificador LDPC, são idênticos e empregam as mesmas técnicas de codificação LDPC e de mapeamento de binário para inteiro (*bit* 0 para $+1$ e *bit* 1 para -1). Nesses modelos, como citado anteriormente, é empregado o canal com RAGB.

6.1.3 RESULTADOS E COMENTÁRIOS

Em uma primeira fase, o desempenho do sistema DH é comparado ao de um decodificador *min-sum* BP de referência que funciona com um número global de iterações igual a 200. Na etapa de decodificação *min-sum* BP do sistema DH são consideradas duas situações: a primeira emprega, no máximo, 25 iterações e a segunda, no máximo, 50 iterações. Nessa análise, para cada um desses referidos códigos LDPC, são geradas quatro curvas de desempenho, sendo a primeira para um decodificador *min-sum* BP com um número reduzido de iterações (25 ou 50 iterações), a segunda e a terceira para o decodificador DH para um ciclo e para dois ciclos de decodificação, respectivamente, e a quarta para o decodificador *min-sum* BP de referência. Nessa análise, para fins de avaliação de desempenho, toma-se uma taxa WER igual a 3×10^{-6} que é o *nível de comparação*. Nos gráficos gerados, o eixo horizontal representa a SNR, em dB, e o eixo vertical representa a taxa de erro por palavra.

Em todos os casos analisados, o número global de iterações da etapa de decodificação *min-sum* BP do sistema DH, mesmo quando opera em dois ciclos, é inferior ao número de iterações do decodificador *min-sum* BP de referência. No caso do sistema DH operando em dois ciclos e tendo uma etapa de decodificação *min-sum* BP com no máximo 25 iterações, o número global de iterações

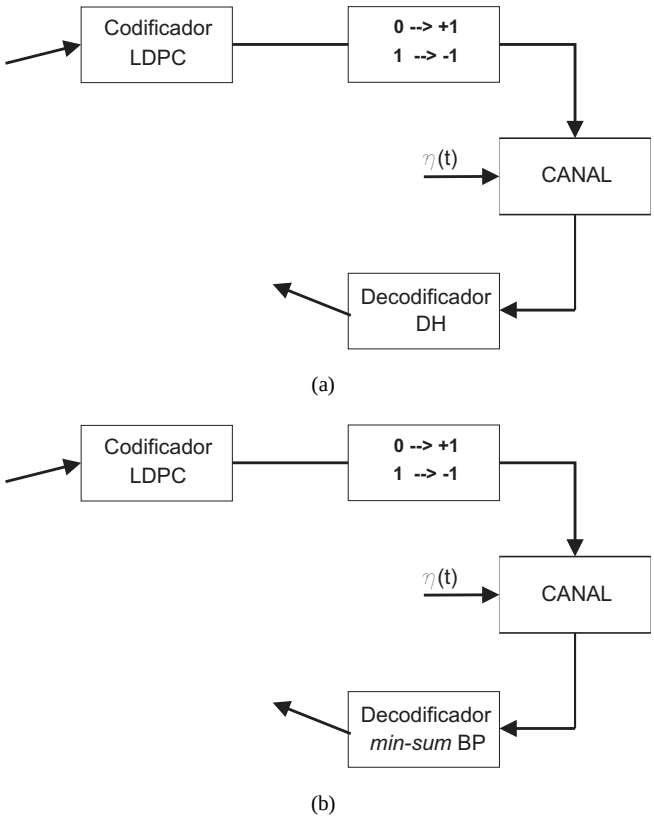


Figura 6.1: *(a) Modelo de simulação do sistema DH em canal com RAGB; (b) Modelo de simulação do decodificador min-sum BP em canal com RAGB.*

é igual a 50, considerando-se apenas aquelas executadas na etapa de decodificação *min-sum* BP. Esse valor corresponde a $1/4$ do número de iterações empregado para o decodificador *min-sum* BP de referência. Do mesmo modo, caso seja empregado um estágio *min-sum* BP com 50 iterações para o sistema DH que opera em dois ciclos, o número global de iterações é igual a 100. Esse valor corresponde à metade do número de iterações do decodificador *min-sum* BP de referência.

As Figuras 6.2 e 6.3 mostram curvas de desempenho para o código LDPC (1944,972). Nesse caso, o decodificador *min-sum* BP de referência atinge um desempenho igual ao *nível de comparação* para uma SNR de aproximadamente $2,33$ dB.

Para conseguir esse desempenho, observa-se na Figura 6.2 que a SNR é de aproximadamente $2,66$ dB para o decodificador *min-sum* BP empregando apenas 25 iterações. Para o sistema DH, usando uma etapa de decodificação *min-sum* BP com 25 iterações, a SNR é de aproximadamente $2,56$ dB para um ciclo de operação e é igual a $2,46$ dB para dois ciclos de operação. Este último resultado é $0,13$ dB inferior ao resultado obtido para a decodificador *min-sum* BP de referência.

Do mesmo modo, para atingir um desempenho igual ao *nível de comparação*, observa-se na

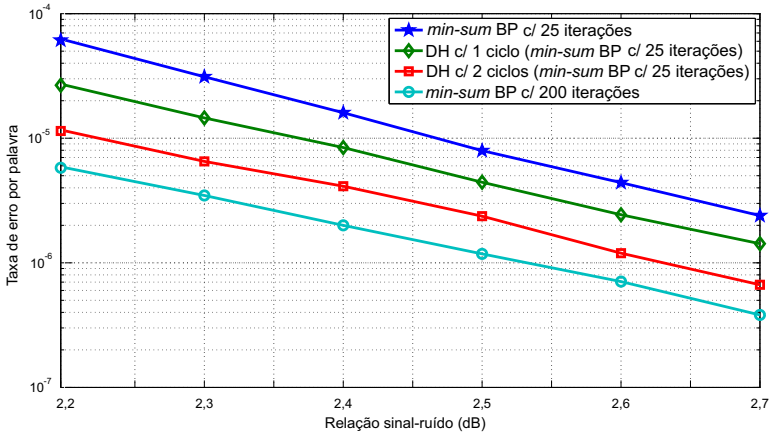


Figura 6.2: *Curvas de desempenho para o código LDPC* $(1944, 972)$ *na região de patamar de erros, para DH usando min-sum BP com* 25 *iterações.*

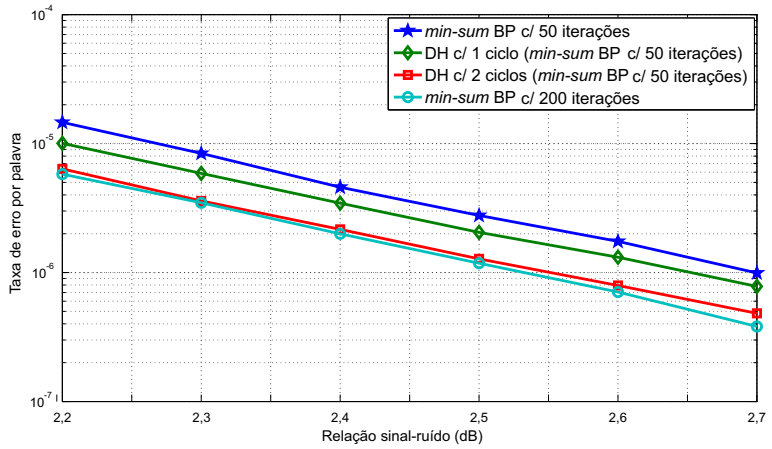


Figura 6.3: *Curvas de desempenho para o código LDPC* $(1944, 972)$ *na região de patamar de erros, para DH usando min-sum BP com* 50 *iterações.*

Figura 6.3 que a SNR é de aproximadamente $2,48$ dB para o decodificador *min-sum* BP com 50 iterações. Para o sistema DH, usando uma etapa de decodificação *min-sum* BP com 50 iterações, a SNR é de aproximadamente $2,42$ dB para um ciclo de operação e é igual a $2,33$ dB para dois ciclos de operação. Este último resultado é equivalente ao resultado obtido para o decodificador *min-sum* BP de referência.

As Figuras 6.4 e 6.5 mostram curvas de desempenho para o código LDPC (1944,1296). Nesse caso, o decodificador *min-sum* BP de referência atinge um desempenho igual ao *nível de comparação* para uma SNR de aproximadamente $2,85$ dB.

Para conseguir esse desempenho, observa-se na Figura 6.4 que a SNR é de aproximadamente $3,10$ dB para o decodificador *min-sum* BP empregando 25 iterações. Para o sistema DH, empregando uma etapa de decodificação *min-sum* BP com 25 iterações, a SNR é de aproximadamente $3,05$ dB

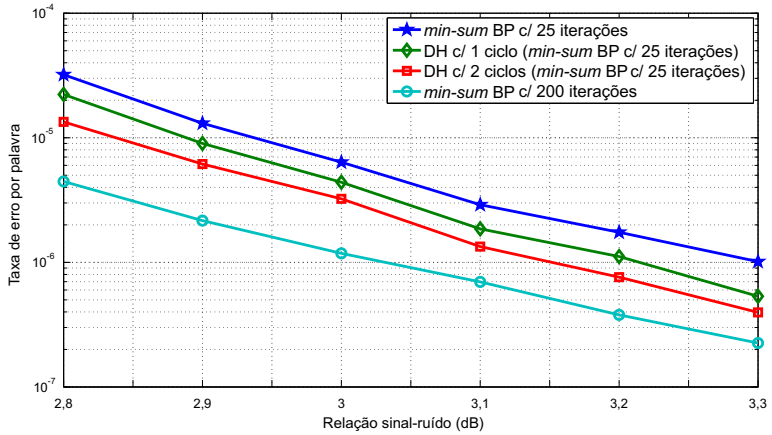


Figura 6.4: *Curvas de desempenho para o código LDPC* $(1944, 1296)$ *na região de patamar de erros, para DH usando min-sum BP com* 25 *iterações.*

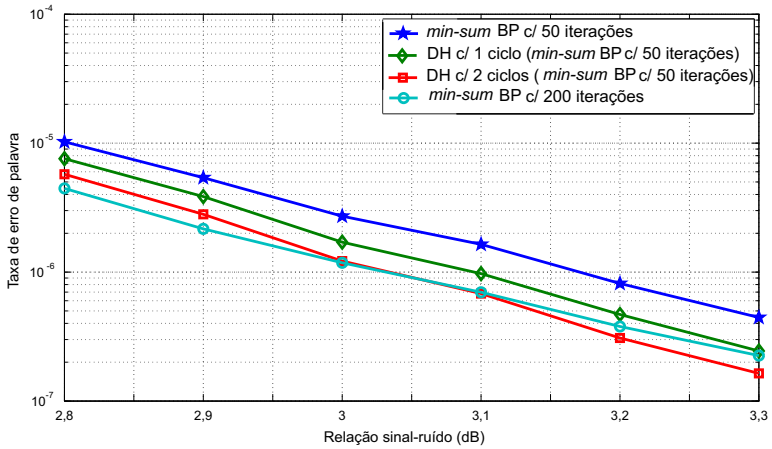


Figura 6.5: *Curvas de desempenho para o código LDPC* $(1944, 1296)$ *na região de patamar de erros, para DH usando min-sum BP com* 50 *iterações.*

para um ciclo de operação e é igual a $3,0$ dB para dois ciclos de operação. Este último resultado é $0,15$ dB inferior ao resultado obtido para o decodificador *min-sum* BP de referência.

Do mesmo modo, para atingir um desempenho igual ao *nível de comparação*, observa-se na Figura 6.5 que a SNR é de aproximadamente $2,97$ dB para o decodificador *min-sum* BP empregando 50 iterações. Para o sistema DH, usando uma etapa de decodificação *min-sum* BP com 50 iterações, a SNR é de aproximadamente $2,93$ dB para um ciclo de operação e é igual a $2,88$ dB para dois ciclos de operação. Este último resultado é $0,03$ dB inferior ao resultado obtido para o decodificador *min-sum* BP de referência.

As Figuras 6.6 e 6.7 mostram curvas de desempenho para o código LDPC (1296,648), as Figuras 6.8 e 6.9 mostram curvas de desempenho para o código LDPC (1296,864), as Figuras 6.10 e 6.11 mostram curvas de desempenho para o código LDPC (648,324) e as Figuras 6.12 e 6.13 mos-

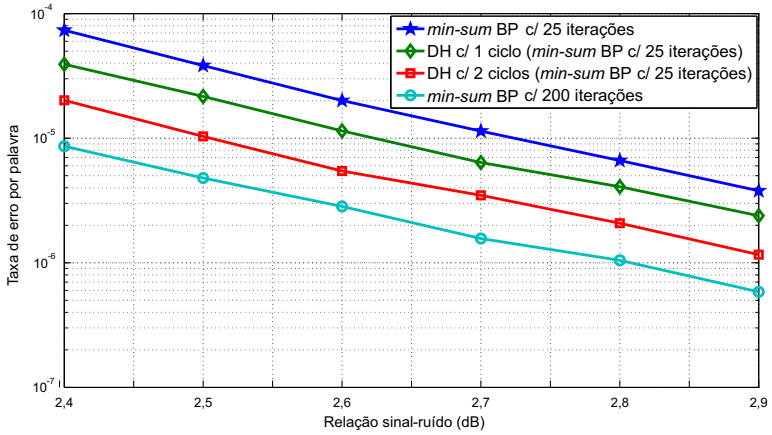


Figura 6.6: *Curvas de desempenho para o código LDPC* $(1296, 648)$ *na região de patamar de erros, para DH usando min-sum BP com* 25 *iterações.*

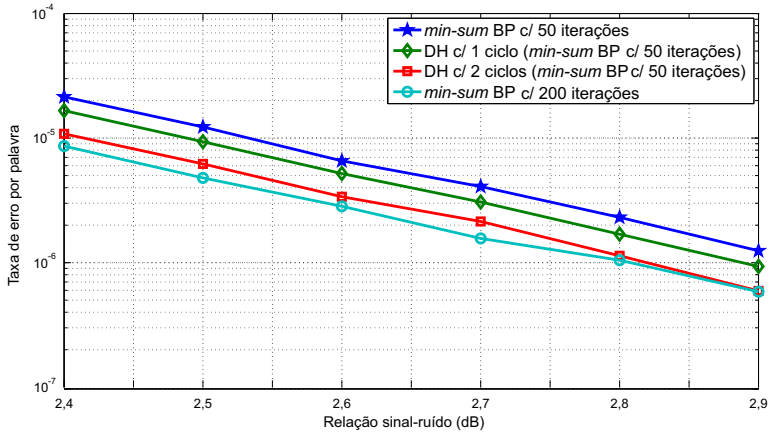


Figura 6.7: *Curvas de desempenho para o código LDPC* $(1296, 648)$ *na região de patamar de erros, para DH usando min-sum BP com* 50 *iterações.*

tram curvas de desempenho para o código LDPC (648,432). Para esses códigos LDPC, a exemplo dos outros analisados anteriormente, o desempenho do sistema DH, operando em dois ciclos e com uma etapa *min-sum* BP com 50 iterações, é equivalente ao do decodificador *min-sum* BP de referência, que é executado com um número máximo de iterações igual a 200.

Em uma segunda fase, uma análise comparativa de desempenho entre o sistema DH e o sistema convencional de decodificação *min-sum* BP é executada para os códigos LDPC (648,324), LDPC (648,432), LDPC (1296,648) e LDPC (1296,864), considerando-se um número ainda menor de iterações na etapa de decodificação *min-sum* BP. Na decodificação dos códigos LDPC (648,324) e LDPC (648,432) são empregados $I_{\mathrm{BP}} = 5$, $I_{\mathrm{BP}} = 12$ e $I_{\mathrm{BP}} = 20$ iterações na etapa de decodificação *min-sum* BP do sistema DH. Para esses casos, o decodificador *min-sum* BP de referência opera, respectivamente, com $I_{\mathrm{BP}} = 20$, $I_{\mathrm{BP}} = 50$ e $I_{\mathrm{BP}} = 80$ iterações, que, portanto, correspondem ao dobro

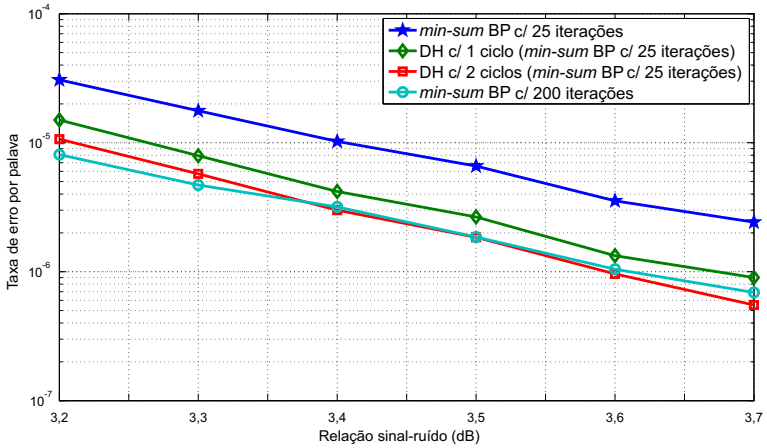


Figura 6.8: *Curvas de desempenho para o código LDPC* $(1296, 864)$ *na região de patamar de erros, para DH usando min-sum BP com* 25 *iterações.*

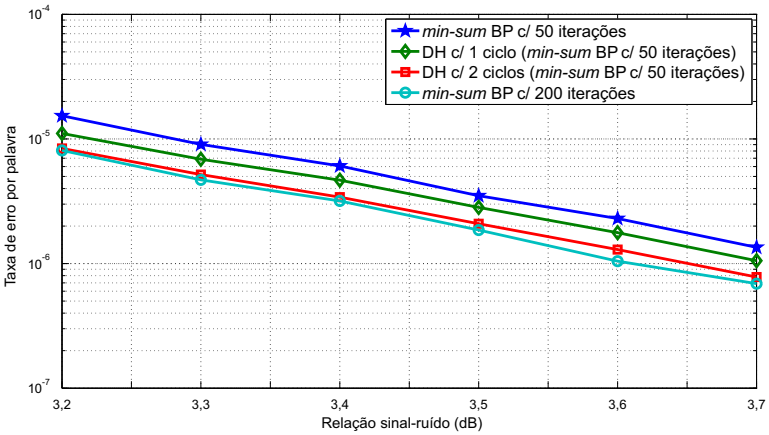


Figura 6.9: *Curvas de desempenho para o código LDPC* $(1296, 864)$ *na região de patamar de erros, para DH usando min-sum BP com* 50 *iterações.*

do que é empregado para o sistema DH operando em dois ciclos. Para os códigos LDPC (1296,648) e LDPC (1296,864) é realizada a simulação apenas para $I_{\text{BP}} = 12$ iterações do estágio *min-sum* BP do sistema DH e tomado como referência o decodificador *min-sum* BP com $I_{\text{BP}} = 50$ iterações.

A Figura 6.14 mostra as curvas de desempenho para o código LDPC (648,324) em quatro situações: decodificação *min-sum* BP empregando $I_{\text{BP}} = 5$ iterações, decodificação empregando o sistema DH com um ciclo de operação e para dois ciclos de operação, em ambos os casos utilizando um estágio de decodificação *min-sum* BP com $I_{\text{BP}} = 5$ iterações, e decodificação *min-sum* BP com $I_{\text{BP}} = 20$ iterações. Este último corresponde ao decodificador de referência. Observa-se que o resultado para a decodificação *min-sum* BP para 5 iterações é inferior em mais de 1 dB ao daquele obtido para a decodificação com sistema DH para um ciclo de operação. Para ilustrar, tomando como referência a taxa de erro por palavra igual a 1×10^{-2}, esse desempenho é conseguido para o deco-

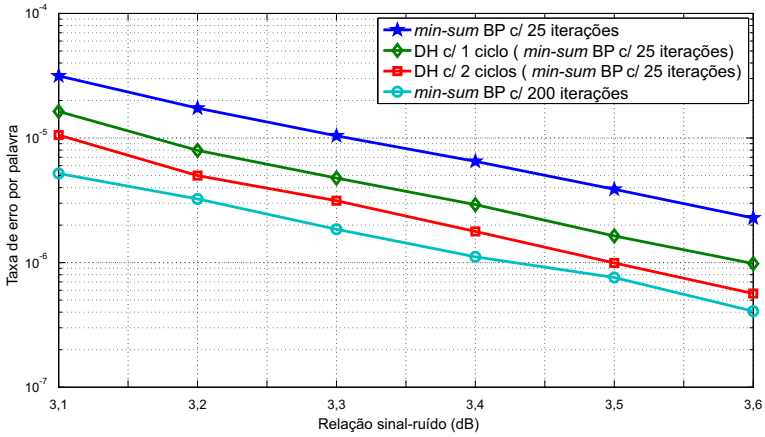


Figura 6.10: *Curvas de desempenho para o código LDPC* $(648, 324)$ *na região de patamar de erros, para DH usando min-sum BP com* 25 *iterações.*

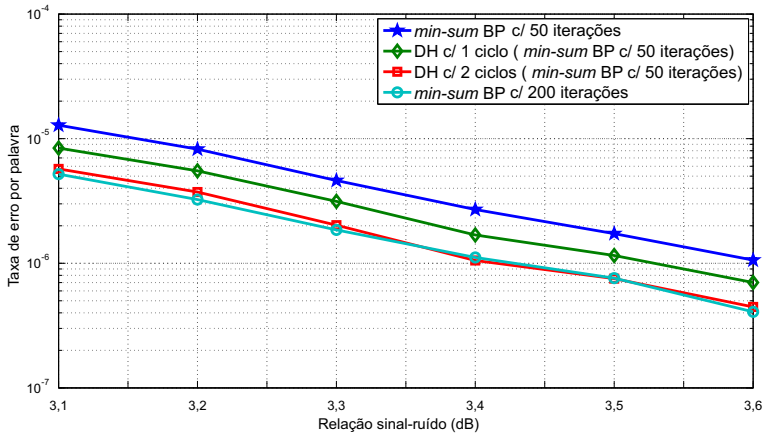


Figura 6.11: *Curvas de desempenho para o código LDPC* $(648, 324)$ *na região de patamar de erros, para DH usando min-sum BP com* 50 *iterações.*

dificador *min-sum* BP com 5 iterações para uma SNR um pouco maior do que 4 dB, ao passo que para o decodificador com o sistema DH para um ciclo de operação isto é atingido para uma SNR um pouco menor do que $3,1$ dB. Do mesmo modo, observa-se que o resultado para o decodificador com sistema DH com dois ciclos de operação é superior em aproximadamente $0,8$ dB em relação ao do decodificador DH com um ciclo de operação. Isto pode ser observado, por exemplo, para uma taxa de erro de palavra igual a 4×10^{-4}. No entanto, o resultado para esse decodificador DH com dois ciclos de operação, empregando no máximo 10 iterações no estágio *min-sum* BP, é cerca de $0,4$ dB inferior ao daquele obtido para o decodificador *min-sum* BP de referência que emprega 20 iterações.

A Figura 6.15 é similar à Figura 6.14 porém exibindo as seguintes curvas de desempenho: decodificação *min-sum* BP empregando $I_{\text{BP}} = 12$ iterações, decodificação empregando o sistema DH, com um estágio de decodificação *min-sum* BP com $I_{\text{BP}} = 12$ iterações, para um e dois ciclos de

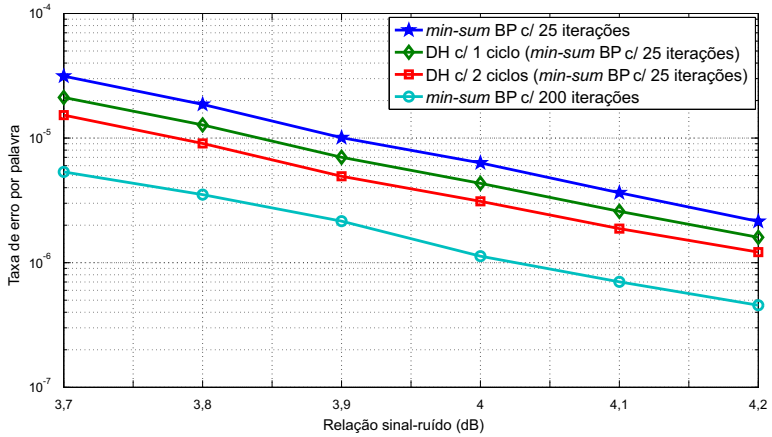


Figura 6.12: *Curvas de desempenho para o código LDPC* $(648, 432)$ *na região de patamar de erros, para DH usando min-sum BP com* 25 *iterações.*

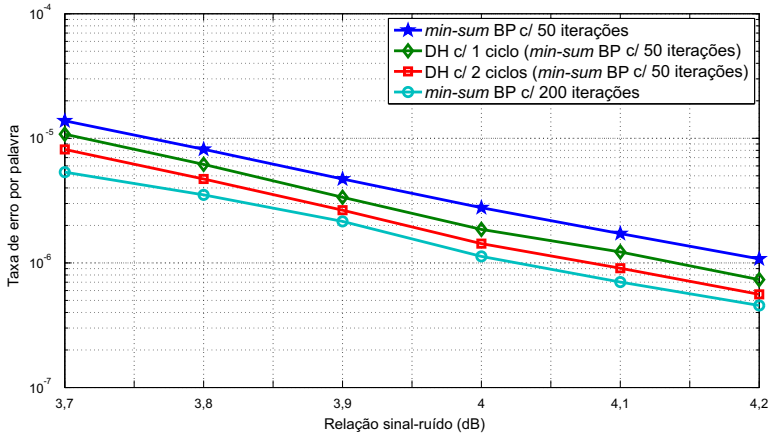


Figura 6.13: *Curvas de desempenho para o código LDPC* $(648, 432)$ *na região de patamar de erros, para DH usando min-sum BP com* 50 *iterações.*

operação e decodificação *min-sum* BP com $I_{\text{BP}} = 50$ iterações. Este último corresponde ao decodificador de referência. Observa-se que o resultado para a decodificação *min-sum* BP para 12 iterações é aproximadamente $0,2$ dB inferior ao do decodificador com sistema DH para um ciclo de operação. Para ilustrar toma-se, por exemplo, como referência a taxa de erro por palavra igual a 1×10^{-5}. Esse desempenho é conseguido para o decodificador *min-sum* BP com 12 iterações para uma SNR de $3,7$ dB, ao passo que para o decodificador com o sistema DH para um ciclo de operação isto é atingido para uma SNR de $3,5$ dB. Do mesmo modo, observa-se que o resultado para o decodificador com sistema DH com dois ciclos de operação é superior em aproximadamente $0,3$ dB em relação ao do decodificador DH com um ciclo de operação. Isto pode ser observado, por exemplo, para uma taxa de erro de palavra igual a 2×10^{-6}. Outrossim, o resultado desse decodificador DH com dois ciclos de operação, empregando no máximo 24 iterações no estágio *min-sum* BP, é equivalente ao daquele

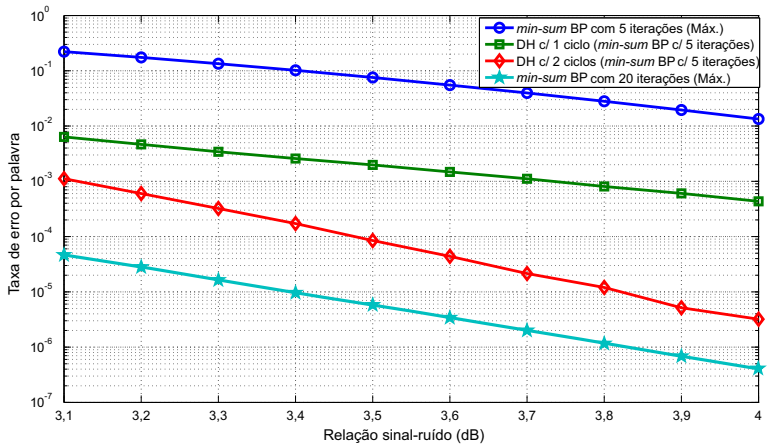


Figura 6.14: *Curvas de desempenho para o código LDPC* $(648, 324)$ *para DH usando min-sum BP com* 5 *iterações.*

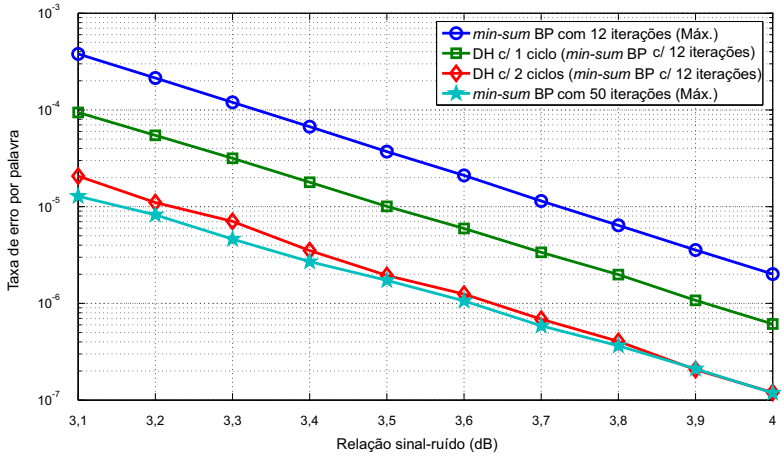


Figura 6.15: *Curvas de desempenho para o código LDPC* $(648, 324)$ *para DH usando min-sum BP com* 12 *iterações.*

obtido para o decodificador *min-sum* BP de referência que emprega 50 iterações, especialmente para valores mais elevados de SNR.

A Figura 6.16 exibe as seguintes curvas de desempenho para o código LDPC $(648, 324)$: decodificação *min-sum* BP empregando $I_{\mathrm{BP}} = 20$ iterações, decodificação empregando o sistema DH para um e dois ciclos de operação, com uma etapa de decodificação *min-sum* BP utilizando $I_{\mathrm{BP}} = 20$ iterações, e decodificação *min-sum* BP com $I_{\mathrm{BP}} = 80$ iterações. Este último corresponde ao decodificador de referência. Observa-se que a decodificação *min-sum* BP para 20 iterações tem resultado aproximadamente $0,1$ dB inferior ao da decodificação com sistema DH com um ciclo de operação. Do mesmo modo, observa-se que o decodificador com sistema DH para dois ciclos de operação tem resultado cerca de $0,1$ dB superior ao do decodificador DH com um ciclo de operação.

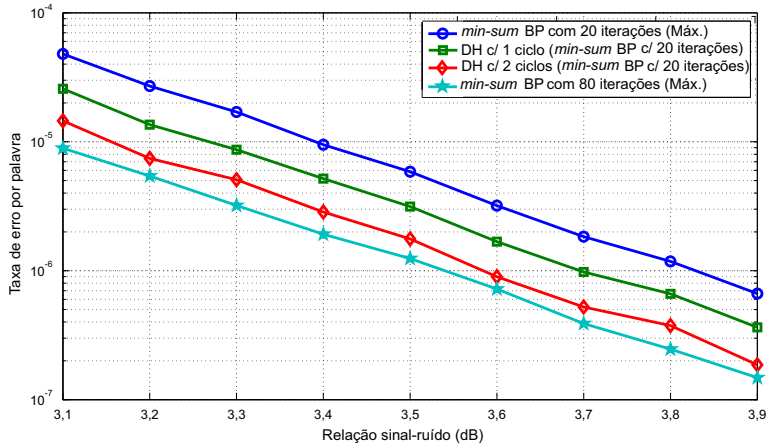


Figura 6.16: *Curvas de desempenho para o código LDPC* $(648, 324)$ *para DH usando min-sum BP com* 20 *iterações.*

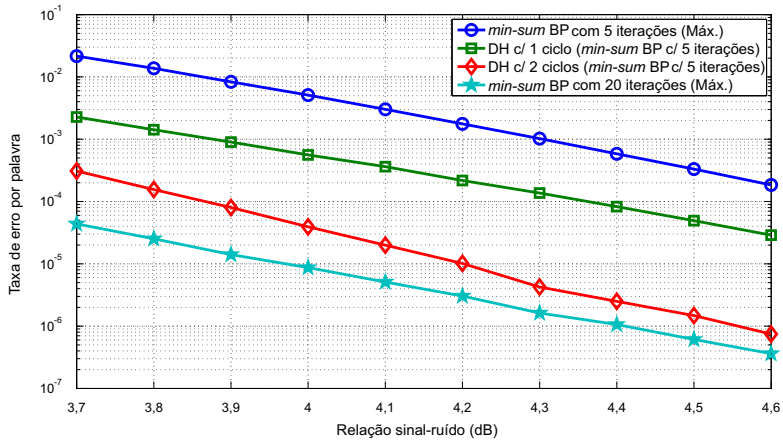


Figura 6.17: *Curvas de desempenho para o código LDPC* $(648, 432)$ *para DH usando min-sum BP com* 5 *iterações.*

Outrossim, o resultado para esse decodificador DH com dois ciclos de operação, empregando no máximo 40 iterações no estágio *min-sum* BP, é ligeiramente inferior ($< 0,1$ dB) ao daquele obtido para o decodificador *min-sum* BP de referência que emprega 80 iterações.

Os gráficos contendo as curvas de desempenho gerados para as outras simulações, no caso para os códigos LDPC $(648, 432)$, LDPC $(1296, 648)$ e LDPC $(1296, 864)$, são bem similares àqueles descritos anteriormente para o código LDPC $(648, 324)$.

Nas Figuras 6.17, 6.18 e 6.19 são mostrados, respectivamente, os resultados de desempenho para o código LDPC $(648, 432)$ para $I_{\mathrm{BP}} = 5$, $I_{\mathrm{BP}} = 12$ e $I_{\mathrm{BP}} = 20$ iterações no estágio de decodificação *min-sum* BP do sistema DH. Diferentemente do código LDPC $(648, 324)$, o desempenho do decodificador DH com dois ciclos só é equivalente ao do decodificador de referência para $I_{\mathrm{BP}} = 20$ iterações.

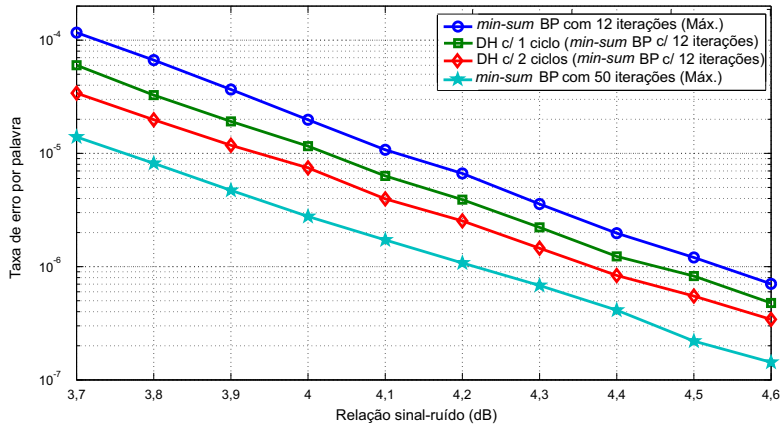


Figura 6.18: *Curvas de desempenho para o código LDPC* $(648, 432)$ *para DH usando min-sum BP com* 12 *iterações.*

Nos demais casos, especialmente para $I_{\mathrm{BP}} = 5$ iterações, há ganho no desempenho do sistema DH com um e dois ciclos sobre o decodificador *min-sum* BP mas que não é suficiente para se igualar ao do decodificador de referência empregando o dobro de iterações.

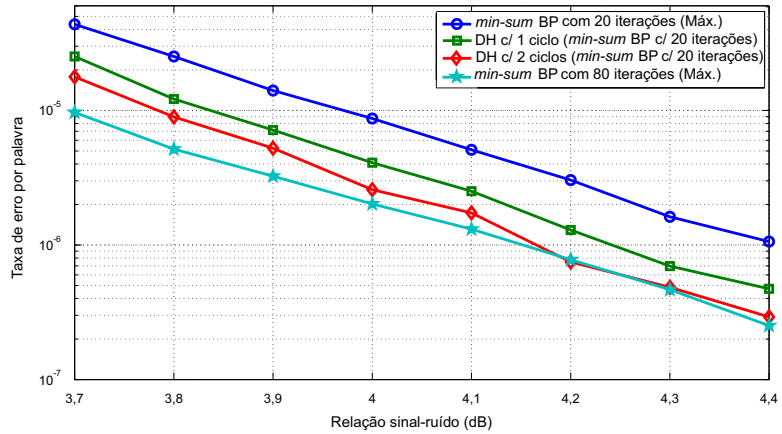


Figura 6.19: *Curvas de desempenho para o código LDPC* $(648, 432)$ *para DH usando min-sum BP com* 20 *iterações.*

Nas Figuras 6.20 e 6.21 são mostrados, respectivamente, as curvas de desempenho para os códigos LDPC $(1296, 648)$ e LDPC $(1296, 864)$ para $I_{\mathrm{BP}} = 12$ iterações no estágio de decodificação *min-sum* BP do sistema DH. Em ambos os casos, o desempenho do decodificador DH com dois ciclos, com um número global de iterações igual a 24, é equivalente ao do decodificador de referência que emprega $I_{\mathrm{BP}} = 50$ iterações.

De uma forma geral, tendo por base os resultados da análise comparativa entre o sistema DH e o decodificador *min-sum* BP, é verificado que, para o sistema DH com $I_{\mathrm{DH}} = 2$ ciclos de operação, obtém-se desempenho semelhante ao do decodificador *min-sum* BP, necessitando, no entanto, um

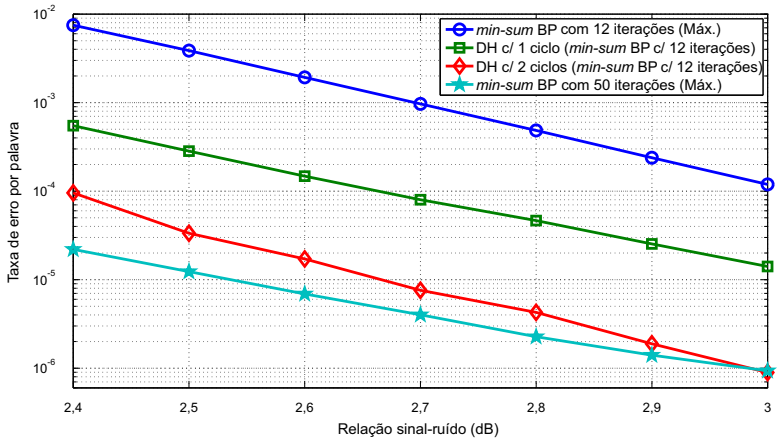


Figura 6.20: *Curvas de desempenho para o código LDPC* $(1296, 648)$ *para DH usando min-sum BP com* 12 *iterações.*

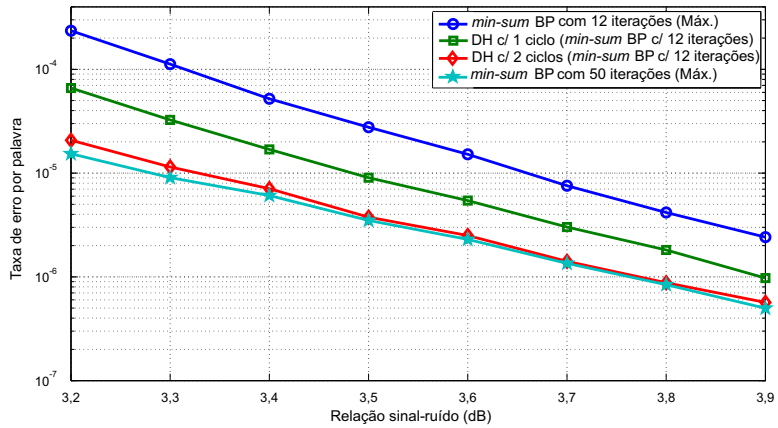


Figura 6.21: *Curvas de desempenho para o código LDPC* $(1296, 864)$ *para DH usando min-sum BP com* 12 *iterações.*

número global inferior de iterações na etapa de decodificação *min-sum* BP.

6.2 Análise Comparativa com Sistemas de Decodificação Concatenada Serial

6.2.1 Introdução

O sistema de decodificação DH pode ser comparado a sistemas de decodificação concatenada na busca de um método alternativo e eficiente de decodificação, uma vez que ambos os sistemas atuam prioritariamente na região de patamar de erros típica dos códigos LDPC [26]. A vantagem da utilização de um sistema de decodificação DH é que este possibilita uma maior taxa de código com desempenho equivalente ao da decodificação concatenada, em termos da taxa WER.

6.2.2 CÓDIGOS CONCATENADOS

Os códigos concatenados foram propostos por Forney [89] como uma forma de combinar dois códigos para se obter um código mais longo, com uma baixa taxa de erros e com uma complexidade de implementação global menor daquela que seria requerida pelo uso de um único código com parâmetros equivalentes. Esta estrutura consiste em uma associação em cascata [38], [16] de um código interno (*inner code*) binário $\mathcal{C}_1(N_1, K_1)$, com um código externo (*outer code*) não-binário $\mathcal{C}_2(N_2, K_2)$, composto por símbolos de $GF(2^{K_1})$. A Figura 6.22 mostra o diagrama em blocos de um sistema de codificação concatenada, representando a parte de transmissão e a parte de recepção com seus respectivos blocos. Os símbolos de $\mathcal{C}_2$ são representados pelos seus correspondentes bytes de K_1 símbolos binários (ou K_1-uplas).

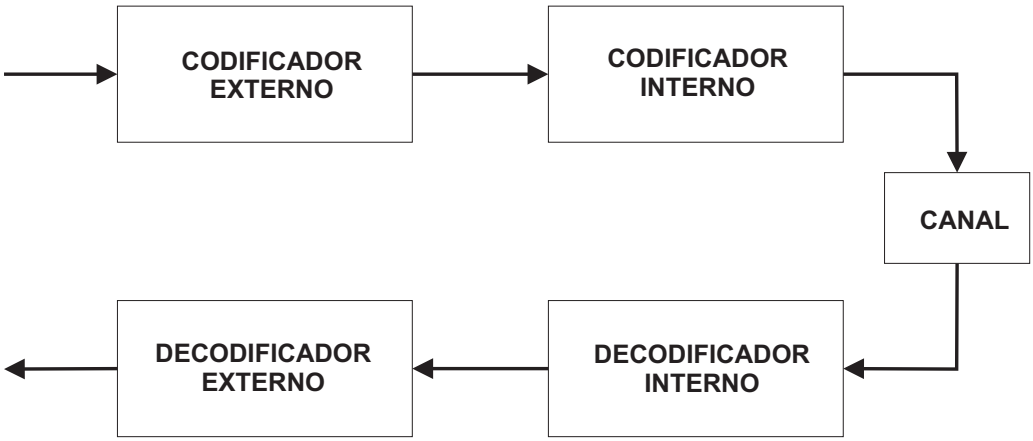


Figura 6.22: *Diagrama em blocos de um sistema de codificação concatenada.*

A codificação consiste em, primeiramente, dividir os K_1K_2 dígitos de informação binária em K_2 *bytes* de K_1 dígitos de informação cada. Estes K_2 *bytes* são codificados de acordo com a regra de codificação de $\mathcal{C}_2$ para formar uma palavra-código de N_2 *bytes*. Em seguida, cada *byte* de K_1 dígitos é codificado em uma palavra-código em $\mathcal{C}_1$, resultando em um conjunto de N_2 palavras-código de $\mathcal{C}_1$, dando um total de N_1N_2 dígitos para a palavra-código resultante. Estes dígitos são então transmitidos ou armazenados.

Uma vez que os códigos $\mathcal{C}_1$ e $\mathcal{C}_2$ são lineares, o código concatenado resultante é um código linear binário (N_1N_2, K_1K_2), com distância mínima igual a $d_{min_1}d_{min_2}$, em que d_{min_1} e d_{min_2} são as distâncias mínimas dos códigos $\mathcal{C}_1$ e $\mathcal{C}_2$, respectivamente, e taxa de código igual a $\frac{K_1 \times K_2}{N_1 \times N_2} = R_1 \times R_2$, em que R_1 e R_2 correspondem às taxas individuais dos códigos empregados na configuração.

Uma combinação clássica usada para esse modelo é formada pela associação do código de bloco Reed-Solomon (RS), como código externo, com um código convolucional, como código interno. Um

exemplo desta aplicação pode ser encontrada no Sistema Brasileiro de TV Digital [90], em que é empregada a codificação concatenada, formada por um código externo encurtado RS $(204, 188)$, capaz de corrigir até 8 *bytes*, combinado com um código convolucional interno com taxa típica $1/2$, podendo ser configurado para taxas maiores como $3/4$, $5/6$ e $7/8$. Outro exemplo, pode ser observado no padrão adotado em 1987 pela NASA, em conjunto com a Agência Espacial Européia, para missões de espaço profundo (*deep space missions*), em que o código RS$(255, 233)$ é usado concatenado com um código convolucional [39].

Nessa linha, surgiu mais recentemente a combinação de um código LDPC com um código de bloco BCH, sendo o primeiro empregado como código interno e o segundo como código externo. Essa solução é atualmente empregada nos padrões DVB-S2 de comunicação por satélite [29] e no padrão chinês de TV digital, DTMB (*Digital Terrestrial Television Multimedia Broadcasting*) [27], [28]. No modelo chinês de TV Digital, o código externo BCH $(762, 752)$, com capacidade de correção de um erro por palavra, é derivado por puncionamento (*puncturing*) de 261 *bits* nulos a partir do código BCH $(1023, 1013)$. Numa possível configuração desse modelo, quatro palavras-código geradas por esse código externo são combinadas formando um grupo de 3048 *bits* que passam a representar a mensagem a ser codificada pelo código interno LDPC $(7493, 3048)$. Como 40 *bits* do total de 3048 constituem *bits* de paridade para o código BCH, a taxa efetiva deste código é de $0,40$. O padrão também prevê outras possíveis configurações para prover maior taxa de dados, como por exemplo LDPC $(7543, 4572)$ com taxa de $0,60$ e LDPC $(7493, 6096)$ com taxa de $0,80$. Esse modelo utiliza a decodificação suave iterativa baseada no SPA para o código interno (LDPC), que em função de seu desempenho permite a dispensa do uso de um código não-binário para o código externo. O código binário BCH (código externo) atua principalmente na melhoria do patamar de erros, característico do código LDPC.

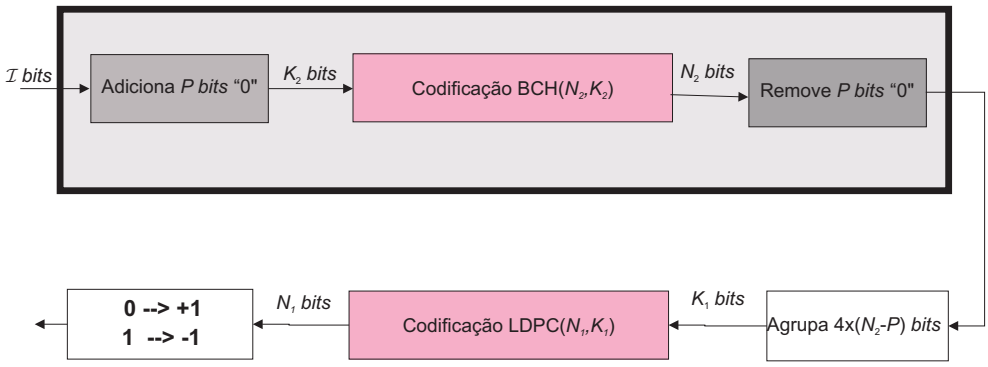


Figura 6.23: *Modelo do sistema de codificação concatenada empregado na simulação.*

Tabela 6.1: *Parâmetros Adotados para as Configurações de Teste da Codificação Concatenada Serial.*

Configuração	$\mathcal{I}$	P	N_2	K_2	N_1	K_1
A	74	46	127	120	648	324
B	208	39	255	247	1296	864

6.2.3 MODELO DE SIMULAÇÃO, RESULTADOS E COMENTÁRIOS

O modelo empregado para a simulação da decodificação do código concatenado serial é mostrado na Figura 6.23 e é baseado no modelo do padrão chinês de TV Digital [27], [28]. São gerados pacotes de $\mathcal{I}$ *bits* de informação que são combinados a P *bits* nulos formando uma mensagem de $K_2 = \mathcal{I} + P$ *bits*. O codificador externo BCH (N_2, K_2) codifica essa mensagem em palavras-código de N_2 *bits*. Após essa codificação, os P *bits* nulos introduzidos são removidos restando apenas $N_2 - P$ *bits*. A cada grupo de 4 pacotes de $N_2 - P$ *bits* é gerada uma mensagem de $K_1 = 4 \times (N_2 - P)$ *bits* a ser codificada pelo codificador interno LDPC (N_1, K_1) do padrão IEEE802.11n com taxa igual K_1/N_1. Ao final, a N-upla binária é convertida para uma sequência de números inteiros pelo estágio mapeador, de modo que o *bit* 0 é convertido para $+1$ e o *bit* 1 para -1. Essa sequência é então enviada por um canal com RAGB. No lado do receptor, executa-se o processo inverso a fim de se recuperar a informação enviada.

Para a simulação do código concatenado serial foram adotadas duas configurações, denotadas **A** e **B**, que correspondem respectivamente, às combinações LDPC $(648, 324)$ com BCH $(127, 120)$ e LDPC $(1296, 864)$ com BCH $(255, 247)$. Os códigos BCH adotados nesses esquemas possuem distância mínima, d_{min}, igual a três, podendo portanto corrigir até $t = 1$ erro aleatório, e são decodificados usando o algoritmo Berlekamp-Massey [39]. Os parâmetros adotados nessas duas configurações estão listados na Tabela 6.1.

Nas simulações realizadas, é comparado o desempenho do sistema de decodificação concatenada serial, para as configurações de teste adotadas, com o do sistema de decodificação DH, tomando-se os mesmos valores de SNR do canal com RAGB. Para que a comparação se torne equânime, o número global de iterações da etapa de decodificação *min-sum* BP do sistema de decodificação DH, computado em todos os ciclos de execução desse sistema, não ultrapassa o número máximo de iterações empregado para o decodificador *min-sum* BP do código concatenado serial.

As Figuras 6.24 e 6.25 mostram os resultados de desempenho da decodificação para a configuração **A** e **B**, respectivamente. Em todos os casos, esses resultados são comparados aos desempenhos obtidos para um sistema de decodificação *min-sum* BP convencional e para um sistema DH. As com-

parações são realizadas para um número de iterações do decodificador *min-sum* BP igual a 25 ou 50.

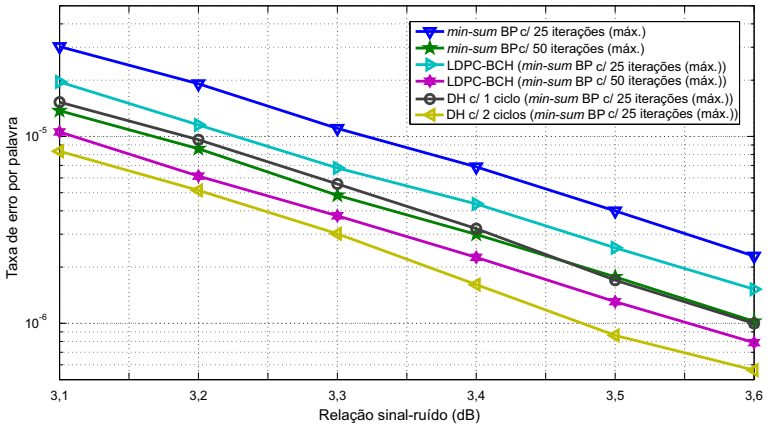


Figura 6.24: *Análise comparativa de desempenho para a configuração* ***A***.

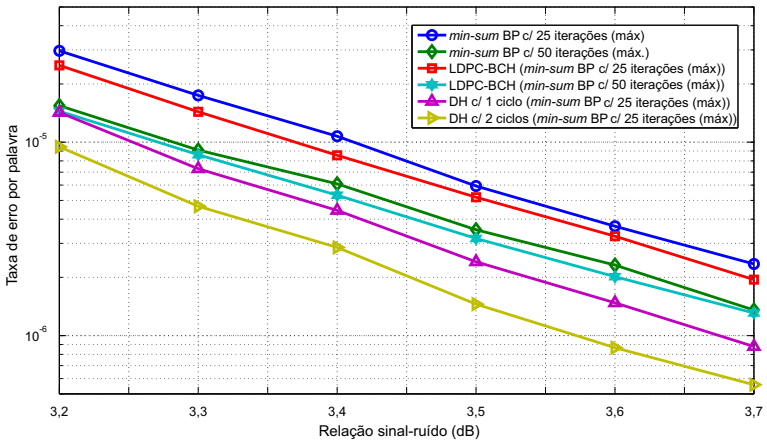


Figura 6.25: *Análise comparativa de desempenho para a configuração* ***B***.

Para todos esses casos, o desempenho do decodificador para o código concatenado serial, considerando o mesmo número de iterações do decodificador *min-sum* BP, 25 ou 50 iterações, é superior ao do decodificador *min-sum* BP e ligeiramente inferior ao do decodificador DH. A título de ilustração, considerando uma taxa de erro por palavra igual a 3×10^{-6} e para um número de iterações máxima do estágio *min-sum* BP igual a 25 em todos os casos, observa-se que na Figura 6.24, esse desempenho é obtido para aproximadamente $3,55$ dB para o decodificador *min-sum* BP, para aproximadamente $3,47$ dB para o código concatenado serial LDPC $(648, 324)$ com BCH $(127, 120)$ e para aproximadamente $3,4$ dB para o sistema de decodificação DH com um ciclo de operação.

Além de apresentar um desempenho superior ao sistema de decodificação de um código concatenado serial, o sistema DH por operar apenas com o código LDPC, possibilita uma taxa de código

Tabela 6.2: *Funcões de Distribuição $\lambda(x)$ para os Códigos LDPC do Padrão IEEE*$802.11n$*.*

Código	$\lambda(\mathbf{x})$
LDPC (1944,972)	$0,256x + 0,314x^2 + 0,046x^3 + 0,384x^{10}$
LDPC (1944,1296)	$0,159x + 0,409x^2 + 0,068x^5 + 0,364x^7$
LDPC (1296,648)	$0,256x + 0,314x^2 + 0,046x^3 + 0,384x^{10}$
LDPC (1296,864)	$0,159x + 0,409x^2 + 0,159x^6 + 0,273x^7$
LDPC (648,324)	$0,25x + 0,341x^2 + 0,409x^{11}$
LDPC (648,432)	$0,159x + 0,273x^2 + 0,227x^3 + 0,068x^5 + 0,273x^{11}$

Tabela 6.3: *Funcões de Distribuição $\rho(x)$ para os Códigos LDPC do Padrão IEEE*$802.11n$*.*

Código	$\rho(\mathbf{x})$
LDPC (1944,972)	$0,814x^6 + 0,186x^7$
LDPC (1944,1296)	x^{10}
LDPC (1296,648)	$0,814x^6 + 0,186x^7$
LDPC (1296,864)	x^{10}
LDPC (648,324)	$0,637x^6 + 0,363x^7$
LDPC (648,432)	x^{10}

superior. No caso da configuração **A**, a taxa do código concatenado é de aproximadamente $0,46$ e é ligeiramente inferior à taxa do código LDPC $(648, 324)$, empregada no sistema DH, que é igual a $0,50$.

6.3 ANÁLISE DO TEMPO DE EXECUÇÃO DO ALGORITMO DH

A decodificação iterativa de apagamentos introduzida no sistema DH, além de ser mais simples do que a decodificação *min-sum* BP, em geral, é executada de forma mais rápida e com um número bem inferior de iterações. A técnica de evolução de densidade (*density evolution*) pode ser empregada para determinar o número mínimo necessário de iterações do decodificador para a correção dos apagamentos, em função da probabilidade de apagamento ε e das funções de distribuição de graus [13]. As Tabelas 6.2 e 6.3 listam, respectivamente, as funções de distribuição que representam a fração de ramos que é conectada a nós de variável, $\lambda(x)$, e as funções de distribuição que representam a fração de ramos que é conectada a nós de verificação de paridade, $\rho(x)$, para alguns dos códigos LDPC adotados pelo padrão IEEE802.11n.

A Figura 6.26 mostra que, para esses códigos LDPC, em aproximadamente três iterações o decodificador iterativo de apagamentos converge. Nesta análise, considera-se uma estimativa de probabilidade de apagamentos do canal $\varepsilon = 0,14$ para o código LDPC (648,324), $\varepsilon = 0,11$ para

o código LDPC (648,432), $\varepsilon = 0,10$ para o código LDPC (1296,648), $\varepsilon = 0,08$ para o código LDPC (1296,864), $\varepsilon = 0,10$ para o código LDPC (1944,972) e $\varepsilon = 0,05$ para o código LDPC (1944,1296). Esses valores de ε são referentes à razão do número de apagamentos $\mathcal{X}$, introduzidos pelo BEC artificialmente criado, pelo comprimento de bloco do código N.

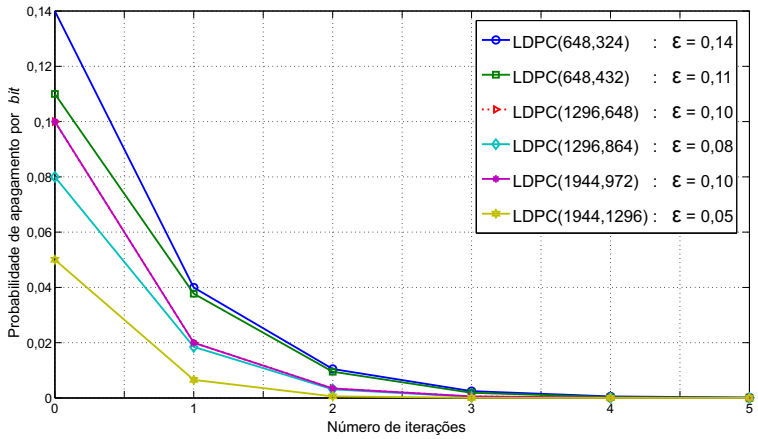


Figura 6.26: *Probabilidade de apagamento por bit como função do número de iterações do decodificador iterativo de apagamentos para os códigos do padrão IEEE*802.11*n.*

A Tabela 6.4 lista os tempos medidos por simulação para o código LDPC (648,324) em decodificação DH contendo etapa *min-sum* BP com 25 ou 50 iterações e em decodificação convencional *min-sum* BP com 200 iterações. Nessa análise, foi considerado um total de 1000 N-uplas em erro processadas simultaneamente pelos dois modelos de simulação, o que permite avaliá-los nas mesmas condições de teste. Para realizar esse levantamento, foi empregado um sistema de rede de computadores numa configuração conhecida por *cluster computing* [91] usando um total de quatro computadores (*nodes*) com processador Intel Core2 Quad série Q9xxx @3Ghz e com memória RAM de 4 GB. Os tempos medidos nos dois casos são listados em função da SNR.

Nessa tabela, a coluna **Iterações** indica o número de iterações empregadas na etapa *min-sum* BP do decodificador DH, a coluna $\mathbf{T}_{\text{DH}}$ indica o tempo total gasto na decodificação de 1000 N-uplas pelo sistema DH para $I_{\text{DH}} = 2$ ciclos de operação, a coluna $\mathbf{T}_{\text{BP}}$ indica o tempo total gasto na decodificação das mesmas 1000 N-uplas pelo algoritmo *min-sum* BP para $I_{\text{BP}} = 200$ iterações e a coluna **G** indica o percentual de redução de tempo de processamento do sistema DH em relação ao do decodificador *min-sum* BP com 200 iterações, e pode ser determinado como

$$\mathbf{G} = \frac{\mathbf{T}_{\text{BP}} - \mathbf{T}_{\text{DH}}}{\mathbf{T}_{\text{BP}}} \times 100. \tag{6.1}$$

Os resultados mostrados na Tabela 6.4 indicam um ganho médio de desempenho de aproxi-

Tabela 6.4: *Comparação dos tempos de decodificação DH e decodificação min-sum BP com* 200 *iterações para o código LDPC* $(648, 324)$ *do Padrão IEEE*$802.11n$*.*

$\mathbf{SNR(dB)}$	**Decodificação DH**		**Decod. *min-sum* BP c/** 200 **iterações**	$\mathbf{G}(\%)$
	Iterações	$\mathbf{T_{DH}}$ **(s)**	$\mathbf{T_{BP}}$ **(s)**	
$3,1$	25	17,455674	41,417113	57,85
	50	38,700695	71,505397	45,87
$3,2$	25	17,263062	38,660725	55,34
	50	28,272169	51,973922	45,60
$3,3$	25	18,134288	40,129897	54,81
	50	21,570460	41,060450	47,46
$3,4$	25	15,627382	35,372799	55,82
	50	38,036613	70,569608	46,10
$3,5$	25	11,687780	25,225864	53,66
	50	33,113285	61,147599	45,84
$3,6$	25	5,706098	13,227305	56,86
	50	24,878528	47,134288	47,21

madamente $55,7\%$ para o decodificador DH, com uma etapa de decodificação *min-sum* BP com 25 iterações, totalizando 50 iterações do algoritmo *min-sum* BP nos dois ciclos do decodificador DH, sobre o decodificador *min-sum* BP com 200 iterações e um ganho médio de desempenho de aproximadamente $46,3\%$ para um decodificador DH, com a etapa de decodificação *min-sum* BP com 50 iterações, totalizando 100 iterações do algoritmo *min-sum* BP nos dois ciclos do decodificador DH, sobre o decodificador *min-sum* BP com 200 iterações. Ressalta-se que, nesse caso, os desempenhos obtidos pelo sistema DH com dois ciclos, empregando uma etapa de decodificação *min-sum* BP com $I_{\mathrm{BP}} = 50$ iterações, e pelo decodificador *min-sum* BP com $I_{\mathrm{BP}} = 200$ iterações são equivalentes.

6.4 Análise do Número Máximo de Ciclos do Algoritmo DH

Em todas as simulações efetuadas para o sistema DH, foi empregado no máximo $I_{\mathrm{DH}} = 2$ ciclos de operação. Isso se deve aos seguintes fatores:

- Os valores estimados de QLLR, mesmo para *bits* em erro, assumem valores elevados a partir do segundo ciclo de operação do sistema DH e já não se consegue detectá-los com eficiência; e
- São introduzidos novos erros a partir do segundo ciclo de operação do sistema DH, por conta dos *bits* não detectados e não corrigidos nos ciclos anteriores.

As Figuras 6.27, 6.28 e 6.29 mostram os resultados da análise do percentual de detecção de padrões de erros no sistema DH, considerando-se que a etapa de decodificação *min-sum* BP emprega, respectivamente, $I_{\mathrm{BP}} = 5$, $I_{\mathrm{BP}} = 12$ e $I_{\mathrm{BP}} = 20$ iterações.

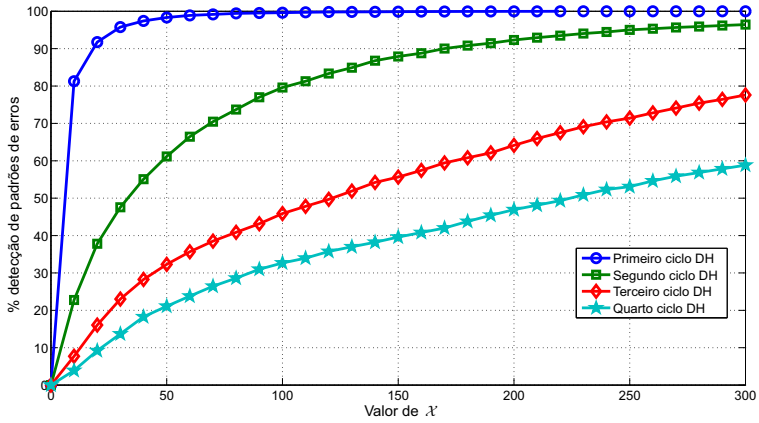


Figura 6.27: *Análise de detecção de padrões de erros, em função de $\mathcal{X}$, para sistema DH empregando estágio min-sum BP com $I_{BP} = 5$ iterações.*

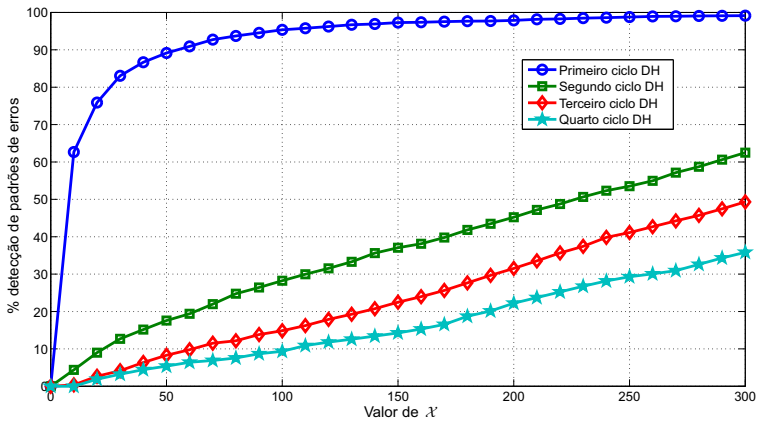


Figura 6.28: *Análise de detecção de padrões de erros, em função de $\mathcal{X}$, para sistema DH empregando estágio min-sum BP com $I_{BP} = 12$ iterações.*

A Figura 6.27 exibe quatro curvas sendo que a primeira corresponde ao percentual de detecção de padrões de erros no primeiro ciclo de operação do sistema DH, após a falha no estágio de decodificação *min-sum* BP; a segunda corresponde ao percentual de detecção de padrões de erros para o segundo ciclo de operação, após a falha de decodificação no primeiro ciclo e após a falha do estágio decodificador *min-sum* BP no segundo ciclo; a terceira corresponde ao percentual de detecção de padrões de erros para o terceiro ciclo de operação, após a falha de decodificação nos ciclos anteriores e após a falha na decodificação no estágio *min-sum* BP do terceiro ciclo e a quarta corresponde ao percentual de detecção de padrões de erros para o quarto ciclo de operação, obtido após a falha

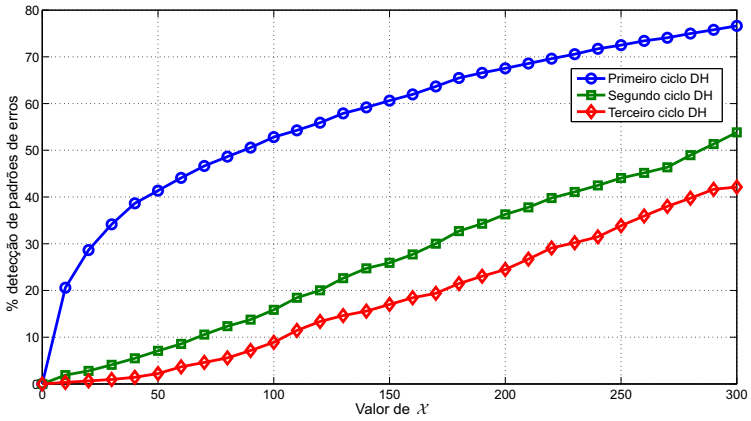


Figura 6.29: *Análise de detecção de padrões de erros, em função de $\mathcal{X}$, para sistema DH empregando estágio min-sum BP com $I_{BP} = 20$ iterações.*

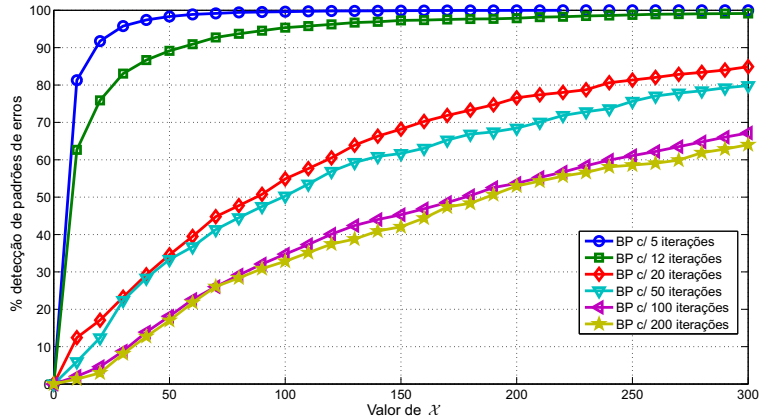


Figura 6.30: *Análise de detecção de padrões de erros para o sistema DH , em função de $\mathcal{X}$ e I_{BP} do estágio min-sum BP.*

de decodificação nos três ciclos anteriores e falha na etapa de decodificação *min-sum* BP do quarto ciclo. A título de exemplo, considerando um valor para $\mathcal{X}$ igual a 90, observa-se que o percentual de detecção para o primeiro ciclo de operação é próximo de 100%, reduz para um pouco menos de 80% para o segundo ciclo de operação, cai para algo em torno de 45% para três ciclos de operação e, finalmente, para quatro ciclos de operação corresponde a aproximadamente 30%. Esse decréscimo na taxa de detecção também pode ser observado nas Figuras 6.28 e 6.29.

Não obstante a maioria dos *bits* em erro assumir valores de QLLR próximos a zero, esses resultados demonstram que alguns deles podem assumir valores elevados de estimativa de QLLR a partir do segundo ciclo de operação do sistema DH, tornando a detecção dos padrões de erros pouco eficiente.

Um segundo aspecto a ser considerado diz respeito à influência do número máximo de iterações do estágio *min-sum* BP, I_{BP}, sobre o percentual de detecção de padrões de erros. É mostrado na

Figura 6.30 que mantido o mesmo valor para $\mathcal{X}$, esse percentual cai à medida em que o número de iterações aumenta. Isso indica que mais e mais valores de QLLR, mesmo para *bits* em erro, assumem valores altos de QLLR à medida que cresce o número de iterações I_{BP}. Isso se reflete sobre o desempenho do sistema DH para o mesmo código LDPC. Tome-se como exemplo o código LDPC $(648, 324)$, com $I_{\text{BP}} = 5$ iterações, para se obter o mesmo desempenho, o sistema DH necessita de um valor inferior a 1 dB de SNR se comparado a um decodificador *min-sum* BP operando com 5 iterações, conforme mostrado na Figura 6.14, ao passo que para $I_{\text{BP}} = 12$ iterações, esse valor cai para $0,2$ dB, conforme mostrado na Figura 6.15, e para $I_{\text{BP}} = 50$ iterações esse valor é menor que $0,1$ dB, conforme mostrado na Figura 6.11.

6.5 Análise e Impacto de Eventos de Erros sobre o Desempenho do Sistema DH

6.5.1 Introdução

O decodificador BP empregado para a decodificação dos códigos LDPC atinge a condição ótima de decodificação MAP (*Maximum a Posteriori*) quando não houver ciclos na representação por grafo de Tanner. Contudo, esses códigos, especialmente com pequeno e médio comprimentos, são caracterizados pela presença desses ciclos que afetam a decodificação dos *bits* neles inclusos [92].

O decodificador BP é executado por um número predefinido de iterações, I_{BP}, e, em caso de falha de decodificação, isso conduz a uma N-upla que não corresponde a uma palavra-código e, consequentemente, a um dado número de erros de *bits*. Essa quantidade de erros de *bits*, para uma dada ocorrência de falha de decodificação, pode variar significativamente ao longo das iterações intermediárias do decodificador BP [93].

Na região de patamar de erros dos códigos LDPC, na qual esses códigos exibem uma súbita saturação na taxa BER e na taxa WER, os erros de *bits* são causados principalmente por conjuntos de armadilhas [25]. Um conjunto de armadilhas TS (ω, ν) contêm ω nós de variável que induzem um subgrafo com ν nós de verificação de paridade de grau ímpar. Nesse caso, concorrem três tipos de eventos de erros ao longo das iterações intermediárias do decodificador BP [37], [14]: (1) Eventos de erros instáveis, os quais mudam dinamicamente de iteração a iteração e os valores de ω e ν são elevados; (2) Conjuntos de armadilhas estáveis, para os quais ω e ν são tipicamente baixos e (3) Conjuntos de armadilhas oscilantes, os quais variam periodicamente com o número de iterações do decodificador. A Figura 6.31 mostra o efeito de cada um desses eventos sobre o número de erros

de *bits* ao longo das iterações intermediárias do decodificador BP.

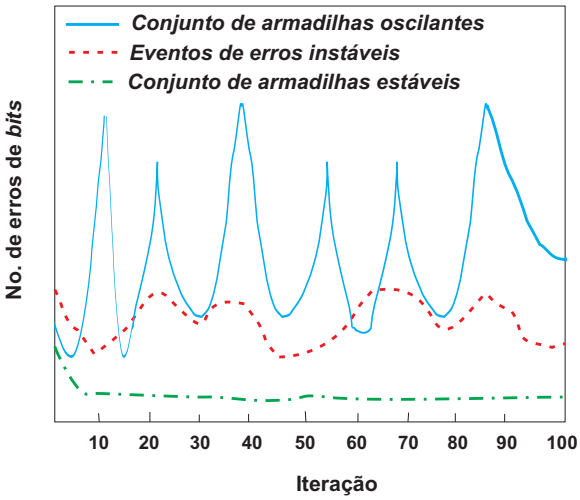


Figura 6.31: *Padrões de erros dos códigos LDPC em função das iterações do decodificador min-sum BP.*

Uma grande gama de pesquisas têm sido realizada com o intuito de projetar decodificadores que mitiguem os erros causados por conjuntos de armadilhas. Em [92], a mensagem obtida na iteração anterior é adicionada à mensagem obtida na iteração atual, caso tenham sinais diferentes. Essa modificação auxilia na redução das oscilações do número de erros de *bits* causadas pelos ciclos nos grafos de Tanner de códigos LDPC de pequeno e médio comprimentos. Em [93], uma modificação no decodificador BP, baseada no rastreamento do número de equações de verificação de paridade não atendidas durante as iterações intermediárias, possibilita a redução do BER para altos valores de SNR. Em [37], é proposto um decodificador bi-modal baseado em pós-processamento com decodificação iterativa de apagamentos que é ativada somente quando o peso da síndrome dos eventos de erros estiver contido em um conjunto de pesos de síndromes de conjuntos de armadilhas pré-selecionados. Em [94], é proposto um decodificador de dois estágios tratando de diferentes tipos de conjuntos de armadilhas. O primeiro estágio opera sobre os conjuntos de armadilhas oscilantes enquanto o segundo lida com os conjuntos de armadilhas elementares.

O sistema DH proposto nesta tese é utilizado para mitigar os efeitos de eventos de erros causados por conjuntos de armadilhas e, a exemplo do que é proposto em [37], é um decodificador bi-modal baseado em pós-processamento com correção de apagamentos, com algumas diferenças, uma delas é que atua sempre que houver a falha da decodificação *min-sum* BP. Como foi descrito no Capítulo 5, os apagamentos são gerados por um canal BEC [40] artificialmente criado e que se baseia nas informações de confiabilidade dos dígitos, geradas ao fim da última iteração da decodificação de erros do tipo *min-sum* BP. O número de apagamentos gerados por esse canal é fixo e predeterminado,

sendo denotado por $\mathcal{X}$. Idealmente, todos os erros de bits remanescentes na N-upla resultante da decodificação *min-sum* BP sem sucesso devem estar inclusos dentre as $\mathcal{X}$ coordenadas selecionadas. Isso é mais provável de ocorrer quando o valor $\mathcal{X}$ aumenta ou quando a cardinalidade do padrão de erros remanescentes é baixa. O aumento do valor de $\mathcal{X}$ em muitos casos pode não ser uma boa solução, em razão, principalmente, do surgimento de conjuntos de parada [25] que impede o sucesso da decodificação iterativa de apagamentos. Por outro lado, caso o código LDPC seja caracterizado por eventos de erros do tipo conjuntos de armadilhas oscilantes em algumas iterações intermediárias do decodificador *min-sum* BP, é possível que existam padrões de erros com baixa cardinalidade. Pode-se, nesses casos, utilizar essa característica para maximizar o desempenho do sistema DH.

6.5.2 ANÁLISE DE EVENTOS DE ERROS DOS CÓDIGOS LDPC DO PADRÃO IEEE802.11N

Simulações de computador são efetuadas para alguns códigos LDPC do padrão IEEE802.11n com o intuito de extrair um número médio de erros de *bits* ao longo das iterações intermediárias do decodificador *min-sum* BP e, dessa forma, identificar quais os tipos de eventos de erros que concorrem mais fortemente para a falha de decodificação. O procedimento usado é assim descrito:

Seja β_i o número de erros presentes na N-upla estimada na i-ésima iteração em uma decodificação *min-sum* BP, para $i \leq I_{\text{BP}}$. Em caso de falha da decodificação *min-sum* BP, então é gerado $\mathbf{x_m} = \{\beta_{1,m}, \beta_{2,m}, \ldots, \beta_{I_{\text{BP}},m}\}$ que corresponde à m-ésima sequência amostrada cujas coordenadas correspondem ao número de erros presentes em cada iteração. As coordenadas da sequência $\mathbf{x_m}$ são então normalizadas pelo maior valor de seus elementos, denotado por ϵ_m, dando origem à sequência normalizada $\tilde{\mathbf{x}}_\mathbf{m} = \{\tilde{\beta}_{1,m}, \tilde{\beta}_{2,m}, \ldots, \tilde{\beta}_{I_{\text{BP}},m}\}$. Extraindo a média aritmética de todas as m sequências normalizadas $\tilde{\mathbf{x}}_\mathbf{m}$ resulta na sequência $\gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_{I_{\text{BP}}}\}$, cuja i-ésima coordenada (correspondendo à i-ésima iteração do decodificador *min-sum* BP) é expressa por

$$\gamma_i = \frac{\sum_{m=1}^{M} \tilde{\beta}_{i,m}}{M}, \tag{6.2}$$

em que M denota o número máximo de sequências amostradas consideradas na análise.

A sequência γ estabelece o comportamento médio da decodificação *min-sum* BP com respeito ao incremento ou decremento da quantidade de erros ao longo das iterações.

6.5.3 IMPACTO DE CONJUNTOS DE ARMADILHAS OSCILANTES SOBRE O SISTEMA DH

A análise dos tipos de eventos de erros é executada sobre os códigos LDPC $(648, 324)$, LDPC $(648, 432)$, LDPC $(1296, 648)$, LDPC $(1296, 864)$, LDPC $(1944, 972)$ e LDPC $(1944, 1296)$

do padrão IEEE$802.11n$, considerando-se $M > 500$ N-uplas resultantes de falha na decodificação *min-sum* BP e $I_{\text{BP}} = 100$ iterações. Os resultados dessa análise são mostrados nas Figuras 6.32 a 6.43.

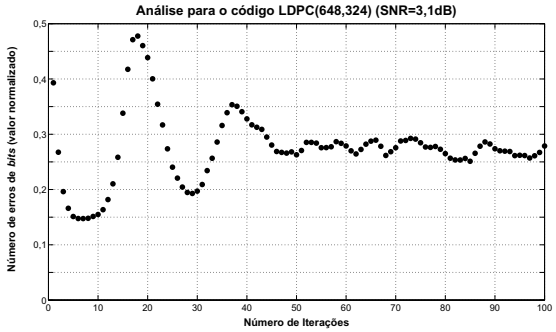


Figura 6.32: *Análise de eventos de erros para o código LDPC (*$648, 324$*) para* $SNR = 3,1$ *dB.*

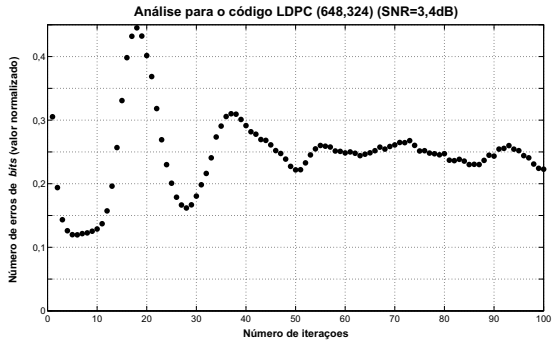


Figura 6.33: *Análise de eventos de erros para o código LDPC (*$648, 324$*) para* $SNR = 3,4$ *dB.*

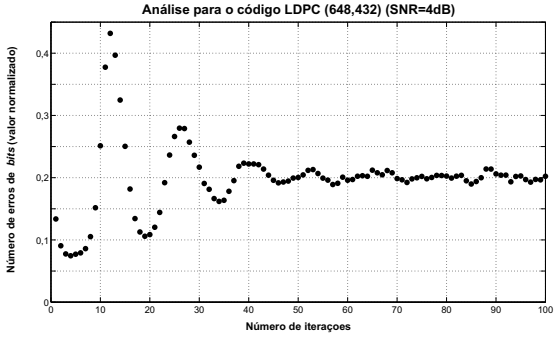


Figura 6.34: *Análise de eventos de erros para o código LDPC (*$648, 432$*) para* $SNR = 4$ *dB.*

Nessas referidas figuras, o eixo horizontal representa o número das iterações, de 1 a I_{BP}, e o eixo vertical representa a quantidade média normalizada de erros de *bits* em cada iteração do algoritmo *min-sum* BP, que é calculado conforme o procedimento descrito anteriormente.

Considerações sobre os resultados:

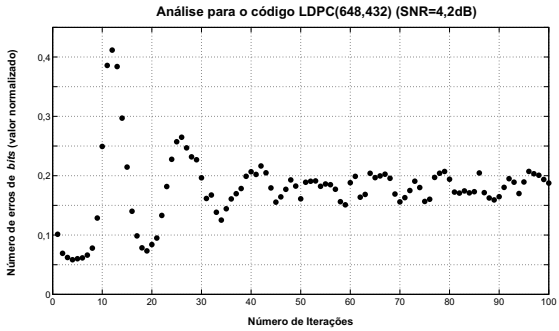


Figura 6.35: *Análise de eventos de erros para o código LDPC (*$648, 432$*) para* $SNR = 4,2$ *dB.*

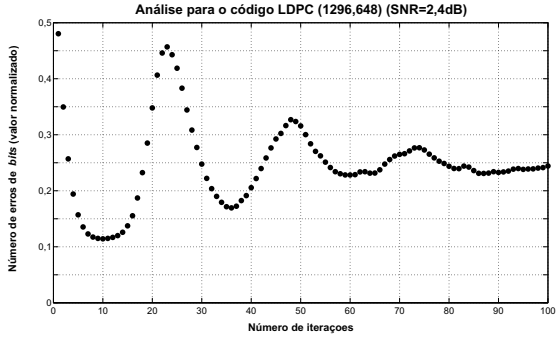


Figura 6.36: *Análise de eventos de erros para o código LDPC (*$1296, 648$*) para* $SNR = 2,4$ *dB.*

- ▷ Em todos os casos investigados, predominam as armadilhas de erros oscilantes para um número baixo de iterações (menor do que 50 iterações) e eventos de erros instáveis para um número elevado de iterações (maior do que 50 iterações);
- ▷ Para os códigos LDPC analisados, o perfil de eventos de erros se repete para uma larga faixa de SNR e apenas alguns dos casos analisados estão reproduzidos nas referidas figuras;
- ▷ Os perfis de eventos de erros traçados para esses códigos LDPC analisados independem da sequência de palavras-código utilizadas na decodificação *min-sum* BP, seja ela originalmente toda-nula ou uma palavra-código aleatoriamente gerada.

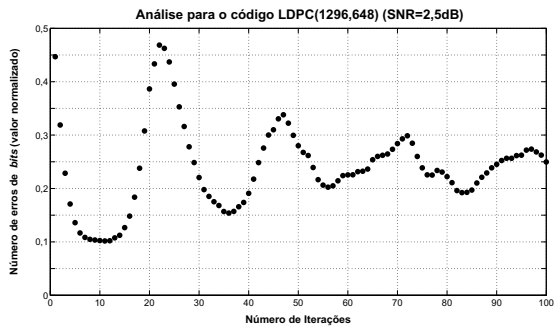


Figura 6.37: *Análise de eventos de erros para o código LDPC (*$1296, 648$*) para* $SNR = 2,5$ *dB.*

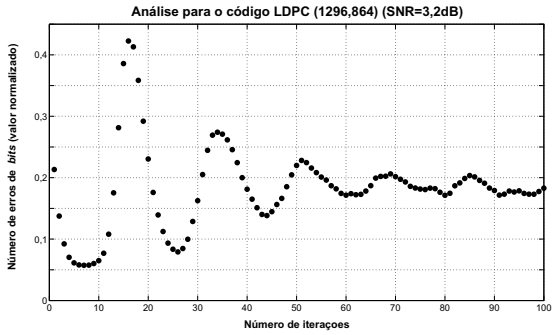


Figura 6.38: *Análise de eventos de erros para o código LDPC (*$1296, 864$*) para* $SNR = 3,2$ *dB.*

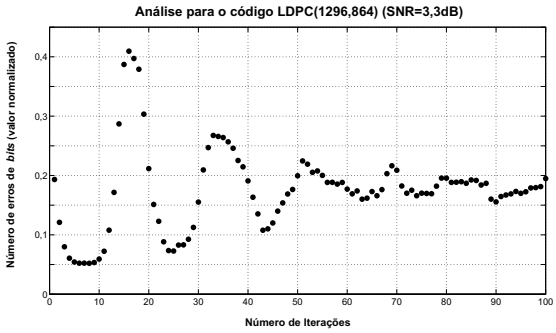


Figura 6.39: *Análise de eventos de erros para o código LDPC (*$1296, 864$*) para* $SNR = 3,3$ *dB.*

Quando predominam os eventos decorrentes de conjuntos de armadilhas oscilantes, o número de erros de *bits* pode alcançar valores relativamente elevados em algumas iterações intermediárias, que são denotados como *pontos de pico*. Também é possível, nessa situação, atingir valores relativamente baixos em algumas iterações intermediárias, que são denotadas como *pontos de vale*. A título de exemplo, observa-se que nas Figuras 6.32 e 6.33, relativas ao comportamento dos eventos de erros para o código LDPC $(648, 324)$, os *pontos de vale*, ao longo das 100 iterações realizadas pelo decodificador *min-sum* BP, ocorrem em torno das iterações 5 e 28, ao passo que os *pontos de pico* ocorrem em torno das iterações 18 e 37.

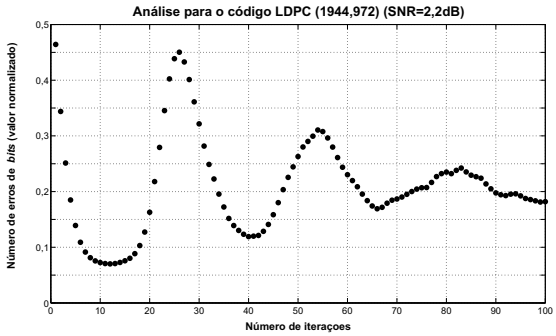


Figura 6.40: *Análise de eventos de erros para o código LDPC (*$1944, 972$*) para* $SNR = 2,2$ *dB.*

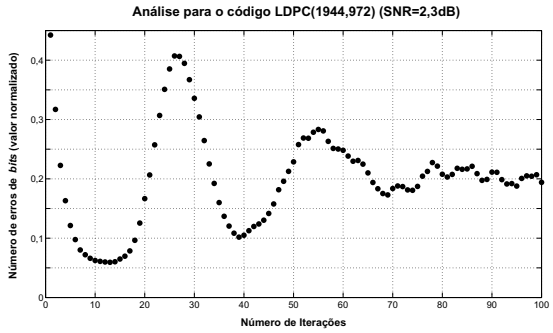


Figura 6.41: *Análise de eventos de erros para o código LDPC (*$1944, 972$*) para* $SNR = 2,3$ *dB.*

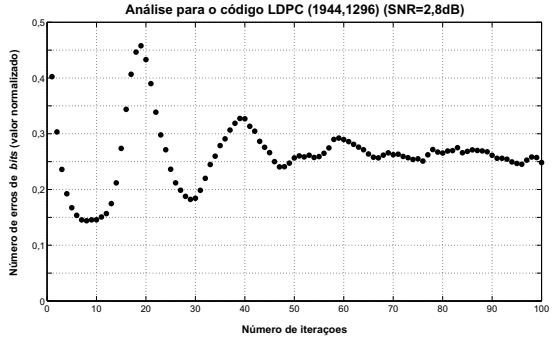


Figura 6.42: *Análise de eventos de erros para o código LDPC (*$1944, 1296$*) para* $SNR = 2,8$ *dB.*

O comportamento dos eventos de erros para os códigos LDPC do padrão IEEE802.11n investigados afeta o desempenho do sistema DH, especialmente quando na sua etapa de decodificação *min-sum* BP é empregado $I_{\text{BP}} < 50$ iterações. Para esses casos, como é observado pela análise dos eventos de erros, há a predominância de conjuntos de armadilhas oscilantes.

Tomando-se, por exemplo, a análise de desempenho para o código LDPC $(648, 324)$, cujas curvas estão plotadas na Figura 6.14, observa-se que o sistema DH para um ciclo de operação, empregando uma etapa *min-sum* BP com $I_{\text{BP}} = 5$ iterações, atinge um desempenho equivalente, em termos da taxa WER, com uma SNR de cerca de 1 dB inferior àquela apresentada na decodificação convencional

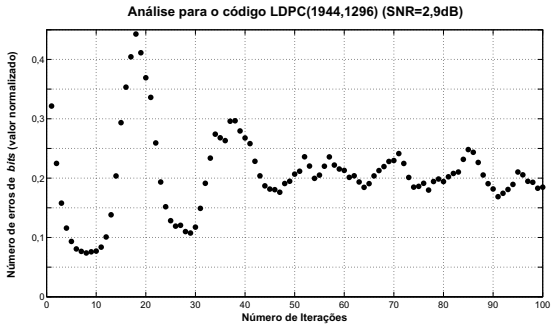


Figura 6.43: *Análise de eventos de erros para o código LDPC (*$1944, 1296$*) para* $SNR = 2,9$ *dB.*

min-sum BP com $I_{\text{BP}} = 5$ iterações. Esse mesmo sistema DH para dois ciclos de operação, totalizando 10 iterações na etapa *min-sum* BP, atinge o mesmo desempenho de decodificador convencional *min-sum* BP com $I_{\text{BP}} = 20$ iterações para uma SNR de cerca de $0,3$ dB superior. De forma similar, a decodificação DH para o código LDPC $(648, 432)$, para dois ciclos de operação e com uma etapa *min-sum* BP com $I_{\text{BP}} = 5$ iterações, como mostrado na Figura 6.17, atinge o mesmo desempenho de um decodificador convencional *min-sum* BP com $I_{\text{BP}} = 20$ iterações para uma SNR de cerca de $0,1$ dB superior. Em ambos os casos, com base na análise de eventos de erros, esse número de iterações adotado para a etapa *min-sum* BP do sistema DH (5 iterações), conforme mostrado nas Figuras 6.32, 6.33, 6.34 e 6.35, corresponde a um *ponto de vale* em que são mais prováveis os padrões de erros com menor cardinalidade.

Por outro lado, os desempenhos obtidos para os códigos LDPC $(648, 324)$ e LDPC $(648, 432)$ em decodificação DH, empregando uma etapa *min-sum* BP com $I_{\text{BP}} = 12$ iterações, são bem distintos, como mostrado nas Figuras 6.15 e 6.18, respectivamente. Para o código LDPC $(648, 324)$, o desempenho do sistema DH para dois ciclos de operação é equivalente ao do decodificador convencional *min-sum* BP com $I_{\text{BP}} = 50$ iterações, enquanto para o código LDPC $(648, 432)$, o desempenho do sistema DH para dois ciclos é cerca $0,15$ dB inferior. Para o código LDPC $(648, 324)$, esse número de iterações adotado para a etapa *min-sum* BP do sistema DH (12 iterações) está associada a um *ponto de vale* enquanto para o código LDPC $(648, 432)$ é associada a um *ponto de pico*.

Esse comportamento se inverte para as curvas de desempenho para os códigos LDPC $(648, 324)$ e LDPC $(648, 432)$ em decodificação DH, empregando uma etapa *min-sum* BP com $I_{\text{BP}} = 20$ iterações, como mostrado nas Figuras 6.16 e 6.19, respectivamente. Para o código LDPC $(648, 432)$, o desempenho do sistema DH para dois ciclos de operação é equivalente ao do decodificador convencional *min-sum* BP com $I_{\text{BP}} = 80$ iterações, especialmente para valores altos de SNR, enquanto para o código LDPC $(648, 324)$, o desempenho do sistema DH para dois ciclos é ligeiramente inferior ($< 0,1$ dB). Para o código LDPC $(648, 432)$, esse número de iterações adotado para a etapa *min-sum* BP do sistema DH (12 iterações) está associada a um *ponto de vale* enquanto para o código LDPC $(648, 324)$ é associada a um *ponto de pico*.

Essas mesmas observações são obtidas para os demais códigos LDPC investigados. Para o código LDPC $(1296, 648)$, o desempenho do sistema DH para dois ciclos de operação, empregando $I_{\text{BP}} = 12$ iterações no seu estágio *min-sum* BP, é equivalente ao decodificador convencional *min-sum* BP com 50 iterações, como mostrado na Figura 6.20, especialmente para valores altos de SNR. No entanto, quando a etapa *min-sum* BP emprega $I_{\text{BP}} = 25$ iterações, o desempenho do sistema

DH é inferior ao de um decodificador convencional *min-sum* BP com 200 iterações, como mostra a Figura 6.6. O número de iterações adotado no primeiro caso para a etapa *min-sum* BP do sistema DH, ou seja, $I_{\mathrm{BP}} = 12$ iterações, é associada a um *ponto de vale*, ao passo que no segundo caso, ou seja, $I_{\mathrm{BP}} = 25$ iterações, é associada a um *ponto de pico*, conforme mostrado nas Figuras 6.36 e 6.37.

Para o código LDPC $(1296, 864)$, o desempenho do sistema DH para dois ciclos de operação, empregando na sua etapa de decodificação *min-sum* BP $I_{\mathrm{BP}} = 12$ iterações ou $I_{\mathrm{BP}} = 25$ iterações, é equivalente ao decodificador convencional *min-sum* BP de referência que emprega um número de iterações bem mais elevado, como mostram as Figuras 6.21 e 6.8, respectivamente. Em ambos os casos, os valores para as iterações empregados pela etapa *min-sum* BP do sistema DH estão associadas a *pontos de vale*, conforme mostrado nas Figuras 6.38 e 6.39.

6.6 Análise da Atuação do Sistema DH na Região de Patamar de Erros

Como é destacado neste trabalho de tese, o sistema DH proposto atua prioritariamente e apresenta um melhor desempenho na região de patamar de erros dos códigos LDPC, que ocorre para altos valores de SNR. Para ilustrar e sem perda de generalidade, a Figura 6.44 mostra algumas curvas de desempenho para o código LDPC $(1944, 972)$.

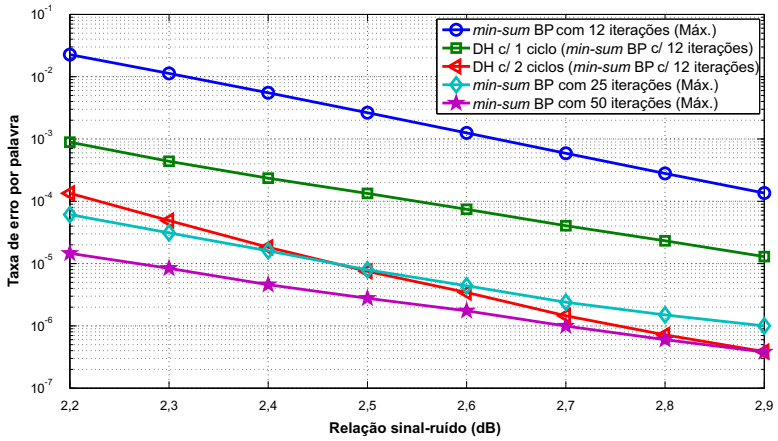


Figura 6.44: *Análise da atuação do sistema DH para o código LDPC (*$1944, 972$*).*

Observa-se que para baixos valores de SNR (valores menores do que $2,5$ dB), o desempenho do sistema DH, com dois ciclos de operação e empregando uma etapa *min-sum* BP com $I_{\mathrm{BP}} = 12$ iterações, perfazendo, para os dois ciclos, um número global de 24 iterações, é inferior ao obtido para o decodificador convencional do tipo *min-sum* BP utilizando 25 iterações, que

corresponde praticamente ao mesmo número de iterações empregado no sistema DH. Esse fraco desempenho do sistema DH se deve a dois fatores:

- ▷ Quando ocorre falha do primeiro estágio de decodificação de erros do sistema DH, os padrões de erros remanescentes são de cardinalidade elevada e, portanto, são de difícil detecção, uma vez que a maior cardinalidade do padrão de erros plenamente detectável é limitada ao valor $\mathcal{X}$;
- ▷ Com a não detecção plena dos padrões de erros remanescentes, a segunda etapa de decodificação iterativa de apagamentos do sistema DH introduz novos erros, uma vez que a correção dos apagamentos por meio da resolução das equações de verificação de paridade pode ser baseada em dígitos errados e que não foram detectados.

Por outro lado, para valores mais elevados de SNR, correspondendo à região de patamar de erros do código LDPC, o desempenho do sistema DH, com dois ciclos de operação e empregando uma etapa *min-sum* BP com $I_{\mathrm{BP}} = 12$ iterações, torna-se superior ao obtido para o decodificador convencional do tipo *min-sum* BP com um número global de 25 iterações e equivalente ao obtido para o decodificador convencional do tipo *min-sum* BP que emprega o dobro das iterações empregadas no sistema DH, ou seja, 50 iterações. Para esse caso, o principal fator que concorre na melhora de desempenho é a redução da cardinalidade dos padrões de erros remanescentes e que, portanto, podem ser plenamente detectados.

CAPÍTULO 7

CONCLUSÕES

Este capítulo descreve brevemente as contribuições desta tese e propõe alguns trabalhos para futuras investigações.

7.1 CONTRIBUIÇÕES DA TESE

Nesta tese, é proposta uma abordagem de decodificação híbrida (DH) que emprega a decodificação iterativa de apagamentos como pós-processamento [37] em caso de falha do decodificador iterativo de erros do tipo *min-sum* BP, possibilitando, de forma indireta, a correção dos erros remanescentes. Por se tratar de uma técnica empregada em canais com RAGB, os apagamentos a serem processados pelo decodificador iterativo de apagamentos são gerados por um canal BEC [40] criado artificialmente e que utiliza as informações de confiabilidade dos símbolos, estimadas na última iteração do decodificador de erros *min-sum* BP. Essas informações, na forma de valores absolutos de QLLR [58], são organizadas em ordem crescente e um total de $\mathcal{X}$ dígitos na N-upla estimada, associados ao menores valores, são selecionados para serem apagados. A análise realizada nesta tese, e corroborada na literatura [83], indica que a maioria dos erros remanescentes de uma decodificação *min-sum* BP sem sucesso apresenta baixo valor de QLLR e, portanto, esses erros têm grande probabilidade de estarem inclusos nas $\mathcal{X}$ coordenadas selecionadas para serem apagadas na N-upla.

Com o intuito de otimizar o desempenho do sistema DH, é realizado preliminarmente um ajuste sobre o BEC artificial de modo que os $\mathcal{X}$ apagamentos introduzidos atendam ao compromisso de obter o maior percentual possível de detecção dos padrões de erros remanescentes da decodificação *min-sum* BP sem sucesso e o maior percentual possível de correção, pelo decodificador iterativo

de apagamentos, desses padrões contendo $\mathcal{X}$ apagamentos. Quanto maior o valor do parâmetro $\mathcal{X}$, maior é o percentual de detecção dos padrões de erros remanescentes. Por outro lado, o aumento da cardinalidade do conjunto de padrões de apagamentos gerados pelo BEC artificial, decorrente do aumento do valor do parâmetro $\mathcal{X}$, provoca o aparecimento de conjuntos de parada [77], o que restringe o desempenho do decodificador iterativo de apagamentos. Esse ajuste do parâmetro $\mathcal{X}$ é realizado para cada código LDPC usado e, uma vez determinado esse valor, ele passa a ser fixo e independente da SNR do canal com RAGB.

Tipicamente, os valores do parâmetro $\mathcal{X}$ determinados para os códigos LDPC do padrão IEEE$802.11n$ [42] analisados nesta tese, são bem maiores do que a distância mínima do código [38], [39], d_{min}, o que possibilita ao sistema DH, por utilizar a decodificação iterativa de apagamentos, se beneficiar do fato de que os códigos de bloco lineares possuem excelente capacidade de corrigir uma grande quantidade de padrões de apagamentos com cardinalidade superior ao limite d_{min} [43].

É importante ressaltar que esse parâmetro $\mathcal{X}$ é limitado à capacidade do BEC, $\mathbb{C}$, que é expressa por $\mathbb{C} = 1 - \varepsilon$, em que ε representa a probabilidade de apagamento deste canal.

Um ciclo de decodificação DH inclui obrigatoriamente uma passagem pelo decodificador *min-sum* BP e, em caso de falha, uma passagem pelo decodificador iterativo de apagamentos. Esse sistema DH tem característica iterativa podendo ser executado em até I_{DH} ciclos de operação, em que I_{DH} corresponde a um número inteiro positivo. Os resultados obtidos neste trabalho mostram que o sistema DH se torna mais eficiente quando executado em até dois ciclos de operação, ou seja, $I_{\mathrm{DH}} = 2$. Acima de dois ciclos de operação, competem os seguintes fatores para a ocorrência de menores ganhos incrementais no desempenho do sistema DH:

- Os valores estimados de QLLR, mesmo para *bits* em erro, assumem valores elevados e já não se consegue detectá-los com eficiência; e
- São introduzidos novos erros pela etapa de decodificação iterativa de apagamentos, devido à presença, nas equações de verificação de paridade, de *bits* que não foram detectados e corrigidos.

Outrossim, o sistema DH atua prioritariamente na região de patamar de erros [25], [16] dos códigos LDPC, porquanto:

- Apresenta um melhor desempenho porque, em geral, os erros remanescentes de uma decodificação *min-sum* BP sem sucesso fazem parte de conjuntos de armadilhas de pequena cardinalidade [25], podendo ser mais facilmente detectados ainda que seja adotado um valor baixo para $\mathcal{X}$.
- Apresenta maior eficiência computacional, uma vez que a quantidade de erros nesta região é sig-

nificativamente baixa [95] e a inclusão do processo de decodificação iterativa de apagamentos não implica custo computacional significativo.

Uma vez que esse método se torna eficaz para a região de patamar de erros dos códigos LDPC, pode ser utilizado como uma alternativa para a codificação concatenada serial [26], que atua na mesma região, com a vantagem de trabalhar com apenas um único código (código LDPC), com taxas maiores e com desempenho e complexidade equivalentes.

Nesta tese, é feita a comparação do sistema DH com um modelo simplificado de codificação concatenada serial, baseado naquele usado no padrão de TV digital DTMB [27], [28], e os resultados de desempenho para esses sistemas são bem próximos.

O fato do sistema DH empregar um estágio decodificador de apagamentos torna possível a redução do número máximo de iterações do estágio de decodificação de erros do tipo *min-sum* BP e, ainda assim, conseguir um desempenho equivalente àquele que seria obtido por um único decodificador *min-sum* BP operando com um número bem maior de iterações. Nesse caso, o tempo de execução do sistema DH se torna relativamente baixo porque, além de haver a redução do número máximo de iterações no estágio de decodificação *min-sum* BP, a etapa de decodificação iterativa de apagamentos é de baixa complexidade porque emprega principalmente operações simples de lógica OU-exclusivo na correção dos apagamentos que são rapidamente resolvidas.

Um outro aspecto investigado nesta tese diz respeito ao comportamento da quantidade de erros de dígitos de uma dada N-upla em função das iterações intermediárias da decodificação *min-sum* BP sem sucesso e o seu impacto sobre o desempenho do sistema DH. É observado que, para os códigos LDPC do padrão IEEE$802.11n$, a curva gerada dessa análise tem característica oscilatória para um número baixo de iterações. Portanto, ocorrem pontos, para alguns valores intermediários do número de iterações, denotados como *pontos de vale*, em que se atinge a menor quantidade possível de erros. Dessa maneira, é possível tornar o sistema DH mais eficaz se o estágio de decodificação *min-sum* BP operar com um número de iterações que implique um *ponto de vale*. Adicionalmente, como neste caso é empregado um número baixo de iterações na etapa *min-sum* BP do sistema DH, o percentual de detecção de padrões de erros remanescentes da decodificação *min-sum* BP é maior, como é demonstrado nesta tese. A combinação desses dois fatores permite que, para o código LDPC $(648, 324)$ por exemplo, o sistema DH, com apenas um ciclo de operação, tenha um desempenho superior a 1 dB em relação ao do decodificador *min-sum* BP, operando com o mesmo número de iterações da etapa de decodificação de erros do sistema DH. Por outro lado, para um número mais elevado de iterações na etapa de decodificação *min-sum* BP, em razão principalmente do baixo percentual de

detecção dos padrões de erros remanescentes, esse ganho se reduz para $0,1$ dB a $0,2$ dB.

7.2 SUGESTÕES DE TRABALHOS FUTUROS

A seguir, são apresentadas algumas sugestões para trabalhos futuros que podem ser realizados com base nos resultados expostos nesta tese.

- ▷ Extensão da análise do sistema DH para outros tipos de códigos LDPC;
- ▷ Investigar o sistema DH para algoritmos de decodificação com decisão suave usando método serial de atualização de mensagens [19]–[23];
- ▷ Análise do sistema DH para outros tipos de canais, em especial para canais com desvanecimento [96];
- ▷ Implementação do sistema DH com tecnologia *Field Programmable Gate Array* (FPGA).

7.3 ARTIGOS PUBLICADOS

- ▷ Guimarães, W. P. S. e da Rocha Jr., V. C.: 'Técnica de decodificação híbrida iterativa para códigos LDPC', Proc. of the XXX Simpósio Brasileiro de Telecomunicações (SBRT2012), Setembro 2012.
- ▷ Guimarães, W. P. S., Lemos-Neto, J. S., and da Rocha Jr., V. C.: 'A hybrid iterative decoder for LDPC codes', Proc. of the 9th International Symposium on Wireless Communication Systems, (ISWCS 2012), August 2012, pp. 979-983.
- ▷ Guimarães, W. P. S., Lemos-Neto, J.S. and da Rocha Jr., V. C.: 'Improved Use of a Hybrid Decoding Technique for LDPC Codes', *IET Electronics Letters*. (submetido)

REFERÊNCIAS

[1] C. E. SHANNON, A mathematical theory of communication, *Bell System Technical Journal*, v. 27, n. 3 and 4, p. 379–423 and 623–656, July and October 1948.

[2] C. BERROU, A. GLAVIEUX, & P. THITIMAJSHIMA, Near Shannon limit error-correcting coding and decoding: Turbo codes, In: **Proceedings of the IEEE International Conference on Communications (ICC´93)**, Geneva, May 1993, p. 1064–1070.

[3] R. G. GALLAGER, Low-density parity-check codes, *IRE Transactions Information Theory*, v. IT-8, p. 21–28, January 1962.

[4] ——, Low-density parity-check codes, Tese, MIT Press, Cambridge, MA, 1963.

[5] R. TANNER, A recursive approach to low complexity codes, *IEEE Transactions on Information Theory*, v. IT-27, n. 5, September 1981.

[6] D. MACKAY & R. NEAL, Near Shannon limit performance of low density parity check codes, *Electronics Letters*, v. 32, n. 18, August 1996.

[7] X. Y. HU, E. ELEFTHERIOU, & D. ARNOLD, Progressive edge-growth Tanner graphs, In: **Proceedings of the IEEE Global Telecommunications Conference, 2001. GLOBECOM '01.**, v. 1, San Antonio, USA, November 2001, p. 995–1001.

[8] T. J. RICHARDSON, M. A. SHOKROLLAHI, & R. L. URBANKE, Design of capacity-approaching irregular low-density parity-check codes, *IEEE Transactions on Information Theory*, v. 47, n. 2, p. 619–637, February 2001.

[9] S. CHUNG, G. FORNEY, T. RICHARDSON, & R. URBANKE, On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit, *IEEE Communications Letters*, v. 5, n. 2, p. 58–60, February 2001.

[10] T. TIAN, C. JONES, J. D. VILLASENOR, & R. D. WESEL, Construction of irregular LDPC codes with low error floors, In: **Proceedings of the IEEE International Conference on Communications**, v. 5, May 2003, p. 3125–3129.

[11] J. XU, L. CHEN, I. DJURDJEVIC, S. LIN, & K. ABDEL-GHAFFAR, Construction of regular and irregular LDPC codes: geometry decomposition and masking, *IEEE Trans. Inform. Theory*, v. 53, n. 1, p. 121–134, January 2007.

[12] T. RICHARDSON & R. URBANKE, **Modern Coding Theory**. Cambridge University Press, 2008.

[13] S. JOHNSON, **Iterative Error Correction Turbo, Low-Density Parity-Check and Repeat-Accumulate Codes**. New York, USA: Cambridge University Press, 2010.

[14] W. RYAN & S. LIN, **Channel Codes: Classical and Modern**. Cambridge University Press, 2009.

[15] T. RICHARDSON & R. URBANKE, The renaissance of Gallager's low-density parity-check codes, *IEEE Communications Magazine*, v. 41, n. 8, p. 126–130, August 2003.

[16] T. K. MOON, **Error Correction Coding: Mathematical Methods and Algorithms**. New Jersey, USA: John Wiley & Sons, Inc., 2005.

[17] J. MOREIRA & P. G. FARRELL, **Essentials of Error-Control Codes**. New York, USA: John Wiley & Sons Ltd., 2006.

[18] F. R. KSCHISCHANG & B. J. FREY, Iterative decoding of compound codes by probability propagation in graphical models, *IEEE Journal on Selected Areas in Communication*, v. 16, n. 2, p. 219–230, February 1998.

[19] E. SHARON, S. LITSYN, & J. GOLDBERGER, An efficient message-passing schedule for LDPC decoding, In: **Proceedings of the 23rd IEEE Convention of Electrical and Electronics Engineers**, Tel-Aviv, Israel, September 2004, p. 223–226.

[20] H. KFIR & I. KANTER, Parallel versus sequential updating for belief propagation decoding, *Physica A*, v. 330, p. 259–270, September 2003.

[21] J. ZHANG & M. FOSSORIER, Shuffled iterative decoding, *IEEE Transactions on Communications*, v. 2, n. 53, p. 209–213, February 2005.

[22] G. HAN & X. LIU, An efficient dynamic schedule for layered belief-propagation decoding of LDPC codes, *IEEE Communications Letters*, v. 13, n. 12, p. 950–952, December 2009.

[23] A. I. VILA CASADO, M. GRIOT, & R. R. WESEL, Informed dynamic scheduling for belief-propagation decoding of LDPC codes, In: **Proceedings of the IEEE ICC'07**, Glasgow, Scotland, June 2007.

[24] IEEE, **LDPC Support in IEEE**802.16**e, IEEE** $C802.16e-05/066r2$, 2005.

[25] T. RICHARDSON, Error-floors of LDPC codes, In: **Proceedings of the 41st Annual Allerton Conference on Communication, Control and Computing**, Monticello, USA, September 2003, p. 1426–1435.

[26] S. BENEDETTO, G. MONTORSI, & D. DIVSALAR, Concatenated convolutional codes with interleavers, *IEEE Communications Magazine*, v. 41, n. 8, p. 102 –109, August 2003.

[27] **Framing Structure, Channel Coding and Modulation for Digital Television Terrestrial Broadcasting System, GB20600-2006**, August 2006.

[28] J. SONG, Z. YANG, L. YANG, K. GONG, C. PAN, J. WANG, & Y. WU, Technical review on chinese digital terrestrial television broadcasting standard and measurements on some working modes, *IEEE Transactions on Broadcasting*, v. 53, n. 1, p. 1–7, March 2007.

[29] ETSI, **Digital Video Broadcasting (DVB); Second generation framing structure, channel coding and modulation systems for Broadcasting, Interactive services, News Gathering and others broadband satellite applications (DVB-S2), EN 302 307**, August 2009.

[30] S. LANDNER & O. MILENKOVIC, Algorithmic and combinatorial analysis of trapping sets in structured LDPC codes, In: **Proceedings of the International Conference on Wireless Networks, Communications and Mobile Computing**, Maui, USA, June 2005, p. 630–635.

[31] M. CHERTKOV, Reducing the error floor, In: **Proceedings of the Information Theory Workshop**, Lake Tahoe, USA, September 2007, p. 230–235.

[32] E. CAVUS & B. DANESHRAD, A performance improvement and error floor avoidance technique for belief propagation decoding of LDPC codes, In: **Proceedings of the 16th IEEE International Symp. Personal, Indoor Mobile Radio Commun.**, v. 4, Berlin, Germany, September 2005, p. 2386–2390.

[33] M. JIANG, C. ZHAO, E. XU, & L. ZHANG, Reliability-based iterative decoding of LDPC codes using likelihood accumulation, *IEEE Communications Letters*, v. 11, n. 8, p. 677–679, August 2007.

[34] J. PEARL, **Probabilistic Reasoning in Intelligent Systems**, 2ª ed. San Francisco, USA: Kaufmann, 1988.

[35] M. P. C. FOSSORIER, Iterative reliability-based decoding of low-density parity check codes, *IEEE Journal Sel. Areas Communications*, v. 19, n. 5, p. 908–917, May 2001.

[36] B. SKLAR, A primer on turbo code concept, *IEEE Communications Magazine*, v. 35, n. 12, December 1997.

[37] Y. HAN & W. RYAN, Low-floor decoders for LDPC codes, *IEEE Transactions on Communications*, v. 57, p. 1663–1673, June 2009.

[38] S. LIN & D. J. COSTELLO JR., **Error Control Coding: Fundamentals and Applications**. New Jersey, USA: Prentice Hall, Inc. Englewood Cliffs, 2004.

[39] S. B. WICKER, **Error Control Systems for Digital Communication and Storage**. Prentice Hall, 1995.

[40] P. ELIAS, Coding for two noisy channels, In: **Proceedings of 3rd London Symposium on Information Theory**, London, England, 1955, p. 61–76.

[41] M. P. C. FOSSORIER, M. MIHALJEVIC, & H. IMAI, Reduced complexity iterative decoding of low density parity check codes based on belief propagation, *IEEE Transactions on Communications*, v. 47, n. 5, p. 673–680, May 1999.

[42] **IEEE 802.11n-D1.0 - IEEE 802.11n Wireless LAN Medium Access Control MAC and Physical Layer PHY specifications**, 2006.

[43] P. R. FREITAS, V. C. DA ROCHA JR., & J. S. LEMOS-NETO, On the iteractive decoding of binary product codes over the binary erasure channel, In: **Proceedings of 8th International Symposium on Wireless Communication Systems**, Aachen, Germany, November 2011, p. 126–130.

[44] A. PRABHAKAR & K. NARAYANAN, Pseudorandom construction of LDPC codes using linear congruential sequences, *IEEE Transactions on Communications*, v. 50, n. 9, p. 1389–1396, September 2002.

[45] Y. KOU, S. LIN, & M. P. C. FOSSORIER, Low-density parity-check codes based on finite geometries: A rediscovery and new results, *IEEE Transactions on Information Theory*, v. 47, n. 7, p. 2711–2736, November 2001.

[46] S. JOHNSON & S. R. WELLER, A family of irregular LDPC codes with low encoding complexity, *IEEE Communications Letters*, v. 7, n. 2, p. 79–81, February 2003.

[47] M. M. VASCONCELOS, Decodificação iterativa de códigos baseados em matrizes de verificação de paridade esparsas, Dissertação (Mestrado em Eng. Elétrica), Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Pernambuco, Recife, PE, 2006.

[48] F. R. KSCHISCHANG, B. J. FREY, & H. LOELIGER, Factor graphs and the sum-product algorithm, *IEEE Transactions on Information Theory*, v. 47, n. 2, p. 498–519, February 2001.

[49] M. A. CORDEIRO, Decodificação iterativa de códigos LDPC em canais discretos com quantização uniforme, Dissertação (Mestrado em Eng. Elétrica), Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Pernambuco, Recife, PE, 2010.

[50] Y. MAO & A. H. BANIHASHEMI, Decoding low-density parity-check codes with probabilistic scheduling, *IEEE Communication Letters*, v. 5, n. 10, p. 414–416, October 2001.

[51] M. A. C. GOMES, Codigos binarios definidos por matrizes de teste de paridade esparsas algoritmos de descodificação, Dissertação (Mestrado em Eng. Elétrica), Departamento de Engenharia Electrotécnica e de Computadores da Faculdade Ciências e Tecnologia da Universidade de Coimbra, Coimbra, Portugal, 2003.

[52] F. R. KSCHISCHANG, Codes defined on graphs, *IEEE Communications Magazine*, p. 118–125, August 2003.

[53] H. LOELIGER, An introduction to factor graphs, *IEEE Signal Processing Magazine*, v. 21, n. 1, p. 28–41, January 2004.

[54] S. A. ABRANTES, Descodificação iterativa de códigos LDPC por transferência de mensagens em grafos de factores, Porto, Portugal, Julho 2005. [Online]. Disponível: http://repositorio-aberto.up.pt/handle/10216/274

[55] G. LECHNER, Convergence of sum-product algorithm for finite length low-density parity-check codes, *Winter School on Coding and Information Theory*, February 2003.

[56] D. J. C. MACKAY, **Information Theory, Inference and Learning Algorithms**. New York, USA: Cambridge University Press, 2003.

[57] B. J. FREY, F. R. KSCHISCHANG, H. A. LOELIGER, & N. WIBERG, Factor graphs and algorithms, In: **Proceedings of the 35th Annual Allerton Conf. on Commun., Control and Computing**, Monticello, USA, September 29-October 1 1997, p. 666–680.

[58] J. CHEN, A. DHOLAKIA, E. ELEFTHERIOU, M. P. C. FOSSORIER, & X. Y. HU, Reduced-complexity decoding of LDPC codes, *IEEE Transactions on Communications*, v. 53, n. 8, p. 1288–1299, August 2005.

[59] J. HAGENAUER, E. OFFER, & L. PAPKE, Iterative decoding of binary block and convolutional codes, *IEEE Transactions on Information Theory*, v. 42, p. 429–445, March 1996.

[60] D. MACKAY & M. POSTOL, Weaknesses of Margulis and Ramanujan-Margulis low-density parity-check codes, *Electronic Notes in Theoretical Computer Science*, v. 74, 2003.

[61] S. CHILAPPAGARI, S. SANKARANARAYANAN, & B. VASIC, Error floors of LDPC codes on the binary symmetric channel, In: **Proceedings of the IEEE International Conference on Communications**, Istambul, Turkey, June 2006, p. 1089–1094.

[62] L. DOLECEK, P. LEE, Z. ZHANG, V. ANANTHARAM, B. NIKOLIC, & M. WAINWRIGHT, Predicting error floors of structured LDPC codes: Deterministic bounds and estimates, *IEEE Journal on Selected Areas in Communications*, v. 27, n. 6, p. 908–917, 2009.

[63] L. RIZZO, Effective erasure codes for reliable computer communication protocols, *ACM SIGCOMM Computer Communication Review*, v. 27, n. 2, p. 24–36, April 1997.

[64] J. CAI, M. TOMLINSON, C. TJHAI, M. AMBROZE, & M. AHMED, Advances in iterative decoding and maximum likelihood decoding for the packet network with comparisons to fountain codes over the erasure channel, In: **Proceedings of 4th International Symposium on Turbo Codes in Connection with 6th International ITG-Conference on Source and Channel Coding**, Munich, Germany, 2006, p. 3–7.

[65] ——, Comparison of concatenated Reed-Solomon coding with hard-decision to soft-decision LDPC coding for the Rayleigh fading channel, In: **Proceedings of IEEE Information Theory Workshop**, Chengdu, China, 2006, p. 135–139.

[66] M. O. RABIN, Efficient dispersal of information for security, load balancing and fault tolerance, *Journal ACM*, v. 36, n. 2, p. 335–348, 1989.

[67] S. CHANGUEL, R. LE BIDAN, & R. PYNDIAH, Iterative decoding of product codes over binary erasure channel, *Electronics Letters*, v. 46, n. 7, p. 503–505, 2010.

[68] E. ROSNES & O. YTREHUS, Turbo decoding on the binary erasure channel: Finite-length analysis and turbo stopping sets, *IEEE Transactions on Information Theory*, v. 53, n. 11, p. 4059–4075, November 2007.

[69] M. G. LUBY, M. MITZENMACHER, M. A. SHOKROLLAHI, & D. A. SPIELMAN, Efficient erasure correcting codes, *IEEE Transactions on Information Theory*, v. 47, n. 2, p. 569–584, February 2001.

[70] W. PETERSON & E. WELDON JR., **Error-Correcting Codes**. Massachussets, USA: MIT Press, 1972.

[71] F. J. MACWILLIAMS & N. J. A. SLOANE, **The Theory of Error-Correcting Codes**. Amsterdam, Netherlands: North-Holland, 1977.

[72] I. DUMER & P. FARRELL, Erasure correction performance of linear block codes, In: **Proceedings of First French-Israeli Workshop on Algebraic Coding**, London, England, 1993, p. 316–326.

[73] R. E. BLAHUT, **Algebraic Codes for Data Transmission**. New York, USA: Cambridge University Press, 2003.

[74] M. TOMLINSON, C. TJHAI, J. CAI, & M. AMBROZE, Analysis of the distribution of the number of erasures correctable by a binary linear code and the link to low-weight codewords, *IET Communications*, v. 1, n. 3, p. 539–548, 2007.

[75] M. G. LUBY, M. MITZENMACHER, M. A. SHOKROLLAHI, & D. SPIELMAN, Analysis of low density codes and improved designs using irregular graphs, In: **Proceedings of thirtieth annual ACM symposium on Theory of Computing (STOC'98)**, New York, USA, 1998, p. 249–258.

[76] T. J. RICHARDSON & R. L. URBANKE, The capacity of low-density parity check codes under message-passing decoding, *IEEE Transactions on Information Theory*, v. 47, n. 2, p. 599–618, February 2001.

[77] C. DI, D. PROIETTI, I. E. TELATAR, T. J. RICHARDSON, & R. L. URBANKE, Finite-length analysis of low-density parity-check codes on the binary erasure channel, *IEEE Transactions on Information Theory*, v. 48, n. 6, p. 1570–1579, June 2002.

[78] N. KASHYAP & A. VARDY, Stopping sets in codes from designs, In: **Proceedings of the IEEE International Symposium on Information Theory**, Yokohama, Japan, June/July 2003, p. 1222.

[79] M. SCHWARTZ & A. VARDY, On the stopping distance and the stopping redundancy of codes, *IEEE Transactions on Information Theory*, v. 52, n. 3, p. 922–932, March 2006.

[80] H. R. ZEIDAN & M. M. ELSABROUTY, Two-stage hybrid decoding for low-density parity check (LDPC) codes, In: **Proceedings of the 4th International Conference on Innovations in Information Technology, IIT '07**, Dubai, November 2007, p. 650–654.

[81] M. ARDAKANI & F. R. KSCHISCHANG, Gear-shift decoding, *IEEE Transactions on Communications*, v. 54, n. 7, p. 1235–1242, July 2006.

[82] J. LI & X. ZHANG, Hybrid iterative decoding for low-density parity-check codes based on finite geometries, *IEEE Communications Letters*, v. 12, n. 1, p. 29–31, January 2008.

[83] Z. AI-MIN & Y. SEN-JIE, A modified belief propagation decoding algorithm of LDPC codes for fast convergence, In: **Proceedings of the International Conference on Communication Software and Networks**, Macau, February 2009, p. 518–522.

[84] E. ROSNES & O. YTREHUS, An efficient algorithm to find small-size stopping sets of low-density parity-check matrices, *IEEE Transactions on Information Theory*, v. 55, n. 9, p. 4167–4178, 2009.

[85] E. ROSNES, O. YTREHUS, M. A. AMBROZE, & M. TOMLINSON, Addendum to "An efficient algorithm to find small-size stopping sets of low-density parity-check matrices", *IEEE Transactions on Information Theory*, v. 58, n. 1, p. 164–171, 2012.

[86] G. RICHTER, Finding small stopping sets in the Tanner graphs of LDPC codes, In: **Proceedings of the 6th International ITG-Conference on Source and Channel Coding**, Munich, Germany, April 2006, p. 1–5.

[87] S. KAKAKHAIL, S. REYNAL, D. DECLERCQ, & V. HEINRICH, An efficient pseudo-codeword search algorithm for belief propagation decoding of LDPC codes, In: **Proceedings of the ICUMT'09**, St. Petersburg, Russia, October 2009.

[88] H. JIN, A. KHANDEKAR, & R. MCELIECE, Irregular repeat-accumulate codes, In: **Proceedings of the 2nd International Symposium on Turbo Codes & Related Topics**, Brest, France, September 2000, p. 1–8.

[89] G. D. FORNEY JR., **Concatenated Codes**. Cambridge, USA: MA: MIT Press, 1966.

[90] ABNT, **ABNT NBR**15601 **: Televisão digital terrestre - Sistema de transmissão**, Abril 2008.

[91] L. L. PETERSON & B. S. DAVIE, **Computer Networks, A Systems Approach**, 2ª ed. Morgan Kaufmann Publishers, 2000.

[92] S. GOUNAI, T. OHTSUKI, & T. KANEKO, Modified belief propagation decoding algorithm for low-density parity check code based on oscillation, In: **Proceedings of the IEEE 63rd Vehicular Technology Conference, 2006. VTC 2006-Spring.**, v. 3, Melbourne, Australia, May 2006, p. 1467–1471.

[93] E. ALGHONAIM, M. DHAHRAN, & A. EL-MALEH, Improving BER performance of LDPC codes based on intermediate decoding results, In: **Proceedings of the IEEE International Conference on Signal Processing and Communications - ICSPC 2007**, Dubai, November 2007, p. 1547–1550.

[94] A. KUMAR & A. DUKKIPATI, A two stage selective averaging LDPC decoding, In: **Proceedings of the 2012 IEEE International Symposium on Information Theory Proceedings (ISIT)**, Cambridge, USA, July 2012, p. 2866–2870.

[95] Y. ZHANG & W. E. RYAN, Toward low LDPC-code floors: A case study, *IEEE Transactions on Communications*, v. 57, n. 6, p. 1566–1573, June 2009.

[96] T. S. RAPPAPORT, **Wireless Communications — Principles and Practice**, 2ª ed. Prentice Hall PTR, 2002.

SOBRE O AUTOR

O autor nasceu em Manaus, Amazonas, no dia 25 de Novembro de 1963. Formou-se em Engenharia Elétrica pela Universidade Federal do Amazonas em 1986 e obteve grau de Mestre em Engenharia Elétrica em 2003 pela Universidade Federal de Pernambuco. Seus interesses de pesquisa incluem Teoria da Informação, Códigos Corretores de Erro, Sistemas de Comunicação Digital e Processamento Digital de Sinais.

Endereço: Rua Raimundo A. Borges, 192, Conj. Petro
69083-150 Aleixo
Manaus – AM
Brasil

e-mail: wguimaraes@uea.edu.br, walter.prado@fucapi.br

Esta tese foi diagramada usando LaTeX 2_ε[1] pelo autor.

[1] LaTeX 2_ε é uma extensão do LaTeX. LaTeX é uma coleção de macros criadas por Leslie Lamport para o sistema TeX, que foi desenvolvido por Donald E. Knuth. TeX é uma marca registrada da Sociedade Americana de Matemática ($\mathcal{AMS}$). O estilo usado na formatação desta tese foi escrito por Dinesh Das, Universidade do Texas. Modificado por Renato José de Sobral Cintra (2001) e por André Leite Wanderley (2005), ambos da Universidade Federal de Pernambuco. Sua última modificação ocorreu em 2010 realizada por José Sampaio de Lemos Neto, também da Universidade Federal de Pernambuco.