UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Códigos LDPC Multinível para Codificação de Rede na Camada Física

Dissertação submetida à
Universidade Federal de Santa Catarina
como parte dos requisitos para a obtenção
do grau de Mestre em Engenharia Elétrica

Paulo Ricardo Branco da Silva

Orientador: Danilo Silva

Florianópolis, 27 de novembro de 2015.

PAULO RICARDO BRANCO DA SILVA

CÓDIGOS LDPC MULTINÍVEL PARA CODIFICAÇÃO DE REDE NA CAMADA FÍSICA

FLORIANÓPOLIS
2015

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Branco da Silva, Paulo Ricardo
Códigos LDPC Multinível para Codificação de Rede na Camada Física / Paulo Ricardo Branco da Silva ; orientador, Danilo Silva - Florianópolis, SC, 2015.
116 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Engenharia Elétrica.

Inclui referências

1. Engenharia Elétrica. 2. Codificação de Rede na Camada Física. 3. Eficiência Espectral. 4. Códigos LDPC. 5. Reticulados. I. Silva, Danilo. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia Elétrica. III. Título.

Paulo Ricardo Branco da Silva

CÓDIGOS LDPC MULTINÍVEL PARA CODIFICAÇÃO DE REDE NA CAMADA FÍSICA

Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Engenharia Elétrica, área de concentração Comunicações e Processamento de Sinais, e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina.

Florianópolis, 27 de novembro de 2015.

Carlos Galup Montoro, Dr.
Coordenador do Programa de Pós-Graduação em Engenharia Elétrica

Banca examinadora

Prof. Danilo Silva, Ph.D.
Universidade Federal de Santa Catarina

Prof. Cecilio José Lins Pimentel, Ph.D.
Universidade Federal de Pernambuco

Prof. Bartolomeu Ferreira Uchôa Filho, Ph.D.
Universidade Federal de Santa Catarina

Prof. Roberto Wanderley da Nóbrega, Dr.
Instituto Federal de Santa Catarina

Dedico este trabalho à minha família
por seu apoio incondicional

Agradecimentos

Desejo expressar meu agradecimento primeiramente a Danilo Silva, por ter sido, além de um grande orientador, uma pessoa que tentou corrigir as minhas falhas e me aconselhar em todos os momentos como eu deveria me organizar para resolver os problemas que apareciam. Agradeço todas as dicas e orientações, que foram imprescindíveis para eu chegar ao fim. Agradeço pelo tempo que dedicou à verificação do meu projeto, muitas vezes em horas fora do seu regime normal de trabalho.

Agradeço também a Ricardo Bohaczuk Venturelli por ter sido, além de um grande amigo, uma fonte de sugestões e dicas para a realização do meu trabalho. Agradeço-lhe especialmente pela ajuda com códigos e pelas valiosas ideias para as minhas apresentações.

Estendo meus agradecimentos igualmente ao Prof. Bartolomeu Ferreira Uchôa Filho, que, além de um ótimo professor, é um exemplo de comportamento. Sempre oferece comentários pertinentes de maneira calma e explica conceitos de forma muito elucidativa.

Gostaria também de mencionar o meu colega e amigo Bruno Fontana da Silva, que sempre fez comentários inteligentes e levantou questões que me fizeram reavaliar o meu trabalho. Sua leitura do meu artigo para a convenção ITS 2014 foi essencial para chegar à versão final do texto.

Dentre meus colaboradores de laboratório agradeço por último a Rodrigo Farias. Foi um grande amigo: acompanhou a minha pesquisa e me deu inúmeras sugestões e contribuições. Estudamos muitas vezes juntos e compartilhamos vários conhecimentos.

Acima de tudo, agradeço à minha família. Apoiaram-me muito e me deram forças para persistir e nunca abandonar o meu objetivo. Sou grato, em particular, à minha mãe, que não mediu esforços para me manter confiante que eu chegaria onde cheguei.

Agradeço à UFSC e ao CNPq os recursos financeiros necessários para a minha subsistência ao longo da pesquisa.

“Não é o trabalho, mas o saber trabalhar, que é o segredo do êxito no trabalho. Saber trabalhar quer dizer: não fazer um esforço inútil, persistir no esforço até o fim, e saber reconstruir uma orientação quando se verificou que ela era, ou se tornou, errada.”

FERNANDO PESSOA

Resumo da Dissertação apresentada à UFSC como parte dos requisitos necessários para obtenção do grau de Mestre em Engenharia Elétrica

CÓDIGOS LDPC MULTINÍVEL PARA CODIFICAÇÃO DE REDE NA CAMADA FÍSICA

Paulo Ricardo Branco da Silva

27 de novembro de 2015

Orientador: Danilo Silva
Área de concentração: Comunicações e Processamento de Sinais
Palavras-chave: Códigos Multinível, Códigos LDPC, Reticulados, Codificação de rede na camada física
Número de páginas: xviii + 98

Códigos de reticulado desempenham um papel fundamental na codificação de rede na camada física (*physical-layer network coding, PNC*), uma técnica de comunicação cooperativa que explora a interferência entre usuários para possibilitar um aumento de *throughput* em redes sem fio. O foco deste trabalho é o projeto de códigos de reticulado de baixa complexidade e alta eficiência espectral, especificamente utilizando códigos LDPC (*low-density parity-check codes*) irregulares aninhados em uma construção multinível. São descritos métodos de codificação e decodificação multi-estágio para códigos aninhados definidos por equações de paridade. Estes métodos permitem, ao menos teoricamente, um desempenho próximo do ótimo com baixa complexidade de decodificação. Dentre as contribuições realizadas destaca-se o projeto de distribuições de graus otimizadas para a decodificação multi-estágio. As distribuições satisfazem restrições de aninhamento, o que é essencial para a construção multinível resultar em um código de reticulado. Em um cenário PNC com dois usuários e desvanecimento Rayleigh, resultados de simulação mostram que o projeto de códigos de reticulado com códigos LDPC é promissor, o que permite concluir que estudos continuados para a melhoria do desempenho são de grande interesse.

Abstract of Dissertation presented to UFSC as a partial fulfillment of the requirements for the degree of Master in Electrical Engineering

LDPC MULTILEVEL CODE DESIGN FOR PHYSICAL-LAYER NETWORK CODING

Paulo Ricardo Branco da Silva

November 27th, 2015

Advisor: Danilo Silva
Area of concentration: Communications and Signal Processing
Keywords: Multilevel Codes, LDPC codes, Lattices, Physical-Layer Network Coding
Number of pages: xviii + 98

Lattice codes play an essential role in physical-layer network coding (PNC), a cooperative communication technique which exploits interference among users to enable an increase in the throughput of wireless networks. This thesis' objective is to design low complexity lattice codes with high spectral efficiency, in particular irregular nested LDPC (low-density parity-check) codes in a multilevel construction. Multi-stage coding and decoding methods intended for nested codes defined by parity check equations are described. These methods allow, at least theoretically, a performance close to optimum with low decoding complexity. Among the project's contributions the design of degree distributions optimized for multi-stage decoding stands out. These distributions meet nesting constraints, which is key for the multilevel construction to yield lattice codes. In a PNC scenario with two users and Rayleigh fading, simulation results show that LDPC lattice codes are promising, which comes to show that it is of great interest to further research these codes and find ways to improve their performance.

Sumário

Notação

Símbolo	Descrição
$\mathcal{B} \subseteq \mathcal{A}$	$\mathcal{B}$ é um subconjunto de $\mathcal{A}$.
$\mathcal{A}^n$	n-ésima potência cartesiana do conjunto $\mathcal{A}$, isto é, o conjunto de todas as n-tuplas com componentes em $\mathcal{A}$.
$\lvert\mathcal{A}\rvert$	Cardinalidade (número de elementos) do conjunto $\mathcal{A}$.
$\mathbb{N}$ ($\mathbb{N}^*$)	Números naturais, incluindo (excluindo) zero.
$\mathbb{Z}$	Números inteiros.
$\mathbb{R}$	Números reais.
$\mathbb{C}$	Números complexos.
$\Pr[A]$	Probabilidade do evento A.
$p(x)$	Probabilidade de a variável aleatória X assumir o valor x.
$p(x\lvert y)$	Probabilidade de a variável aleatória X assumir o valor x dado que Y assume o valor y.
$\mathrm{E}[X]$	Valor esperado da variável aleatória X.
$I(X;Y)$	Informação mútua entre as variáveis aleatórias X e Y.
$\lceil x \rceil$	Função teto de $x \in \mathbb{R}$ (menor inteiro maior ou igual a x).

CAPÍTULO 1

Introdução

1.1 Motivação e Contexto Geral

Há uma grande demanda pelo aumento das taxas de comunicação em redes sem fio. Para tanto, técnicas capazes de aumentar a eficiência espectral são necessárias.

Entre as várias estratégias existentes destaca-se a *Physical-Layer Network Coding* (PNC), um esquema de comunicação cooperativa que explora a interferência entre sinais em um canal sem fio. A ideia da PNC é extrair funções lineares das mensagens das fontes a partir de sinais recebidos combinados entre si e com ruído [20].

Sinais em uma rede sem fio estão sujeitos a desvanecimento, o qual pode ser modelado por meio de coeficientes de ganho do canal. A PNC tenta decodificar uma ou mais combinações lineares inteiras das mensagens [22], sendo que os coeficientes inteiros podem ser escolhidos livremente. Porém, alcança-se maior confiabilidade na decodificação quando são escolhidos próximos dos coeficientes do canal [9].

A busca por funções lineares leva ao uso de estruturas lineares, como os reticulados, subgrupos discretos de $\mathbb{R}^n$ fechados sob combinações lineares inteiras. Nazer e Gastpar demonstraram que códigos

de reticulado, subconjuntos de reticulado utilizados como códigos no espaço euclidiano, fornecem a estrutura espacial necessária para a decodificação da PNC [22].

Feng *et al.* [9] usaram códigos de reticulado cujos espaços de mensagens são módulos sobre inteiros gaussianos. Esta abordagem algébrica, chamada *Lattice Network Coding* (LNC), tem como maior vantagem não supor a priori nenhuma construção específica para os códigos de reticulado, o que lhe confere generalidade. Além disso, Feng *et al.* forneceram orientações práticas para a criação de códigos com as construções de reticulado A e D [9]. Em [31] encontramos uma análise detalhada e prática da LNC para a Construção A, através da qual se particularizam definições para $\mathbb{Z}[\mathrm{i}]$ e se obtém uma métrica de decodificação por mínima distância adequada para o Algoritmo de Viterbi.

A Construção A também foi objeto de estudo de [4], tendo sido empregada na criação (através de códigos turbo binários) de códigos de reticulado transmitidos via modulação 4-QAM. Os resultados, no entanto, se limitaram a eficiências espectrais inferiores a 1 bit por dimensão complexa. Na verdade, a Construção A só é capaz de alcançar alta eficiência espectral para códigos q-ários, onde $q > 2$ [9]. Porém, desejamos códigos simples e facilmente decodificáveis, ou seja, queremos códigos binários. Outras construções são necessárias para atender todas as nossas exigências.

No canal gaussiano uma maneira eficiente de aumentar as taxas efetivas é por meio de esquemas de codificação multinível. Desenvolvidos por Imai e Hirakawa, separam dados em níveis distintos, sobre os quais códigos potencialmente distintos [16] são aplicados. A grande vantagem se encontra na decodificação simples (a decodificação multi-estágio) capaz de atingir a capacidade do canal [37]. No entanto, não costumam construir reticulados. Para isto faz-se necessário o uso de construções de reticulado multinível, incluindo as construções D e D$'$ [5, 18].

A construção D$'$ é definida em função das equações de paridade de seus códigos componentes [5]. A sua definição naturalmente sugere o uso de códigos facilmente representados por matrizes de paridade. Uma extensão natural é o uso de códigos LDPC (*Low-Density Parity-Check*), os quais são definidos por matrizes de paridade esparsas. Além de possuírem estrutura compatível com a Construção D$'$, os LDPCs são códigos de alto desempenho. Os LDPCs irregulares, em particular, são

interessantes por propiciarem maior aproximação da capacidade. Em [13] encontramos, por exemplo, um estudo do uso de LDPCs irregulares com codificação multinível em canal gaussiano, unindo, portanto alto desempenho à eficiência espectral. Neste trabalho, desejamos dar um passo a mais e combinar LDPCs irregulares projetadas via *Density Evolution* [27] com reticulados.

1.2 Objetivos

O principal objetivo deste trabalho é projetar códigos de reticulado para o esquema PNC que sejam de fácil decodificação, permitam alta eficiência espectral e, ainda, usem códigos binários componentes cujos projeto e características gerais sejam bem conhecidos.

Para alcançar a meta de uma decodificação simples, trabalhamos na formulação de uma decodificação multi-estágio de reticulados. A decodificação multi-estágio de esquemas multinível tradicionais é bem conhecida [37], mas a sua generalização para reticulados é um problema pouco abordado. Desejamos não só desenvolver uma métrica de decodificação de reticulado, senão também entender as implicações do método.

Com o intuito de criar reticulados dedicamos atenção à Construção D e a códigos convolucionais. No entanto, na busca por melhor desempenho e maior flexibilidade de projeto, adotamos o uso de códigos LDPC, pressupondo, assim, o uso da Construção D′. Reticulados baseados em LDPCs regulares são propostos em [30]. Em [2], por sua vez, são discutidos reticulados criados com LDPCs irregulares otimizados para a decodificação conjunta. Este trabalho difere dos anteriores na medida em que projetamos códigos LDPC irregulares otimizados para a decodificação multi-estágio sobre reticulados.

1.3 Contribuições

Começamos o desenvolvimento com a criação de reticulados a partir de códigos convolucionais. Construções multinível de reticulado exigem o aninhamento dos códigos componentes, mas códigos convolucionais não são facilmente aninhados. Além disso, não possuem grande flexibilidade de projeto de taxas. A técnica mais usual para se obter uma taxa

desejada é por puncionamento, mas este procedimento compromete o aninhamento entre códigos. Por estas razões decidimos construir reticulados com códigos LDPC: códigos eficientes, com projeto de taxas flexível e de aninhamento relativamente mais fácil.

A codificação para um código de reticulado multinível é mais complicada que para um esquema multinível tradicional, pois níveis anteriores influenciam os posteriores. Por este motivo, a decodificação multi-estágio precisa encontrar uma forma de cancelar estas influências. Para os códigos convolucionais a decodificação é relativamente simples, pois permite o uso do Algoritmo de Viterbi, que, como visto em [9, 31], pode ser relacionado com a decodificação de reticulado. Com os códigos LDPC torna-se mais complexa, pois exige o conhecimento da codificação com a Construção D′. Mostramos nesta dissertação como codificar e decodificar códigos de reticulado criados com as construções D e D′. Demonstramos que a Construção D′ pode ser manipulada para a criação de códigos de reticulado codificados segundo a definição da Construção D. Mostramos, por fim, que as regras de codificação e decodificação para as duas construções são na verdade praticamente idênticas.

Destacamos também o projeto de códigos LDPC aninhados para a codificação multinível. A caracterização de canais equivalentes permite descrever o projeto como um problema de otimização da probabilidade de erro de códigos aninhados. Isto demanda adaptações à *Density Evolution*, que pode ser entendida grosso modo como a estimativa de desempenho de um código LDPC. A *Density Evolution* adaptada é utilizada internamente em um algoritmo de otimização. Escolhemos a *Differential Evolution*, um algoritmo genético bastante utilizado em projetos de códigos LDPC [13, 36], para buscar os parâmetros ótimos dos códigos. Nas condições de aninhamento e codificação multinível apresentadas, a *Differential Evolution* também requer adaptações. Adaptamos neste trabalho as técnicas genéticas mencionadas.

Por último, exibimos resultados de simulação com as Construções D e D′ que comprovam ser possível obter alta eficiência espectral usando o esquema PNC. Mostramos também a viabilidade do uso de um sistema LDPC multinível neste esquema de comunicação de rede.

1.4 Estrutura do Texto

No capítulo 2 apresentamos conhecimentos preliminares, que formam a base teórica deste trabalho. Revisamos esquemas de codificação multinível e reticulados, com ênfase nas construções de reticulado baseadas em códigos lineares. No capítulo 3 focamos nas nossas contribuições para a codificação e decodificação de códigos de reticulado aninhados. Detalhamos o projeto de códigos de alta eficiência espectral a partir da Construção D e um procedimento de geração de reticulados definidos pela Construção D′ através de geradores de reticulado (tal como observado na Construção D). No capítulo 4 revisamos os códigos LDPC, incluindo o projeto de códigos através da *Density Evolution*, o problema de otimização relacionado e o algoritmo de otimização global *Differential Evolution*. No capítulo 5 usamos as ferramentas de projeto da modulação codificada e de códigos LDPC em conjunto, adaptamolas e criamos um projeto de códigos multinível LDPC de reticulado. No capítulo 6 discutimos conceitos de PNC fundamentais para o nosso trabalho. Abordamos o procedimento de codificação e decodificação, incluindo as alterações que a PNC impõe à decodificação multi-estágio de códigos de reticulado. No capítulo 7 realizamos simulações de probabilidade de erro em dois cenários: em canais gaussiano e multi-acesso (MAC). Apresentamos e avaliamos os resultados de simulação. Por fim, no capítulo 8 apresentamos nossas conclusões. Encerramos o texto com propostas de projetos embasadas em oportunidades de pesquisa verificadas ao longo do desenvolvimento e em limitações observadas na nossa proposta atual.

CAPÍTULO 2

Preliminares

Neste capítulo discutimos na seção 2.1 elementos essenciais da teoria de códigos multinível, incluindo o conceito de canais equivalentes e a codificação e a decodificação multi-estágio. Na seção 2.2 revisamos conceitos de reticulados, incluindo as definições mais importantes para este trabalho. Por fim, na seção 2.3 introduzimos as construções de reticulado baseadas em códigos lineares. Nesta seção abordamos a construção mononível A e as construções multinível D e D′.

2.1 Códigos Multinível

O aumento da eficiência espectral em comunicações exige modificações ao projeto tradicional de códigos. O campo de estudo dedicado a encontrar códigos capazes de atingir alta eficiência espectral recebe o nome de modulação codificada.

Entre as técnicas de modulação codificada mais usadas encontram-se a *Trellis Coded Modulation* (TCM) e a *Bit Interleaved Coded Modulation* (BICM). A TCM se baseia no uso de códigos convolucionais e de um particionamento adequado dos pontos da constelação para aumentar a distância Euclidiana entre os símbolos, o que melhora o de-

sempenho do decodificador [35]. Neste esquema um certo número k_c de bits é codificado, enquanto que os demais $k - k_c$ bits permanecem sem codificação. A decodificação é feita em treliça por meio do Algoritmo de Viterbi com métricas Euclidianas [35]. A BICM, que é normalmente a forma mais prática de se realizar a modulação codificada por precisar de apenas um codificador, codifica os dados por meio de um único código e entrelaça aleatoriamente os símbolos codificados para depois modulá-los [8]. No decodificador o entrelaçamento é desfeito antes de se efetuar a decodificação. Sabe-se que a BICM obtém seus melhores resultados para o mapeamento Gray [8].

Nem a TCM nem a BICM, no entanto, são boas candidatas para o nosso trabalho. Desejamos um esquema capaz de alcançar alta eficiência espectral, mas que, além disso, seja capaz de construir reticulados e possa ser decodificado facilmente. A TCM não produz códigos capazes de atingir a capacidade do canal, porque ela demanda o uso de códigos convolucionais [35]. Além disso, a codificação da TCM restringe as taxas de projeto. A BICM, por sua vez, entrelaça os símbolos e usa mapeamento Gray [8]. Estas características da BICM impedem que ela construa reticulados. Para atender todas as nossas exigências decidimos usar codificação multinível.

Este capítulo focará em esquemas de codificação multinível. Introduzimos o modelo do canal analisado. Em seguida, discutimos o conceito de canal equivalente. Continuamos com as definições de codificação e decodificação multi-estágio. Por fim, discorremos sobre o cálculo de probabilidade de erro para a codificação multinível.

2.1.1 Definições Básicas

Um esquema multinível consiste na separação de dados em níveis, na codificação de cada nível por um código específico e na modulação para um dado instante de tempo do vetor formado pelos bits codificados dos diferentes níveis [16].

Seja $m = 2^L$ e $\mathcal{X} = \{0, 1, \ldots, M-1\}$. A representação dos m símbolos de modulação a partir de L bits exige um mapeamento bijetivo $\mathcal{M} : \{0,1\}^L \leftrightarrow \mathcal{X}$. Denotamos as componentes da representação binária de um símbolo de modulação x como $x^{(0)}, x^{(1)}, \ldots, x^{(L-1)}$, i.e.,

$$x = \mathcal{M}(x^{(0)}, \ldots, x^{(L-1)}) \longleftrightarrow (x^{(0)}, \ldots, x^{(L-1)}) = \mathcal{M}^{-1}(x).$$

Como exemplo, um dos mapeamentos $\mathcal{M}$ mais comuns é o Particionamento de Ungerböck. Ele separa a constelação $\mathcal{X}$ iterativamente em partições binárias, criando níveis nos quais as partições constituem subconjuntos da constelação [35]. Os rótulos binários atribuídos aos pontos da constelação visam a aumentar a distância entre estes subconjuntos [35, 37].

Uma representação simples para o particionamento de Ungerböck é a construção de x a partir da multiplicação dos bits $x^{(i)}$ dos diferentes níveis i por potências correspondentes de 2, i.e.,

$$x = \mathcal{M}(x^{(0)}, \ldots, x^{(L-1)}) = \sum_{i=0}^{L-1} 2^i x^{(i)},$$

onde as somas são realizadas sobre os inteiros.

Tomemos como exemplo a constelação 4-PAM e a palavra binária $\left(x^{(0)}, x^{(1)}\right) = (1, 0)$ no esquema multinível de 2 níveis apresentado na figura 2.1. Usando o particionamento de Ungerböck obtemos

$$\mathcal{M}\,(1, 0) = 1 \cdot 2^0 + 0 \cdot 2^1 = 1.$$

No que segue, é útil considerar a seguinte notação:

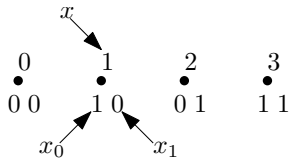
$$\mathcal{X}(x^{(0)}, \ldots, x^{(i)}) \triangleq \Big\{ \mathcal{M}(x^{(0)}, \ldots, x^{(i)}, x^{(i+1)}, \ldots, x^{(L-1)}) : (x^{(i+1)}, \ldots, x^{(L-1)}) \in \{0, 1\}^{L-i-1} \Big\}, \qquad (2.1)$$

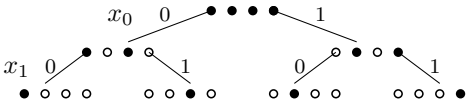
que é o subconjunto da constelação $\mathcal{X}$ quando os bits $x^{(0)}, \ldots, x^{(i)}$ estão fixos.

Referindo-nos novamente à figura 2.1 e com base na constatação de que a expressão $\mathcal{X}(0)$ pode ser expressa como $\mathcal{M}\left(0, x^{(1)}\right)$, onde $x^{(1)} = 0, 1$, notamos que $\mathcal{X}(0) = \{0, 2\}$.

2.1.2 Canais Equivalentes

Considere o modelo de tempo discreto de um canal real sem memória. Para um sinal transmitido $\mathbf{s}$ e um recebido $\mathbf{y}$, ambos de comprimento n, dados por $\mathbf{s} = (s_1, s_2, \ldots, s_n)$ e $\mathbf{y} = (y_1, y_2, \ldots, y_n)$, respectivamente, o canal é especificado por uma função de densidade de probabilidade (pdf) condicional $p(\mathbf{y}|\mathbf{s})$ conforme a seguir:





$$x = \mathcal{M}\left(x^{(0)}, x^{(1)}\right) = x_0 + 2 \cdot x_1 \qquad \mathcal{X}(0) = \mathcal{M}\left(0, x^{(1)}\right) = \{0, 2\}$$

$$0 = 0 + 2 \cdot 0 \qquad 2 = 0 + 2 \cdot 1$$
$$1 = 1 + 2 \cdot 0 \qquad 3 = 1 + 2 \cdot 1 \qquad \mathcal{X}(1) = \mathcal{M}\left(1, x^{(1)}\right) = \{1, 3\}$$

Figura 2.1: Representação do particionamento Ungerböck para uma constelação 4-PAM

$$p(\mathbf{y}|\mathbf{s}) = \prod_{i=1}^{n} p(y_i|s_i).$$

Como $\mathcal{M}$ é uma função bijetora, o mapeamento específico dos bits para o símbolo x não influencia nos resultados posteriores. Adotamos as variáveis aleatórias $X^{(i)}$, X e Y para $x^{(i)}$, x e y, respectivamente, onde $i = 0, 1, \ldots, L-1$. Relembramos a seguir a Regra da Cadeia para a Informação Mútua [6, 37]:

$$\begin{aligned} I(Y;X) =& I\left(Y; \mathcal{M}\left(X^{(0)}, X^{(1)}, \ldots, X^{(L-1)}\right)\right) \\ =& I\left(Y; X^{(0)}, X^{(1)}, \ldots, X^{(L-1)}\right) \\ =& I\left(Y; X^{(0)}\right) + I\left(Y; X^{(1)} \,|\, X^{(0)}\right) + \ldots \\ &+ I\left(Y; X^{(L-1)} \,|\, X^{(0)}, X^{(1)}, \ldots, X^{(L-2)}\right). \end{aligned} \tag{2.2}$$

A equação em (2.2) indica que a transmissão de x no canal físico pode se separar na transmissão paralela de bits $x^{(i)}$ em L canais equivalentes, dado que os bits anteriores $x^{(0)}, x^{(1)}, \ldots, x^{(i-1)}$ são conhecidos.

É possível definir a capacidade equivalente $C^{(i)}$ do canal i com base nos termos da soma na equação (2.2) [37]. Para o cálculo de capacidade devemos maximizar a informação mútua em função das probabilidades a priori dos sinais de entrada. Porém, as probabilidades a priori não podem ser otimizadas independentemente em função da dependência em níveis anteriores [37]. Por isso, a definição de $C^{(i)}$ se baseia nas

probabilidades a priori específicas $\Pr[x]$ dos símbolos x de modulação. Consequentemente, a definição de capacidade equivalente é definida como a informação mútua equivalente [37]

$$C^{(i)} = I\left(Y; X^{(i)} \,|\, X^{(0)}, X^{(1)}, \ldots, X^{(i-1)}\right).$$

A partir da Regra da Cadeia para a Informação Mútua podemos obter a capacidade $C^{(i)}$ do canal equivalente i de forma alternativa [37] como

$$\begin{aligned} C^{(i)} = &I\left(Y; X^{(i)}, \ldots, X^{(L-1)} \,|\, X^{(0)}, \ldots, X^{(i-1)}\right) \\ &- I\left(Y; X^{(i+1)}, \ldots, X^{(L-1)} \,|\, X^{(0)}, \ldots, X^{(i)}\right). \end{aligned} \tag{2.3}$$

A densidade de probabilidade $p\left(y \,|\, x^{(i)}, x^{(0)}, \ldots, x^{(i-1)}\right)$ caracteriza o canal equivalente i. Para um nível $i < L-1$ com i níveis anteriores conhecidos, o bit $x^{(i)}$ tem representação múltipla na constelação $\mathcal{X}$. Consequentemente, a pdf $p\left(y \,|\, x^{(i)}, x^{(0)}, \ldots, x^{(i-1)}\right)$ é dada pelo valor esperado de $p(y \,|\, x)$ sobre os sinais $x \in \mathcal{X}(x^{(0)}, \ldots, x^{(i)})$ [37]:

$$\begin{aligned} p\left(y \,|\, x^{(i)}, x^{(0)}, \ldots, x^{(i-1)}\right) &= \mathrm{E}_{x \in \mathcal{X}(x^{(0)}, \ldots, x^{(i)})}\left[p(y \,|\, x)\right] \\ &= \frac{\sum\limits_{x \in \mathcal{X}(x^{(0)}, \ldots, x^{(i)})} \Pr[x] \cdot p(y \,|\, x)}{\Pr\left[X \in \mathcal{X}(x^{(0)}, \ldots, x^{(i)})\right]}. \end{aligned} \tag{2.4}$$

De (2.3), da definição da informação mútua condicionada e da expressão em (2.4) podemos calcular a capacidade $C^{(i)}$ do canal equivalente i através da fórmula [37]

$$\begin{aligned} C^{(i)} = &\mathrm{E}_{x^{(0)}, \ldots, x^{(i-1)}}\left[C\left(\mathcal{X}\left(x^{(0)}, \ldots, x^{(i-1)}\right)\right)\right] \\ &- \mathrm{E}_{x^{(0)}, \ldots, x^{(i)}}\left[C\left(\mathcal{X}\left(x^{(0)}, \ldots, x^{(i)}\right)\right)\right], \end{aligned} \tag{2.5}$$

onde $C\left(\mathcal{X}\left(x^{(0)}, \ldots, x^{(i)}\right)\right)$ denota a capacidade calculada com o subconjunto $\mathcal{X}\left(x^{(0)}, \ldots, x^{(i)}\right)$, cujos elementos possuem probabilidades a priori $\Pr[x] / \Pr\left[\mathcal{X}\left(x^{(0)}, \ldots, x^{(i)}\right)\right]$. Uma definição análoga vale para $C\left(\mathcal{X}\left(x^{(0)}, \ldots, x^{(i-1)}\right)\right)$, na qual apenas substituímos todas as instâncias de $x^{(i)}$ por $x^{(i-1)}$.

Já que a Regra da Cadeia da Informação Mútua permite a separação de um esquema em níveis, a derivação da capacidade de um esquema multinível segue naturalmente do resultado apresentado a seguir [37]:

Teorema 2.1 (Teorema da Capacidade de Esquemas Multinível)**.** *A capacidade $C = C(\mathcal{X})$ de um esquema de modulação composto por sinais $x \in \mathcal{X}$, cuja probabilidade a priori é $\Pr[x]$, é igual à soma das capacidades $C^{(i)}$ dos canais equivalentes i de um esquema multinível:*

$$C = \sum_{i=0}^{L-1} C^{(i)}.$$

A prova do teorema é apresentada em [15].

2.1.3 Codificação Multi-Estágio

Assumimos um vetor de dados $\mathbf{u} = (u_1, u_2, \ldots, u_k)$ com k símbolos. Estes símbolos podem ser divididos em L níveis, formando vetores $\mathbf{u}^{(i)}$ de comprimento $k^{(i)}$, $i = 0, 1, \ldots, L-1$, tal que $k = \sum_{i=0}^{L-1} k^{(i)}$.

Pela Regra da Cadeia da Informação Mútua um canal equivalente i pode ser descrito pela pdf $p\left(\mathbf{y} \,|\, \mathbf{x}^{(i)}, \mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(i-1)}\right)$ [37]. A descrição indica que podemos aplicar a cada nível i um codificador $\mathcal{E}^{(i)}$ baseado em um código $\mathbf{C}^{(i)}\left(n, k^{(i)}\right)$ e nos dados dos níveis anteriores: $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(i-1)}$. Assim, temos

$$\mathbf{x}^{(i)} = \mathcal{E}^{(i)}\left(\mathbf{u}^{(i)} \,|\, \mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(i-1)}\right), \tag{2.6}$$

onde $\mathbf{x}^{(i)}$ é o símbolo codificado de $\mathbf{u}^{(i)}$ com a informação lateral dos dados dos níveis i', tal que $0 \leq i' < i$. Exibimos na figura 2.2 a ilustração de um codificador multi-estágio para 3 níveis, na qual visualizamos as dependências.

É interessante notar que um caso particular de (2.6) é a codificação independente, i.e., sem a informação de níveis anteriores. A equação (2.6) se torna

$$\mathbf{x}^{(i)} = \mathcal{E}^{(i)}\left(\mathbf{u}^{(i)}\right).$$

A codificação independente é a forma mais comum de codificação em esquemas multinível [16, 37], mas ela não permite a construção de

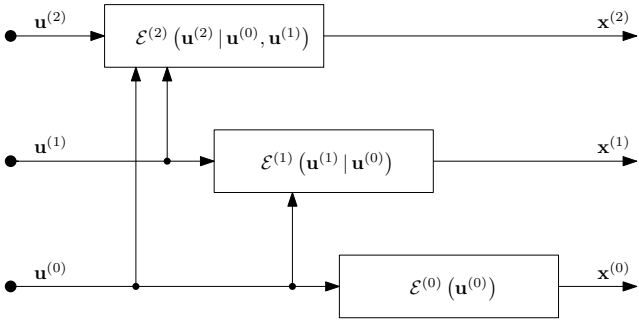


Figura 2.2: Codificador multi-estágio para 3 níveis

reticulados [18] e, por isso, não a utilizamos neste trabalho.

2.1.4 Decodificação Multi-Estágio

Vimos que o conceito de canais equivalentes valida a técnica de codificação multi-estágio. A codificação multi-estágio, por sua vez, implica a decodificação multi-estágio. Se um nível pode ser codificado com base na informação de níveis anteriores, sabe-se que existe dependência entre eles, i.e., a informação mútua é não nula [6]. E como um esquema multinível pressupõe a independência de um nível com relação aos níveis posteriores (vide a expressão da Regra da Cadeia da Informação Mútua em (2.2)), podemos supor uma operação de decodificação que só depende do nível atual e de anteriores. Trata-se da decodificação multi-estágio.

Para decodificar $\mathbf{u}^{(i)}$, para $i = 0, 1, \ldots, L-1$, o decodificador precisa dos dados de níveis passados, i.e., de $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(i-1)}$. Para o sinal de saída $\mathbf{y} \in \mathbb{R}^n$, a operação $\mathcal{D}^{(i)}$ do decodificador multi-estágio no nível i é dada por

$$\hat{\mathbf{u}}^{(i)} = \mathcal{D}^{(i)}\left(\mathbf{y} \mid \hat{\mathbf{u}}^{(0)}, \hat{\mathbf{u}}^{(1)}, \ldots, \hat{\mathbf{u}}^{(i-1)}\right). \tag{2.7}$$

Apresentamos na figura 2.3 um diagrama ilustrativo da decodificação multi-estágio aplicada a 3 níveis, que indica a existência de influência vinda de níveis anteriores.

Ao considerar codificadores multi-estágio dependentes, o decodificador multi-estágio demanda a codificação das estimativas obtidas por

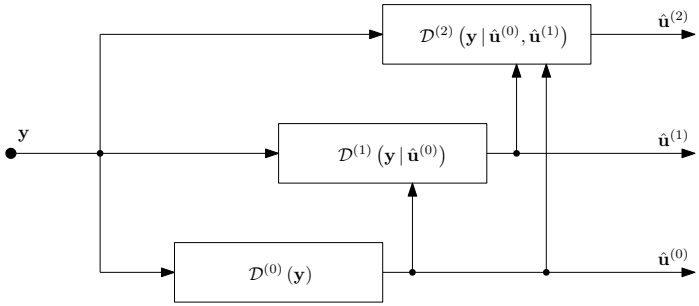


Figura 2.3: Decodificador multi-estágio para 3 níveis

decodificadores de níveis anteriores para eliminar a influência destes níveis no atual.

Sabemos de [37] que o decodificador multi-estágio é capaz de alcançar a capacidade do canal físico em um esquema multinível se as taxas de projeto $R^{(i)}$ dos códigos componentes forem escolhidas de acordo com as capacidades equivalentes $C^{(i)}$. Ou seja, o decodificador multi-estágio pode alcançar a capacidade quando

$$R^{(i)} = C^{(i)}. \tag{2.8}$$

Este resultado tem relação direta com a Regra da Cadeia para a Informação Mútua e com o Teorema 2.1.

2.1.5 Probabilidade de Erro

Seja $\mathfrak{e}_i = \{\hat{\mathbf{u}}^{(i)} \neq \mathbf{u}^{(i)}\}$ o evento de erro no nível i e seja

$$P_e^{(i)} = \Pr[\mathfrak{e}_i \mid \mathfrak{e}_0^c \cap \cdots \cap \mathfrak{e}_{i-1}^c]$$

a probabilidade de erro no nível i dadas decisões corretas nos níveis anteriores, isto é, a probabilidade de erro referente exclusivamente ao canal equivalente i. A probabilidade de erro total do esquema multiní-

vel pode ser expressa como

$$\begin{aligned} P_e &= \Pr[\mathfrak{e}_0 \cup \cdots \cup \mathfrak{e}_{L-1}] \\ &= \sum_{i=0}^{L-1} \Pr[\mathfrak{e}_i \mid \mathfrak{e}_0^c \cap \cdots \cap \mathfrak{e}_{i-1}^c] \cdot \Pr[\mathfrak{e}_0^c \cap \cdots \cap \mathfrak{e}_{i-1}^c]. \end{aligned}$$

Como $\Pr[\mathfrak{e}_0^c \cap \cdots \cap \mathfrak{e}_{i-1}^c] \leq 1$, podemos obter o seguinte limitante superior para P_e:

$$P_e \leq P_e^{(0)} + P_e^{(1)} + \cdots + P_e^{(L-1)}.$$

Portanto, uma maneira conveniente de otimizar esquemas de codificação multinível é por meio da minimização da soma das probabilidades de erro de cada código.

2.2 Reticulados

Nesta seção apresentamos os conceitos de reticulado mais importantes para a explicação das contribuições realizadas. Incluímos as definições de reticulado, subreticulado, grupo quociente, região fundamental, quantização, e, principalmente, de código de reticulado aninhado.

Um reticulado é um subgrupo discreto de $\mathbb{R}^n$ com a operação aditiva. Ele pode ser completamente definido por n vetores de base [5]. Por exemplo, o reticulado n-dimensional $\mathbb{Z}^n$ é completamente descrito pelos vetores linha de uma matriz identidade $\mathbf{I}_n$ [25]. Portanto, uma representação válida para um reticulado é sua matriz geradora $\mathbf{G}_\Lambda$, na qual os vetores linha $\mathbf{g}_i \in \mathbb{R}^n$, $i = 1, 2, \ldots, n$, da matriz definem uma base para Λ. Um reticulado admite a definição de um grupo contendo todas as combinações lineares inteiras dos vetores de base $\mathbf{g}_i$ [9], i.e.,

$$\Lambda = \{a_1\mathbf{g}_1 + a_2\mathbf{g}_2 + \ldots + a_n\mathbf{g}_n \ : \ a_i \in \mathbb{Z}, i \in \{1, 2, \ldots, n\}\}.$$

Assumindo um vetor linha $\mathbf{a} = (a_1, a_2, \ldots, a_n) \in \mathbb{Z}^n$ adotamos para $\boldsymbol{\lambda}$ a definição $\boldsymbol{\lambda} = \mathbf{a}\mathbf{G}_\Lambda$, onde $\boldsymbol{\lambda} \in \Lambda$.

Um subreticulado Λ' de Λ é um subconjunto de Λ que também constitui um reticulado. Dois reticulados Λ' e Λ são aninhados se Λ' for um subreticulado de Λ, i.e., $\Lambda' \subseteq \Lambda$. Podemos generalizar esta definição para L reticulados, onde $L \in \mathbb{N}^*$: $\Lambda_0 \subseteq \Lambda_1 \subseteq \cdots \subseteq \Lambda_{L-1}$ são

reticulados aninhados [22].

Assumindo os reticulados Λ e Λ', para cada $\boldsymbol{\lambda} \in \Lambda$, um deslocamento do tipo $\boldsymbol{\lambda} + \Lambda'$ é chamado de coset de Λ' em Λ. Duas cosets $\boldsymbol{\lambda}_1 + \Lambda'$ e $\boldsymbol{\lambda}_2 + \Lambda'$, tal que $\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2 \in \Lambda$, podem ser ou idênticas ou disjuntas: se $\boldsymbol{\lambda}_1 - \boldsymbol{\lambda}_2 \in \Lambda'$, verifica-se a identidade, mas, se $\boldsymbol{\lambda}_1 - \boldsymbol{\lambda}_2 \notin \Lambda'$, observa-se a disjunção [9]. Cosets normalmente são representadas por um membro do grupo, que recebe o nome de líder. Para uma coset $\boldsymbol{\lambda} + \Lambda'$, por exemplo, o líder é $\boldsymbol{\lambda}$ [7].

O quociente Λ/Λ' é um grupo formado pelo conjunto das cosets de Λ' em Λ [9, 7], ou seja,

$$\Lambda/\Lambda' = \{\boldsymbol{\lambda} + \Lambda', \boldsymbol{\lambda} \in \Lambda\}.$$

A região fundamental $\mathcal{R}_\Lambda$ de um reticulado é o conjunto $\mathcal{R}_\Lambda \subseteq \mathbb{R}^n$ com $\mathbf{0} \in \mathcal{R}_\Lambda$, cujas translações $\mathbf{x} + \mathcal{R}_\Lambda = \{\mathbf{x} + \mathbf{y} : \mathbf{y} \in R_\Lambda\}$ tomadas sobre todos os $\mathbf{x} \in \Lambda$ formam uma partição de $\mathbb{R}^n$. Ou seja, $\mathcal{R}_\Lambda$ é uma estrutura em $\mathbb{R}^n$ que, se repetida muitas vezes, cobre todo o espaço com apenas um ponto do reticulado em cada cópia.

Podemos também definir o quantizador de um reticulado como um mapeamento $\mathcal{Q}_\Lambda : \mathbb{R}^n \to \Lambda$ tal que $\mathcal{Q}_\Lambda(\mathbf{0}) = \mathbf{0}$ e $\mathcal{Q}_\Lambda(\boldsymbol{\lambda}+\mathbf{x}) = \boldsymbol{\lambda}+\mathcal{Q}_\Lambda(\mathbf{x})$, para todo $\boldsymbol{\lambda} \in \Lambda$. Ao quantizador $\mathcal{Q}_\Lambda$ podemos associar a operação módulo Λ. Para um vetor $\mathbf{y} \in \mathbb{R}^n$ esta operação é dada por [22, 9]

$$\mathbf{y} \bmod \Lambda = \mathbf{y} - Q_\Lambda(\mathbf{y}).$$

Dados os reticulados Λ e $\Lambda' \subseteq \Lambda$, seja um código de reticulado aninhado $\mathbf{C}(\Lambda, \Lambda')$ um mapeamento de dados para os $|\Lambda/\Lambda'|$ pontos de Λ delimitados pela região fundamental de Λ', o que se pode expressar através da seguinte fórmula [9]:

$$\mathbf{C}(\Lambda, \Lambda') = \Lambda \cap \mathcal{R}_{\Lambda'}.$$

Alternativamente, $\mathbf{C}(\Lambda, \Lambda')$ permite a interpretação algébrica

$$\mathbf{C}(\Lambda, \Lambda') = \Lambda \bmod \Lambda' = \{\boldsymbol{\lambda} \bmod \Lambda' \ : \ \boldsymbol{\lambda} \in \Lambda\}.$$

Isto significa que o código $\mathbf{C}(\Lambda, \Lambda')$ pode ser definido como o conjunto das cosets do grupo quociente Λ/Λ' [22].

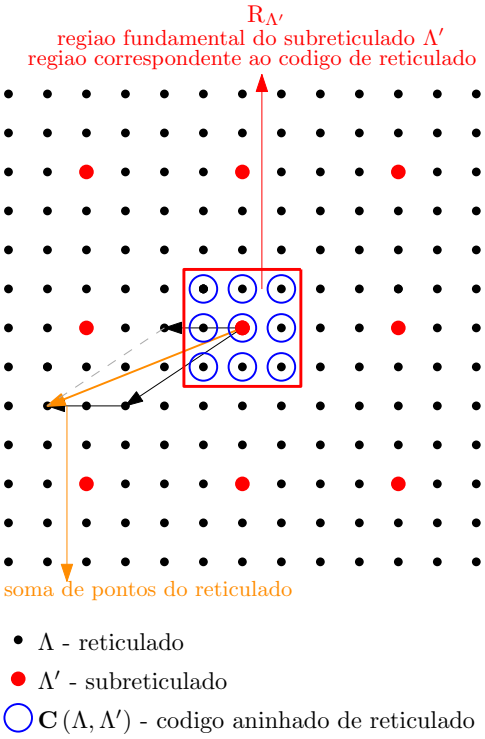


Figura 2.4: Ilustração dos principais conceitos de reticulado

2.3 Construções de Reticulado Baseadas em Códigos Lineares

Reticulados podem ser obtidos de códigos lineares através de certas construções. Códigos de reticulado podem ser definidos a partir destas construções através do uso de reticulados aninhados [9, 31].

Nesta seção introduzimos as definições gerais das construções A, D e D′. A Construção A contém o mecanismo básico de construção de um código de reticulado aplicado a apenas um nível e um código linear [9, 5]. As construções D e D′ são construções multinível e aplicam o mecanismo a vários níveis e códigos [10]. Como veremos em maiores detalhes adiante, a Construção D possui uma definição baseada em uma matriz geradora de reticulados. A Construção D′, por sua vez, possui uma definição baseada em congruências definidas pelas equações de

paridade de um código.

No que segue, restringimos todas as definições e análises apenas ao caso de códigos binários.

Seja $\varphi : \mathbb{Z} \to \mathbb{F}_2$ o mapeamento que representa a projeção natural $\varphi(x) = x + 2\mathbb{Z}$ e $\tilde{\varphi} : \mathbb{F}_2 \to \mathbb{Z}/2\mathbb{Z} \cong \{0, 1\} \subseteq \mathbb{Z}$ o mapeamento de imersão $\tilde{\varphi}(a + 2\mathbb{Z}) = a$, onde $a \in \{0, 1\}$. Ambos os mapeamentos podem ser estendidos para vetores e matrizes aplicando-os componente-a-componente.

2.3.1 Construção A

A Construção A gera reticulados com base em um único código linear binário $\mathbf{C} \subseteq \mathbb{F}_2^n$. O reticulado é dado por [5, 31]

$$\Lambda \triangleq \{\boldsymbol{\lambda} \in \mathbb{Z}^n : \varphi(\boldsymbol{\lambda}) \in \mathbf{C}\}. \tag{2.9}$$

O código formado pelas palavras-código $\mathbf{c} \in \mathbf{C}$ mapeadas componente a componente de $\mathbb{F}_2$ para $\{0, 1\}$ é o código de reticulado $\mathbf{C}_\Lambda$, ou seja, $\mathbf{C}_\Lambda = \tilde{\varphi}(\mathbf{C})$ [31]. A equação (2.9) é equivalente a

$$\Lambda \triangleq \mathbf{C}_\Lambda + 2\mathbb{Z}^n.$$

Note que, fazendo $\Lambda' = 2\mathbb{Z}^n$ e considerando a região $[0, 2)^n$ como a região fundamental de Λ', temos que $\mathbf{C}_\Lambda$ é um código de reticulado $\mathbf{C}_\Lambda = \Lambda \cap \mathcal{R}_{\Lambda'}$, isomorfo ao quociente Λ/Λ' [9].

Tomemos como exemplo o código de bloco linear

$$\mathbf{C} = \{000, 011, 101, 110\} \subseteq \mathbb{F}_2^3.$$

Em vista da equação (2.9), observamos que o reticulado gerado a partir de $\mathbf{C}$ pode ser representado como

$$\Lambda = \{000, 011, 101, 110, 020, 431, 563, 758, \ldots\} \subseteq \mathbb{Z}^3.$$

O subreticulado é dado por $\Lambda' = 2\mathbb{Z}^3$, e o código aninhado de reticulado consiste dos líderes das cosets do grupo quociente Λ/Λ', tal que

$$\mathbf{C}_\Lambda = \{000, 011, 101, 110\} \subseteq \mathbb{Z}^3.$$

Decodificação no quociente de reticulados

No artigo [31], demonstra-se que a decodificação de mínima distância em quocientes de reticulado criados com a Construção A pode ser realizada por meio de uma busca em $\mathbf{C}$ e de uma métrica modificada. Seja $\mathbf{y} = (y_1, y_2, \ldots, y_n) \in \mathbb{R}^n$ o vetor recebido, $\mathbf{c} = (c_1, c_2, \ldots, c_n) \in \mathbb{F}_2^n$ uma palavra-código de $\mathbf{C}$ e $\hat{\mathbf{c}} \in \mathbf{C}$ a estimativa do decodificador.

De acordo com [31], a busca em $\mathbf{C}$ é dada por

$$\hat{\mathbf{c}}\,(\mathbf{y}) = \underset{\mathbf{c}\in\mathbf{C}}{\operatorname{argmin}} \sum_{j=1}^{n} |\,(y_j - \tilde{\varphi}\,(c_j)) \bmod 2\mathbb{Z}^n|^2,$$

a qual pode ser realizada pelo Algoritmo de Viterbi [31].

2.3.2 Construção D

A Construção D é uma construção multinível de reticulados composta por códigos lineares aninhados [5, 18].

Seja $\mathbf{C}^{(0)} \subseteq \mathbf{C}^{(1)} \subseteq \cdots \subseteq \mathbf{C}^{(L-1)} \subseteq \mathbb{F}_2^n$ uma família de códigos lineares binários aninhados, onde $\mathbf{C}^{(i)}$ possui parâmetros (n, k_i), para $i = 0, 1, \ldots, L-1$.

Sejam $\mathbf{g}_1, \ldots, \mathbf{g}_{k_{L-1}} \in \mathbb{Z}^n$ e

$$\mathbf{G}_i = \begin{bmatrix} \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k_i} \end{bmatrix}$$

tais que $\varphi\,(\mathbf{G}_i) \in \mathbb{F}_2^{k_i \times n}$ é uma matriz geradora de $\mathbf{C}^{(i)}$, $i = 0, \ldots, L-1$.

A aplicação da Construção D a essa família de códigos resulta no reticulado [5, 18]

$$\Lambda \triangleq \mathbf{C}_\Lambda + 2^L\mathbb{Z}^n, \tag{2.10}$$

onde

$$\mathbf{C}_\Lambda \triangleq \left\{ \sum_{i=0}^{L-1} \sum_{j=1}^{k_i} 2^i u_j^{(i)} \mathbf{g}_j \;:\; u_j^{(i)} \in \{0,1\} \right\} \bmod 2^L\mathbb{Z}^n. \tag{2.11}$$

Note que, fazendo $\Lambda' = 2^L\mathbb{Z}^n$ e considerando a região $[0, 2^L)^n$ como

a região fundamental de Λ', temos que $\mathbf{C}_\Lambda$ é um código de reticulado $\mathbf{C}_\Lambda = \Lambda \cap \mathcal{R}_{\Lambda'}$, isomorfo ao quociente Λ/Λ' [9].

Equivalentemente, a equação (2.11) pode ser expressa como

$$\mathbf{C}_\Lambda = \left\{ \sum_{i=0}^{L-1} 2^i \mathbf{u}^{(i)} \mathbf{G}_i \ : \ \mathbf{u}^{(i)} \in \{0,1\}^{k_i} \right\} \bmod 2^L \mathbb{Z}^n. \tag{2.12}$$

Devido à definição dos geradores $\mathbf{g}_j$ como vetores em $\mathbb{Z}$, a definição do código de reticulado construído através da Construção D apresentada na expressão (2.11), ou, equivalentemente, em (2.12) é, na verdade, uma generalização da definição normalmente encontrada na literatura para a Construção D (vide [5] e [18]). O intuito é obter uma expressão genérica capaz também de acomodar a definição do código de reticulado obtido através da Construção D$'$. Na Construção D tradicional, os geradores $\mathbf{g}_j \in \{0,1\}^n$, mas na Construção D$'$, como veremos adiante, $\mathbf{g}_j \in \{0, \ldots, 2^L - 1\}^n$.

Tomemos como exemplo um esquema multinível com dois níveis constituído pelos códigos lineares de bloco aninhados $\mathbf{C}_0 \subseteq \mathbf{C}_1$ dados a seguir:

$$\begin{aligned} \mathbf{C}_0 &= \{(0,1,1)\} \subseteq \mathbb{F}_2^3 \\ \mathbf{C}_1 &= \{(0,1,1),(1,0,1)\} \subseteq \mathbb{F}_2^3. \end{aligned}$$

Sejam $\{\mathbf{g}_1, \mathbf{g}_2\}$ geradores de $\mathbf{C}_1$, tais que

$$\left.\begin{array}{l} \mathbf{g}_1 = (0,1,1)\} \text{ gera } \mathbf{C}_0 \\ \mathbf{g}_2 = (1,0,1) \end{array}\right\} \text{ gera } \mathbf{C}_1.$$

Então, com base na equação (2.11), temos que

$$\Lambda = \left\{ \begin{array}{l} \left(u_1^{(0)}(0,1,1)\right) + \\ +2 \cdot \left(u_1^{(1)}(0,1,1) + u_2^{(1)}(1,0,1)\right) + \\ +4\mathbb{Z}^3 \end{array} \right| \quad u_j^{(i)} \in \{0,1\} \Bigg\}.$$

Listamos no que segue alguns pontos pertencentes ao reticulado Λ:

$$\{(0,3,7),(2,5,7),(2,6,4),(6,1,3),(6,7,9)\} \subset \Lambda \subset \mathbb{Z}^3.$$

De acordo com a equação (2.10), o subreticulado é dado por $\Lambda' = 4\mathbb{Z}^3$.

O código aninhado de reticulado $\mathbf{C}_\Lambda$ consiste dos líderes das cosets do grupo quociente Λ/Λ', de modo que

$$\mathbf{C}_\Lambda = \left\{ \left(u_1^{(0)}(0,1,1) \right) + 2 \cdot \left(u_1^{(1)}(0,1,1) + u_2^{(1)}(1,0,1) \right) \;\middle|\; u_j^{(i)} \in \{0,1\} \right\}.$$

Utilizando-nos da operação $\bmod 4\mathbb{Z}^3$, listamos a seguir os pontos do código de reticulado aninhado $\mathbf{C}_\Lambda$ correspondentes aos pontos listados anteriormente para Λ:

$$\{(0,3,3),(2,1,3),(2,2,0),(2,1,3),(2,3,1)\} \subset \mathbf{C}_\Lambda \subset \mathbb{Z}^3.$$

2.3.3 Construção D'

Seja $\mathbf{C}^{(0)} \subseteq \mathbf{C}^{(1)} \subseteq \cdots \subseteq \mathbf{C}^{(L-1)} \subseteq \mathbb{F}_2^n$ uma família de códigos lineares binários aninhados, onde $\mathbf{C}^{(i)}$ possui parâmetros (n, k_i), para $i = 0, 1, \ldots, L-1$. Para $i = 1, \ldots, L-1$, definimos $m_i = n - k_i$ como o número de equações de paridade para o código $\mathbf{C}^{(i)}$. Por conveniência, definimos também $m_L = 0$.

Sejam $\mathbf{h}_1, \ldots, \mathbf{h}_{m_L - 1} \in \mathbb{Z}^n$ e

$$\mathbf{H}_i = \begin{bmatrix} \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_{m_i} \end{bmatrix}$$

tais que $\varphi(\mathbf{H}_i) \in \mathbb{F}_2^{m_i \times n}$ é uma matriz de verificação de paridade para $\mathbf{C}^{(i)}$, $i = 0, \ldots, L-1$.

A aplicação da Construção D′ a essa família de códigos resulta no reticulado Λ constituído de todos os vetores $\mathbf{x} \in \mathbb{Z}^n$ que satisfazem as seguintes congruências [5, 18]:

$$\mathbf{x}\,\mathbf{h}_j^{\mathrm{T}} \equiv 0 \pmod{2^{i+1}\mathbb{Z}^n}, \tag{2.13}$$

onde $m_{i+1} < j \leq m_i$, $i = 0, \ldots, L-1$.

Percebemos que, se multiplicarmos as equações modulares em (2.13) por potências apropriadas de 2, então $\mathbf{x} \in \Lambda$ se e somente se [30]

$$2^{L-1-i}\mathbf{x}\,\mathbf{h}_j^{\mathrm{T}} \equiv 0 \pmod{2^L\mathbb{Z}^n}, \tag{2.14}$$

para $m_{i+1} < j \leq m_i$, $i = 0, \ldots, L-1$.

Definindo

$$\mathbf{H}_\Lambda = \begin{bmatrix} \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_{m_{L-1}} \\ 2\mathbf{h}_{m_{L-1}+1} \\ \vdots \\ 2\mathbf{h}_{m_{L-2}} \\ \vdots \\ 2^{L-2}\mathbf{h}_{m_1} \\ 2^{L-1}\mathbf{h}_{m_1+1} \\ \vdots \\ 2^{L-1}\mathbf{h}_{m_0} \end{bmatrix},$$

denominada matriz de paridade do reticulado, podemos reescrever a relação (2.14) como

$$\mathbf{x}\,\mathbf{H}_\Lambda^{\mathrm{T}} \equiv 0 \pmod{2^L\mathbb{Z}^n}.$$

A relação anterior resulta no código de reticulado

$$\mathbf{C}_\Lambda = \left\{\mathbf{x} \in \{0,\ldots,2^L-1\}^n : \mathbf{x}\,\mathbf{H}_\Lambda^{\mathrm{T}} \equiv 0 \pmod{2^L\mathbb{Z}^n}\right\}.$$

Desta forma, podemos expressar o reticulado como

$$\Lambda = \mathbf{C}_\Lambda + 2^L\mathbb{Z}^n. \tag{2.15}$$

Note que, fazendo $\Lambda' = 2^L\mathbb{Z}^n$ e considerando a região $[0, 2^L)^n$ como a região fundamental de Λ', temos que $\mathbf{C}_\Lambda$ é um código de reticulado $\mathbf{C}_\Lambda = \Lambda \cap R_{\Lambda'}$, isomorfo ao quociente Λ/Λ' [9].

Reaproveitamos os códigos aninhados $\mathbf{C}_0 \subseteq \mathbf{C}_1$ utilizados na exemplificação da Construção D para ilustrar a Construção D′. Sejam $\{\mathbf{h}_1, \mathbf{h}_2\} \in \mathbb{F}_2^3$ vetores de paridade de $\mathbf{C}_0$, tais que

$$\left.\begin{array}{l} \mathbf{h}_1 = (1,1,1)\} \text{ checa } \mathbf{C}_1 \\ \mathbf{h}_2 = (0,1,1) \end{array}\right\} \text{ checa } \mathbf{C}_0.$$

Então, com base na equação (2.13) temos que

$$\Lambda = \left\{ \mathbf{x} \in \mathbb{Z}^3 \;\middle|\; \begin{array}{ll} \mathbf{x}\,(0,1,1)^{\mathrm{T}} \equiv 0 & (\mathrm{mod}\ 2\mathbb{Z}^3)\ \mathrm{e} \\ \mathbf{x}\,(1,1,1)^{\mathrm{T}} \equiv 0 & (\mathrm{mod}\ 4\mathbb{Z}^3) \end{array} \right\}.$$

A seguir listamos alguns pontos pertencentes ao reticulado Λ:

$$\{(0,2,2),(2,1,1),(4,3,1),(6,1,5),(8,4,4)\} \subset \Lambda \subset \mathbb{Z}^3.$$

De acordo com a equação (2.15), o subreticulado é dado por $\Lambda' = 4\mathbb{Z}^3$. O código aninhado de reticulado $\mathbf{C}_\Lambda$ consiste dos líderes das cosets do grupo quociente Λ/Λ', de modo que

$$\mathbf{C}_\Lambda = \left\{ \mathbf{x} \bmod 4\mathbb{Z}^3 \;\middle|\; \mathbf{x} \in \Lambda \right\}.$$

Mostramos, enfim, os pontos pertencentes ao código de reticulado aninhado $\mathbf{C}_\Lambda$ correspondentes aos pontos de Λ listados previamente:

$$\{(0,2,2),(2,1,1),(0,3,1),(2,1,1),(0,0,0)\} \subseteq \mathbf{C}_\Lambda \subset \mathbb{Z}^3.$$

CAPÍTULO 3

Códigos de Reticulado Multinível

Apresentamos neste capítulo contribuições originais, feitas com relação à decodificação e codificação de códigos de reticulado multinível. Na seção 3.1 expomos a decodificação multi-estágio para a Construção D, mostrando de forma detalhada por que a decodificação multi-estágio pode ser empregada para esta construção e apresentando um decodificador multi-estágio muito simples, apesar de subótimo. Na seção 3.2 discorremos sobre o projeto de códigos de reticulado usando a Construção D e códigos convolucionais. Focamos, em especial, no projeto de taxas. Por fim, na seção 3.3 manipulamos as definições da Construção D′ para criar geradores de reticulado semelhantes ao da Construção D.

3.1 Decodificação Multi-Estágio para a Construção D

Seja $\mathbf{x} = \mathcal{E}\left(\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(L-1)}\right) \in \mathbb{Z}^n$ o sinal transmitido, codificado a partir dos dados $\mathbf{u}^{(0)}, \ldots, \mathbf{u}^{(L-1)} \in \mathbb{F}_2^n$ segundo o código de reticulado criado com a Construção D, i.e,

$$\mathbf{x} = \sum_{i=0}^{L-1} 2^i \mathbf{u}^{(i)} \mathbf{G}_i \pmod{2^L \mathbb{Z}^n}.$$

Seja $\mathbf{y}$ o sinal recebido pelo decodificador. Podemos então escrever $\mathbf{y}$ como

$$\mathbf{y} = \underbrace{\left(\sum_{i=0}^{L-1} 2^i \mathbf{u}^{(i)} \mathbf{G}_i\right) \bmod 2^L \mathbb{Z}^n}_{\mathbf{x}} + \mathbf{z},$$

onde $\mathbf{z} \in \mathbb{R}^n$ é o ruído do canal.

Buscamos a partir de $\mathbf{y}$ o vetor de dados $\mathbf{u}^{(i)}$ do nível i, ou, equivalentemente, o sinal codificado $\mathbf{x}^{(i)}$. Para tal, exploramos a natureza multinível da Construção D: a aplicação da operação mod $2^{i+1}\mathbb{Z}^n$ a $\mathbf{x}$ elimina a influência de todos os níveis $i' > i$, i.e.,

$$\mathbf{y} = \sum_{i=0}^{L-1} 2^i \mathbf{u}^{(i)} \mathbf{G}_i \bmod 2^L + \mathbf{z} \pmod{2^{i+1}\mathbb{Z}^n}$$

$$\mathbf{y} \bmod 2^{i+1}\mathbb{Z}^n = \left(\sum_{i'=0}^{i} 2^{i'} \mathbf{u}^{(i')} \mathbf{G}^{(i')} + \mathbf{z}\right) \bmod 2^{i+1}\mathbb{Z}^n. \tag{3.1}$$

O descarte de níveis superiores é possível porque a Construção D é um esquema de particionamento Ungerböck que admite este tipo de redução [10].

Seja um decodificador $\mathcal{D}^{(i)}$, específico ao código $\mathbf{C}^{(i)}$ aplicado ao nível i. Através deste decodificador e da equação em (3.1) podemos decodificar $\mathbf{u}^{(i)}$ a partir de $\mathbf{y}$ mod $2^{i+1}\mathbb{Z}^n$:

$$\left(\mathbf{y} - \sum_{i'=0}^{i-1} 2^{i'} \mathbf{u}^{(i')} \mathbf{G}_{i'}\right) \bmod 2^{i+1}\mathbb{Z}^n = \left(2^i \mathbf{u}^{(i)} \mathbf{G}_i + \mathbf{z}\right) \bmod 2^{i+1}\mathbb{Z}^n$$

$$\frac{\mathbf{y} - \sum_{i'=0}^{i-1} 2^{i'} \mathbf{u}^{(i')} \mathbf{G}_{i'}}{2^i} \bmod 2^{i+1}\mathbb{Z}^n = \left(\mathbf{u}^{(i)} \mathbf{G}_i + \frac{\mathbf{z}}{2^i}\right) \bmod 2^{i+1}\mathbb{Z}^n$$

$$\frac{\mathbf{y} - \sum_{i'=0}^{i-1} 2^{i'} \mathbf{u}^{(i')} \mathbf{G}_{i'}}{2^i} \bmod 2\mathbb{Z}^n = \left(\mathbf{u}^{(i)} \mathbf{G}_i + \frac{\mathbf{z}}{2^i}\right) \bmod 2\mathbb{Z}^n. \tag{3.2}$$

Sabemos que $\mathbf{u}^{(i)} \mathbf{G}_i \bmod 2\mathbb{Z}^n$ é uma palavra código $\mathbf{c}^{(i)} \in \mathbf{C}^{(i)}$.

Com a definição de $\mathbf{y}^{(i)}$ igual a

$$\mathbf{y}^{(i)} = \frac{\mathbf{y} - \sum_{i'=0}^{i-1} 2^{i'} \mathbf{u}^{(i')} \mathbf{G}_{i'}}{2^i}$$

podemos reescrever (3.2) como

$$\mathbf{y}^{(i)} \mod 2\mathbb{Z}^n = \left(\mathbf{c}^{(i)} + \frac{\mathbf{z}}{2^i}\right) \mod 2\mathbb{Z}^n.$$

O propósito do decodificador $\mathcal{D}^{(i)}$ é extrair $\mathbf{c}^{(i)}$ a partir da expressão $\left(\mathbf{c}^{(i)} + \mathbf{z}/2^i\right) \mod 2\mathbb{Z}^n$, ou seja, eliminar o ruído. Equivalentemente, podemos afirmar que $\mathcal{D}^{(i)}$ estima a informação $\mathbf{u}^{(i)}$ associada a $\mathbf{c}^{(i)}$. Obtemos por meio de $\mathcal{D}^{(i)}$ uma estimativa $\hat{\mathbf{u}}^{(i)}$ segundo

$$\hat{\mathbf{u}}^{(i)} = \mathcal{D}^{(i)} \left(\mathbf{y}^{(i)} \mod 2\mathbb{Z}^n\right).$$

Na prática, não utilizamos os dados $\mathbf{u}^{(i')}$ na decodificação, mas sim os sinais $\hat{\mathbf{u}}^{(i')}$ estimados com os decodificadores $D^{(i')}$ de níveis anteriores, i.e.,

$$\hat{\mathbf{u}}^{(i)} = \mathcal{D}^{(i)} \left(\frac{\mathbf{y} - \sum_{i'=0}^{i-1} 2^{i'} \hat{\mathbf{u}}^{(i')} \mathbf{G}_{i'}}{2^i} \mod 2\mathbb{Z}^n \right). \tag{3.3}$$

Notamos em (3.3) que as estimativas de níveis anteriores a i são recodificadas através da construção D (somatória em (3.3)) a fim de cancelar o efeito que exercem no nível i.

O decodificador $\mathcal{D}^{(i)}$ utiliza $\mathbf{y}$ e $\hat{\mathbf{u}}^{(0)}, \ldots, \hat{\mathbf{u}}^{(i-1)}$ para a decodificação do nível i, i.e., $\mathcal{D}^{(i)}$ admite a representação por meio da expressão $\mathcal{D}^{(i)}\left(\mathbf{y} \,|\, \hat{\mathbf{u}}^{(0)}, \hat{\mathbf{u}}^{(1)}, \ldots, \hat{\mathbf{u}}^{(i-1)}\right)$. Como o decodificador depende de $\mathbf{y}$ e dos dados de níveis anteriores, respeita o paradigma de (2.7). É, consequentemente, um decodificador multi-estágio.

Mostramos na figura 3.1 uma exemplificação de um decodificador multi-estágio para três níveis, onde a codificação é multi-estágio e dependente, de acordo com (2.6). Observamos na figura as etapas de recodificação e eventual subtração de influências passadas do sinal de modo que ele possa ser processado pelo decodificador do nível seguinte.

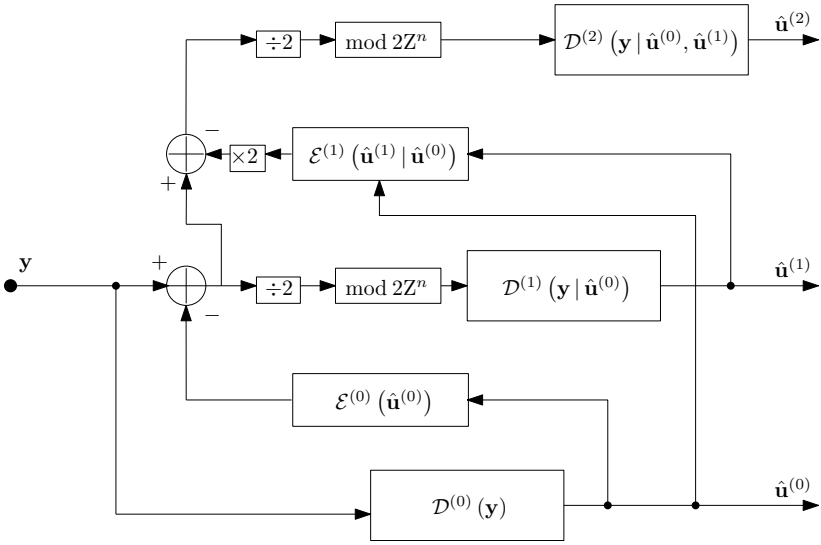


Figura 3.1: Decodificador multi-estágio de 3 níveis, assumindo codificação multi-estágio dependente

A análise da decodificação de mínima distância para o código de reticulado $\mathbf{C}_\Lambda$ é similar à da Construção A na seção 2.3. Desejamos encontrar o ponto do reticulado mais próximo do sinal recebido $\mathbf{y}$. Para isso projetamos o quantizador $Q_\Lambda(\mathbf{y}) \triangleq \underset{\boldsymbol{\lambda}\in\Lambda}{\operatorname{argmin}} \|\mathbf{y}-\boldsymbol{\lambda}\|$ [24].

O decodificador $\mathcal{D}^{(i)}$ pode ser interpretado como um decodificador de mínima distância [24], tal que

$$\hat{\mathbf{c}}^{(i)} = \underset{\mathbf{c}^{(i)}\in\mathbf{C}^{(i)}}{\operatorname{argmin}} \left\| \left(\mathbf{y}^{(i)} - \mathbf{c}^{(i)}\right) \bmod 2\mathbb{Z}^n \right\|, \tag{3.4}$$

onde o argumento da norma é a expressão do decodificador multi-estágio que deduzimos.

Sabe-se de [5] que a Construção D de um único nível reduz para a Construção A. A equação acima se resume a um único nível, já que os efeitos de níveis anteriores são cancelados e já que a operação módulo descarta os níveis superiores. O decodificador de (3.4) é o decodificador ótimo para a Construção A construída com o código $\mathbf{C}^{(i)}$. No entanto, na presença de mais de um nível, como no caso da Construção

D, a operação módulo desconsidera níveis superiores no processo de decodificação. Não usamos toda a informação disponível e por isso a decodificação perde em otimalidade. Porém, é um decodificador bastante simples, o que justifica a sua escolha.

3.2 Projeto usando Códigos Convolucionais

Podemos projetar códigos de reticulado multinível capazes de atingir a capacidade do canal com a Construção D e com a decodificação multiestágio vista na seção anterior. Para isso as taxas escolhidas devem seguir a regra da capacidade definida pela equação $R^{(i)} = C^{(i)}$, cuja explicação detalhada se encontra na subseção 2.1.4.

Como vimos na subseção 2.3.2, a Construção D demanda códigos aninhados. No entanto, códigos convolucionais são códigos difíceis de aninhar, já que são descritos por um número infinito de palavras-código infinitamente longas, ou seja, por matrizes geradoras infinitas. Códigos convolucionais possuem ainda uma outra dificuldade: são códigos usualmente definidos para um número restrito de taxas. A maneira mais prática de se obter códigos com taxas flexíveis é através de puncionamento. Contudo, este procedimento pode comprometer o aninhamento entre os códigos. Portanto, a obtenção de aninhamento para códigos convolucionais é de extrema dificuldade. Por isso simplificamos o projeto, buscando apenas códigos com níveis superiores "não codificados", eliminando assim as restrições de aninhamento sobre os códigos.

Usamos projetos desenvolvidos em [37] com níveis superiores de taxa aproximadamente igual a 1 bit por dimensão (b/dim). Citamos como exemplo dois projetos de taxas utilizados neste trabalho:

- $R^{(0)} = 0,50$ b/dim e $R^{(1)} = 1,00$ b/dim, para as capacidades equivalentes $C^{(0)} = 0,52$ b/dim e $C^{(1)} = 0,98$ b/dim. O projeto resulta na eficiência espectral de $1,50$ b/dim ($3,00$ bits por dimensão complexa - b/cdim);

- $R^{(0)} = 0,50$ b/dim, $R^{(1)} = 1,00$ b/dim e $R^{(2)} = 1,00$ b/dim, para as capacidades equivalentes $C^{(0)} = 0,52$ b/dim, $C^{(1)} = 0,98$ b/dim e $C^{(2)} = 1,00$ b/dim. O projeto resulta na eficiência espectral de $2,50$ b/dim ($5,00$ b/cdim).

As capacidades foram calculadas para esquemas de modulação PAM

com mapeamento Ungerböck, seguindo a fórmula apresentada em (2.5) e, em especial, as equações (16) a (22) em [15]. Apresentamos na figura 3.2 os gráficos das capacidades equivalentes em função da SNR para esquemas de modulação 4-PAM e 8-PAM.

Os códigos de reticulado multinível projetados a partir de códigos convolucionais são códigos simples, de fácil implementação e decodificação, mas possuem duas limitações importantes:

- Não são códigos de alto desempenho, uma vez que os códigos convolucionais não são capazes de se aproximar da capacidade do canal;

- Como mencionado acima e visto no exemplo, as simplificações necessárias para o aninhamento, assim como os projetos padrões para códigos convolucionais, restringem as taxas dos códigos componentes e a eficiência espectral total.

Vemos que, para atender as exigências de alto desempenho e flexibilidade de projeto de taxas imposta pela a regra de projeto para a capacidade, os códigos convolucionais não são adequados. Apresentaremos adiante uma alternativa viável: o uso de códigos LDPC.

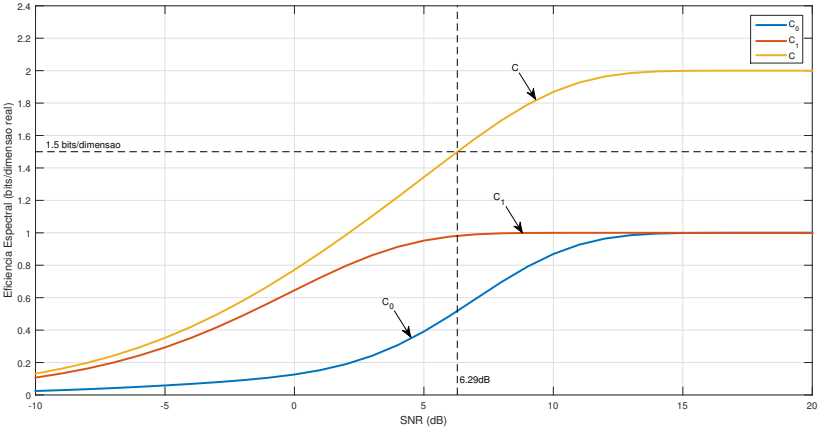
3.3 Convertendo a Construção D′ em Construção D

A Construção D′ não especifica a codificação. Precisamos além do reticulado uma técnica de codificação linear que nos permita obter geradores $\mathbf{g}_j$ do reticulado, para $j = 1, \ldots, k_{L-1}$. Nosso objetivo é manipular as definições da Construção D′ de forma a obter um código de reticulado de definição igual à vista para a Construção D em (2.11), a qual já embute a codificação linear.

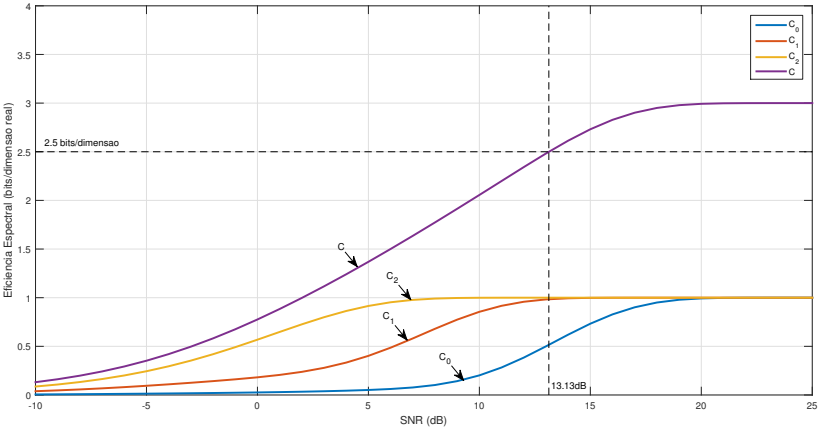
Sejam $\mathbf{x}^{(0)}, \ldots, \mathbf{x}^{(L-1)} \in \{0,1\}^n$ as componentes do sinal transmitido $\mathbf{x}$, tal que

$$\mathbf{x} = \sum_{i=0}^{L-1} 2^i \cdot \mathbf{x}^{(i)}. \tag{3.5}$$

Seja Λ um reticulado construído pela Construção D', conforme a subseção 2.3.3. De acordo com (2.13), para um vetor $\mathbf{x} \in \mathbb{Z}^n$, sabemos



(a) Capacidades equivalentes em função da SNR para os dois níveis de um esquema multinível com Mapeamento Ungerböck usando a modulação 4-PAM. O valor mínimo de SNR para o qual é possível obter uma eficiência espectral de $1,5$ bits/dimensão real é $6,29$dB.



(b) Capacidades equivalentes para os três níveis de um esquema multinível com Mapeamento Ungerböck usando a modulação 8-PAM. O valor mínimo de SNR para o qual é possível obter uma eficiência espectral de $2,5$ bits/dimensão real é $13,13$dB.

Figura 3.2: Capacidades totais e equivalentes para modulações 4-PAM e 8-PAM em função da SNR. A notação C se refere à capacidade total do esquema multinível, ao passo que C_i, para $i = 1, 2, \ldots, L$ e L correspondente ao número de níveis do esquema multinível, denota as capacidades equivalentes dos diferentes níveis.

que $\mathbf{x} \in \Lambda$ se e somente se

$$\mathbf{x}\,\mathbf{h}_j^{\mathrm{T}} \equiv 0 \pmod{2^{i+1}\mathbb{Z}}$$

para $m_{i+1} < j \leq m_i$, $i = 0, \ldots, L-1$.

Como uma congruência módulo 2^{i+1} é também válida módulo $2^{\ell+1}$, para todo $\ell \leq i$, temos que, para $\ell = 0, \ldots, L-1$,

$$\mathbf{x}\,\mathbf{h}_j^{\mathrm{T}} \equiv 0 \pmod{2^{\ell+1}\mathbb{Z}},$$

onde $m_{i+1} < j \leq m_i$ e $i = \ell, \ldots, L-1$ ou, equivalentemente, onde $0 < j \leq m_\ell$. Assim sendo, podemos reescrever (2.13) como

$$\mathbf{x}\,\mathbf{H}_\ell^{T} \equiv 0 \pmod{2^{\ell+1}\mathbb{Z}}$$

para $\ell = 0, \ldots, L-1$.

Analisando as congruências para os níveis 0 e 1, podemos definir uma regra geral de codificação binária para um nível i qualquer.

Para o nível 0 temos

$$\begin{aligned} \mathbf{x}\,\mathbf{H}_0^{\mathrm{T}} &\equiv 0 \pmod 2 \\ \left(\mathbf{x}^{(0)} + 2\cdot\mathbf{x}^{(1)} + \ldots + 2^{L-1}\cdot\mathbf{x}^{(L-1)}\right)\mathbf{H}_0^{\mathrm{T}} &\equiv 0 \pmod 2. \end{aligned} \tag{3.6}$$

Como os termos $\mathbf{x}^{(1)}$ até $\mathbf{x}^{(L-1)}$ são multiplicados por potências de 2 o módulo 2 retorna 0. Desta forma, podemos simplificar (3.6) para

$$\mathbf{x}^{(0)}\,\mathbf{H}_0^{\mathrm{T}} \equiv 0 \pmod 2.$$

No nível 1 observamos

$$\begin{aligned} &\mathbf{x}\,\mathbf{H}_1^{\mathrm{T}} \equiv 0 \pmod{2^2} \\ &\left(\mathbf{x}^{(0)} + 2\cdot\mathbf{x}^{(1)} + 2^2\cdot\mathbf{x}^{(2)} + \ldots + 2^{L-1}\cdot\mathbf{x}^{(L-1)}\right)\mathbf{H}_1^{\mathrm{T}} \equiv 0 \pmod{2^2} \\ &\left(\mathbf{x}^{(0)} + 2\cdot\mathbf{x}^{(1)}\right)\mathbf{H}_1^{\mathrm{T}} \equiv 0 \pmod{2^2} \\ &2\cdot\mathbf{x}^{(1)}\mathbf{H}_1^{\mathrm{T}} \equiv \mathbf{x}^{(0)}\mathbf{H}_1^{\mathrm{T}} \pmod{2^2} \\ &\mathbf{x}^{(1)}\,\mathbf{H}_1^{\mathrm{T}} \equiv \frac{\mathbf{x}^{(0)}\,\mathbf{H}_1^{\mathrm{T}}}{2} \pmod 2. \end{aligned}$$

É possível efetuar a generalização das equações anteriores para um nível $0 \leq i \leq L-1$ qualquer:

$$\begin{aligned}
&\mathbf{x}\,\mathbf{H}_i^{\mathrm{T}} \equiv 0 \pmod{2^{i+1}} \\
&\left(\mathbf{x}^{(0)} + 2 \cdot \mathbf{x}^{(1)} + \ldots + 2^i \cdot \mathbf{x}^{(i)}\right) \mathbf{H}_i^{\mathrm{T}} \equiv 0 \pmod{2^{i+1}} \\
&\mathbf{x}^{(i)}\,\mathbf{H}_i^{\mathrm{T}} \equiv \frac{\left(\sum_{i'=0}^{i-1} 2^{i'} \cdot \mathbf{x}^{(i')}\right) \mathbf{H}_i^{\mathrm{T}}}{2^i} \pmod{2}. \qquad (3.7)
\end{aligned}$$

Para simplificar a notação, podemos reescrever (3.7) como

$$\mathbf{x}^{(i)}\,\mathbf{H}_i^{\mathrm{T}} \equiv \mathbf{d}^{(i)} \pmod{2},$$

onde, para $i = 1, \ldots, L-1$,

$$\mathbf{d}^{(i)} = \frac{\left(\sum_{i'=0}^{i-1} 2^{i'} \mathbf{x}^{(i')}\right) \mathbf{H}_i^{\mathrm{T}}}{2^i} \bmod 2 \qquad (3.8)$$

e $\mathbf{d}^{(0)} = 0$.

Considere um nível $i \in \{0, \ldots, L-1\}$. Podemos expressar $\mathbf{x}^{(i)}$ por sua forma sistemática

$$\mathbf{x}^{(i)} = \left[\mathbf{u}^{(i)} \quad \mathbf{v}^{(i)}\right],$$

onde $\mathbf{u}^{(i)}$ representa o vetor de dados de comprimento k_i e $\mathbf{v}^{(i)}$, o vetor de paridades de comprimento m_i.

A matriz $\mathbf{H}_i$ pode ser dividida em $\mathbf{A}_i$ e $\mathbf{B}_i$ conforme

$$\mathbf{H}_i = [\mathbf{A}_i \quad \mathbf{B}_i],$$

onde $\mathbf{A}_i$ é a submatriz relacionada aos dados, cuja dimensão é $m_i \times k_i$, e $\mathbf{B}_i$, a submatriz inversível relacionada às paridades, de dimensão m_i.

De posse de $\mathbf{u}^{(i)}$ e $\mathbf{d}^{(i)}$, podemos calcular $\mathbf{v}_i$. Empregando (3.8), temos

$$\mathbf{d}_i \equiv \mathbf{x}^{(i)}\,\mathbf{H}_i^{\mathrm{T}} \pmod{2} = \mathbf{u}^{(i)}\mathbf{A}_i^{\mathrm{T}} + \mathbf{v}^{(i)}\mathbf{B}_i^{\mathrm{T}} \pmod{2}.$$

Ao resolver para $\mathbf{v}_i$ obtemos

$$\begin{aligned}\mathbf{v}^{(i)} &\equiv \left(\mathbf{d}^{(i)} - \mathbf{u}^{(i)}\,\mathbf{A}_i^{\mathrm{T}}\right)\left(\mathbf{B}_i^{-1}\right)^{\mathrm{T}} \pmod 2\\ \mathbf{x}^{(i)} &= \left[\mathbf{u}^{(i)} \quad \left(\mathbf{d}^{(i)} - \mathbf{u}^{(i)}\,\mathbf{A}_i^{\mathrm{T}}\right)\left(\mathbf{B}_i^{-1}\right)^{\mathrm{T}}\right] \bmod 2. \end{aligned} \tag{3.9}$$

Recapitulando, iniciamos com a codificação de $\mathbf{u}^{(0)}$ em $\mathbf{x}^{(0)}$ através da equação (3.9) particularizada para $i = 0$, onde $\mathbf{d}^{(i)} = \mathbf{0}$. Para $0 < i \leq L$ encontramos $\mathbf{d}^{(i)}$ segundo a equação (3.8) e em seguida calculamos $\mathbf{x}^{(i)}$ através da equação (3.9). Após repetir os passos anteriores até o nível $L-1$, aplicamos os sinais codificados $\mathbf{x}^{(0)}, \ldots, \mathbf{x}^{(L-1)}$ na equação (3.5) para obter $\mathbf{x}$. Denotamos essa regra de codificação por

$$\mathbf{x} = \mathcal{E}_{\mathrm{D}'}(\mathbf{u}^{(0)}, \ldots, \mathbf{u}^{(L-1)}).$$

A regra de codificação descrita pelas equações (3.8) e (3.9) não é adequada para obter uma representação do código de reticulado como a apresentada em (2.11). Para chegar à expressão de (2.11) descrevemos a seguir um procedimento que converte a Construção D′ em um caso particular da Construção D por meio da construção de geradores do reticulado $\mathbf{g}_j$, onde $j = 1, \ldots, k_{L-1}$.

Seja $k_{-1} = 0$. Para $j = 1, \ldots, k_{L-1}$, seja $i_j \in \{0, \ldots, L-1\}$ tal que $k_{i_j-1} < j \leq k_{i_j}$. Para encontrar $\mathbf{g}_1, \ldots, \mathbf{g}_{k_{L-1}}$ definimos k_{L-1} vetores de dados $\mathbf{u}_j^{(i_j)}$, para $j = 1, \ldots, k_{L-1}$, tais que

$$u_{j,t}^{(i_j)} = \begin{cases} 0, & \text{se } t \neq j \\ 1, & \text{se } t = j \end{cases} \tag{3.10}$$

para $t = 1, \ldots, k_i$.

Para $j = 1, \ldots, k_{L-1}$, seja

$$\boldsymbol{\lambda}_j = \mathcal{E}_{D'}(\mathbf{0}, \ldots, \mathbf{0}, \mathbf{u}_j^{(i_j)}, \mathbf{0}, \ldots, \mathbf{0}).$$

De acordo com (3.9), podemos expressar $\boldsymbol{\lambda}_j$ através de suas componen-

tes binárias

$$\boldsymbol{\lambda}_j^{(\ell)} = \begin{cases} \mathbf{0}, & \text{para } 0 \leq \ell < i_j \\ \begin{bmatrix} \mathbf{u}_j^{(\ell)} & -\mathbf{u}_j^{(\ell)} \left(\mathbf{B}_\ell^{-1}\mathbf{A}_\ell\right)^{\mathrm{T}} \end{bmatrix}, & \text{para } \ell = i_j \\ \begin{bmatrix} \mathbf{0}_{k_\ell} & \mathbf{d}_j^{(\ell)} \left(\mathbf{B}_\ell^{-1}\right)^{\mathrm{T}} \end{bmatrix}, & \text{para } i_j < \ell \leq L-1, \end{cases} \tag{3.11}$$

onde

$$\mathbf{d}_j^{(i)} = \frac{\left(\sum_{i'=0}^{i-1} 2^{i'} \boldsymbol{\lambda}_j^{(i')}\right) \mathbf{H}_i^{\mathrm{T}}}{2^i} \bmod 2. \tag{3.12}$$

Consequentemente, podemos expressar $\boldsymbol{\lambda}_j$ como

$$\boldsymbol{\lambda}_j = \sum_{i=i_j}^{L-1} 2^i \boldsymbol{\lambda}_j^{(i)} = 2^{i_j} \mathbf{g}_j, \tag{3.13}$$

tal que

$$\mathbf{g}_j = \sum_{i=0}^{L-1-i_j} 2^i \boldsymbol{\lambda}_j^{(i+i_j)}. \tag{3.14}$$

Finalmente, podemos utilizar os geradores $\mathbf{g}_j$ em uma definição equivalente a (2.11):

$$\mathbf{C}_\Lambda \triangleq \left\{ \sum_{i=0}^{L-1} \sum_{j=1}^{k_i} 2^i u_j^{(i)} \mathbf{g}_j \,:\, u_j^{(i)} \in \{0,1\} \right\} \bmod 2^L \mathbb{Z}^n. \tag{3.15}$$

Vale mencionar que, diferentemente da Construção D convencional, os geradores $\mathbf{g}_j$ podem ser não binários, i.e., $\mathbf{g}_j \in \{0, \ldots, 2^L - 1\}^n$.

3.3.1 Exemplo

Seja

$$\mathbf{H}_\Lambda = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 2 & 0 & 0 & 2 \end{bmatrix}$$

a matriz de paridade do código de reticulado $\mathbf{C}_\Lambda$, construído a partir dos códigos binários aninhados $\mathbf{C}^{(0)} \subseteq \mathbf{C}^{(1)} \subset \mathbb{F}_2^n$, cujas matrizes de

paridade, respectivamente, são

$$\mathbf{H}_0 = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}, \qquad \mathbf{H}_1 = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}.$$

Podemos escrever $\mathbf{H}_i$, $i = 0, 1$, como $\begin{bmatrix} \mathbf{A}_i & \mathbf{B}_i \end{bmatrix}$, tal que $\mathbf{B}_i$ é uma matriz inversível. Desta forma, temos

$$\mathbf{A}_0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \qquad \mathbf{B}_0 = \mathbf{B}_0^{-1} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix},$$

onde $\mathbf{B}_0$ e a sua inversa $\mathbf{B}_0^{-1}$ são iguais. Da mesma maneira, para $\mathbf{H}_1$ dispomos das matrizes

$$\mathbf{A}_1 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \qquad \mathbf{B}_1 = \mathbf{B}_1^{-1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}.$$

Criaremos com as matrizes apresentadas e a lei de codificação descrita por (3.11), (3.12), (3.13) e (3.14) geradores para o reticulado Λ.

A observação de $\mathbf{H}_0$ e $\mathbf{H}_1$ indica que $k_0 = 1$ e $k_1 = 2$, $j = 1, 2$ e, adotando a notação i_j, $i_1 = 0$ e $i_2 = 1$. De acordo com (3.15), precisamos de k_1 geradores, i.e., de $\mathbf{g}_1$ e $\mathbf{g}_2$. Consequentemente, necessitamos dos vetores de dados $\mathbf{u}_1^{(0)} = \begin{bmatrix} 1 \end{bmatrix}$ e $\mathbf{u}_2^{(1)} = \begin{bmatrix} 0 & 1 \end{bmatrix}$, os quais são vetores canônicos linearmente independentes definidos por (3.10) correspondentes ao índice j de informação.

Iniciamos a obtenção de $\mathbf{g}_1$ com o cálculo de $\boldsymbol{\lambda}_1^{(0)}$ segundo (3.11):

$$\begin{aligned} \boldsymbol{\lambda}_1^{(0)} &= \begin{bmatrix} \mathbf{u}_1^{(0)} & \mathbf{u}_1^{(0)} \left(\mathbf{B}_0^{-1}\mathbf{A}_0\right)^{\mathrm{T}} \end{bmatrix} \\ \boldsymbol{\lambda}_1^{(0)} &= \begin{bmatrix} \mathbf{u}_1^{(0)} & \mathbf{u}_1^{(0)} \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \end{bmatrix} \\ \boldsymbol{\lambda}_1^{(0)} &= \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}. \end{aligned}$$

Em seguida, partimos para o cálculo de $\mathbf{d}_1^{(1)}$ via (3.12):

$$\mathbf{d}_1^{(1)} = \frac{\boldsymbol{\lambda}_1^{(0)}\mathbf{H}_1^{\mathrm{T}}}{2} \bmod 2$$
$$\mathbf{d}_1^{(1)} = \begin{bmatrix} 1 & 1 \end{bmatrix}.$$

Obtemos, então, $\boldsymbol{\lambda}_1^{(1)}$, mas como não há transmissão de dados associada ao nível 1, i.e., como $\ell > i_j$ em (3.11), consideramos a transmissão de um vetor nulo $\mathbf{0}$. Isto nos leva à terceira equação de (3.11):

$$\boldsymbol{\lambda}_1^{(1)} = \begin{bmatrix} \mathbf{0} & \mathbf{d}_1^{(1)} \left(\mathbf{B}_1^{-1}\right)^{\mathrm{T}} \end{bmatrix}$$
$$\boldsymbol{\lambda}_1^{(1)} = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}.$$

Chegamos a $\mathbf{g}_1$ através de (3.14), para $L = 2$ e $i_1 = 0$:

$$\mathbf{g}_1 = \sum_{i=0}^{L-1-i_1} 2^i \boldsymbol{\lambda}_1^{(i+i_1)} = \sum_{i=0}^{1} 2^i \boldsymbol{\lambda}_1^{(i+i_1)}$$
$$\mathbf{g}_1 = \boldsymbol{\lambda}_1^{(0)} + 2\boldsymbol{\lambda}_1^{(1)} = \begin{bmatrix} 1 & 1 & 1 & 3 \end{bmatrix}.$$

Calculamos agora o gerador $\mathbf{g}_2$. Primeiramente, notamos que ele só está definido para o nível 1 e que não há dados no nível 0. Portanto, para este nível, com base em (3.11), $\boldsymbol{\lambda}_2^{(0)} = \mathbf{0}$.

Com a segunda equação de (3.11) obtemos $\boldsymbol{\lambda}_2^{(1)}$:

$$\boldsymbol{\lambda}_2^{(1)} = \begin{bmatrix} \mathbf{u}_2^{(1)} & -\mathbf{u}_2^{(1)} \left(\mathbf{B}_1^{-1}\mathbf{A}_1\right)^{\mathrm{T}} \end{bmatrix}$$
$$\boldsymbol{\lambda}_2^{(1)} = \begin{bmatrix} 0 & 1 & 1 & 1 \end{bmatrix}.$$

Recorremos, então, à equação (3.14) para obter $\mathbf{g}_2$:

$$\mathbf{g}_2 = \sum_{i=0}^{L-1-i_2} 2^i \boldsymbol{\lambda}_2^{(i+i_2)} = \sum_{i=0}^{0} 2^i \boldsymbol{\lambda}_2^{(i+i_2)}$$
$$\mathbf{g}_2 = \boldsymbol{\lambda}_2^{(1)} = \begin{bmatrix} 0 & 1 & 1 & 1 \end{bmatrix}.$$

Por fim, definimos a regra de codificação de um reticulado definido pela matriz de paridade $\mathbf{H}_\Lambda$. Utilizando a expressão (3.15) e lembrando

que $L = 2$, $k_0 = 1$ e $k_1 = 2$, chegamos a

$$\mathbf{C}_\Lambda = \left\{ \sum_{i=0}^{1} \sum_{j=1}^{k_i} 2^i u_j^{(i)} \mathbf{g}_j \,:\, u_j^{(i)} \in \{0,1\} \right\} \bmod 2^2\mathbb{Z}^n$$

$$\mathbf{C}_\Lambda = \left\{ u_1^{(0)}\mathbf{g}_1 + u_1^{(1)}\mathbf{g}_1 + u_2^{(1)}\mathbf{g}_2 \,:\, u_1^{(0)}, u_1^{(1)}, u_2^{(1)} \in \{0,1\} \right\} \bmod 4\mathbb{Z}^n$$

$$\mathbf{C}_\Lambda = \left\{ \left(u_1^{(0)} + u_1^{(1)}\right) \begin{bmatrix} 1 & 1 & 1 & 3 \end{bmatrix} + u_2^{(1)} \begin{bmatrix} 0 & 1 & 1 & 1 \end{bmatrix} : \right.$$
$$\left. u_1^{(0)}, u_1^{(1)}, u_2^{(1)} \in \{0,1\} \right\} \bmod 4\mathbb{Z}^n.$$

CAPÍTULO 4

Códigos LDPC

Códigos LDPC são códigos de alto desempenho, i.e., que possuem a propriedade de se aproximar da capacidade. Além do bom desempenho, veremos que códigos LDPC são flexíveis com relação às taxas, ou seja, são ideais para o projeto através da regra da capacidade da equação (2.8).

Neste capítulo apresentamos na seção 4.1 aspectos teóricos fundamentais dos códigos LDPC. Na seção 4.2 expomos a *Belief Propagation* (BP), o algoritmo mais utilizado na decodificação destes códigos. A descrição da BP é importante, porque dela derivam ferramentas importantes para a análise e projeto de códigos LDPC.

O projeto envolve duas etapas principais: a estimativa de desempenho, que se trata do valor aproximado de probabilidade de erro de bit de códigos LDPC para uma dada SNR, e a otimização, que busca os código de melhor desempenho. Em 4.3 discutimos métodos estimadores de desempenho, em especial, focamos na Density Evolution e na sua aproximação gaussiana. Na seção 4.4 fornecemos detalhes do projeto de distribuição de graus, em particular com a descrição do problema de otimização, inclusive de suas restrições, e do algoritmo de otimização global conhecido como *Differential Evolution*.

Por último, na seção 4.5, mencionamos o método de construção de matrizes de paridade a partir dos parâmetros obtidos na etapa de otimização. Como neste trabalho não realizamos quaisquer contribuições com relação ao processo construtivo, apenas mencionamos os conceitos principais.

4.1 Conceitos Básicos

Seja n o comprimento de palavras-código e k o comprimento de vetores de dados para um código $\mathbf{C}(n,k)$. Um código LDPC (*Low-Density Parity-Check Code*) $\mathbf{C}(n,k)$ é um código linear descrito por uma matriz de paridade $\mathbf{H}$ esparsa, i.e., com uma proporção de 0s muito superior à de 1s [11].

A codificação segue o esquema comum a códigos lineares. Da matriz de paridade $\mathbf{H} = [\mathbf{I}_{n-k}\ \mathbf{A}]$ geramos uma matriz geradora $\mathbf{G} = \left[\mathbf{A}^{\mathrm{T}}\ \mathbf{I}_k\right]$, com a qual codificamos dados $\mathbf{u}$ em palavras-código $\mathbf{x}$ através da equação $\mathbf{x} = \mathbf{uG}$ [25].

É útil representar os códigos LDPC via grafos biparticionados não direcionados conhecidos como grafos de Tanner (vide a figura 6.2 e a contraposição à matriz de paridade $\mathbf{H}_{\mathrm{Ham}}$ abaixo). Nesta representação, os bits c_j de uma palavra-código $\mathbf{c} = (c_1, \ldots, c_n)$ com n componentes são representados por nós de variável v_j. Definimos o conjunto dos nós de variável como $\mathcal{V}_v = \{v_j : j \in \{1, \ldots, n\}\}$ [34].

$$\mathbf{H}_{\mathrm{Ham}} = \begin{array}{c} \\ p_1 \\ p_2 \\ p_3 \end{array} \begin{array}{c} \begin{array}{ccccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

Supondo um código com k bits de informação, i.e., com uma taxa $R = k/n$, as $n - k$ equações de paridade que definem o código linear são representadas por nós de paridade p_q. Definimos o conjunto dos nós de paridade como $\mathcal{V}_p = \{p_q : q \in \{1, \ldots, n-k\}\}$.

Nos grafos a conexão entre nós de variável e de paridade é feita por arestas. Uma aresta que conecta o nó de variável v_j ao nó de paridade p_q é descrita por e_{v_j,p_q}. Definimos o conjunto das arestas do grafo como

$$\mathcal{E} = \left\{e_{v_j,p_q} : j \in \{1, \ldots, n\}, q \in \{1, \ldots, n-k\}\right\}.$$

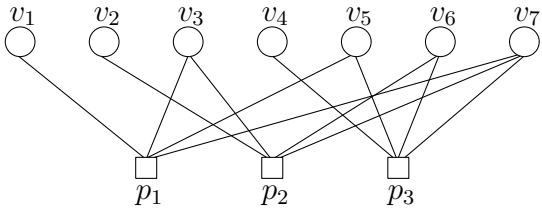


Figura 4.1: Grafo correspondente à matriz de paridade do código de Hamming $\mathbf{C}_{\mathrm{Ham}}(7,4)$

As arestas e_{v_j,p_q} que saem de nós de variável $v_j \in \mathcal{V}_v$ e entram em um nó de paridade $p_q \in \mathcal{V}_p$ definem os bits c_j que participam da equação de paridade q. O nó p_q representa a soma módulo 2 dos bits c_j incidentes [25, 17].

Conceitos úteis para a representação gráfica e para o projeto de códigos LDPC incluem a vizinhança e o grau de um nó.

Seja $\mathcal{V} = \mathcal{V}_v \cup \mathcal{V}_p$ o conjunto de todos os nós de um grafo bipartido. A vizinhança $\mathcal{N}_v$ do nó $v \in \mathcal{V}$ é o conjunto dos nós diretamente conectados a ele. A vizinhança $\mathcal{N}_v$ possui ordem $|\mathcal{N}_v| = d_v$, onde d_v, o grau do nó v, indica o número de conexões diretas [28].

A distribuição dos graus em um código LDPC dá origem a duas classificações: LDPC regulares e irregulares.

(i) Códigos regulares são aqueles com nós v_j, $j = 1, \ldots, n$, e p_q, $q = 1, \ldots, n-k$, com graus $d_{v_1} = \cdots = d_{v_n} = d_v$ e $d_{p_1} = \cdots = d_{p_{n-k}} = d_p$. Códigos regulares apresentam um *gap* significativo para a capacidade de um canal, mesmo para valores assintoticamente grandes dos blocos de informação, i.e, $k, n \to \infty$ [17].

(ii) Códigos irregulares possuem um *gap* menor da capacidade na comparação com os regulares. Na verdade, para comprimentos longos, aproximam-se bastante da capacidade [26]. Eles são chamados de irregulares porque os graus de variável d_{v_j} e os graus de paridade d_{p_q} não são constantes para diferentes valores de j e q. Uma maneira prática de descrever códigos irregulares é por meio de distribuições de graus, como veremos a seguir [28].

Existem duas perspectivas para as distribuições de graus [28]. A perspectiva de nós contabiliza o número de nós com grau i. Associa-se

ao grau i, $i = 1, \ldots, \ell$ (ℓ é o grau máximo para os nós de variável), a fração α_i dos nós de variável, dentre todos os nós $\in \mathcal{V}_v$, com este grau. Podemos juntar as frações no vetor $\boldsymbol{\alpha} = (\alpha_2, \ldots, \alpha_\ell)$ [28]. Há definições idênticas para os nós de paridade, com a diferença que trocamos todas as instâncias de α por β e ℓ por r, sendo r o grau máximo para os nós de paridade.

Usamos o código $\mathbf{C}_{\mathrm{Ham}}(7, 4)$, cujo grafo biparticionado está exposto na figura 6.2, para exemplificar a perspectiva de nós. Os nós de variável apresentam os graus 1, 2 e 3, ao passo que os nós de paridade possuem, todos, grau 4. Seguindo a perspectiva de nós, temos a seguinte distribuição de graus:

$$\alpha_1 = \frac{3}{7}, \quad \alpha_2 = \frac{3}{7}, \quad \alpha_3 = \frac{1}{7}, \quad \beta_4 = 1, \quad \beta_i = 0 \text{ para } i \neq 4$$
$$\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \alpha_3) = \left(\frac{3}{7}, \frac{3}{7}, \frac{1}{7}\right)$$
$$\boldsymbol{\beta} = (\beta_1, \beta_2, \beta_3, \beta_4) = (0, 0, 0, 1)\,.$$

A outra perspectiva é a de arestas. Nesta abordagem contabilizamos o número total n_e de arestas do grafo

$$n_e = n \cdot \sum_{i=2}^{\ell} \alpha_i \cdot i = (n - k) \cdot \sum_{i=2}^{r} \beta_i \cdot i.$$

Então, para o caso dos nós de variável, associamos ao grau i, $i = 1, \ldots, \ell$, a fração λ_i, dentre as n_e arestas existentes, de arestas conectadas a nós de variável $v \in \mathcal{V}_v$ com grau i. De forma análoga, para os nós de paridade associamos ao grau i, $i = 2, \ldots, r$, a fração de arestas ρ_i conectadas a nós de paridade $p \in \mathcal{V}_p$ com grau i [28, 17]. Exemplificamos a perspectiva de arestas usando novamente o código $\mathbf{C}_{\mathrm{Ham}}(7, 4)$

representado na figura 6.2:

$$
\begin{aligned}
&n_e = 12 \\
&\lambda_1 = \frac{3}{12} = \frac{1}{4},\ \lambda_2 = \frac{6}{12} = \frac{1}{2},\ \lambda_3 = \frac{3}{12} = \frac{1}{4},\ \rho_4 = 1,\ \rho_i = 0 \text{ para } i \neq 4 \\
&\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \lambda_3) = \left(\frac{1}{4}, \frac{1}{2}, \frac{1}{4}\right) \\
&\boldsymbol{\rho} = (\rho_1, \rho_2, \rho_3, \rho_4) = (0, 0, 0, 1)\,.
\end{aligned}
$$

Existe uma conexão direta entre $\boldsymbol{\lambda}$ e $\boldsymbol{\alpha}$ e entre $\boldsymbol{\rho}$ e $\boldsymbol{\beta}$. Assim, podemos converter da perspectiva de arestas para a de nós da seguinte maneira [28, 17]:

$$
\boldsymbol{\alpha}_j = \frac{\lambda_j / j}{\sum_{i=2}^{m} \lambda_i / i}, \qquad \boldsymbol{\beta}_j = \frac{\rho_j / j}{\sum_{i=2}^{m} \rho_i / i}
$$

para $j = 2, \ldots, m$.

Ilustramos a conversão entre as perspectivas com base nas distribuição obtidas para o código de Hamming:

$$
\begin{aligned}
\alpha_1 &= \frac{\lambda_1/1}{\lambda_1/1 + \lambda_2/2 + \lambda_3/3} = \frac{\frac{1/4}{1}}{\frac{1/4}{1} + \frac{1/2}{2} + \frac{1/4}{3}} = \frac{3}{7} \\
\alpha_2 &= \frac{\lambda_2/2}{\lambda_1/1 + \lambda_2/2 + \lambda_3/3} = \frac{\frac{1/2}{2}}{\frac{1/4}{1} + \frac{1/2}{2} + \frac{1/4}{3}} = \frac{3}{7} \\
\alpha_3 &= \frac{\lambda_3/3}{\lambda_1/1 + \lambda_2/2 + \lambda_3/3} = \frac{\frac{1/4}{3}}{\frac{1/4}{1} + \frac{1/2}{2} + \frac{1/4}{3}} = \frac{1}{7} \\
\beta_4 &= \frac{\lambda_4/4}{\lambda_4/4} = 1.
\end{aligned}
$$

O par de distribuições de graus $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ ou, equivalentemente, $(\boldsymbol{\lambda}, \boldsymbol{\rho})$, define uma família de códigos chamada ensemble. As análises adiante são médias calculadas sobre o ensemble e não valem para um código em particular (veremos mais detalhes no capítulo 5) [28].

4.2 Decodificação

A BP é usada aqui como um algoritmo de decodificação de bits do tipo *message passing* [34, 19]. Isto significa que na BP mensagens circulam iterativamente no grafo para calcular a confiabilidade dos valores dos bits x_j de uma palavra código $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbf{C} \subseteq \mathbb{F}_2^n$ com base em bits y_j de um vetor recebido $\mathbf{y} = (y_1, \ldots, y_n) \in \mathbb{R}^n$ [29]. Uma maneira geral de calcular confiabilidade é através do uso de probabilidades a posteriori de bit (APPs) $\Pr[x_j \,|\, \mathbf{y}]$. Podemos observar que a probabilidade $\Pr[x_j \,|\, \mathbf{y}]$ é uma marginal da probabilidade a posteriori conjunta $\Pr[\mathbf{x} \,|\, \mathbf{y}]$ [19].

Escrevendo $\Pr[\mathbf{x} \,|\, \mathbf{y}]$ como

$$\Pr[\mathbf{x} \,|\, \mathbf{y}] = \frac{\Pr[\mathbf{x}]\, p(\mathbf{y} \,|\, \mathbf{x})}{p(\mathbf{y})}$$

e assumindo palavras-código equiprováveis, vemos que há proporcionalidade com $p(\mathbf{y} \,|\, \mathbf{x})$. Retomando a hipótese de canal sem memória, podemos fatorar $p(\mathbf{y} \,|\, \mathbf{x})$ da seguinte maneira:

$$p(\mathbf{y} \,|\, \mathbf{x}) = \prod_{j=1}^{n} p(y_j \,|\, x_j)\,.$$

Para levar em conta em uma só expressão as probabilidades condicionais quando $x_j = 0$ e quando é 1, definimos a LLR $\mathsf{L}(y_j)$ [28]:

$$\mathsf{L}(y_j) = \ln\left(\frac{p(y_j \,|\, x_j = 0)}{p(y_j \,|\, x_j = 1)}\right). \tag{4.1}$$

A BP toma, então, como entrada as LLRs $\mathsf{L}(y_j)$ e retorna após um dado número de iterações razões conhecidas como *log a posteriori probability ratios* (LAPPRs), relacionadas às APPs de forma semelhante à forma como as LLRs se relacionam às probabilidades condicionais $p(y_j \,|\, x_j)$ [19]. Definimos a LAPPR para o bit x_j como

$$\mathrm{LAPPR}_j = \ln\left(\frac{\Pr[x_j = 0 \,|\, \mathbf{y}]}{\Pr[x_j = 1 \,|\, \mathbf{y}]}\right).$$

Podemos interpretar as LAPPRs como *soft bits*. Elas contêm informação sobre a confiabilidade do valor de um bit x_j após o recebimento

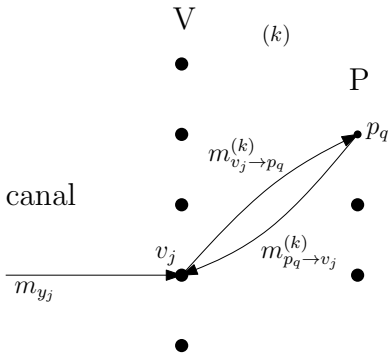


Figura 4.2: Mensagens circuladas na BP

de $\mathbf{y}$ (tal qual as APPs). De fato, podemos usá-las na decodificação [29]. O sinal de uma LAPPR determina o valor do bit: sinal positivo corresponde ao bit 0 e negativo, ao bit 1. A magnitude da LAPPR, por sua vez, indica o grau de confiança com relação ao valor do bit [17].

Como a BP tem acesso a $\Pr[\mathbf{x} \,|\, \mathbf{y}]$ via $p(y_j \,|\, x_j)$ e como calcula ao término de suas iterações $\Pr[x_j \,|\, \mathbf{y}]$ é possível interpretar a BP como um algoritmo de marginalização (ver [19]).

Para cacular as LAPPRs devemos levar em conta a estrutura do código. Sabemos que um código LDPC é um grafo de Tanner e que, de acordo com [34], podemos aplicar a BP nele para calcular as estimativas das LAPPRs desejadas. A BP aplicada a um grafo de Tanner passa mensagens entre nós de variável e de paridade, as quais estão relacionadas com as LLRs definidas em (4.1). Veremos as expressões exatas destas mensagens mais adiante.

Seja $m^{(k)}_{v_j \to p_q}$ a mensagem transmitida de um nó de variável v_j para um nó de paridade p_q, onde $k \in \mathbb{N}^*$ indica a iteração do algoritmo. Analogamente, $m^{(k)}_{p_q \to v_j}$ é a mensagem de p_q para v_j. As mensagens iniciais $m^{(0)}_{v_j \to p_q}$ para quaisquer valores de $j \in \{1, \ldots, n\}$ e $q \in \{1, \ldots, n-k\}$ são dadas por $\mathsf{L}(y_j)$, enquanto que todas as mensagens na direção inversa, tal como $m^{(0)}_{p_q \to v_j}$, são iguais a 0. As mensagens vindas do canal, simbolizadas por m_{y_j}, são iguais às LLRs de entrada $\mathsf{L}(y_j)$. Ilustramos as mensagens passadas na BP na figura 4.2.

No contexto de códigos LDPC, a BP possui dois estágios: o de paridades e o de variáveis. Na etapa de paridades, a mensagem transmitida

de um nó de paridade para um nó de variável é dada por [17]

$$m^{(k)}_{p_q \to v_j} = 2 \cdot \operatorname{atanh}\left(\prod_{v' \in \mathcal{N}_{p_q} \setminus \{v_j\}} \tanh\left(\frac{m^{(k-1)}_{v' \to p_q}}{2} \right) \right). \tag{4.2}$$

Quando lidamos com bits (*hard decoding*), ao invés de indicadores de confiabilidade (*soft decoding*), verificamos que um nó de paridade retorna como mensagem a soma módulo 2 dos bits vindos de nós de variável vizinhos, exceto pelo nó que receberá a mensagem. A equação (4.2) é a operação equivalente no domínio real, com mensagens relacionadas às LLRs [17].

Na etapa de variáveis, a mensagem transmitida de um nó de variável para um nó de paridade é definida como [17]

$$m^{(k)}_{v_j \to p_q} = m_{y_j} + \sum_{p' \in \mathcal{N}_{v_j} \setminus \{p_q\}} m^{(k)}_{p' \to v_j}. \tag{4.3}$$

Isto significa que esta mensagem consiste da soma da mensagem do canal e das mensagens dos nós de paridade vizinhos, exceto pelo nó para o qual se deseja enviar a mensagem.

Com base nas equações (4.2) e (4.3) é possível mostrar que, para uma árvore de profundidade k, as LAPPRs podem ser calculadas de forma exata [19, 29] através de

$$\mathrm{LAPPR}_j = m_{y_j} + \sum_{p' \in \mathcal{N}_{v_j}} m^{(k)}_{p' \to v_j}. \tag{4.4}$$

Para um grafo com ciclos o lado direito da equação (4.4) fornece uma boa estimativa da LAPPR após um número suficiente de iterações [19, 29, 17], i.e.,

$$\widehat{\mathrm{LAPPR}}_j = m_{y_j} + \sum_{p' \in \mathcal{N}_{v_j}} m^{(k)}_{p' \to v_j}.$$

4.3 Estimativa de Desempenho

Uma das técnicas de estimativa de desempenho para um código LDPC é a *Density Evolution*. Ela fornece uma estimativa da probabilidade de erro (de bit) para códigos LDPC com base nas pdfs das LLRs do canal condicionadas em $\mathbf{x} = \mathbf{0}$ [27]. O algoritmo utiliza premissas de simetria que permitem obter o erro de um código após um dado número de iterações com a análise apenas da palavra-código nula $\mathbf{x} = \mathbf{0}$. Além disso, a simetria permite considerar as LLRs do canal variáveis aleatórias i.i.d. [27]. Logo, a *Density Evolution* considera uma única pdf condicionada como entrada.

Como o algoritmo se fundamenta na BP, itera igualmente entre os estágios de paridade e de variável. Mas, em virtude das hipóteses de simetria, podemos considerar as pdfs das mensagens de variável e de paridade em uma mesma iteração k invariáveis com relação ao índice dos nós, i.e., temos na iteração k uma única pdf de mensagem na direção variável-paridade e uma outra na direção inversa. Para maiores detalhes sobre a *Density Evolution* referimos o leitor a [28].

A *Density Evolution* apresenta uma matemática complexa e de difícil implementação. Existem aproximações que tornam seu cálculo mais eficiente, como a Aproximação Gaussiana (*Gaussian Approximation*, GA), a qual descrevemos abaixo.

4.3.1 Aproximação Gaussiana

Consideremos um canal com ruído aditivo gaussiano $Z \sim \mathcal{N}(0, N_0/2)$, tal que para um sinal transmitido x, um ruído z e um sinal recebido y, temos a relação

$$y = x + z.$$

Como veremos a seguir, a hipótese de canal gaussiano é interessante para a análise da GA pois em um canal gaussiano as LLRs são também gaussianas [28]. Com o canal gaussiano, a pdf de entrada para a *Density Evolution* $p\left(\mathrm{LLR}(y) \,|\, \mathbf{x} = \mathbf{0}\right)$ é dada por

$$p(LLR(y) \,|\, \mathbf{x} = \mathbf{0}) = \sqrt{\frac{\sigma^2}{8\pi}} \mathrm{e}^{-\frac{\left(y - \frac{2}{\sigma^2}\right)^2 \sigma^2}{8}},$$

cuja média é $2/\sigma^2$ e a variância é $4/\sigma^2$, i.e., $\mu = 2\sigma^2$.

A ideia da GA é aproximar as pdfs das mensagens trocadas na BP por pdfs gaussianas [29]. A grande vantagem é a facilidade de descrição. Uma distribuição gaussiana é completamente determinada por apenas dois parâmetros, a média (μ) e a variância $\left(\sigma^2\right)$. Portanto, durante a evolução das pdfs, basta acompanhar a evolução destes dois parâmetros ao longo das iterações k. Além disso, é comum fazer uma outra simplificação: considerar a suposição de consistência da distribuição gaussiana. Assumindo a pdf de uma mensagem genérica m, de variável aleatória M, a hipótese de consistência requer que [29]

$$p(m) = \mathrm{e}^m \, p(-m). \tag{4.5}$$

Se considerarmos uma distribuição gaussiana $M \sim \mathcal{N}\left(\mu, \sigma^2\right)$, (4.5) reduz para $\sigma^2 = 2\mu$, o que implica $M \sim \mathcal{N}\left(\mu, 2\mu\right)$ [29]. A vantagem de se supôr consistência é que só é necessário acompanhar a evolução de uma única variável: a média ou a variância da pdf. O problema se torna unidimensional, tal como no caso do canal com apagamento [27], onde a única variável que deve ser acompanhada ao longo das iterações da BP é o apagamento ϵ do canal [28]. Contudo, o uso da GA ignora as mudanças sofridas pelas pdfs ao longo da Density Evolution através das operações de convolução entre as mensagens. A aproximação, portanto, diminui a precisão da predição de desempenho. Além disso, quando a pdf de entrada não é gaussiana, a suposição de pdfs gaussianas torna a aproximação ainda menos precisa.

A GA adota as mesmas premissas de simetria da *Density Evolution*. Elas permitem considerar as pdfs $p\left(m_{v_j \to p_q}^{(k)}\right)$ e $p\left(m_{p_q \to v_j}^{(k)}\right)$ invariáveis para os diferentes valores de $j \in \{1, \ldots, n\}$ e $q \in \{1, \ldots, n-k\}$ [17]. Em outras palavras, em uma mesma iteração k, todas as pdfs de mensagens saindo de nós de paridade são iguais. A mesma constatação vale para as pdfs de mensagens emitidas por nós de paridade. Portanto, passaremos a omitir a direção da mensagem na notação das pdfs, assim como os índices j e q onde não forem necessários. Desta maneira criamos os símbolos $p\left(m_v^{(k)}\right)$ para a pdf de uma mensagem saindo de um nó de variável qualquer na iteração k e $p\left(m_p^{(k)}\right)$ para a pdf correspondente de um nó de paridade. Para as mensagens vindas do canal, supomos $p\left(m_y\right)$ como a pdf das LLRs condicionadas em $\mathbf{x} = \mathbf{0}$, o que significa que $p\left(m_y\right) = p_L\left(\ell \,|\, \mathbf{x} = \mathbf{0}\right)$, onde ℓ é um valor genérico de LLR do

canal e L é a variável aleatória associada. As pdfs $p\left(m_v^{(k)}\right)$, $p\left(m_p^{(k)}\right)$ e $p\left(m_y\right)$ têm médias $\mu_v^{(k)}$, $\mu_p^{(k)}$ e μ_y, respectivamente.

As equações iterativas da GA são semelhantes às equações da BP. Elas são obtidas, na verdade, do valor esperado E (avaliado com base em uma pdf normal e consistente $\mathcal{N}\left(\mu, 2\mu\right)$) destas equações [29].

Assumindo códigos LDPC regulares, obtemos para nós de variável v o resultado a seguir [29]:

$$\mathrm{E}\left[m_{v_j\to p_q}^{(k)}\right] = \mathrm{E}\left[m_y + \sum_{p'\in\mathcal{N}_{v_j}\setminus\{p_q\}} m_{p'\to v_j}^{(k)}\right]$$
$$\mu_v^{(k)} = \mu_y + \left(d_v - 1\right)\mu_p^{(k)}.$$

Fazendo a média sobre a distribuição de graus $\lambda(x)$, obtemos para códigos LDPC irregulares a expressão

$$\mu_v^{(k)} = \mu_y + \sum_i \lambda_i \left(i-1\right)\mu_p^{(k)}. \tag{4.6}$$

Para os nós de paridade, novamente assumindo códigos LDPC regulares, temos [29]

$$\mathrm{E}\left[m_{p_q\to v_j}^{(k)}\right] = \mathrm{E}\left[2\cdot \mathrm{atanh}\left(\prod_{v'\in\mathcal{N}_{p_q}\setminus\{v_j\}} \tanh\left(\frac{1}{2}m_{v'\to p_q}^{(k-1)}\right)\right)\right]. \tag{4.7}$$

Com base em [29], sabemos que o valor esperado de (4.7) resulta na equação

$$\Phi\left(\mu_p^{(k)}\right) = 1 - \left[1 - \Phi\left(\mu_v^{(k-1)}\right)\right]^{d_p-1}. \tag{4.8}$$

Para $\mu \geq 0$ define-se [29]

$$\Phi\left(\mu\right) \triangleq 1 - \frac{1}{\sqrt{4\pi\mu}}\int_{-\infty}^{\infty} \tanh\left(\tau/2\right)\mathrm{e}^{-(\tau-\mu)^2/(4\mu)}\mathrm{d}\tau.$$

Seja $\mathcal{M}: \{0,1\} \to \{1,-1\}$ o mapeamento do bit 0 para +1 e do bit 1 para −1. A função $\Phi\left(\mu\right)$ só está definida para valores não negativos de μ porque supomos a transmissão de uma palavra-código nula, ou seja, só símbolos reais +1. Isto significa que as médias das mensagens

devem ser positivas.

A partir da equação (4.8) isolamos μ_p^k:

$$\mu_p^{(k)} = \Phi^{-1}\left(1 - \left[1 - \Phi\left(\mu_v^{(k-1)}\right)\right]^{d_p - 1}\right).$$

Fazendo a média para a distribuição de graus $\rho(x)$, obtemos para códigos LDPC irregulares a expressão [29]

$$\mu_p^{(k)} = \sum_i \rho_i \Phi^{-1}\left(1 - \left[1 - \Phi\left(\mu_v^{(k-1)}\right)\right]^{i-1}\right). \tag{4.9}$$

Pelas hipóteses de simetria adotadas anteriormente para a Density Evolution, a cada iteração a GA retorna apenas uma média, comum a cada nó de variável. A média retornada pela GA a cada iteração k depende apenas da média μ_y das LLRs do canal condicionadas em $\mathbf{x} = \mathbf{0}$ e da parametrização com relação a $\lambda(x)$ e $\rho(x)$. A expressão da média na saída da GA é idêntica a (4.6). Podemos ainda substituir $\mu_p^{(k)}$ em (4.6) pela expressão em (4.9) para criar a equação recursiva

$$\mu_v^{(k)} = \mu_y + \sum_i \lambda_i (i-1)\left[\sum_j \rho_j \Phi^{-1}\left(1 - \left[1 - \Phi\left(\mu_v^{(k-1)}\right)\right]^{j-1}\right)\right].$$

Recordando a suposição de transmissão de uma palavra-código nula e que os bits 0 são mapeados para símbolos reais +1, podemos afirmar que a probabilidade de erro de bit $P_e^{(k)}$ para a k-ésima iteração pode ser estimada pela equação [17]

$$\hat{P}_e^{(k)} = \Pr\left[m_v^{(k)} < 0\right],$$

onde $p\left(m_v^{(k)}\right)$ é uma pdf do tipo $\mathcal{N}\left(\mu_v^{(k)}, 2\mu_v^{(k)}\right)$.

Em resumo, a GA tem como entrada a média μ_y das pdfs das LLRs do canal condicionadas em $\mathbf{x} = \mathbf{0}$. No caso gaussiano este μ_y é facilmente calculável à partir da variância σ^2 do ruído. Junto com as distribuições de grau $\boldsymbol{\lambda}$ e $\boldsymbol{\rho}$, a GA usa μ_y para o cálculo da P_e [29].

4.4 Projeto de Distribuição de Graus

O projeto de um código LDPC binário $\mathbf{C}$ de um único nível determina o par de distribuições de graus, ou seja, um vetor $(\boldsymbol{\lambda}, \boldsymbol{\rho})$, que minimiza a probabilidade de erro de $\mathbf{C}$ [26]. A função objetivo da otimização é dada pela *Density Evolution* para um número fixo k de iterações [17, 28]. Este problema de otimização possui restrições como as citadas acima: as frações assumidas pelos parâmetros devem somar a 1 para formar distribuições de graus $\boldsymbol{\lambda}$ e $\boldsymbol{\rho}$ válidas; as distribuições devem respeitar uma equação definida pela taxa R de $\mathbf{C}$; e cada parâmetro deve pertencer a $[0, 1]$ para ser uma fração válida de uma distribuição de probabilidade [28]. Fornecemos a seguir o problema completo [28, 13]:

$$
\begin{aligned}
&(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \underset{(\boldsymbol{\alpha},\boldsymbol{\beta})}{\operatorname{argmin}} \; P_e^{(k)}, \qquad \text{sujeito a:} \\
&\sum_{i=1}^{\ell} \alpha_i = 1, \quad \sum_{i=1}^{r} \beta_i = 1 \\
&\sum_{i=1}^{\ell} i \cdot \alpha_i = (1 - R) \cdot \sum_{i=1}^{r} i \cdot \beta_i \\
&0 \leq \alpha_i \leq 1 \qquad \text{para } i = 1, 2, \ldots, \ell \\
&0 \leq \beta_i \leq 1 \qquad \text{para } i = 1, 2, \ldots, r.
\end{aligned}
$$

Deseja-se que a otimização descrita acima seja global para obter a menor probabilidade de erro possível [26]. A otimização pode, por exemplo, ser realizada com algoritmos de busca direta [32]. São métodos que criam variações sobre os vetores de parâmetros e testam se os vetores gerados são admissíveis e se melhoram ou não o valor da função objetivo. Outra estratégia é o uso de algoritmos genéticos, pela qual vetores de parâmetros formam populações. Nesta populações procedimentos de mutação de vetores e recombinação de parâmetros privilegiam os membros com menores valores de função objetivo [32].

Um método de otimização eficiente e muito utilizado no projeto de códigos LDPC é a *Differential Evolution* [12, 13, 36, 38], cuja abordagem mistura características de algoritmos de busca direta com algoritmos genéticos.

4.4.1 Projeto via *Differential Evolution*

A *Differential Evolution* opera com populações contendo NP vetores do tipo $\mathbf{x} = (x_1, x_2, \ldots, x_D)$. O tamanho da população é fixo, e os vetores evoluem ao longo das gerações de acordo com procedimentos conhecidos como mutação e recombinação [32]. Os parâmetros recebem normalmente a notação $x_{i,G}$, onde i se refere ao índice dos D parâmetros e G à geração.

O algoritmo começa com a geração de uma população inicial. Vetores são gerados aleatoriamente por meio de uma distribuição uniforme [12, 33, 36]. O ideal é buscar uma população inicial que cubra de forma uniforme o espaço do parâmetro vetorial $\mathbf{x}$ [33].

Calculamos a função objetivo $f_{\text{obj}}\left(\mathbf{x}^{(j)}\right)$ (baseada na *Density Evolution*) do j-ésimo vetor $\mathbf{x}^{(j)}$ da população para $j = 1, 2, \ldots, NP$ [12, 36, 38]. Selecionamos como o melhor vetor aquele com o menor valor de f_{obj}. Este vetor recebe a notação $\mathbf{x}^{\text{best}}$ [32].

Após a primeira geração e para todas as demais aplicamos aos vetores $\mathbf{x}^{(j)}$ os procedimentos de mutação e recombinação. Os vetores mutados recebem a notação $\mathbf{v}$ e os recombinados (também conhecidos como vetores de teste) a notação $\mathbf{u}$ [32].

Para cada vetor alvo $\mathbf{x}_G^{(j)}$ a *Differential Evolution* normalmente cria um vetor de mutação $\mathbf{v}_{G+1}^{(j)} = \left(v_{1,G+1}^{(1)}, \ldots, v_{D,G+1}^{(j)}\right)$ com D parâmetros através da soma de uma combinação linear de vetores distintos da população (entre si e ao vetor sofrendo a mutação) com o melhor vetor da geração atual [12, 33, 36]. Calculamos o vetor de mutação da seguinte maneira [12, 32, 36]:

$$\mathbf{v}_{G+1}^{(j)} = \mathbf{x}_G^{\text{best}} + F \cdot \left(\mathbf{x}_G^{(r_1)} - \mathbf{x}_G^{(r_2)} + \mathbf{x}_G^{(r_3)} - \mathbf{x}_G^{(r_4)}\right), \tag{4.10}$$

onde j descreve o índice do vetor na população, i.e., $j = 1, \ldots, \text{NP}$, G a geração atual e $G+1$ a futura, F a constante real que controla a amplificação da variação diferencial (a escolha padrão é F=0,5 [32]), $\mathbf{x}_G^{\text{best}}$ o melhor vetor da geração atual e r_1, r_2, r_3 e r_4 inteiros distintos escolhidos aleatoriamente dentro do intervalo $[1, n]$, tais que $r_1, r_2, r_3, r_4 \neq j$ [32].

Os parâmetros do vetor de mutação são então combinados com o vetor alvo ($\mathbf{x}$) para gerar o vetor de teste [12, 32, 36]. Para a descrição da recombinação, seja

$$\mathbf{u}_G^{(j)} = \left(u_{1,G}^{(j)}, \ldots, u_{i,G}^{(j)}, \ldots, u_{D,G}^{(j)}\right)$$

o vetor recombinado, onde $i \in \{1, \ldots, D\}$ é o índice das componentes do vetor $\mathbf{u}_G^{(j)}$.

A recombinação segue a seguinte expressão [12, 33, 36]:

$$u_{i,G+1}^{(j)} = \begin{cases} v_{i,G+1}^{(j)}, & \text{se } \gamma(i) \leq \text{CR ou } i = d(j) \\ x_{i,G}^{(j)}, & \text{caso contrário .} \end{cases} \tag{4.11}$$

Nas expressões acima, γ refere-se à i-ésima realização de um gerador aleatório uniforme cujos resultados $\in [0, 1]$ [33], CR à constante de recombinação $\in [0, 1]$ e d à função de escolha aleatória uniforme de um índice $\in \{1, \ldots, D\}$ para o vetor j, que assegura que $\mathbf{u}_{G+1}^{(j)}$ receberá ao menos um parâmetro de $\mathbf{v}_{G+1}^{(i)}$ [33].

Para decidir se um vetor de teste deve ou não se tornar um membro da geração futura, o vetor $\mathbf{u}_{G+1}^{(j)}$ é comparado com o vetor alvo $\mathbf{x}_G^{(j)}$. Se $\mathbf{u}_{G+1}^{(j)}$ produzir um valor de f_{obj} menor que o de $\mathbf{x}_G^{(j)}$, então $\mathbf{u}_{G+1}^{(j)}$ toma o lugar de $\mathbf{x}_G^{(j)}$ na geração $G + 1$. Caso contrário, o valor atual $\mathbf{x}_G^{(j)}$ é mantido. Esta operação recebe o nome de seleção [12, 33, 36]. Cada vetor da população tem que servir como vetor alvo, de forma que haja NP competições durante uma geração.

4.5 Construção

Chamamos de construção o método pelo qual particularizamos o ensemble para um código específico por meio da definição de um grafo de Tanner. Entre as várias formas de se construir códigos LDPC escolhemos o algoritmo PEG (*Progressive Edge Growth*), porque ele é capaz de produzir bons códigos para blocos relativamente curtos de forma simples [14].

O PEG tem como entrada a distribuição de graus $\boldsymbol{\lambda}$ e o o comprimento das palavras-código n. Ele retorna o grafo de Tanner, i.e., a matriz de paridade $\mathbf{H}$ do código LDPC [14, 29].

Vale reparar que o PEG não depende de $\boldsymbol{\rho}$. Na verdade, com base em um valor específico de $\boldsymbol{\lambda}$, o PEG define a distribuição de graus $\boldsymbol{\rho}$,

normalmente tornando-a bem homogênea, i.e., concentrada em dois (no máximo três) graus consecutivos [14].

CAPÍTULO 5

Projeto de Códigos LDPC Multinível

Um esquema de codificação multinível com códigos LDPC envolve a divisão dos dados em níveis, de modo a utilizar em cada nível um código LDPC com uma taxa apropriada [37]. O projeto de taxas segue a regra da capacidade (2.8), pela qual a taxa $R^{(i)}$ para um canal i é selecionada igual à capacidade equivalente $C^{(i)}$. Esta abordagem é genérica e independe de um modelo de canal.

Como estamos lidando com canais equivalentes de um esquema multinível, as LLRs para a decodificação BP são alteradas, conforme veremos na seção 5.2. A distribuição de probabilidade delas, usadas na entrada da *Density Evolution*, também devem ser modificadas, mudando ligeiramente o processo de estimativa de desempenho visto na seção 4.3. Discutimos as mudanças necessárias na seção 5.3.

Nosso enfoque é a obtenção de esquemas capazes de criar reticulados. Portanto, de acordo com as construções da seção 2.3, os códigos componentes devem estar aninhados. A otimização do nosso esquema multinível utiliza como função objetivo a soma das probabilidades de erro calculadas para os pares $\left(\boldsymbol{\lambda}^{(i)}, \boldsymbol{\rho}^{(i)}\right)$, de $i = 0$ até $L - 1$. Esta otimização se submete não somente às restrições mencionadas na seção 4.4, mas também às restrições impostas pelo aninhamento. Explica-

mos em detalhes a otimização adaptada ao aninhamento de códigos na seção 5.4.

Em [30] foi criado um algoritmo multinível para a construção de códigos LDPC, i.e., para a definição de matrizes de paridade $\mathbf{H}$ à partir de ensembles $(\boldsymbol{\lambda}, \boldsymbol{\rho})$ projetadas. Explicamos brevemente este processo de construção na seção 5.5.

5.1 Modelo do Canal

Baseamos o modelo de canal utilizado ao longo deste capítulo nos conceitos expostos na subseção 2.1.2.

Seja um canal real sem memória com ruído aditivo $Z \sim \mathcal{N}(0, N_0/2)$. Sendo $x \in \mathbb{R}$ um símbolo na entrada do canal e $y \in \mathbb{R}$ o símbolo correspondente na saída, temos

$$y = x + z.$$

O esquema de codificação multinível permite a criação de canais equivalentes. Lembramos da seção 2.1 que o símbolo x corresponde ao mapeamento $\mathcal{M}$ de um conjunto de L bits, i.e., $x = \mathcal{M}\left(x^{(0)}, \ldots, x^{(L-1)}\right)$. Supomos a transmissão de um bit $x^{(i)}$ no canal equivalente i para $i = 0, \ldots, L-1$.

Observamos um ruído aditivo equivalente $z^{(i)}$, cuja variância é $\left(\sigma^{(i)}\right)^2$ e não possui necessariamente relação direta com $N_0/2$ do canal físico. As distribuições de probabilidade dos canais equivalentes são, em geral, distintas à gaussiana.

Seja $y^{(i)}$ o sinal recebido de um canal equivalente i. Este sinal é dado por

$$y^{(i)} = x^{(i)} + z^{(i)}.$$

5.2 Decodificação

O decodificador BP de um canal equivalente i exige uma LLR como entrada. Como vimos, os símbolos transmitidos são compostos por bits de L níveis. Observa-se que um bit $x^{(i)}$ possui dependência nos bits anteriores: $x^{(0)}, \ldots, x^{(i-1)}$. Assumindo o sinal $\mathbf{x} = (x_1, \ldots, x_n)$, para um símbolo transmitido x, um símbolo recebido y e bits $x^{i'}$ conhecidos,

onde $0 \leq i' < i$, definimos a LLR $\mathsf{L}^{(i)}(y)$ (de variável aleatória $L^{(i)}$) como

$$\mathsf{L}^{(i)}(y) = \ln\left(\frac{p\left(y \,|\, x^{(i)}=0, x^{(0)}, \ldots, x^{(i-1)}\right)}{p\left(y \,|\, x^{(i)}=1, x^{(0)}, \ldots, x^{(i-1)}\right)}\right). \tag{5.1}$$

Podemos reescrever a equação (5.1) usando a definição do conjunto $\mathcal{X}\left(x^{(0)}, \ldots, x^{(i)}\right)$ fornecida em (2.1):

$$\mathsf{L}^{(i)}(y) = \ln\left(\frac{\sum\limits_{x \in \mathcal{X}\left(x^{(0)},\ldots,x^{(i-1)},0\right)} p(y \,|\, x)}{\sum\limits_{x \in \mathcal{X}\left(x^{(0)},\ldots,x^{(i-1)},1\right)} p(y \,|\, x)}\right).$$

Sabemos que $\mathsf{L}^{(i)}(y)$ é uma função de y. Conhecemos as pdfs do tipo $p(y \,|\, x)$: com base no modelo suposto na seção 5.1, $p(y \,|\, x)$ constituem pdfs gaussianas de média x e variância σ^2. Ou seja, com base no nosso modelo, a LLR de um canal equivalente corresponde ao logaritmo da razão de somas de exponenciais.

Com (5.1) prosseguimos então com os passos da BP padrão, vista na seção 4.2.

5.3 Estimativa de Desempenho

Para aplicar a *Density Evolution* a canais equivalentes precisamos especificar a entrada do algoritmo. Antes, na seção 4.3, assumimos como entrada a pdf da LLR do canal mononível. Supondo um canal gaussiano, sabemos de [28] que esta pdf é uma gaussiana. No entanto, a expressão da LLR em (5.1) não admite esta hipótese. Devemos calcular novas pdfs.

Com base nas premissas de simetria e nas considerações que elas permitem com relação às pdfs das mensagens da BP (ver subseção 4.3.1), as variáveis aleatórias $L_j^{(i)}$ para $j = 1, 2, \ldots, n$ são i.i.d e iguais à $L^{(i)}$. Só precisamos, então, levar em conta a pdf $p_{L^{(i)}}(\ell \,|\, \mathbf{x} = \mathbf{0})$ (como vemos, independente do índice j) de uma LLR qualquer de entrada, condicionada na transmissão do vetor nulo $\mathbf{x} = \mathbf{0}$. Esta pdf é importante porque é a entrada da *Density Evolution* para o canal equivalente i. Fora isto, não há qualquer outra mudança com relação ao caso mononível da seção 4.3.

A pdf $p_{L^{(i)}}\left(\ell \,|\, \mathbf{x} = \mathbf{0}\right)$ não é conhecida a priori, mas como as LLRs são calculadas em função de y, cuja variável aleatória Y possui uma pdf condicionada $p_Y\left(y \,|\, \mathbf{x} = 0\right)$ conhecida, é possível calcular a pdf das LLRs através do cálculo da pdf de uma função de variável aleatória. Encontramos o procedimento de cálculo em [23].

5.3.1 Aproximação Gaussiana

Como vimos na subseção 4.3.1, a GA é uma simplificação da *Density Evolution* que assume pdfs gaussianas consistentes para as mensagens m da BP.

Como nem sequer as pdfs das LLRs do canal (a entrada da GA) são gaussianas, encontramos a média $\mu^{(i)}$ e a variância $\left(\sigma^{(i)}\right)^2$ para a pdf $p_{L^{(i)}}\left(\ell|\mathbf{x} = \mathbf{0}\right)$.

Como a suposição de que a distribuição original de $L^{(i)}$ se assemelha a uma distribuição do tipo $\mathcal{N}\left(\mu, 2\mu\right)$ não é verossímil, é necessário forçar que a pdf simplificada gaussiana seja o mais parecida possível com $p_{L^{(i)}}\left(\ell \,|\, \mathbf{x} = \mathbf{0}\right)$. Por meio de testes e análise empíricas baseadas no grau de semelhança de uma pdf gaussiana com a pdf real de entrada, verifica-se que uma maneira simples de atingir este objetivo é definir um $\mu^{(i)'}$ igual à média geométrica de $\mu^{(i)}$ e $\left(\sigma^{(i)}\right)^2$ originais:

$$\mu^{(i)'} \triangleq \sqrt{\mu^{(i)} \cdot 2\left(\sigma^{(i)}\right)^2}. \tag{5.2}$$

Ilustramos na figura 5.1 um exemplo de pdf na entrada da *Density Evolution* para um canal equivalente. Comparamos a curva com a pdf gaussiana obtida de acordo com a aproximação (5.2).

A média aproximada $\mu^{(i)'}$ é utilizada como a entrada da GA para o nível i. O algoritmo prossegue tal como descrito na subseção 4.3.1.

5.4 Projeto de Distribuição de Graus

Como um projeto LDPC multinível envolve o projeto de códigos para L canais equivalentes, um bom código LDPC multinível precisa obedecer à regra de otimização enunciada para a codificação multinível na

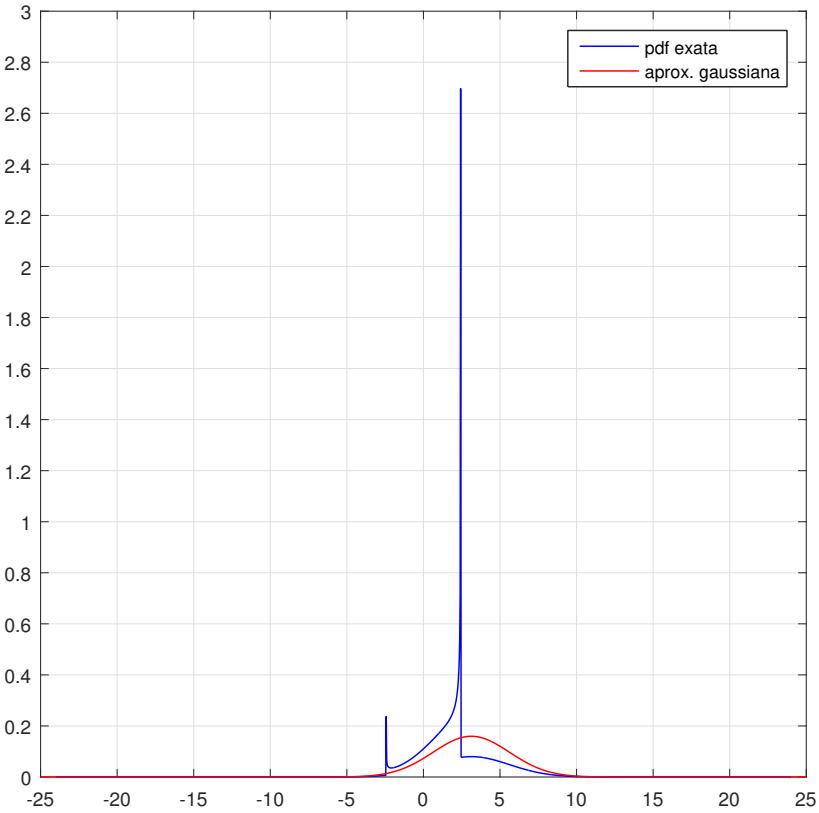


Figura 5.1: Comparação da pdf real na entrada da Density Evolution com a definição da aproximação gaussiana definida fornecida em (5.2).

subseção 2.1.5. Para isso utilizamos a função objetivo

$$f_{\text{obj}} = \sum_{i=0}^{L-1} P_e^{(i)}. \tag{5.3}$$

De acordo com [37] o projeto ótimo de esquemas multinível requer que os códigos componentes de cada nível tenham performance igual, i.e., probabilidades de erro iguais:

$$P_e^{(0)} = P_e^{(1)} = \ldots = P_e^{(L-1)}.$$

Este objetivo de projeto tenta obter bons códigos em todos os níveis, de forma que nenhum código componente degrade a performance geral do esquema multinível. A função objetivo em (5.3) tenta alcançar estas metas de projeto, minimizando, para uma dada SNR, as probabilidades de erro dos códigos e também as tornando iguais.

5.4.1 Restrições

Temos que levar em consideração algumas restrições para a otimização de códigos LDPC. Primeiramente, reiteramos as restrições lineares já expostas na seção 4.4, mas adaptadas aqui para os canais equivalentes. Consideremos as distribuições de graus $\boldsymbol{\alpha}^{(i)} = \left(\alpha_2^{(i)}, \alpha_3^{(i)}, \ldots, \alpha_{\ell^{(i)}}^{(i)}\right)$ e $\boldsymbol{\beta}^{(i)} = \left(\beta_2^{(i)}, \beta_3^{(i)}, \ldots, \beta_{r^{(i)}}^{(i)}\right)$ para a perspectiva de nós e a taxa $R^{(i)}$ para um código equivalente i, $i = 0, 1, \ldots, L-1$.

Para facilitar a otimização e o aninhamento dos códigos (o que é necessário para a construção de reticulados - vide as Construções D e D′ no capítulo 2), observa-se que bons códigos costumam ter distribuições $\boldsymbol{\beta}^{(i)}$ com apenas dois graus consecutivos: $r^{(i)} - 1$ e $r^{(i)}$ [26]. Além disso, o método de construção escolhido, o PEG, geralmente produz a partir da distribuição de graus dos nós de variável uma distribuição de graus dos nós de paridade com graus consecutivos [14]. Com base em $\boldsymbol{\beta}^{(i)}$ dado por $\left(\beta_{r^{(i)}-1}^{(i)}, \beta_{r^{(i)}}^{(i)}\right)$ e na equação (4.1), notamos que podemos definir tudo a respeito de $\boldsymbol{\beta}^{(i)}$ se conhecermos a distribuição $\boldsymbol{\alpha}^{(i)}$ correspondente e a taxa $R^{(i)}$ do código. A partir de (4.1) obtemos

$$\sum_{j=1}^{r^{(i)}} j \cdot \beta_j^{(i)} = \frac{1}{\left(1 - R^{(i)}\right)} \sum_{j=1}^{\ell^{(i)}} j \cdot \alpha_j^{(i)}. \tag{5.4}$$

Supondo $\boldsymbol{\alpha}^{(i)} = \left(\alpha_2^{(i)}, \alpha_3^{(i)}, \ldots, \alpha_{\ell^{(i)}}^{(i)}\right)$ conhecido, o lado direito da equação (5.4) é uma constante, cujo valor é igual a $\bar{r}^{(i)}$, i.e., o grau médio dos nós de paridade. Como estamos usando um $\boldsymbol{\beta}^{(i)}$ simplificado que considera apenas os graus $r^{(i)} - 1$ e $r^{(i)}$, podemos escrever

$$\beta_{r^{(i)}-1}^{(i)}(r^{(i)} - 1) + \beta_{r^{(i)}}^{(i)} r^{(i)} = \bar{r}^{(i)}. \tag{5.5}$$

Sabemos também que

$$\beta^{(i)}_{r^{(i)}-1} + \beta^{(i)}_{r^{(i)}} = 1. \tag{5.6}$$

Isto significa que temos duas equações, (5.5) e (5.6), e três variáveis, $r^{(i)}$, $\beta^{(i)}_{r^{(i)}-1}$ e $\beta^{(i)}_{r^{(i)}}$. Por termos apenas dois graus consecutivos para $\boldsymbol{\beta}^{(i)}$ sabemos ainda que

$$\left\lceil \bar{r}^{(i)} \right\rceil = r^{(i)}. \tag{5.7}$$

Com base em (5.6) e (5.7), o parâmetro $\beta^{(i)}_r$ pode ser deduzido da seguinte forma:

$$\begin{aligned}
&\beta^{(i)}_{r^{(i)}-1}\left(r^{(i)}-1\right) + \beta^{(i)}_{r^{(i)}} r^{(i)} = \bar{r}^{(i)} \\
&(1-\beta^{(i)}_{r^{(i)}})(\left\lceil \bar{r}^{(i)} \right\rceil - 1) + \beta^{(i)}_{r^{(i)}} \left\lceil \bar{r}^{(i)} \right\rceil = \bar{r}^{(i)} \\
&\beta^{(i)}_{r^{(i)}} = \bar{r}^{(i)} - \left\lceil \bar{r}^{(i)} \right\rceil + 1.
\end{aligned} \tag{5.8}$$

Outra simplificação muito útil é restringir os graus de distribuições $\boldsymbol{\alpha}^{(i)}$ a apenas três valores, i.e., concentrar a massa de probabilidade apenas nos graus 2, 3 e $\ell^{(i)}$. Nota-se em vários trabalhos que códigos construídos com base nesta restrição possuem bom desempenho [26, 13]. Admitimos, portanto, que

$$\alpha^{(i)}_2 + \alpha^{(i)}_3 + \alpha^{(i)}_{\ell^{(i)}} = 1.$$

Só consideramos até agora o aspecto multinível dos códigos. No entanto, para permitir que os códigos $\mathbf{C}^{(i)}$ dos canais equivalentes sejam aninhados, tal que $\mathbf{C}^{(0)} \subseteq \mathbf{C}^{(1)} \subseteq \ldots \subseteq \mathbf{C}^{(L-1)}$, é necessário preencher as restrições de aninhamento das construções D e D′.

Restringindo-nos a apenas dois códigos $\mathbf{C}^{(0)}$ e $\mathbf{C}^{(1)}$, as relações ge-

néricas de aninhamento [1] são

$$\beta_j^{(0)} \geq \frac{1-R^{(1)}}{1-R^{(0)}} \cdot \beta_j^{(1)} \quad \text{para } j = r^{(1)}-1, \ldots, r^{(1)}$$
$$\sum_{k=j}^{\ell^{(0)}} \alpha_k^{(0)} \geq \sum_{k=j}^{\ell^{(1)}} \alpha_k^{(1)} \quad \text{para } j = 2, 3, \ell^{(1)}.$$

Incorporamos a condição de dois graus consecutivos para $\boldsymbol{\beta}^{(0)}$ e $\boldsymbol{\beta}^{(1)}$. E para facilitar o aninhamento dos dois códigos e o processo de otimização adotamos uma nova simplificação: consideramos tanto para $\boldsymbol{\beta}^{(0)}$ como para $\boldsymbol{\beta}^{(1)}$ o mesmo grau máximo r, i.e., consideramos os mesmos graus $r-1$ e r para as duas distribuições. Essa consideração cria subotimalidade, pois restringimos os graus de liberdade do problema e, consequentemente, diminuímos o espaço de busca das possíveis distribuição. No entanto, apresenta grandes vantagens implementacionais e agiliza a convergência do algoritmo de otimização. Com base nestas simplificações temos as seguintes relações entre as distribuições:

$$\beta_r^{(0)} \geq \frac{1-R^{(1)}}{1-R^{(0)}} \cdot \beta_r^{(1)} \tag{5.9}$$
$$\beta_{r-1}^{(0)} \geq \frac{1-R^{(1)}}{1-R^{(0)}} \cdot \beta_{r-1}^{(1)}. \tag{5.10}$$

Desenvolvemos relações a partir das desigualdades em (5.9) e (5.10). Podemos reescrever (5.9) como

$$\beta_r^{(1)} \leq \frac{1-R^{(0)}}{1-R^{(1)}} \cdot \beta_r^{(0)}. \tag{5.11}$$

Usamos novamente o fato que $\beta_{r-1}^{(0)} + \beta_r^{(0)} = 1$ e que $\beta_{r-1}^{(1)} + \beta_r^{(1)} = 1$, com o qual podemos reescrever (5.10) como

$$1-\beta_r^{(1)} \leq \frac{1-R^{(0)}}{1-R^{(1)}} \cdot \left(1-\beta_r^{(0)}\right)$$
$$\beta_r^{(1)} \geq 1 - \frac{1-R^{(0)}}{1-R^{(1)}} \cdot \left(1-\beta_r^{(0)}\right). \tag{5.12}$$

Combinando (5.11) e (5.12), temos

$$1 - \frac{1 - R^{(0)}}{1 - R^{(1)}} \cdot \left(1 - \beta_r^{(0)}\right) \leq \beta_r^{(1)} \leq \frac{1 - R^{(0)}}{1 - R^{(1)}} \cdot \beta_r^{(0)}. \tag{5.13}$$

Retomando (5.8) e (5.4), podemos reexpressar $\beta_r^{(1)}$ da seguinte forma:

$$\beta_r^{(1)} = \frac{2 \cdot \alpha_2^{(1)} + 3 \cdot \alpha_3^{(1)} + \ell^{(1)} \cdot \alpha_{\ell^{(1)}}^{(1)}}{\left(1 - R^{(1)}\right)} - (r - 1)\,. \tag{5.14}$$

Então:

$$\begin{aligned}
&1 - \frac{1 - R^{(0)}}{1 - R^{(1)}} \cdot \left(1 - \beta_r^{(0)}\right) \leq \frac{2 \cdot \alpha_2^{(1)} + 3 \cdot \alpha_3^{(1)} + \ell^{(1)} \cdot \alpha_{\ell^{(1)}}^{(1)}}{1 - R^{(1)}} - \\
&- (r - 1) \leq \frac{\beta_r^{(0)} \left(1 - R^{(0)}\right)}{1 - R^{(1)}} \\
&\left(1 - R^{(1)}\right) \cdot r - \left(1 - R^{(0)}\right) \left(1 - \beta_r^{(0)}\right) \leq 2 \cdot \alpha_2^{(1)} + 3 \cdot \alpha_3^{(1)} + \\
&+ \ell^{(1)} \cdot \alpha_{\ell^{(1)}}^{(1)} \leq \left(1 - R^{(0)}\right) \cdot \beta_r^{(0)} + (r - 1) \left(1 - R^{(1)}\right).
\end{aligned} \tag{5.15}$$

Sabemos de [28] que $\bar{\ell}^{(1)} = 2 \cdot \alpha_2^{(1)} + 3 \cdot \alpha_3^{(1)} + \ell^{(1)} \cdot \alpha_{\ell^{(1)}}^{(1)}$ e substituímos a expressão à direita da segunda desigualdade em (5.15) por z, tal que

$$z + R^{(0)} - R^{(1)} \leq \bar{\ell}^{(1)} \leq z, \tag{5.16}$$

onde $z = (1 - R^{(0)}) \cdot \beta_r^{(0)} + (r - 1)\left(1 - R^{(1)}\right)$.

Seguimos a mesma abordagem para $\beta_r^{(0)}$:

$$\begin{aligned}
&\beta_r^{(0)} \geq \frac{1 - R^{(1)}}{1 - R^{(0)}} \cdot \beta_r^{(1)} \\
&\beta_r^{(0)} \leq 1 - \frac{1 - R^{(1)}}{1 - R^{(0)}} \cdot \left(1 - \beta_r^{(1)}\right) \\
&\frac{1 - R^{(1)}}{1 - R^{(0)}} \cdot \beta_r^{(1)} \leq \beta_r^{(0)} \leq 1 - \frac{1 - R^{(1)}}{1 - R^{(0)}} \cdot \left(1 - \beta_r^{(1)}\right).
\end{aligned}$$

Contudo, tal como em (5.14), podemos escrever $\beta_r^{(0)}$ de outra forma:

$$\beta_r^{(0)} = \frac{2 \cdot \alpha_2^{(0)} + 3 \cdot \alpha_3^{(0)} + \ell^{(0)} \cdot \alpha_{\ell^{(0)}}^{(0)}}{1 - R^{(0)}} - (r-1)\,.$$

Obtemos agora um par de relações semelhantes às de (5.15):

$$\begin{aligned} &\left(1 - R^{(1)}\right) \cdot \beta_r^{(1)} + \left(1 - R^{(0)}\right)(r-1) \leq \bar{\ell}^{(0)} \leq \\ &\leq \left(1 - R^{(0)}\right) \cdot r - \left(1 - R^{(1)}\right)\left(1 - \beta_r^{(1)}\right) \\ &z' + R^{(0)} - R^{(1)} \leq \bar{\ell}^{(0)} \leq z', \end{aligned} \tag{5.17}$$

onde $z' = \left(1 - R^{(0)}\right) \cdot r - \left(1 - R^{(1)}\right)\left(1 - \beta_r^{(1)}\right)$.

Podemos simplificar as desigualdades encontradas para $\bar{\ell}^{(0)}$ e $\bar{\ell}^{(1)}$. Analisando a desigualdade esquerda de (5.15), torna-se válido desprezar $\beta_r^{(0)} \cdot \left(1 - R^{(0)}\right)$, já que $0 \leq \beta_r^{(0)} \leq 1$ e $0 \leq \left(1 - R^{(0)}\right) \leq 1$. Ou seja, $\beta_r^{(0)} \cdot \left(1 - R^{(0)}\right) \geq 0$ e o seu módulo $|\beta_r^{(0)} \cdot \left(1 - R^{(0)}\right)|$ é pequeno quando comparado com o resto da expressão. Concluímos, portanto, que

$$\bar{\ell}^{(1)} > (r-1)\left(1 - R^{(1)}\right) + \left(R^{(0)} - R^{(1)}\right). \tag{5.18}$$

Analisando agora a desigualdade esquerda em (5.17) observamos que podemos proceder como antes e desprezar o termo $\beta_r^{(1)} \cdot \left(1 - R^{(1)}\right)$ já que $0 \leq \beta_r^{(1)} \cdot \left(1 - R^{(1)}\right) \leq 1$ e possui módulo bem inferior aos outros termos da expressão. Podemos, então, escrever

$$\bar{\ell}^{(0)} > (r-1)\left(1 - R^{(0)}\right). \tag{5.19}$$

Com base em (5.18), em (5.19) e no fato que $R^{(0)} \leq R^{(1)}$ (por consequência da regra de projeto de capacidade, da regra de cálculo para as capacidades equivalentes e do Mapeamento Ungerböck adotado), observamos que $\bar{\ell}^{(0)} > \bar{\ell}^{(1)}$. Como pelas nossas simplificações tanto $\bar{\ell}^{(0)}$ e $\bar{\ell}^{(1)}$ dependem apenas dos graus $2, 3$ e $\ell^{(0)}$ ou $\ell^{(1)}$ (respectivamente), concluímos que

$$\ell^{(0)} > \ell^{(1)}.$$

Incluímos agora o uso apenas dos graus 2, 3 e $\ell^{(i)}$ para $\boldsymbol{\alpha}^{(i)}$, $i = 0, 1$.

Com isso, podemos escrever as seguintes relações entre as distribuições $\boldsymbol{\alpha}^{(0)}$ e $\boldsymbol{\alpha}^{(1)}$:

$$\alpha_2^{(0)} + \alpha_3^{(0)} + \alpha_{\ell^{(0)}}^{(0)} \geq \alpha_2^{(1)} + \alpha_3^{(1)} + \alpha_{\ell^{(1)}}^{(1)} \tag{5.20}$$

$$\alpha_3^{(0)} + \alpha_{\ell^{(0)}}^{(0)} \geq \alpha_3^{(1)} + \alpha_{\ell^{(1)}}^{(1)} \tag{5.21}$$

$$\alpha_{\ell^{(0)}}^{(0)} \geq \alpha_{\ell^{(1)}}^{(1)}. \tag{5.22}$$

Admitindo $\boldsymbol{\alpha}^{(0)}$ e $R^{(0)}$ conhecidos, calculamos $\boldsymbol{\beta}^{(0)}$ e r. Como discutimos anteriormente, queremos que o grau máximo r seja igual para $\boldsymbol{\beta}^{(0)}$ e $\boldsymbol{\beta}^{(1)}$. Se supormos $\beta_r^{(1)}$ como um grau de liberdade do sistema, podemos definir uma distribuição $\boldsymbol{\alpha}^{(1)}$ e encontrar à partir dela o r buscado. Para isso, escolhemos apenas a variável $\alpha_{\ell^{(1)}}^{(1)}$ de $\boldsymbol{\alpha}^{(1)}$ aleatoriamente. A variável $\alpha_2^{(1)}$ depende da restrição de soma unitária para as componentes de $\boldsymbol{\alpha}^{(1)}$: $\alpha_2^{(1)} = 1 - \alpha_3^{(1)} - \alpha_{\ell^{(1)}}^{(1)}$. Quanto à componente $\alpha_3^{(1)}$, ela é obtida à partir de (5.14) segundo o desenvolvimento a seguir:

$$\beta_r^{(1)} = \frac{2 \cdot \alpha_2^{(1)} + 3 \cdot \alpha_3^{(1)} + \ell^{(1)} \cdot \alpha_{\ell^{(1)}}^{(1)}}{(1 - R^{(1)})} - (r-1)$$

$$(1 - R^{(1)}) \left[\beta_r^{(1)} + (r-1)\right] = 2 \cdot \alpha_2^{(1)} + 3 \cdot \alpha_3^{(1)} + \ell^{(1)} \cdot \alpha_{\ell^{(1)}}^{(1)}.$$

Mas como $\alpha_2^{(1)} = 1 - \alpha_3^{(1)} - \alpha_{\ell^{(1)}}^{(1)}$:

$$\alpha_3^{(1)} = (1 - R^{(1)}) \left[\beta_r^{(1)} + (r-1)\right] - \alpha_{\ell^{(1)}}^{(1)} \left(\ell^{(1)} - 2\right) - 2.$$

E como sabemos de (5.4) que $(1 - R^{(1)}) \left[\beta_r^{(1)} + (r-1)\right] = \bar{r}^{(1)}$, temos

$$\alpha_3^{(1)} = \bar{r}^{(1)} - \alpha_{\ell^{(1)}}^{(1)} \left(\ell^{(1)} - 2\right) - 2. \tag{5.23}$$

Adaptamos, por fim, as restrições de fronteira sobre as frações das distribuições de graus para os canais equivalentes. Em ambas perspectivas, as frações obedecem as mesmas limitações:

$$0 \leq \alpha_j^{(i)}, \lambda_j^{(i)} \leq 1$$

$$0 \leq \beta_j^{(i)}, \rho_j^{(i)} \leq 1.$$

5.4.2 Adaptações à *Differential Evolution* para Códigos Aninhados

Para a otimização de um par de códigos aninhados $\mathbf{C}^{(0)} \subseteq \mathbf{C}^{(1)}$ geramos inicialmente uma população de pares de vetores aninhados. A seguir aproveitamos as definições e deduções da subseção anterior para desenvolver uma metodologia de criação de vetores iniciais possíveis para o nosso problema:

(*i*) Escolhemos graus máximos $\ell^{(0)}$ para $\boldsymbol{\alpha}^{(0)}$ e $\ell^{(1)}$ para $\boldsymbol{\alpha}^{(1)}$ tal que $\ell^{(0)} > \ell^{(1)}$;

(*ii*) Escolhemos arbitrariamente frações $\left(\alpha_2^{(0)}, \alpha_3^{(0)}\right)$ (lembrando que consideramos a existência de apenas três graus - 2, 3 e $\ell^{(0)}$) de forma que $0 \leq \alpha_j^{(0)} \leq 1$, para $j = 2, 3$;

(*iii*) Calculamos $\alpha_{\ell^{(0)}}^{(0)}$ a partir de $\alpha_{\ell^{(0)}}^{(0)} = 1 - \left(\alpha_2^{(0)} + \alpha_3^{(0)}\right)$;

(*iv*) Verificamos se $0 \leq \alpha_{\ell^{(0)}}^{(0)} \leq 1$ e repetimos os passos 2 e 3 até que esta condição seja satisfeita, ou seja, até que tenhamos um vetor $\boldsymbol{\alpha}^{(0)} = \left(\alpha_2^{(0)}, \alpha_3^{(0)}, \alpha_{\ell^{(0)}}^{(0)}\right)$ válido;

(*v*) Encontramos a partir de $\boldsymbol{\alpha}^{(0)}$ e das equações (5.4), (5.5), (5.6), (5.7) e (5.8) as variáveis r e $\beta_r^{(0)}$, das quais obtemos a distribuição $\boldsymbol{\beta}^{(0)} = \left(\beta_{r-1}^{(0)}, \beta_r^{(0)}\right)$;

(*vi*) Adotamos $\beta_r^{(1)}$ como um grau de liberdade do problema. Escolhemos um valor arbitrário que respeita as restrições de (5.13);

(*vii*) Selecionamos arbitrariamente um valor para $\alpha_{\ell^{(1)}}^{(1)}$ que se encontra dentro do intervalo $[0, 1]$ e que satisfaz a condição (5.22). Calculamos $\alpha_3^{(1)}$ com base neste valor por meio da fórmula (5.23) e $\alpha_2^{(1)}$ por meio de $1 - \alpha_3^{(1)} - \alpha_{\ell^{(1)}}^{(1)}$;

(*viii*) Verificamos se $\alpha_2^{(1)}$ e $\alpha_3^{(1)}$ pertencem ao intervalo $[0, 1]$ e se $\bar{\ell}^{(1)}$ respeita as relações em (5.16) e $\bar{\ell}^{(0)}$ as relações em (5.17). Caso contrário, repetimos todos passos a partir de 2.

Seja NP o número de membros de uma população. Após a criação dos vetores iniciais, separamos as populações nos conjuntos $\mathcal{X}^{(0)} =$

$\left\{\mathbf{x}^{(0,1)}, \mathbf{x}^{(0,2)}, \ldots, \mathbf{x}^{(0,\mathrm{NP})}\right\}$ e $\mathcal{X}^{(1)} = \left\{\mathbf{x}^{(1,1)}, \mathbf{x}^{(1,2)}, \ldots, \mathbf{x}^{(1,\mathrm{NP})}\right\}$ pertencentes aos níveis 0 e 1, respectivamente, ou, de forma equivalente, aos códigos correspondentes $\mathbf{C}^{(0)}$ e $\mathbf{C}^{(1)}$. Também definimos $D^{(i)}$, $i = 0, 1$, como o número de parâmetros dos vetores de cada conjunto, os quais contêm apenas as variáveis independentes de otimização. Os demais elementos são encontrados pelas relações de igualdade deduzidas anteriormente.

Expandimos os vetores $\mathbf{x}_j^{(i)}$ resolvendo as dependências, $i = 0, 1$ e $j = 1, \ldots, \mathrm{NP}$. Calculamos a função objetivo $f_{\mathrm{obj}}\left(\mathbf{x}^{(0,j)}, \mathbf{x}^{(1,j)}\right)$ (baseada na *Density Evolution* ou na GA) para cada par de vetores. Selecionamos como melhor par da primeira geração aquele com o menor valor de f_{obj}, o qual recebe a notação $\left(\mathbf{x}^{(0,\mathrm{best})}, \mathbf{x}^{(1,\mathrm{best})}\right)$.

Para todas as demais gerações aplicamos aos vetores $\mathbf{x}^{(0,j)}$ e $\mathbf{x}^{(1,j)}$ as etapas de mutação e recombinação da *Differential Evolution* (ver subseção 4.4.1). Notamos que aplicaremos equações e definições praticamente idênticas às da subseção 4.4.1 (vide (4.10) e (4.11)), apenas adaptando-as para diferenciar os vetores relacionados a canais equivalentes distintos. O índice i indica a procedência do vetor, i.e., se vem de $\mathcal{X}^{(0)}$ ou de $\mathcal{X}^{(1)}$. Deste modo, calculamos os vetores mutados $(\mathbf{v}^{(i,j)})$ segundo

$$\mathbf{v}_{G+1}^{(i,j)} = \mathbf{x}_G^{(i,\mathrm{best})} + F \cdot \left(\mathbf{x}_G^{(i,r_1)} - \mathbf{x}_G^{(i,r_2)} + \mathbf{x}_G^{(i,r_3)} - \mathbf{x}_G^{(i,r_4)}\right), \qquad (5.24)$$

onde, expandindo os vetores $\mathbf{x}_G^{(i,j)}$ e $\mathbf{v}_G^{(i,j)}$ em seus parâmetros componentes, temos

$$\mathbf{x}_G^{(i,j)} = \left(x_{1,G}^{(i,j)}, \ldots, x_{k,G}^{(i,j)}, \ldots, x_{D^{(i)},G}^{(i,j)}\right)$$
$$\mathbf{v}_G^{(i,j)} = \left(v_{1,G}^{(i,j)}, \ldots, v_{k,G}^{(i,j)}, \ldots, v_{D^{(i)},G}^{(i,j)}\right).$$

Para a descrição da recombinação seja

$$\mathbf{u}_G^{(i,j)} = \left(u_{1,G}^{(i,j)}, \ldots, u_{k,G}^{(i,j)}, \ldots, u_{D^{(i)},G}^{(i,j)}\right)$$

o vetor recombinado. Ele é obtido da seguinte forma:

$$u_{k,G+1}^{(i,j)} = \begin{cases} v_{k,G+1}^{(i,j)}, & \text{se } \gamma^{(i)}(k) \leq \mathrm{CR} \text{ ou } k = d^{(i)}(j) \\ x_{k,G}^{(i,j)}, & \text{caso contrário,} \end{cases}$$

onde $k = 1, \ldots, D^{(i)}$, $D^{(i)}$ sendo o número de parâmetros dos vetores $\in \mathcal{X}^{(i)}$.

A mutação e a recombinação desfazem o aninhamento existente entre o par de vetores original $\left(\mathbf{x}_G^{(0,j)}, \mathbf{x}_G^{(1,j)}\right)$. Através destas operações criamos de $\left(\mathbf{x}_G^{(0,j)}, \mathbf{x}_G^{(1,j)}\right)$ o par $\left(\mathbf{u}_{G+1}^{(0,j)}, \mathbf{u}_{G+1}^{(1,j)}\right)$. Mas $\mathbf{u}_{G+1}^{(0,j)}$ e $\mathbf{u}_{G+1}^{(1,j)}$ via de regra não estão aninhados. Para o processo de seleção consideramos todas as quatro possíveis combinações dos vetores originais com os recombinados:

$$\left\{\left(\mathbf{x}_G^{(0,j)}, \mathbf{x}_G^{(1,j)}\right), \left(\mathbf{x}_G^{(0,j)}, \mathbf{u}_{G+1}^{(1,j)}\right), \left(\mathbf{u}_{G+1}^{(0,j)}, \mathbf{x}_G^{(1,j)}\right), \left(\mathbf{u}_{G+1}^{(0,j)}, \mathbf{u}_{G+1}^{(1,j)}\right)\right\}.$$

Com exceção do primeiro par de vetores, devemos forçar o aninhamento dos demais pares. Modificamos suas respectivas distribuições de graus $\boldsymbol{\alpha}^{(i)} = \left(\alpha_2^{(i)}, \ldots, \alpha_{\ell^{(i)}}^{(i)}\right)$ e $\boldsymbol{\beta}^{(i)} = \left(\beta_{r-1}^{(i)}, \beta_r^{(i)}\right)$, $i = 0, 1$.

Para obter $\boldsymbol{\beta}^{(i)}$ ajustado, dado por $\boldsymbol{\beta}^{(i)'} = \left(\beta i'_{r-1}, \beta_r^{(i)'}\right)$, definimos o erro de aninhamento ϵ_β para $\beta_\rho^{(i)}$, $\rho = r-1, r$. Este erro corresponde à infração da condição de aninhamento por parte de um parâmetro da distribuição $\boldsymbol{\beta}^{(i)}$. A expressão para ϵ_β é dada por

$$\epsilon_\beta = \frac{1 - R^{(1)}}{1 - R^{(0)}} \cdot \left(\beta_\rho^{(1)} - \beta_\rho^{(0)}\right).$$

Os parâmetros $\beta_\rho^{(0)}$, ajustados para obedecer as condições de aninhamento (5.9) e (5.10), são dados pela equação

$$\beta_\rho^{(0)'} = \beta_\rho^{(0)} + \frac{\epsilon_\beta}{2}. \tag{5.25}$$

E os parâmetros $\beta_\rho^{(1)}$, igualmente ajustados para se conformar às restrições (5.9) e (5.10), são calculados por

$$\beta_\rho^{(1)'} = \beta_\rho^{(0)} - \left(\frac{\epsilon_\beta}{2} \cdot \frac{1}{\left(\frac{1-R^{(1)}}{1-R^{(0)}}\right)}\right). \tag{5.26}$$

Na verdade, pela relação $\beta_{r-1}^{(i)'} + \beta_r^{(i)'} = 1$, só precisamos ajustar uma das componentes.

Para distribuições $\boldsymbol{\alpha}$ só definimos o erro de aninhamento ϵ_α para os graus $\ell^{(0)}$ e $\ell^{(1)}$, porque $\alpha_2^{(1)}$ e $\alpha_3^{(1)}$ são parâmetros dependentes. Portanto, o erro ϵ_α corresponde à infração da condição de aninhamento explicitada em (5.22). Definimos a modificação $\alpha_{\ell^{(i)}}^{(i)'}$ de $\alpha_{\ell^{(i)}}^{(i)}$ como

$$\epsilon_\alpha = \alpha_{\ell^{(0)}}^{(0)} - \alpha_{\ell^{(1)}}^{(1)}$$

$$\alpha_{\ell^{(0)}}^{(0)'} = \alpha_{\ell^{(0)}}^{(0)} + \frac{\epsilon_\alpha}{2} \tag{5.27}$$

$$\alpha_{\ell^{(1)}}^{(1)'} = \alpha_{\ell^{(1)}}^{(1)} - \frac{\epsilon_\alpha}{2}. \tag{5.28}$$

Com as definições e relações anteriores definimos uma série de passos para o aninhamento dos vetores mutados e recombinados:

(*i*) Modificamos $\beta_r^{(0)}$ e $\beta_r^{(1)}$ associados aos pares de vetores em (5.24) (com a exceção do par original) por meio das equações (5.25) e (5.26). Consequentemente, como os parâmetros das distribuições somam a 1, obtemos também $\beta_{r-1}^{(0)}$ e $\beta_{r-1}^{(1)}$;

(*ii*) Modificamos $\alpha_{\ell^{(0)}}^{(0)}$ e $\alpha_{\ell^{(1)}}^{(1)}$ de acordo com as equações (5.27) e (5.28);

(*iii*) Calculamos $\bar{r}^{(i)}$ e encontramos $\alpha_3^{(i)}$, $i = 0, 1$, conforme (5.23);

(*iv*) Encontramos $\alpha_2^{(i)}$ através da relação $1 - \alpha_3^{(i)} - \alpha_{\ell^{(i)}}^{(i)}$;

(*v*) Verificamos se as variáveis $\beta_{r-1}^{(i)}$, $\beta_r^{(i)}$, $\alpha_2^{(i)}$, $\alpha_3^{(i)}$, $\alpha_{\ell^{(i)}}^{(i)} \in [0, 1]$. Caso alguma das variáveis não satisfaça esta exigência, repetimos os passos anteriores.

Desta forma, criamos um novo conjunto de pares, desta vez respeitando as restrições de aninhamento. São eles

$$\left\{ \left[\mathbf{x}_G^{(0,j)}, \mathbf{x}_G^{(1,j)} \right], \left[\left(\mathbf{x}_G^{(0,j)} \right)', \left(\mathbf{u}_G^{(1,j)} \right)' \right], \right.$$
$$\left. \left[\left(\mathbf{u}_G^{(0,j)} \right)', \left(\mathbf{x}_G^{(1,j)} \right)' \right], \left[\left(\mathbf{u}_G^{(0,j)} \right)', \left(\mathbf{u}_G^{(1,j)} \right)' \right] \right\}.$$

A notação linha indica que o vetor foi modificado para permitir o aninhamento. Daremos a um par genérico com aninhamento garantido para a próxima geração $G+1$ a notação $\bar{\mathbf{a}}_{G+1} = \left(\mathbf{a}_{G+1}^{(0,j)}, \mathbf{a}_{G+1}^{(1,j)}\right)$.

Dos quatro pares de vetores criados escolhemos para a próxima geração $G+1$ o par com o menor valor da função objetivo f_{obj}, i.e.,

$$\left(\mathbf{x}_{G+1}^{(0,j)}, \mathbf{x}_{G+1}^{(1,j)}\right) = \underset{\bar{\mathbf{a}}_{G+1}}{\operatorname{argmin}} \left(f_{\text{obj}}\left(\bar{\mathbf{a}}_{G+1}\right)\right).$$

Este processo modificado de seleção é o que gera a evolução, e consequente otimização, das distribuições de graus na *Differential Evolution*.

5.5 Construção

A construção de códigos LDPC em esquemas multinível também sofre adaptações. Apresentamos na seção 4.5 o algoritmo PEG para a construção de grafos. O PEG pode ser usado em conjunto com códigos multinível aninhados mas precisa ser adaptado para uma versão estendida, cujo desenvolvimento se encontra em [30]. Trata-se de um algoritmo PEG no qual os graus de nós de variável são definidos para os diferentes níveis da codificação multinível em função das restrições de aninhamento. Os nós de paridade elegíveis para conexões com os nós de variável também variam com o nível por causa do aninhamento. Para maiores detalhes sobre o PEG estendido recomendamos [30] como referência.

CAPÍTULO 6

PNC

A codificação de rede na camada física (*Physical-Layer Network Coding* - PNC) é uma estratégia de comunicação cooperativa que explora a interferência entre sinais em uma rede sem fio. A PNC funciona em um sistema com múltiplos nós transmitindo pacotes de informação codificada simultaneamente, tal que sinais transmitidos se sobrepõem de forma ruidosa em um canal de múltiplo acesso (MAC). O objetivo da PNC é extrair de uma combinação ruidosa de sinais uma equação linear dos pacotes de informação. Para atingir este objetivo, a PNC pode utilizar códigos de reticulado aninhados, os quais garantem a possibilidade de decodificação de combinações lineares de sinais.

Apresentamos neste capítulo o modelo do canal utilizado para a análise de esquemas PNC na seção 6.1, o procedimento de codificação em esquemas PNC na seção 6.2, as taxas computacionais de Nazer e Gastpar, que constituem um fundamento para o entendimento da decodificação, na seção 6.3, o procedimento de decodificação na seção 6.4 e as adaptações necessárias à decodificação via BP ocasionadas pelo ruído efetivo visto na saída de um esquema PNC na seção 6.5.

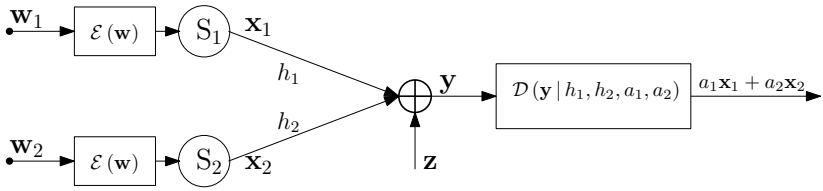


Figura 6.1: Esquema PNC

6.1 Modelo do Canal

Supomos uma rede com L usuários enviando sinais $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_L \in \mathbb{C}^n$, codificados a partir de mensagens $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_L$, respectivamente, e submetidos a desvanecimento Rayleigh de bloco, representado pelos coeficientes constantes de ganho do canal $h_1, h_2, \ldots, h_L \in \mathbb{C}$, e a ruído aditivo gaussiano circularmente simétrico $\mathbf{z}$ de distribuição $Z \sim \mathcal{CN}(\mathbf{0}, N_0\mathbf{I}_n)$. O sinal $\mathbf{y}$ recebido é dado por

$$\mathbf{y} = \sum_{\ell=1}^{L} h_\ell \mathbf{x}_\ell + \mathbf{z}. \tag{6.1}$$

Nosso modelo supõe transmissores idênticos. Consideramos as potências dos sinais de entrada do canal iguais, de maneira a permitir a definição da potência média por $P \triangleq \frac{1}{n} E\left[\|\mathbf{x}_\ell\|^2\right]$, onde $\ell = 1, 2, \ldots, L$. Definimos também $\text{SNR} \triangleq P/N_0$.

Definimos a taxa de mensagem R_{mes} com base na codificação multinível. Se considerarmos a divisão de blocos de mensagem em níveis i para $i = 0, \ldots, L-1$, cujos comprimentos correspondem a $k_0, \ldots, k_{L-1}$ bits de informação, podemos definir a taxa de mensagem (com base em [24]) da seguinte maneira:

$$\begin{aligned} R_{\text{mes}} &= \frac{1}{n} \sum_{i=0}^{L-1} k_i \\ R_{\text{mes}} &= R_{\text{mes}}^{(0)} + R_{\text{mes}}^{(1)} + \ldots + R_{\text{mes}}^{(L-1)}, \end{aligned} \tag{6.2}$$

onde $R_{\text{mes}}^{(i)}$ é a taxa de mensagem correspondente a cada nível, de valor dado por $R_{\text{mes}}^{(i)} = 2\frac{k_i}{n} = 2R_i$, e onde R_i é a taxa do código $\mathbf{C}_i$ do nível i.

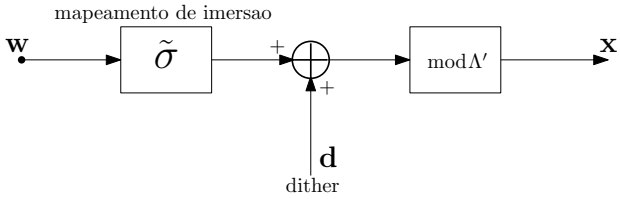


Figura 6.2: Codificação para o esquema PNC

Se codificarmos, transmitirmos e decodificarmos sinais independentemente em eixos ortogonais, em quadratura e em fase, ou seja, se realizarmos o levantamento (*lift*), podemos dobrar R_{mes}, de forma que

$$R_{\text{mes}}^{\text{lift}} = 2 \cdot R_{\text{mes}}. \tag{6.3}$$

Por fim, definimos o *throughput* (TP) como o complemento do *frame error rate* (FER) multiplicado pela taxa de mensagem R_{mes} do código:

$$\text{TP} = \text{R}_{\text{mes}}(1 - \text{FER}).$$

6.2 Codificação

Sejam $\Lambda \subseteq (\mathbb{Z}[\text{i}])^n \subseteq \mathbb{C}^n$ e $\Lambda' \subseteq \Lambda$ reticulados. Sabemos de [9] que o quociente Λ/Λ' é isomorfo ao espaço de mensagens W e ao código de reticulado aninhado $\mathbf{C}_\Lambda$. Seja $\sigma : \Lambda \to \mathbf{C}_\Lambda$ um mapeamento linear do reticulado Λ para um ponto do código de reticulado e $\tilde{\sigma} : \mathbf{C}_\Lambda \to \Lambda$ o mapeamento de imersão, tal que $\sigma(\tilde{\sigma}(\mathbf{w})) = \mathbf{w}$ para algum ponto $\mathbf{w} \in \mathbf{C}_\Lambda$.

Consideramos agora a codificação de uma mensagem $\mathbf{w} \in \mathbf{C}_\Lambda$ para $\mathbf{x} \in \mathbb{C}^n$ através do mapeamento de imersão $\tilde{\sigma}$. Este mapeamento leva $\mathbf{w}$ para um ponto $\lambda \in \Lambda$. Queremos que este ponto tenha a menor energia possível. Por isso o restringimos a $\mathcal{R}_{\Lambda'}$ e o deslocamos com um vetor de *dither* $\mathbf{d} \in \mathbb{C}^n$ apropriado. O nosso codificador $\mathcal{E} : \mathbf{C}_\Lambda \to \mathbb{C}^n$ é dado pela expressão $\mathcal{E}(\mathbf{w}) \triangleq [\tilde{\sigma}(\mathbf{w}) + \mathbf{d}] \cap \mathcal{R}_{\Lambda'} = [\tilde{\sigma}(\mathbf{w}) + \mathbf{d}] \bmod \Lambda'$.

6.3 Taxas Alcançáveis

A seguir apresentamos as taxas alcançáveis de Nazer e Gastpar. Este resultado é imprescindível para o processo de decodificação utilizado

no esquema PNC.

Teorema 6.1 (Teorema das Taxas Alcançáveis [22]). *Para um $\epsilon > 0$ qualquer, um n suficientemente grande e um primo $p \in \mathbb{Z}[\mathrm{i}]$ tal que $n/p \to 0$ quando $n \to \infty$, existe um esquema PNC linear em $\mathbb{Z}[i]$ com um espaço de mensagens W isomorfo ao código de reticulado aninhado $\mathbf{C}_\Lambda$ gerado com a Construção A a partir de um código $\mathbf{C}$ definido sobre o anel $\mathbb{Z}_p[\mathrm{i}]$, tal que, para um canal com L fontes e um vetor de ganhos $\mathbf{h} = (h_1, h_2, \ldots, h_L) \in \mathbb{C}^L$, uma equação linear descrita pelo vetor de coeficientes $\mathbf{a} = (a_1, a_2, \ldots, a_L) \in \mathbb{Z}[\mathrm{i}]^L$ pode ser decodificada com probabilidade de erro tão pequena quanto se deseje, i.e., com $P_e(\mathbf{h}, \mathbf{a}) < \epsilon$, desde que $R_{\mathrm{mes}} \leq R_{\mathrm{comp}}(\mathbf{h}, \mathbf{a})$, onde*

$$R_{\mathrm{comp}} \triangleq \max_{\alpha \in \mathbb{C}} \log_2 \left(\frac{\mathrm{SNR}}{\|\alpha \mathbf{h} - \mathbf{a}\|^2 \mathrm{SNR} + |\alpha|^2} \right),$$

e onde o valor ótimo de α da expressão acima é

$$\alpha_{\mathrm{opt}} = \frac{\mathbf{a}\mathbf{h}^{\mathrm{H}} \mathrm{SNR}}{\|\mathbf{h}\|^2 \mathrm{SNR} + 1},$$

o que resulta em

$$R_{\mathrm{comp}}(\mathbf{h}, \mathbf{a}) = \log_2 \left(\frac{\mathrm{SNR}}{\mathbf{a}\mathbf{M}\mathbf{a}^{\mathrm{H}}} \right),$$

com $\mathbf{M}$ dado por

$$\mathbf{M} = \mathrm{SNR}\mathbf{I}_L - \frac{\mathrm{SNR}^2}{\mathrm{SNR}\|h\|^2 + 1}\mathbf{h}^{\mathrm{H}}\mathbf{h}.$$

6.4 Decodificação

As mensagens codificadas são combinadas e distorcidas como em (6.1). A PNC tem por meta decodificar a partir de $\mathbf{y}$ uma combinação linear das mensagens originais: $\mathbf{w} = \sum_{\ell=1}^{L} a_\ell \mathbf{w}_\ell$, onde $a_1, a_2, \ldots, a_L$ são coeficientes $\in \mathbb{Z}[i]$. Dado os vetores $\mathbf{h} = (h_1, h_2, \ldots, h_L) \in \mathbb{C}^L$ e $\mathbf{a} = (a_1, a_2, \ldots, a_L) \in \mathbb{Z}[\mathrm{i}]^L$ (tal como já vistos no teorema 6.1), podemos decodificar $\tilde{\mathbf{w}}$ a partir de $\mathbf{y}$, onde $\tilde{\mathbf{w}}$ é a estimativa de $\mathbf{w}$. O

decodificador pode ser expresso como $\tilde{\mathbf{w}} = \mathcal{D}(\mathbf{y}|\mathbf{h}, \mathbf{a})$.

Seguimos [9] e definimos uma operação de decodificação em três passos. Multiplicamos o sinal recebido $\mathbf{y}$ por um escalar $\alpha \in \mathbb{C}$, um fator de escalonamento obtido em função de $\mathbf{h}$ e $\mathbf{a}$, cujo intuito é aproximar os coeficientes de ganho do canal em $\mathbf{h}$ dos coeficientes lineares em $\mathbf{a}$ [9]. Esta etapa tem uma estreita relação com o compromisso existente entre auto-interferência (a diferença entre $\mathbf{a} \in \mathbb{Z}[\mathrm{i}]^L$ e $\mathbf{h} \in \mathbb{C}^L$) e o ruído gaussiano [9]. Esta relação e o próprio funcionamento do decodificador $\mathcal{D}$ podem ser melhor explicados através dos conceitos de canal e ruído equivalentes. Observando atentamente a operação de escalonamento percebemos que

$$\alpha\mathbf{y} = \sum_{\ell=1}^{L} \mathbf{a}_\ell \mathbf{x}_\ell + \sum_{\ell=1}^{L} (\alpha\mathbf{h}_\ell - \mathbf{a}_\ell)\, \mathbf{x}_\ell + \alpha\mathbf{z} = \underbrace{\sum_{\ell=1}^{L} \mathbf{a}_\ell \mathbf{x}_\ell}_{\lambda \in \Lambda} + \mathbf{z}', \tag{6.4}$$

onde λ é um ponto no reticulado Λ e $\mathbf{z}'$ é o ruído efetivo, uma mistura de ruído gaussiano e auto-interferência.

Já que incluímos no nosso codificador o efeito de deslocamento dos pontos do reticulado através dos vetores de *dither*, é necessário deslocá-los de volta ao reticulado para realizar a decodificação:

$$\mathbf{y}' = \alpha\mathbf{y} - \sum_{\ell=1}^{L} \mathbf{a}_\ell \mathbf{d}_\ell. \tag{6.5}$$

O resultado $\mathbf{y}'$ acima é a entrada do quantizador Q_Λ. O quantizador elimina o ruído efetivo $\mathbf{z}'$, assim como eliminaria um outro ruído aditivo qualquer, como, por exemplo, o ruído aditivo gaussiano. Como consequência, notamos que a decodificação de reticulado fornecida nesta seção para esquemas PNC vale também para canais ponto a ponto e que, fora a natureza do ruído e as devidas adaptações da potência do ruído para uma dada SNR, o processo de decodificação para um canal ponto a ponto com ruído gaussiano é igual. Discutiremos mais a respeito desta consideração na próxima seção, na qual abordaremos em detalhes as modificações necessárias para adaptar a decodificação por BP para o esquema PNC.

Por fim, o último passo no procedimento de decodificação é a as-

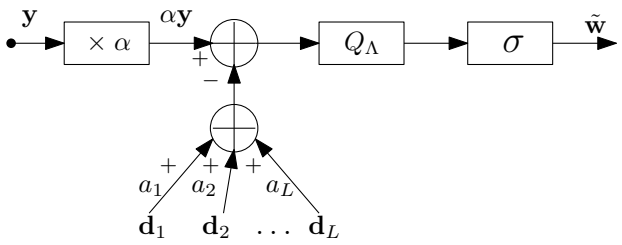


Figura 6.3: Passos seguidos pelo decodificador de reticulado no esquema PNC

sociação dos pontos de reticulado ao espaço de mensagens W, o qual notamos na seção 2.3 ser isomorfo ao código de reticulado aninhado $\mathbf{C}_\Lambda$ através do mapeamento linear $\sigma : \Lambda \to W$.

Os passos de decodificação descritos podem ser resumidos pela expressão

$$\mathcal{D}\left(\mathbf{y}|\mathbf{h},\mathbf{a}\right) \triangleq \sigma\left(Q_\Lambda\left(\alpha\mathbf{y} - \sum_{\ell=1}^{L}\mathbf{a}_\ell\mathbf{d}_\ell\right)\right). \tag{6.6}$$

Eles podem ser observados igualmente na figura 6.3.

6.5 Decodificação de Sinais em Esquemas PNC via BP

Repetimos por conveniência a seguir a equação (6.4) do canal equivalente para o esquema PNC:

$$\alpha\mathbf{y} = \sum_{\ell=1}^{L}\mathbf{a}_\ell\mathbf{x}_\ell + \sum_{\ell=1}^{L}\left(\alpha\mathbf{h}_\ell - \mathbf{a}_\ell\right)\mathbf{x}_\ell + \alpha\mathbf{z} = \underbrace{\sum_{\ell=1}^{L}\mathbf{a}_\ell\mathbf{x}_\ell}_{\lambda\in\Lambda} + \mathbf{z}'.$$

Identificamos a partir da equação acima que o ruído efetivo, representado por $\mathbf{z}' = (z'_1, z'_2, \ldots, z'_n) \in \mathbb{C}^n$, é dado por

$$\mathbf{z}' = \sum_{\ell=1}^{L}\left(\alpha\mathbf{h}_\ell - \mathbf{a}_\ell\right)\mathbf{x}_\ell + \alpha\mathbf{z}. \tag{6.7}$$

Como já havíamos observado na seção anterior, este ruído contém uma parcela proveniente do erro de estimação dos coeficientes de ganho do canal e outra do ruído deste canal. Portanto, mesmo que o ruído seja

gaussiano, o ruído equivalente não o é.

Buscamos a variância de cada componente escalar z'_j de $\mathbf{z}'$, para $j = 1, 2, \ldots, n$. Assumimos que os sinais $\mathbf{x}_i$, $i = 1, ..., L$, são estatisticamente independentes e possuem componentes $x_{i,j}$, $j = 1, ..., n$, com a mesma energia média, dada por $\mathrm{E}[|x_{i,j}|^2] = P$. O vetor de ruído, como um processo estocástico discreto com variáveis aleatórias i.i.d., possui componentes com variâncias $var(z_1) = var(z_2) = \cdots = var(z_n) = N_0$. Podemos então ignorar o índice j referente às componentes e denotar para um j qualquer a componente z' do vetor do ruído equivalente, conforme

$$var\left(z'\right) = P \cdot \sum_{\ell=1}^{L} \left(\alpha h_\ell - a_\ell\right)^2 + |\alpha|^2 N_0.$$

Definimos o vetor $\alpha\mathbf{h} - \mathbf{a} = (\alpha h_1 - a_1, \alpha h_2 - a_2, \ldots, \alpha h_L - a_L)$. A equação anterior então se transforma em

$$var\left(z'\right) = \left(\|\alpha\mathbf{h} - \mathbf{a}\|^2 \,\mathrm{SNR} + |\alpha|^2\right) N_0.$$

Considerando uma decodificação multi-estágio com decodificadores BP a cada nível, faz-se necessário avaliar a expressão das LLRs de entrada. Por simplicidade, para o cálculo de $\mathsf{L}^{(i)}\left(y\right)$ consideramos que os canais equivalentes após os cancelamentos de níveis anteriores e as operações mod $2^{(i+1)}\mathbb{Z}^n$ efetuados pelo decodificador multi-estágio são aproximadamente gaussianos binários (BAWGNC), o que permite utilizar a fórmula da LLR deste canal. Supondo um ruído genérico de variância σ^2 e um símbolo recebido y, temos que [28]

$$\mathsf{L}_{\mathrm{BAWGNC}}(y) = \frac{2y}{\sigma^2}.$$

Com esta aproximação estamos desprezando ainda os efeitos que múltiplos usuários ocasionam sobre o cálculo da LLR, o que justificamos pela complexidade do cálculo envolvido.

Assumindo o ruído N_0 para o canal físico, calculamos a LLR $\mathsf{L}^{(i)}$ do canal equivalente i com o ruído efetivo e com o ruído visto pelo canal equivalente (as operações modulares dividem o ruído por potências de 2, como pode ser visto na seção 3.1, em particular na fórmula (3.2)). Usamos na definição o símbolo $y^{(i)}$ definido na equação (3.1) para explicar a decodificação multi-estágio. Ele simboliza o sinal re-

cebido menos as influências dos canais anteriores. Assim, o cálculo da LLR em presença de ruído efetivo se torna

$$\mathrm{L}^{(i)} = \frac{2\left(y^{(i)}/2 \mod 2\right)}{\left(\|\alpha\mathbf{h} - \mathbf{a}\|^2 \,\mathrm{SNR} + |\alpha|^2\right) N_0/2^i}. \tag{6.8}$$

Desta forma, concluímos que a decodificação em um cenário PNC é idêntica ao caso gaussiano, exceto pelo cálculo da LLR dada por (6.8).

CAPÍTULO 7

Simulações e Resultados

Apresentamos neste capítulo os cenários de simulação e os resultados obtidos. Incluímos casos para o canal gaussiano e para o canal de múltiplo acesso (MAC), no qual adotamos um esquema cooperativo de comunicação.

7.1 Canal Gaussiano

Nesta seção detalhamos os cenários de simulação e os resultados obtidos com o canal gaussiano .

Assumimos um canal sem memória com ruído aditivo gaussiano $\mathbf{Z} \sim \mathcal{N}(\mathbf{0}, N_0/2\,\mathbf{I})$. Sendo $\mathbf{x} \in \mathbb{R}^n$ um vetor de símbolos na entrada do canal, $\mathbf{z}$ o vetor de ruído e $\mathbf{y} \in \mathbb{R}^n$ o vetor de símbolos correspondente na saída, podemos representar o canal através da equação

$$\mathbf{y} = \mathbf{x} + \mathbf{z}. \tag{7.1}$$

Para a codificação usamos a proposta de codificação da subseção 3.3, e para a decodificação usamos a BP com entradas fornecidas pelas LLRs dos canais equivalentes, dadas por (5.1), em conjunto com a decodificação multi-estágio (3.3) da seção 3.1.

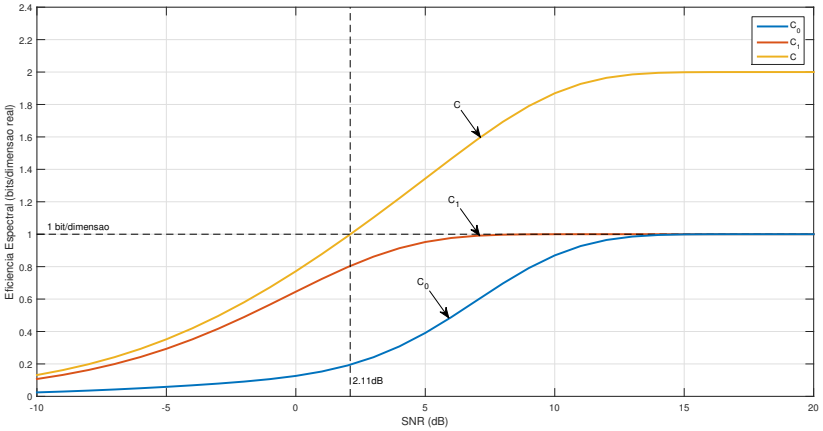


Figura 7.1: Capacidades equivalentes em função da SNR para os dois níveis de um esquema multinível com Particionamento de Ungerböck e modulação 4-PAM. A capacidade de 1 bit/dimensão real é obtida com $2,11$dB. Os símbolos C_0 e C_1 designam as capacidades equivalentes, enquanto que C corresponde à capacidade total do esquema multinível.

7.1.1 Resultados de Simulação

Nesta seção mencionamos apenas resultados obtidos com a Construção D′ e códigos LDPC.

Escolhemos tentar reproduzir os resultados de [13]. Utilizamos uma modulação 4-PAM e blocos de comprimento $n = 10000$. Selecionamos uma eficiência espectral de 1 bit/dimensão real (b/dim). Porém, ao invés do mapeamento Gray usado em [13], adotamos o Particionamento de Ungerböck. Como vimos na subseção 2.1.2 isto não altera a capacidade do esquema multinível, mas sim as capacidades dos canais equivalentes, calculadas segundo as equações em (2.5) e [35, 37], tal como observado no gráfico das capacidades equivalentes do esquema multinível com mapeamento Ungerböck e modulação 4-PAM da figura 7.1. Obtivemos $C^{(0)} \approx 0,19$ b/dim e $C^{(1)} \approx 0,81$ b/dim. Nas simulações otimizamos a probabilidade de erro com relação às distribuições de graus e às taxas. Os melhores projetos foram obtidos com taxas próximas às capacidades dos canais equivalentes, de acordo com a regra de projeto da subseção 2.1.2.

Garantimos que os parâmetros respeitam as restrições de otimiza-

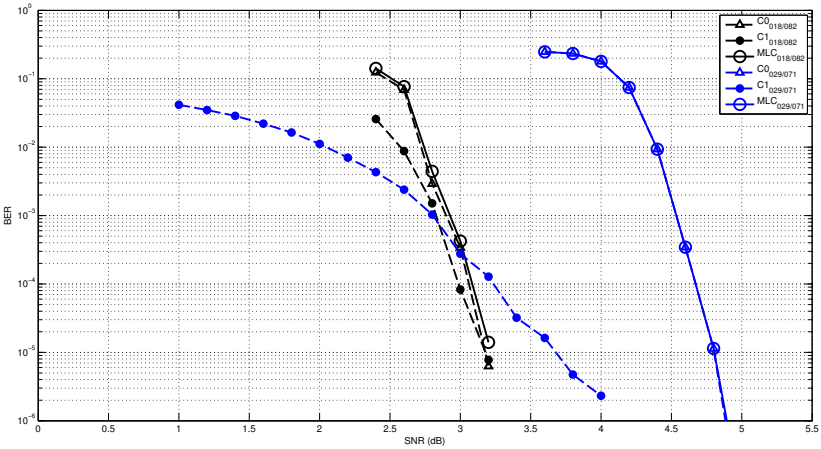


Figura 7.2: Comparação de desempenho de BER para esquemas multinível de dois níveis com blocos de comprimento $n = 10000$ em canal gaussiano. As curvas negras representam o esquema sem restrições de aninhamento com códigos de taxas $0,18$ b/dim e $0,82$ b/dim, e as azuis o esquema com restrições sobre as distribuições de graus e taxas $0,29$ b/dim e $0,71$ b/dim

ção citadas na subseção 5.4.1 por meio de barreiras. Penalizamos as violações de forma linearmente proporcional ao valor absoluto da violação. Por exemplo para um parâmetro λ, seja o valor otimizado $\tilde{\lambda}$. A penalização ξ_λ por barreira é dada por

$$\xi_\lambda = |\lambda - \tilde{\lambda}|.$$

Comparamos a curva de probabilidade de erro de bit de [13] com o nosso esquema na figura 7.2. Utilizamos um decodificador BP para 80 iterações. Observamos para uma SNR de 3.2dB uma probabilidade de erro de bit (BER) de aproximadamente 10^{-5} (ver curvas negras), próxima dos resultados de [13], que obteve resultados semelhantes para 3dB. Isto indica que a função objetivo dada pela soma das probabilidades de erro dos códigos é uma boa métrica de otimização.

Em seguida testamos esquemas para o mesmo cenário mas com distribuições de graus que obedecem as restrições de aninhamento da subseção 5.4.1.

Usando as hipóteses simplificadoras definidas no capítulo 5 para as

distribuições de graus, concluímos que distribuições que respeitam as condições de aninhamento não convergem para valores baixos da função objetivo para as taxas de projeto calculadas anteriormente. Construindo os códigos separadamente sem o uso da PEG estendida, conseguimos os melhores resultados de BER para distribuições de graus condicionadas pelo aninhamento e para uma eficiência espectral fixada em 1 b/dim usando códigos aninhados com as taxas $0,29$ b/dim para o nível 0 e $0,71$ b/dim para o nível 1. Na figura 7.2 apresentamos as probabilidades de erro correspondentes (curvas azuis). Observamos que o esquema atinge BER de 10^{-5} em torno de 4.8 dB.

Por último, criamos não só distribuições de graus adequadas ao aninhamento mas também construímos códigos de fato aninhados através do algoritmo PEG estendido. Constatamos que, dependendo das distribuições de grau $\boldsymbol{\alpha}$, este algoritmo gera matrizes de paridade com um número muito elevado de dependências lineares e que o código resultante tem baixo desempenho. As restrições naturais do aninhamento, o projeto de distribuições aproximado com a GA (principalmente no nível 0, que apresenta canal equivalente com pdf não gaussiana) e as premissas simplificadoras para as distribuições de grau adotadas no capítulo 5 degradam o desempenho. Além disso, o PEG parece também limitar a performance. Distribuições que preenchem as restrições de aninhamento e possuem bom desempenho com a construção não aninhada, degeneram quando aninhadas. Este foi o caso para o código com taxas $0,29$ b/dim e $0,71$ b/dim.

Os melhores códigos aninhados gerados e testados possuem taxas de $0,25$ b/dim e $0,75$ b/dim. O esquema resultante é sabidamente subótimo e atinge valores de 1% para a FER (taxa de erros de quadro) para um comprimento $n = 2000$ em um valor de SNR de 6.9dB. O melhor esquema com distribuições de graus condicionadas pelas restrições do aninhamento mas sem aninhamento propriamente dito (o esquema multinível com códigos componentes $\mathbf{C}^{(0)}$ e $\mathbf{C}^{(1)}$ de taxas $0,29$ b/dim e $0,71$ b/dim, respectivamente) obteve FER de 1% para $n = 2000$ em 5.1dB.

Apesar do resultado ruim, o esquema obtido com os códigos de taxas $0,25$ b/dim e $0,75$ b/dim foi o esquema prático com matriz de paridades de posto completo de melhor desempenho. Deste modo, foi escolhido para o uso nas análises do PNC.

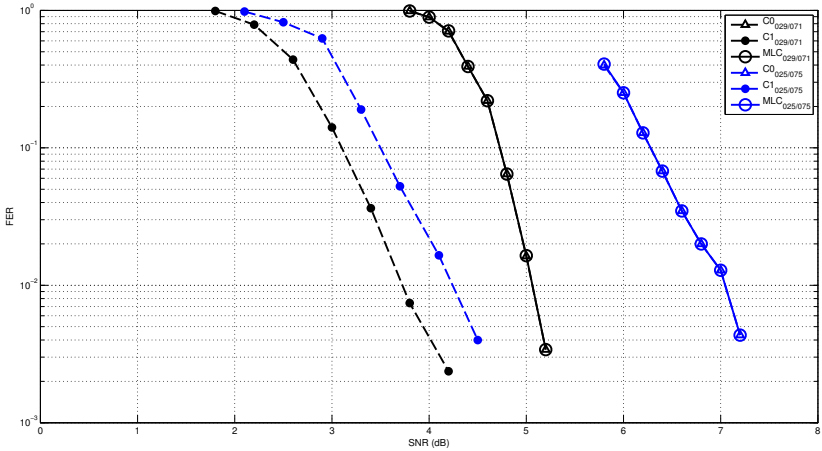


Figura 7.3: Comparação de desempenho de FER para esquemas multinível de dois níveis com blocos de comprimento $n = 2000$ em canal gaussiano para a análise do efeito do aninhamento. As curvas negras indicam o esquema com códigos de taxas $0,29$ b/dim e $0,71$ b/dim e distribuições de graus projetadas com aninhamento mas com transmissão independente, isto é sem a construção de matrizes de paridade aninhadas. As curvas azuis representam o esquema multinível com códigos de taxas $0,25$ b/dim e $0,75$ b/dim construído com matrizes de paridade de fato aninhadas.

Na tabela 7.1 apresentamos a lista de distribuições de graus usadas nas simulações e nos testes efetuados. Essas distribuições foram usadas tanto nas simulações do caso ponto a ponto quanto no caso do esquema PNC.

7.2 Canal MAC

Considerando dois usuários, expomos nesta seção simulações e resultados obtidos com o canal MAC e o esquema cooperativo *Physical-Layer Network Coding* (PNC).

Como visto na seção 6.1, o modelo do canal para o esquema PNC é similar ao modelo do canal gaussiano, diferindo deste apenas com relação ao valor do ruído efetivo. Podemos, portanto, tratar a decodificação de forma análoga, adaptando os cálculos das LLRs na entrada de decodificadores BP para a variância de $\mathbf{z}'$, dada por (6.8). Do ponto

A&T	$\alpha(x)$	$\beta(x)$
na, $0,18$	$0,6090x^2 + 0,3098x^3 + 0,0013x^{14} + 0,0004x^{16} + 0,0049x^{17} + 0,0724x^{18} + 0,0009x^{19} + 0,0005x^{20}$	$0.6181x^4 + 0.3819x^5$
na, $0,82$	$0,3662x^2 + 0,5026x^3 + 0,0008x^4 + 0,0009x^{15} + 0,0013x^{17} + 0,0036x^{18} + 0,12098x^{20}$	$0.3026x^{26}+0.6974x^{27}$
da, $0,29$	$0,7054x^2 + 0,2212x^3 + 0,0733x^{35}$	$0.4633x^6 + 0.5367x^7$
da, $0,71$	$0,9808x^2 + 0,0142x^3 + 0,0051x^4$	$0.0197x^6 + 0.9803x^7$
a, $0,25$	$0.3427x^2 + 0.5331x^3 + 0.1242x^{30}$	$0.9840x^8 + 0.0160x^9$
a, $0,75$	$0.9963x^2 + 0.0027x^3 + 0.0010x^4$	$0.9813x^8 + 0.0187x^9$

Tabela 7.1: Tabela com as distribuições de graus dos códigos utilizados neste trabalho. As siglas utilizadas na caracterização das distribuições significam: A&T - tipo de aninhamento e taxas; na - não aninhado; da - distribuição aninhada; a - aninhado. A notação $\alpha(x)$ representa a notação polinomial da distribuição de graus de variável, enquanto que $\beta(x)$ faz referência à distribuição de graus de paridade. A notação polinomial foi escolhido por sua sucintez, e a sua interpretação pode ser encontrada em [28].

de vista do decodificador, isto corresponde a aproximar o canal equivalente por um canal gaussiano, o que se justifica dado que o cálculo das LLRs verdadeiras em um cenário com PNC e modulação codificada é muito complexo.

Para aumentar a eficiência espectral usamos levantamento na codificação, ou seja, transmitimos sinais em eixos ortogonais, o que permite o uso das Construções de reticulado D e D′ em duas dimensões reais. Geramos, na verdade, constelações QAM com base nas constelações PAM dos códigos de reticulado.

7.2.1 Resultados de Simulação

Nesta subseção mostramos resultados de simulação para esquemas PNC criados com as construções D e D′. Apresentamos a seguir as especifi-

cações dos esquemas de codificação utilizados.

Realizamos simulações, primeiramente, com a Construção D em combinação com códigos convolucionais. Utilizamos modulações do tipo M-PAM, para $M = 2, 4, 8$. Elas foram então submetidas ao levantamento, dando origem a modulações M^2-QAM.

Com a Construção D criamos esquemas multinível equivalentes às modulações 4-QAM, 16-QAM e 64-QAM. Usamos nas simulações quadros com $n = 200$ símbolos reais por componente (em fase e em quadratura). Utilizamos nas estruturas dos três esquemas o mesmo código convolucional $\mathbf{C}_{\text{conv}}$ de taxa $0,5$ b/dim. Ele é dado por $[1 + \mathrm{D} + \mathrm{D}^3 + \mathrm{D}^4 + \mathrm{D}^6,\ 1 + \mathrm{D}^3 + \mathrm{D}^4 + \mathrm{D}^5 + \mathrm{D}^6]$ e possui cauda de comprimento 6.

Listamos a seguir os esquemas multinível desenvolvidos com a Construção D: 1) código gerado à partir de $\mathbf{C}_{\text{conv}}$ com taxa efetiva de $0,94$ bits por dimensão complexa (b/cdim); 2) código com $\mathbf{C}_{\text{conv}}$ no primeiro nível, código de taxa 1 b/dim no segundo e taxa efetiva total de 2.94 b/cdim; 3) código com $\mathbf{C}_{\text{conv}}$ no primeiro nível, acrescido de códigos de taxa 1 b/dim no segundo e terceiro níveis e taxa efetiva total perfazendo 4.94 b/cdim.

Em seguida, criamos um esquema com a Construção D′. Construímos um reticulado com os códigos LDPC aninhados com as taxas $0,25$ b/dim e $0,75$ b/dim da subseção anterior. A transmissão é feita com base em uma modulação 16-QAM, produzida com o levantamento de uma 4-PAM. A eficiência espectral é de 2 b/cdim. Usamos quadros com $n = 2000$ símbolos reais por componente ortogonal.

Em todos os esquemas simulados nesta seção as duas fontes de transmissão utilizam o mesmo código. Como a taxa de erro de quadro (FER) não é adequada para a comparação de códigos com diferentes taxas de mensagem R_{mes}, calculamos um outro parâmetro: o *throughput* (TP).

Atentamos que nas análises que se seguem utilizaremos para a comparação com os valores teóricos de N&G valores de simulação correspondentes à FER de 1%.

Cenário com Coeficientes Fixos

Neste cenário supomos um sistema no qual os coeficientes de ganho do canal são fixos e iguais a $\mathrm{h}_1 = -1.17 + \mathrm{i}2.15$ e $\mathrm{h}_2 = 1.25 - \mathrm{i}1.63$.

As linhas tracejadas verticais que aparecem nos gráficos correspondem ao valor mínimo teórico de SNR para alcançar um dado TP; ou

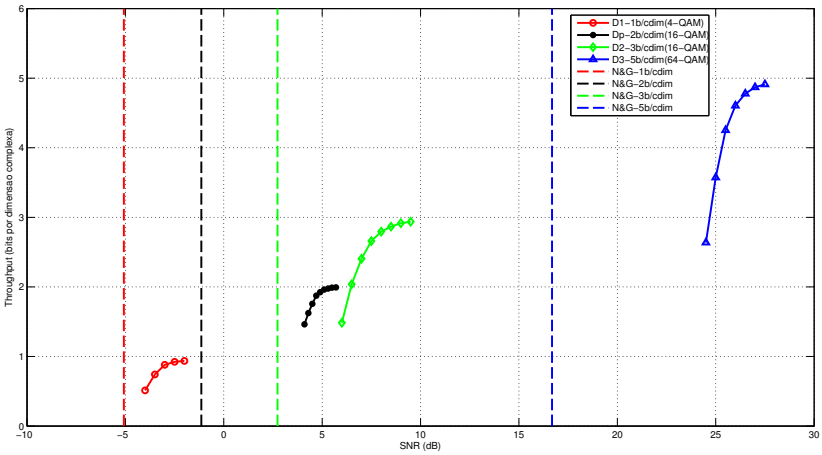


Figura 7.4: Comparação de TP de esquemas em um canal multi-usuário com coeficientes de ganho do canal fixos dados por $h_1 = 1.17 + \mathrm{i}\,2.15$ e $h_2 = 1.25 - \mathrm{i}\,1.63$. As curvas cheias correspondem aos códigos de reticulado desenvolvidos com as construções D e D′. As linhas verticais indicam as taxas computacionais de N&G para eficiências espectrais correspondentes às dos esquemas analisados.

seja, são as taxas computacionais de Nazer e Gastpar (N&G), dadas segundo as equações em [22].

Na figura 7.4 os esquemas criados com a Construção D apresentam as seguintes distâncias para as taxas computacionais de N&G: 1) 3.08dB para a 4-QAM; 2) 6.77dB para a 16-QAM; 3) $10,82$dB para a 64-QAM.

Também na figura 7.4 apresentamos a curva de throughput do código de reticulado LDPC. Colocamos esta curva em destaque, mostrando também em separado as componentes dos níveis 0 e 1, na figura 7.5. A distância da taxa computacional foi de 6.84dB para $\mathbf{C}^{(1)}$ e 3.74dB para $\mathbf{C}^{(0)}$. Vemos que o código do nível inferior prejudica o desempenho do esquema, visto que $\mathbf{C}^{(1)}$ consegue uma distância relativamente pequena em termos de SNR com relação à taxa computacional.

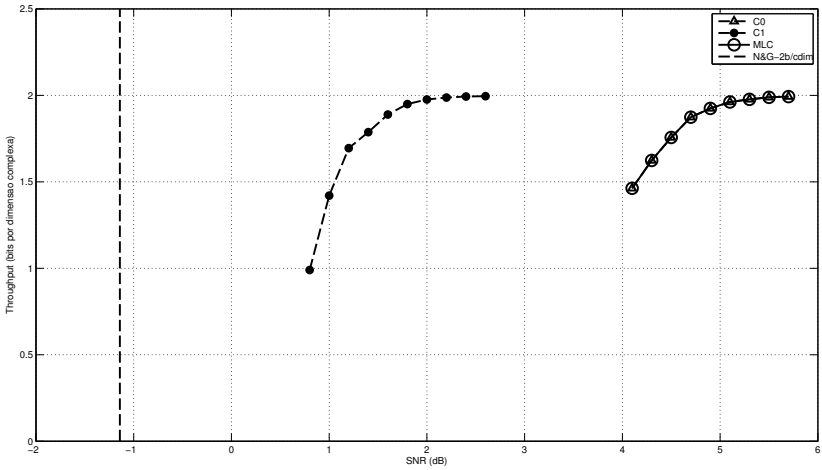


Figura 7.5: Particularização da figura anterior só para o código LDPC de reticulado. As curvas tracejadas representam os códigos componentes e a cheia o comportamento geral do esquema multinível. A curva vertical se refere à taxa computacional de N&G para eficiência espectral de 2 b/cdim.

Cenário com Desvanecimento Rayleigh

Neste segundo cenário assumimos um canal com desvanecimento de bloco modelado por uma distribuição Rayleigh.

As curvas tracejadas para os gráficos desta subseção representam o TP do esquema N&G para as taxas efetivas desejadas. Os resultados de N&G se baseiam em $\mathrm{R_{mes}} \geq \log_2\left(\mathrm{SNR}/\mathbf{aMa}^{\mathrm{H}}\right)$, i.e., representam probabilidades de *outage*.

Na figura 7.6 supomos desvanecimento Rayleigh. Atingimos distâncias com relação aos TPs de N&G de: 1) 3.5 dB para a 4-QAM; 2) 6.5 dB para a 16-QAM; 3) 7.5 dB para a 64-QAM.

Nesta mesma figura apresentamos, em particular, o desempenho do esquema de codificação de reticulado LDPC. Na figura 7.7 destacamos este esquema sob o mesmo cenário de desvanecimento. Incluímos adicionalmente o TP dos códigos componentes $\mathbf{C}^{(0)}$ e $\mathbf{C}^{(1)}$. O código $\mathbf{C}^{(1)}$ se distancia de 4dB de N&G, enquanto que $\mathbf{C}^{(0)}$ se distancia de mais de 8dB.

Observamos que o código $\mathbf{C}^{(1)}$ possui desempenho muito superior a

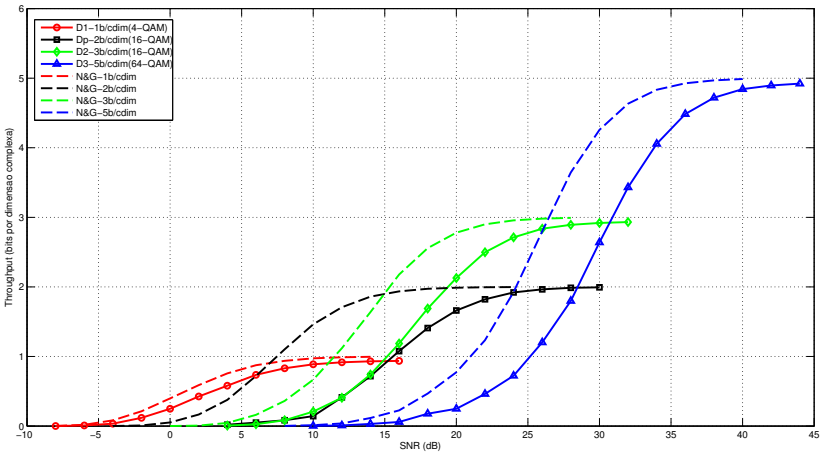


Figura 7.6: Comparação de TP entre esquemas de codificação em um canal MAC com desvanecimento Rayleigh. As curvas tracejadas correspondem ao esquema de transmissão de N&G, que representa probabilidades de *outage*.

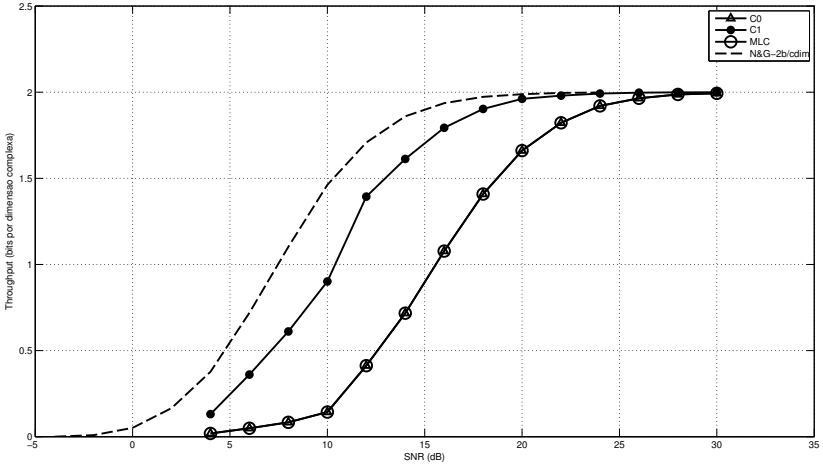


Figura 7.7: Neste gráfico mostramos em detalhes o desempenho de TP em um canal com desvanecimento Rayleigh para o código LDPC de reticulado, destacando a curva negra da figura 7.6. Incluímos neste gráfico também o TP dos códigos componentes.

$\mathbf{C}^{(0)}$, o qual limita o esquema. Como citamos na seção anterior, o código simulado não é ótimo. O comprimento curto de apenas 2000 símbolos, as restrições do aninhamento e o projeto aproximado de distribuições de graus, todos, resultaram na perda expressiva de desempenho. No entanto, os bons resultados de $\mathbf{C}^{(1)}$ apontam para a existência de uma grande margem de melhoria.

CAPÍTULO 8

Conclusão

A maior contribuição desta dissertação compreende o desenvolvimento de esquemas multinível de reticulado. Em particular, construímos esquemas com códigos LDPC unindo em um projeto os seguintes benefícios: uso de códigos poderosos capazes de se aproximar da capacidade do canal, construção de reticulados em $\mathbb{R}^n$, alta eficiência espectral e decodificação simples. A necessidade de códigos com estas características em esquemas PNC motivou o estudo. A PNC explora a estrutura linear de reticulados para converter relações entre sinais transmitidos em combinações lineares de mensagem, e é interessante que as taxas efetivas alcançadas em uma rede sem fio sejam altas.

Na tentativa de construir códigos com as características anteriores, criamos uma maneira simples de otimizar distribuições de graus para um esquema multinível com códigos aninhados. Modificamos ligeiramente a técnica de estimativa de desempenho *Density Evolution* para levar em conta as distintas funções densidade de probabilidade associadas associados aos canais equivalentes. Também alteramos a técnica de otimização *Differential Evolution* para adaptá-la às necessidades do projeto aninhado e à otimização simultânea de múltiplos códigos. Implementamos um otimizador modificado, capaz de gerar populações

de distribuições de graus iniciais aninhadas e com um procedimento de correção dos vetores mutados e recombinados para garantir a cada nova geração populações de pares apropriadamente aninhados. Criamos também um método de seleção aumentado, baseado na função objetivo combinada dos dois códigos.

Para a PNC especificamente, mostramos com a Construção D e códigos convolucionais a possibilidade de se obter alta eficiência espectral, o que demonstra a utilidade e a viabilidade do uso de esquemas multinível de reticulado. Em seguida, mostramos com a Construção D′ a possibilidade da construção de reticulados com códigos LDPC irregulares aninhados, o que parece ser um caminho promissor para obter esquemas que se aproximem das taxas teóricas obtidas por Nazer & Gastpar.

Trabalhos futuros

Como futuras linhas de pesquisa, sugere-se o seguinte:

(*i*) Expandir o aninhamento para mais de dois códigos, encontrando, portanto, as relações entre os códigos necessárias para satisfazer todas as restrições de aninhamento resultantes do número superior de níveis;

(*ii*) Alterar as premissas das distribuições de graus no capítulo 4 e permitir um maior número de graus, adaptando as restrições de projeto e, consequentemente, o procedimento de otimização, em particular, o método de reaninhamento de vetores mutados. A expectativa é obter melhores distribuições e maior flexibilidade de projeto;

(*iii*) Estudar a relação do PEG com códigos LDPC aninhados e verificar por que o aninhamento causa o aumento do número de dependências lineares;

(*iv*) Usar outras análises de desempenho, incluindo EXIT charts. Teoricamente, os EXIT charts produzem resultados mais fidedignos que a Aproximação Gaussiana e possuem complexidade computacional muito inferior à da Density Evolution. Os EXIT charts constituem uma alternativa interessante porque, de acordo com

[28] podem ser usados para descrever um problema de otimização unidimensional parecido ao do canal com apagamento, mas sem aproximações sobre as distribuições de probabilidade feitas pela Aproximação Gaussiana;

(*v*) Acrescentar estrutura aos códigos LDPC utilizados, tornando a codificação linear. Códigos como os LDPCs quasi-cíclicos (QC) ou os códigos *Repeat Accumulate* são alternativas interessantes para obter códigos com codificação de menor complexidade. O uso de QCs é particularmente interessante, pois o projeto de códigos é algébrico. A dificuldade é construir QCs aninhados, um problema pouco discutido na literatura;

(*vi*) Buscar outras formas de construção de reticulados com códigos, tal como os LDLCs;

(*vii*) Pesquisar o comportamento da Density Evolution para comprimentos de bloco n finitos, possibilitando a estimativa de desempenho para os tamanhos dos pacotes com os quais desejamos trabalhar. De fato, há sempre uma grande disparidade entre os resultados teóricos estimados pela Density Evolution para blocos com $n \to \infty$ e códigos de comprimento médio ou pequeno. Além disso, é bem provável que distribuições ótimas para comprimentos infinitos não sejam as mais adequadas para tamanhos práticos.

Referências Bibliográficas

[1] D. Bi and L. C. Pérez, "Rate-compatible low-density parity-check codes with rate-compatible degree profiles," *Electronic Letters*, vol. 42, no. 1, Jan. 5 2006.

[2] I.-J. Baik and S.-Y. Chung, "Irregular Low-Density Parity-Check Lattices," in *ISIT 2008*, Toronto, Canada, July 6-11, 2008.

[3] S. Boyd and L. Vandenberghe, *Convex Optimization*, 2nd ed. Cambridge University Press, 2004.

[4] M. C. Castro, D. Silva, and B. F. Uchôa-Filho, "Códigos Turbo para Codificação de Rede na Camada Física sobre Inteiros Gaussianos," in *XXXI Simpósio Brasileiro de Telecomunicações - SBrT2013*, Fortaleza, CE, Sep. 2013.

[5] J. H. Conway and N. J. A. Sloane, *Sphere Packings, Lattices and Groups*. New York: Springer, 1998.

[6] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. Wiley-Interscience, 2006.

[7] D. S. Dummit and R. M. Foote, *Abstract Algebra*, 3rd ed. John Wiley and Sons, 2004.

[8] A. Guillén i Fàbregas, A. Martinez and G. Caire, *Bit-Interleaved Coded Modulation*, Foundations and Trends® in Communications and Informaiton Theory. Now Publishers Inc., vol. 5, nos. 1-2, pp. 1–153, 2008.

[9] C. Feng, D. Silva, and F. R. Kschischang, "An algebraic approach to physical-layer network coding," *IEEE Transactions on Information Theory*, vol. 59, no. 11, pp. 7576–7596, Nov. 2013.

[10] G. D. Forney Jr., M. D. Trott and S-Y. Chung, "Sphere-Bound-Achieving Coset Codes and Multilevel Coset Codes," *IEEE Transactions on Information Theory*, vol. 46, no. 3, pp. 820–850, May 2000.

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

[12] J. Hou, P. H. Siegel, L. B. Milstein and H. D. Pfister, "Performance Analysis and Code Optimization of Low Density Parity-Check Codes on Rayleigh Fading Channels," *IEEE Transactions on Information Theory*, vol. 19, no. 5, pp. 924–934, May 2001.

[13] J. Hou, P. H. Siegel, L. B. Milstein and H. D. Pfister, "Capacity-Approaching Bandwidth-Efficient Coded Modulation Schemes Based on Low-Density Parity-Check Codes," *IEEE Transactions on Information Theory*, vol. 49, no. 9, pp. 2141–2155, Sep. 2003.

[14] X. Y. Hu, E. Eleftheriou, and D. H. Arnold, "Regular and Irregular Progressive Edge-Growth Tanner Graphs," *IEEE Transactions on Information Theory*, vol. 51, pp. 386–398, Jan. 2005.

[15] J. Huber, "Multilevel codes: Distance profiles and channel cpacity,"in *ITG-Fachbericht 130, Conf. Rec.*, München, Germany, Oct. 1994. pp. 305–319.

[16] H. Imai and S. Hirakawa, "A New Multilevel Coding Method Using Error-Correcting Codes," *IEEE Transactions on Information Theory*, vol. IT-23, no. 3, 1977.

[17] S. J. Johnson, *Iterative Error Correction*. Cambridge University Press, 2010.

[18] W. Kositwattanarerk and F. Oggier, "Connections Between Construction D and Related Constructions of Lattices," *CoRR*, vol. abs/1308.6175, 2014. [Online]. Available: http://arxiv.org/abs/1308.6175

[19] F. R. Kschischang, L. J. Frey, and H. A. Loeliger, "Factor Graphs and the Sum-Product Algorithm," *IEEE Transactions on Information Theory*, vol. 47, pp. 498–519, Feb. 2001.

[20] S. C. Liew, S. Zhang, and L. Lu, "Physical-layer network coding: Tutorial, survey, and beyond," *Physical Communication*, vol. 6, pp. 4–42, May 2013.

[21] T. K. Moon, *Error Correction Coding: Mathematical Methods and Algorithms*. Wiley, June 2005.

[22] B. Nazer and M. Gastpar, "Compute-and-forward: Harnessing interference through structured codes," *IEEE Transactions on Information Theory*, vol. 57, no. 10, pp. 6463–6486, Oct. 2011.

[23] A. Papoulis and S. U. Pillai, *Probability, Random Variables and Stochastic Processes* 4th ed. McGraw-Hill, 2002.

[24] P. R. Branco da Silva and D. Silva, "Design of Lattice Network Codes Based on Construction D," in *International Telecommunications Symposium ITS 2014*, São Paulo, Aug. 17 2014.

[25] J. G. Proakis, M. Salehi, *Digital Communications* 5th ed. McGraw-Hill Higher Education, 2007.

[26] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of Capacity-Approaching Irregular Low-Density Parity-Check Codes," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 619–637, Feb. 2001.

[27] T. J. Richardson and R. L. Urbanke, "The Capacity of Low-Density Parity-Check Codes Under Message-Passing Decoding," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 599–618, Feb. 2001.

[28] ——, *Modern Coding Theory*. Cambridge University Press, 2008.

[29] W. E. Ryan and S. Lin, *Channel Codes - Classical and Modern*. Cambridge University Press, 2009.

[30] M. R. Sadeghi, A. H. Banihashemi, and D. Panario, "Low-Density Parity-Check Lattices: Construction and Decoding Analysis," *IEEE Transactions on Information Theory*, vol. 52, pp. 4481–4495, Oct. 2006.

[31] D. Silva, "Codificação de Rede na Camada Física via Reticulados," *XXXI Simpósio Brasileiro de Telecomunicações - SBrT2013*, pp. 1–5, Fortaleza, CE, Sep. 2013.

[32] R. Storn and K. Price, "Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces," *Journal of Global Optimization*, vol. 11, pp. 341–359, 1997.

[33] K. Price, R. M. Storn and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*. Springer, 2005.

[34] R. Tanner, "A recursive approach to low complexity codes," *Combinatorica*, vol. 2, no. 1, pp. 71–78, 1982.

[35] G. Ungerboeck, "Channel Coding with Multilevel/Phase Signals," *IEEE Transactions on Information Theory*, vol. IT-28, no 1, Jan. 1982.

[36] R. Upadrashta and A. Thangaraj, "Key Reconciliation Using Nested LDPC Codes," *National Conference on Communications NCC 2008*, Feb. 2008, pp. 1–5.

[37] U. Wachsmann, R. F.H. Fischer, and J. B. Huber, "Multilevel Codes: Theoretical Concepts and Practical Design Rules," *IEEE Transactions on Information Theory*, vol. 45, no. 5, pp. 1361–1391, July 1999.

[38] M. Yang, W. E. Ryan, and Y. Li, "Design of Efficiently Encodable Moderate-Length High-Rate Irregular LDPC Codes," *IEEE Transactions on Communications*, vol. 52, no. 4, pp. 564–571, April 2004.