UNICAMP - Universidade Estadual de Campinas
FEEC - Faculdade de Engenharia Elétrica e de Computação
DECOM - Departamento de Comunicações

Canal M-APSK Não-Coerente de Bloco: Capacidade e Proposta de Codificação para Receptores Iterativos

Autor: Daniel Carvalho da Cunha

Orientador: Prof. Dr. Jaime Portugheis

Tese de Doutorado apresentada à Faculdade de Engenharia Elétrica e de Computação como parte dos requisitos para obtenção do título de Doutor em Engenharia Elétrica. Área de concentração: **Telecomunicações e Telemática**.

Banca Examinadora

Prof. Dr. Jaime Portugheis ..... DECOM/FEEC/UNICAMP
Prof. Dr. Bartolomeu Ferreira Uchôa Filho ..... GPqCom/EEL/UFSC
Prof. Dr. Weiler Alves Finamore ..... DEE/PUC-RIO
Prof. Dr. Dalton Soares Arantes ..... DECOM/FEEC/UNICAMP
Prof. Dr. Reginaldo Palazzo Júnior ..... DT/FEEC/UNICAMP
Prof. Dr. Renato Baldini Filho ..... DECOM/FEEC/UNICAMP

Campinas-SP, Maio de 2006

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA E ARQUITETURA - BAE - UNICAMP

C914c

Cunha, Daniel Carvalho da
Canal M-APSK não-coerente de bloco: capacidade e proposta de codificação para receptores iterativos. / Daniel Carvalho da Cunha. --Campinas, SP: [s.n.], 2006.

Orientador: Jaime Portugheis.
Tese (doutorado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Códigos de controle de erros (Teoria da Informação). 2. Teoria da Codificação. I. Portugheis, Jaime. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Titulo em Inglês: Blockwise noncoherent M-APSK channel: capacity and coding scheme for iterative receivers
Palavras-chave em Inglês: Channel capacity, Noncoherent channels, M-APSK constellations, Iterative decoding, Factor graphs
Área de concentração: Telecomunicações e Telemática.
Titulação: Doutor em Engenharia Elétrica
Banca examinadora: Bartolomeu Ferreira Uchôa Filho, Weiler Alves Finamore, Dalton Soares Arantes, Reginaldo Palazzo Júnior e Renato Baldini Filho
Data da defesa: 26/05/2006

Aos meus pais,
Ivaldo (in memoriam) e Heloísa

"Jamais permita que seus objetivos sejam maiores do que seus sonhos."

César Souza

Resumo

Em vários sistemas de transmissão passa-faixa, uma recepção coerente satisfatória é difícil de ser alcançada. Para alguns destes sistemas, é comum supor que a rotação de fase introduzida pelo canal é constante durante um bloco de L símbolos e que ela varia de maneira independente de bloco a bloco. Este canal é denominado canal não-coerente de bloco. Investigamos a capacidade de um canal não-coerente de bloco utilizando a modulação M-APSK (do inglês, *M-ary Amplitude Phase Shift Keying*). Apresentamos a caracterização da distribuição de entrada que atinge a capacidade e obtivemos limitantes superiores e inferiores para a mesma. Adicionalmente, desenvolvemos um algoritmo que simultaneamente fornece a distribuição de entrada e os parâmetros da modulação M-APSK que maximizam a informação mútua com recepção coerente. A investigação da capacidade mostrou que o aumento de L faz a capacidade não-coerente convergir para a coerente. Além disso, o uso de codificação diferencial torna a convergência mais rápida. Motivados por este comportamento, apresentamos um esquema de codificação eficiente em faixa. Este esquema é formado pela concatenação serial de um código LDPC (do inglês, *Low-Density Parity Check*), um entrelaçador e um codificador diferencial. Para o esquema apresentado, o receptor iterativo é descrito por um grafo-fator. Os desempenhos do esquema com diferentes tamanhos de códigos LDPC são comparados.

Palavras-chave: Capacidade de canal, Canais não-coerentes, Constelações M-APSK, Decodificação iterativa, Grafos-fatores.

Abstract

Coherent reception is not possible for many bandpass transmission systems. In some of these systems, it is commonly assumed that the unknown carrier phase rotation is constant over a block of L symbols and it is independent from block to block. This channel is denominated blockwise noncoherent channel. The blockwise noncoherent channel capacity using *M-ary Amplitude and Phase Shift Keying* (M-APSK) modulation is investigated. The characterization of the input distribution achieving capacity is presented. Upper and lower bounds to this capacity are derived. In addition, an algorithm for simultaneously computing the input distribution and the M-APSK constellation parameters which maximizes the mutual information with coherent reception is developed. The investigation of the capacity showed that as L increases, the noncoherent capacity converges to the coherent one. Besides that, the use of differential encoding makes this convergence faster. Motivated by this fact, a bandwidth efficient coding scheme is presented. This scheme is composed of a serial concatenation of a *Low-Density Parity Check* (LDPC) code, an interleaver, and a differential encoder. For this scheme, the iterative receiver is described by a factor graph. The scheme performances for different lengths of LDPC codes are compared.

Keywords: Channel capacity, Noncoherent channels, M-APSK constellations, Iterative decoding, Factor graphs.

Agradecimentos

Agradeço

A Deus, por me proporcionar inúmeros momentos felizes na vida.

Ao Prof. Jaime, os ensinamentos, a paciência nos momentos difíceis, a cumplicidade, a fé no trabalho, e principalmente, a amizade. Tenho a certeza de que, mais que um orientador, tive um amigo nestes cinco anos de convivência no Mestrado e Doutorado na FEEC. Fica a esperança de que um dia ele volte às origens e torça para o time certo.

À minha família, pelo amor, incentivo e confiança na minha capacidade.

À Maria Juliana, pelo amor, dedicação e principalmente, pela paciência em suportar a distância e as contas de "quadrado e bola".

Aos amigos Divanilson Campelo, Charles Casimiro e Fábio Mulheque, pela excelente convivência dentro e fora da Unicamp.

Ao amigo "irmão"Miguel Cosac, pelo companheirismo e amizade.

Aos amigos do DECOM e demais departamentos.

Aos companheiros do LTIA, pela colaboração e preciosa convivência.

A todos os funcionários da FEEC e da CPG, que direta ou indiretamente colaboraram para a realização deste trabalho.

À CAPES e à FAPESP, pelo financiamento deste trabalho.

Aos Profs. Bartolomeu Uchôa Filho, Weiler Finamore, Dalton Arantes, Reginaldo Palazzo Jr. e Renato Baldini Filho, pelas sugestões e contribuições à redação final deste trabalho.

A todos que de alguma forma contribuíram para o sucesso deste trabalho.

Sumário

Lista de Figuras

Lista de Tabelas

1 Introdução

1.1 Contexto e Objetivos do Trabalho

Nos tempos atuais, os sistemas de comunicações estão presentes em nossas vidas de diversas maneiras. Os telefones, os rádios e as televisões, assim como os computadores conectados à Internet, são capazes de fornecer comunicação rápida em qualquer parte do planeta. Além disso, o advento das comunicações celulares e sem fio melhoraram a disseminação da informação, introduzindo o conceito de comunicação a qualquer hora e em qualquer lugar. Na realidade, existe uma lista extensa de aplicações que poderiam ser mencionadas. Todavia, um fato comum em todas é a busca por melhores serviços e melhores desempenhos dos sistemas por parte dos projetistas, diante das limitações impostas em cada uma das aplicações.

Assim, o engenheiro de comunicações basicamente se concentra em resolver duas questões principais. A primeira delas é o projeto de esquemas de codificação que forneçam comunicação confiável, dados os recursos limitados disponíveis, como potência de transmissão, largura de faixa (banda), complexidade etc. A segunda questão é o estudo dos limites teóricos de taxa de transmissão que o sistema pode atingir com as restrições que possui.

Para analisar um sistema de comunicações, é comum se representar o transmissor, o meio físico de transmissão e o receptor por um modelo de canal. Em seu trabalho pioneiro, Claude E. Shannon [1] mostrou que, dado um modelo de canal, existe um parâmetro chamado *capacidade de canal* , que relaciona a potência transmitida, a largura de faixa e a taxa máxima com que a informação pode ser enviada confiavelmente por um canal.

Nesta tese, investigamos o problema de comunicação para o canal com ruído aditivo que introduz uma rotação de fase aleatória, desconhecida para o transmissor e o receptor. A motivação para o estudo deste canal se baseia, por exemplo, em sistemas celulares TDMA (do inglês, *Time Division Multiple Access*) que utilizam técnicas de salto em frequência FH (do inglês, *Frequency Hopping*), nos quais é difícil manter a coerência de fase de salto para salto. Um outro fator motivador é que o estudo deste canal pode ser considerado como uma preparação para a abordagem de um canal com desvanecimento.

Basicamente, existem duas soluções para o descasamento de fase encontrado nos sistemas de comunicações. Uma delas é utilizar esquemas de modulação M-FSK (do inglês, *M-ary Frequency Shift Keying*) não-coerentes, que são imunes à rotação de fase, porém pouco eficientes em largura de faixa. A outra solução é utilizar esquemas que estimem a fase introduzida pelo canal (uso de símbolos-piloto) ou eliminem a necessidade de estimação da fase (codificação diferencial).

O objetivo do presente trabalho é propor um esquema de codificação eficiente em faixa para canais AWGN (do inglês, *Additive White Gaussian Noise*) com ruído de fase. Para isso, faremos uso de códigos LDPC (do inglês, *Low-Density Parity Check*) e um esquema de modulação denominado M-APSK (do inglês, *M-ary Amplitude Phase Shift Keying*) concatenados serialmente. Adicionalmente, faremos um estudo da capacidade de canal para obter uma referência de desempenho para o esquema proposto.

1.2 Organização da Tese

No Capítulo 2, são apresentados conceitos básicos de Teoria da Informação, como a regra da cadeia para infomação mútua e a capacidade de canal. Conceitos sobre Teoria da Codificação, como códigos de bloco lineares e a representação destes por treliças, também são expostos. O objetivo deste capítulo é familiarizar o leitor no contexto do trabalho e facilitar o entendimento do mesmo.

No Capítulo 3, é apresentado um estudo sobre a capacidade de canal AWGN não-coerente de bloco utilizando um esquema de modulação com número finito de sinais. Inicialmente, são apresentadas as constelações utilizadas no transmissor e considerações importantes a respeito da capacidade de canal são mencionadas. Em seguida, apresentamos uma proposta para o cálculo de limitantes de capacidade do canal não-coerente, como também resultados numéricos deste cálculo.

No Capítulo 4, apresentamos as definições básicas sobre os grafos-fatores e a descrição do algoritmo Soma-Produto. Em seguida, o capítulo aborda como os grafos-fatores podem ser utilizados para modelar sistemas codificados. O algoritmo de decodificação aplicado a estes sistemas é tratado, enfatizando-se os grafos sem ciclos e os que possuem ciclos. Finalmente, o capítulo apresenta de maneira resumida os códigos LDPC, destacando suas formas de representação e seus algoritmos de decodificação iterativa nos domínios das probabilidades e dos logaritmos.

No Capítulo 5, propomos um esquema de codificação eficiente em faixa utilizando códigos LDPC e modulações M-APSK. Inicialmente, os componentes do transmissor são descritos. Em seguida, definimos conceitos importantes, como função objetivo e distribuições canônicas, como pré-requisitos para a descrição do grafo que representa o receptor. As mensagens que compõem o algoritmo Soma-Produto e o seu cronograma de execução são exibidos posteriormente. Por fim, resultados de simulação são apresentados e comentados.

Finalmente, no Capítulo 6, são apresentadas as conclusões que foram obtidas com o estudo e as perspectivas de trabalhos futuros.

1.3 Resumo das Principais Contribuições

As principais contribuições desta tese são:

- Caracterização da distribuição de entrada que atinge capacidade para o canal M-APSK não-coerente de bloco. A capacidade é atingida por uma variável de entrada cuja distribuição de fases é uniforme, independente e identicamente distribuída (u.i.i.d.) e independente da distribuição conjunta de amplitudes.
- Proposta do cálculo de limitantes superiores e inferiores de capacidade para o canal M-APSK não-coerente de bloco. Mostra-se que os limitantes convergem para a capacidade coerente com o aumento do intervalo de coerência do canal.
- Generalização de um algoritmo de otimização proposto na literatura, com o objetivo de obter simultaneamente os parâmetros da constelação M-APSK e a distribuição de entrada que atingem a capacidade do canal coerente.
- Mostrar que o uso de codificação diferencial na transmissão leva os limitantes de capacidade a convergirem mais rapidamente para a capacidade coerente. Este resultado, juntamente com a configuração da constelação, nos motivou a considerar a utilização de codificação diferencial no esquema proposto.
- Proposta de um algoritmo de demodulação e decodificação conjuntas com base em grafos-fatores. A descrição de um receptor iterativo por meio de um grafo-fator, que considera o uso de codificação diferencial e modulações não-binárias, é algo que ainda não existe na literatura.
- Apresentação de um esquema de codificação eficiente em faixa, para taxas acima de 1 bit por uso do canal, que ainda não foi proposto na literatura.

2 Conceitos Básicos

Um projeto bem elaborado de um sistema de comunicações parte da premissa do bom conhecimento das ferramentas matemáticas disponíveis para sua concepção. Além disso, também é fundamental ter o conhecimento dos conceitos básicos que tais ferramentas podem manipular. Este capítulo aborda sucintamente alguns conceitos básicos da engenharia de comunicações, como a missão do projetista e algumas ferramentas que ele dispõe para propor soluções no campo da transmissão digital, que são a Teoria da Informação e da Codificação. Sobre estas teorias, definições elementares são apresentadas para possibilitar o entendimento dos estudos feitos no decorrer desta tese.

2.1 O Papel do Engenheiro de Comunicações

A principal atribuição do engenheiro de comunicações é projetar sistemas de comunicações conforme a demanda e a necessidade dos usuários de maneira que a transmissão da informação seja a mais eficiente possível para uma dada confiabilidade. Podemos enumerar alguns objetivos do engenheiro ao projetar um sistema de comunicações, tais como:

1. Maximizar a taxa de transmissão;
2. Minimizar a probabilidade de erro;

3. Minimizar a potência transmitida;
4. Minimizar a largura de faixa utilizada;
5. Minimizar a complexidade e o custo do sistema.

Dos objetivos mencionados, dois deles lidam com os recursos primários de todo sistema de comunicações, quais sejam, a potência média do sinal transmitido e a largura de faixa disponível. Em geral, a missão do projetista é encontrar soluções técnicas que utilizem estes dois recursos de maneira eficiente.

No entanto, para projetar tais sistemas, sejam eles analógicos ou digitais, é necessário lidar com compromissos, isto é, melhorar certos aspectos do sistema em detrimento de outros. Por exemplo, é possível minimizar a probabilidade de erro de um sistema ao preço de um aumento em sua complexidade. Desta forma, dependendo da aplicação, o projetista pode dar ênfase a certos recursos sacrificando outros.

Outro ponto importante a se considerar no projeto é a presença inevitável do ruído no sistema de comunicações. O ruído se caracteriza pela presença de sinais indesejáveis, provenientes de fontes internas ou externas, que prejudicam a transmissão e o processo de obtenção da informação. Desta forma, além de gerenciar os recursos disponíveis do sistema, o engenheiro de comunicações deve desenvolver formas de melhorar a comunicação em canais sujeitos às diversas formas de ruído.

2.2 Comunicação através de Canais Ruidosos

Podemos citar como exemplos de canais de comunicação ruidosos:

- Uma linha telefônica analógica pela qual dois modems digitais trocam informações;
- Um enlace de comunicação de rádio entre um telefone celular e uma estação rádio-base;
- O disco rígido de um computador.

Nos exemplos mencionados anteriormente, assim como em outros, se transmitirmos informação através dos canais, existe uma probabilidade de que a mensagem recebida não seja idêntica à mensagem transmitida. Como resolver tal problema?

Uma das soluções seria melhorar as características físicas do canal de comunicação com o objetivo de reduzir sua probabilidade de erro. Por exemplo, o desempenho das operações de leitura e escrita em um disco rígido poderia ser melhorado com o uso de componentes eletrônicos mais confiáveis, ou com o uso de trilhas magnéticas mais largas para representar cada bit, ou ainda com o uso de dispositivos de resfriamento nos circuitos para reduzir o ruído térmico. Enfim, todas estas modificações representariam a mudança das características físicas do canal, porém acarretariam num aumento significativo dos custos.

Para evitar tal desvantagem, as Teorias da Informação e da Codificação oferecem uma abordagem alternativa, na qual o canal ruidoso é aceito como ele é e sistemas de comunicações são utilizados para detectar e corrigir os erros introduzidos pelo canal. Além disso, tais sistemas também permitem que o canal seja usado de forma eficiente, contribuindo ainda mais para a redução dos custos. Enquanto as soluções físicas aumentam os custos do canal, as soluções sistêmicas podem transformar canais ruidosos em canais confiáveis, porém ao preço de um aumento de processamento computacional.

Um sistema digital de comunicações típico é apresentado na Figura 2.1. Ao se projetar um sistema de comunicações digitais, normalmente tem-se a fonte de informação, o canal de comunicações e o usuário final já especificados. O grande desafio é conceber transmissor e receptor de maneira que a transmissão de informação da fonte ao usuário final seja feita rápida e confiavelmente sob o menor custo possível. Os blocos funcionais do transmissor e do receptor são representados pelo Codificador-Decodificador de Fonte, Codificador-Decodificador de Canal e Modulador-Demodulador.

A função do codificador de fonte é reduzir a redundância da fonte, diminuindo a largura de faixa necessária à transmissão da informação. Uma vez que a sequência de símbolos na saída do codificador de fonte pode sofrer erros durante a transmissão, o codificador de canal tem a função de introduzir uma redundância controlada nesta sequência para tornar a transmissão confiável. Completando o bloco transmissor, o modulador tem a função de casar a saída do codificador de canal com o canal de transmissão. Ele converte a sequência de símbolos em uma sequência de sinais apropriados à transmissão pelo meio físico, que é analógico. No receptor,

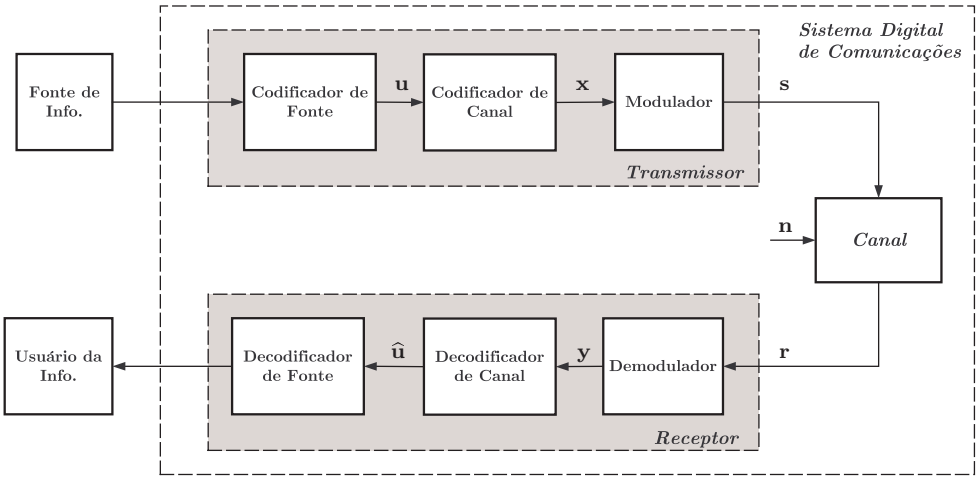


Figura 2.1: Sistema digital de comunicações [2].

o sinal observado na saída do canal é processado na ordem inversa do transmissor, com o objetivo de recuperar a informação original.

2.3 Teoria da Informação

A Teoria da Informação lida com os aspectos mais fundamentais dos sistemas de comunicações, investigando suas limitações teóricas através de modelos matemáticos dos canais físicos. Ela fornece relações que indicam os compromissos existentes na construção de codificadores e decodificadores para um determinado sistema e fornece subsídios para a construção de modelos detalhados de fontes e canais reais.

2.3.1 Modelos Matemáticos de Canais

Um sistema de comunicação pode ser descrito por um modelo, que é uma representação matemática definida para descrever como os sinais são processados e como eles são afetados pelo ambiente de comunicação real. A seguir, vamos mencionar alguns modelos de canais clássicos utilizados em sistemas de comunicações.

Canal sem Memória

O modelo de canal sem memória é definido pelo conjunto de entradas do modulador, o conjunto de saídas do demodulador e um conjunto de probabilidades condicionais chamadas probabilidades de transição, que relacionam as possíveis saídas com cada possível entrada. Dada uma sequência de entrada $\mathbf{x} = [x_1, x_2, ..., x_J]$, a probabilidade de observar uma sequência de saída $\mathbf{y} = [y_1, y_2, ..., y_J]$ é dada por

$$p(\mathbf{y}|\mathbf{x}) = \prod_{k=1}^{J} p(y_k|x_k) \ . \tag{2.1}$$

A descrição do canal como sem memória se refere ao fato de que o símbolo de saída a qualquer instante depende estatisticamente apenas do símbolo de entrada naquele mesmo instante.

Um exemplo de modelo de canal sem memória é o canal DMC (do inglês, *Discrete Memoryless Channel*), que se caracteriza pelos alfabetos de entrada e saída do canal serem finitos.

Canal Binário Simétrico

O canal BSC (do inglês, *Binary Symmetric Channel*) é um exemplo importante de modelo de canal DMC, em que os símbolos de entrada e de saída pertencem a um alfabeto binário e as probabilidades de transição que caracterizam o canal assumem os valores p e $(1-p)$, como probabilidade de ocorrência de erro e de acerto, respectivamente. A Figura 2.2 ilustra a representação do canal BSC mencionado.

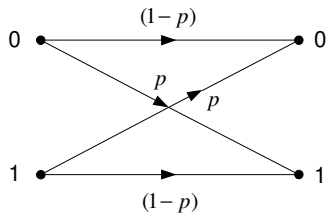


Figura 2.2: Representação de um canal binário simétrico.

Canal AWGN

O canal AWGN é descrito em termos da entrada x e da saída y pela equação

$$y = x + n \ , \tag{2.2}$$

na qual n é uma variável aleatória (v. a.) gaussiana de média zero e variância σ^2. Quando a entrada x assume um de K valores discretos, tais que $K \geq 2$, temos um canal AWGN com entrada restrita. A função densidade de probabilidade condicional da saída y, dado que a entrada x assume um valor x_i, é definida por

$$p(y|x = x_i) = \frac{1}{\sqrt{2\pi}\sigma} \ \exp\left(-\frac{(y - x_i)^2}{2\sigma^2}\right) \ . \tag{2.3}$$

2.3.2 Entropia, Entropia Condicional e Informação Mútua

No início dos estudos sobre Teoria da Informação, buscava-se uma caracterização matemática da quantidade de informação que poderia ser transmitida de maneira confiável sobre um dado canal físico [3]. Para solucionar este problema, tornou-se necessário estabelecer uma medida matemática para a informação. Naturalmente, a função logarítmica surgiu como uma boa sugestão. Desta forma, o conteúdo de informação de uma mensagem gerada a partir de um conjunto de K mensagens igualmente prováveis é $\log K$, em que a base do logaritmo [1] depende da unidade básica de informação, que usualmente é denominada bit.

Considere que a fonte de informação produz K símbolos equiprováveis e que cada um deles é independente de um intervalo de símbolo para outro. Uma vez que sabemos como medir a quantidade de informação transmitida por uma fonte, temos que a taxa de informação vale $\log K$ bits por símbolo. Todavia, com o que foi mencionado até agora, este conceito de medida de informação não pode ser atribuído a fontes cujos símbolos não são equiprováveis.

Para estabelecer uma medida de informação para esta classe de fontes, Shannon definiu o conceito de *entropia* de uma fonte de informação [1]. Considere a fonte representada por uma v. a. X, que pode assumir um dos K símbolos de um alfabeto $\mathcal{X}$ com uma função de

[1] Nesta tese, todos os logaritmos são tomados na base 2, a menos que seja especificado o contrário.

distribuição de probabilidade $P(x_i)$. Daí, a entropia de X é definida como

$$H\left(X\right) \triangleq \sum_{i=1}^{K} P\left(x_i\right) \log\left(\frac{1}{P\left(x_i\right)}\right) \ . \tag{2.4}$$

Observe que a equação (2.4) também enquadra as fontes equiprováveis, pois neste caso $H = -\log P(x_i) = \log K$, conforme mencionado anteriormente. Desta maneira, a função entropia fornece uma medida da quantidade média de informação produzida por símbolo da fonte.

Considere agora uma v. a. Y com Q valores possíveis em um alfabeto $\mathcal{Y}$. Conhecido o conceito de entropia, este pode ser estendido às variáveis X e Y, caracterizando a definição de entropia conjunta

$$H\left(X,Y\right) \triangleq \sum_{i=1}^{K}\sum_{j=1}^{Q} P\left(x_i,y_j\right) \log\left(\frac{1}{P\left(x_i,y_j\right)}\right) , \tag{2.5}$$

em que $P(x_i, y_j)$ é a função de distribuição de probabilidade conjunta das variáveis X e Y.

No contexto das comunicações, podemos considerar que a saída Y é a versão ruidosa da entrada X do canal. Uma vez que sabemos que $H(X)$ é a medida da incerteza *a priori* de X, como podemos medir a incerteza de X após a observação de Y ? Para responder esta questão, definimos o conceito de *entropia condicional* de X dado que $Y = y_j$ pela seguinte equação

$$H\left(X\left|Y=y_j\right.\right) = \sum_{i=1}^{K} P\left(x_i|Y=y_j\right) \log\left(\frac{1}{P\left(x_i|Y=y_j\right)}\right) \ . \tag{2.6}$$

Podemos então obter a entropia condicional de X dado Y como o valor esperado da equação (2.6) em todos os possíveis valores de Y, conforme

$$\begin{aligned} H\left(X\left|Y\right.\right) &\triangleq \sum_{j=1}^{Q} P\left(y_j\right) H\left(X\left|Y=y_j\right.\right) \\ &= \sum_{i=1}^{K}\sum_{j=1}^{Q} P\left(x_i,y_j\right) \log\left(\frac{1}{P\left(x_i|y_j\right)}\right) . \end{aligned} \tag{2.7}$$

De posse das definições de $H(X)$, $H(X,Y)$ e $H(X|Y)$, podemos conceituar a *Regra da Cadeia para entropia* da seguinte forma

$$H(X,Y) = H(X) + H(Y|X) = H(Y) + H(X|Y) \ . \tag{2.8}$$

Sabe-se que o condicionamento de uma v. a. reduz a sua incerteza. Sendo assim, temos que $H(X|Y) \leq H(X)$, com igualdade se, e somente se, X e Y forem independentes. Portanto,

$$H(X,Y) = H(Y) + H(X|Y) \leq H(X) + H(Y) \ .$$

Uma vez que a entropia $H(X)$ representa a incerteza a cerca da entrada do canal antes da observação da saída e que a entropia condicional $H(X|Y)$ representa a incerteza da entrada após a observação da saída do canal, podemos interpretar $H(X) - H(X|Y)$ como uma redução da incerteza da entrada a partir da observação da saída. Esta quantidade é denominada *informação mútua* do canal entre X e Y e é definida por

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) \ . \tag{2.9}$$

$I(X;Y)$ é a quantidade de informação que X traz a respeito de Y, ou vice-versa, daí o nome de informação "mútua". Além disso, $I(X;Y)$ é um valor não-negativo e igual a zero se, e somente se, X e Y forem independentes.

A definição de informação mútua pode ser estendida naturalmente para mais de duas v. a., assim como para vetores aleatórios. Considere as v. a. X_1 , X_2 e Y, em que X_1 e X_2 podem ser interpretadas como duas v. a. distintas ou como componentes de um vetor aleatório $\mathbf{x}$. Logo, podemos dizer que

$$I(X_1\,, X_2\,; Y) = H(X_1\,, X_2) - H(X_1\,, X_2|Y) \ . \tag{2.10}$$

Assim como a entropia, também é possível condicionar a informação mútua ao conhecimento de uma v. a.. O conceito de *informação mútua condicional* é obtido de maneira simples a partir da definição de informação mútua

$$I(X;Y|Z) = H(X|Z) - H(X|Y,Z) \ . \tag{2.11}$$

Sabe-se que

$$H(X|Z) = \sum_k P\left(z_k\right) H\left(X\,|Z = z_k\right) \ ,$$

e

$$H(X|Y,Z) = \sum_k P\left(z_k\right) H\left(X\,|Y, Z = z_k\right) \ .$$

Portanto, percebe-se facilmente que

$$I(X;Y|Z) = \sum_k P(z_k)\, I(X;Y|Z=z_k) \ . \tag{2.12}$$

Retornando às v. a. X_1, X_2 e Y, temos que

$$\begin{aligned} I(X_1, X_2; Y) &= H(X_1, X_2) - H(X_1, X_2|Y) \\ &= H(X_1) + H(X_2|X_1) - [H(X_1|Y) + H(X_2|X_1, Y)] \\ &= I(X_1; Y) + I(X_2; Y|X_1) \ . \end{aligned}$$

Obtemos então a *Regra da Cadeia para informação mútua*

$$I(X_1, X_2; Y) = I(X_1; Y) + I(X_2; Y|X_1) \ . \tag{2.13}$$

2.3.3 Capacidade de um Canal Discreto sem Memória

Vimos que um canal discreto sem memória com entrada X e saída Y é caracterizado por seu conjunto de probabilidades de transição $p(y_j|x_i)$, tal que $i = 1, 2, ..., K$ e $j = 1, 2, ..., Q$. A informação mútua deste canal é definida pela equação (2.9) e pode ser reescrita como

$$I(X;Y) = \sum_{i=1}^{K} \sum_{j=1}^{Q} P(x_i, y_j) \log \left(\frac{p(y_j|x_i)}{P(y_j)} \right) \ . \tag{2.14}$$

Da Teoria de Probabilidades, sabe-se que

$$P(x_i, y_j) = p(y_j|x_i)\, P(x_i) \tag{2.15}$$

e

$$P(y_j) = \sum_{i=1}^{K} P(x_i, y_j) = \sum_{i=1}^{K} p(y_j|x_i)\, P(x_i) \ . \tag{2.16}$$

A partir das equações (2.14), (2.15) e (2.16), percebe-se que o cálculo da informação mútua $I(X;Y)$ necessita do conhecimento da distribuição de probabilidades dos símbolos de entrada do canal. Em outras palavras, a informação mútua depende tanto do canal quanto da maneira como ele é usado. Como as probabilidades de transição não podem ser manipuladas por quem

projeta o sistema de comunicações, a maximização de $I(X;Y)$ é feita por meio da escolha da distribuição de entrada. Sendo assim, define-se a capacidade de um canal DMC como a informação mútua máxima por utilização do canal, levando-se em conta todas as possíveis distribuições de entrada, ou seja,

$$C = \max_{\{P(x_i)\}} I\left(X;Y\right) \tag{2.17}$$

em que C é medida em bits por uso do canal [4]. O cálculo de C implica na maximização de $I(X;Y)$ com relação às variáveis de entrada $\{P(x_i)\}$ sob as restrições

$$P\left(x_i\right) \geq 0, \ \forall \ i$$

e

$$\sum_{i=1}^{K} P\left(x_i\right) = 1 \ .$$

A seguir, vamos mencionar teoremas indispensáveis no estudo da Teoria da Informação, que lidam com a capacidade de canal e estabelecem limites teóricos para os sistemas de comunicações.

2.3.4 Teorema da Codificação de Canal

A recuperação da informação original por parte do usuário final necessita que a probabilidade de erro de símbolo seja arbitrariamente pequena. Uma vez que a codificação de canal tem a finalidade de elevar o nível da confiabilidade da transmissão, ou pelo menos mantê-la num nível aceitável, será que existe um esquema de codificação de canal que forneça uma probabilidade de erro tão pequena quanto se deseja? A resposta é sim e é dada pelo Teorema da Codificação de Canal proposto por Shannon [1].

Considere uma fonte discreta sem memória, representada pela v. a. X com entropia $H(X)$ e que gera um símbolo a cada T_s segundos. Isto significa que a fonte possui uma taxa média de informação igual à $H(X)/T_s$ bits por segundo (*bits/s*). Considere também um canal discreto sem memória com capacidade C, que é usado uma vez a cada T_c segundos, isto é, a capacidade

do canal por unidade de tempo vale C/T_c *bits/s*. O Teorema da Codificação de Canal para um canal DMC diz que:

1. Se
$$\frac{H(X)}{T_s} \leq \frac{C}{T_c} \tag{2.18}$$
existe um esquema de codificação pelo qual a fonte pode transmitir informação pelo canal e esta pode ser recuperada pelo usuário final com uma probabilidade de erro bem pequena.
2. Todavia, se
$$\frac{H(X)}{T_s} > \frac{C}{T_c} \tag{2.19}$$
não será possível transmitir informação pelo canal e recuperá-la com probabilidade de erro pequena.

O Teorema da Codificação de Canal estabelece um limite para o valor da taxa de transmissão da informação sobre um canal ruidoso. Ele determina que existem bons códigos que garantem uma transmissão confiável, entretanto não revela nada sobre a construção de tais códigos.

2.3.5 Teorema da Capacidade de Canal

Até o presente momento, não foram consideradas as restrições de potência e de faixa do canal na transmissão da informação. Vamos agora aplicar os conceitos de informação mútua para a obtenção do Teorema da Capacidade de Canal para canais AWGN limitados em faixa e potência [1].

Primeiramente, é necessário estender os conceitos de Teoria da Informação para v. a. contínuas, pois até então consideramos apenas as v. a. discretas. Considere uma v. a. contínua X com função densidade de probabilidade $p(x)$. De maneira análoga à entropia de uma v. a. discreta, podemos definir a *entropia diferencial* de X como

$$h\left(X\right) = \int_{-\infty}^{+\infty} p\left(x\right) \log\left(\frac{1}{p\left(x\right)}\right) dx \ . \tag{2.20}$$

Com base nas equações (2.5), (2.7) e (2.14), podemos definir a entropia diferencial conjunta, a entropia diferencial condicional e a informação mútua, substituindo as funções de distribuição de probabilidade pelas respectivas densidades de probabilidade, e os somatórios por integrais.

Seja $X(t)$ um processo estacionário, de média zero e limitado a uma faixa de $W\,Hz$. Podemos denominar de X_k , as v. a. contínuas obtidas a partir da amostragem uniforme de $X(t)$ a uma taxa de $2W$ amostras por segundo. Estas amostras são transmitidas em T segundos sobre um canal ruidoso, também limitado a $W\,Hz$. Desta forma, o número de amostras K_p é dado por

$$K_p = 2WT. \tag{2.21}$$

A variável X_k representa as amostras do sinal transmitido, enquanto a variável Y_k representa as amostras do sinal recebido, conforme a equação

$$Y_k = X_k + N_k\,,\ k = 1, 2, \ldots, K_p \tag{2.22}$$

na qual N_k representa o ruído AWGN de média zero e variância $\sigma^2 = N_0 W$. A densidade espectral unilateral de potência do ruído é denotada por N_0 e as variáveis X_k e N_k são consideradas estatisticamente independentes.

Considerando o canal AWGN sem memória, discreto no tempo, com saída dada pela equação (2.22), temos um custo atribuído à transmissão de informação pelo canal. Normalmente, este custo é caracterizado pela limitação de potência do transmissor, que é descrita matematicamente por

$$E\left[X_k^2\right] = \int_{-\infty}^{+\infty} \left[X\left(t\right)\right]^2 dt = P\ . \tag{2.23}$$

Os canais AWGN limitados em potência possuem grande importância teórica e prática, pois modelam canais de comunicação tradicionais, como por exemplo, os canais de rádio de visada direta e os canais de comunicação via satélite.

A capacidade do canal AWGN limitado em potência é definida como

$$C = \max_{p_{X_k}(x):E[X_k^2]=P} I\left(X_k; Y_k\right)\ , \tag{2.24}$$

em que $I(X_k;Y_k)$ é a informação mútua entre X_k e Y_k. Ela deve ser maximizada em todas as possíveis distribuições da entrada X_k sob a restrição de potência dada pela equação (2.23).

A informação mútua $I(X_k;Y_k)$ pode ser escrita como

$$I(X_k;Y_k) = h(Y_k) - h(Y_k|X_k) \ . \tag{2.25}$$

Sabendo que X_k e N_k são independentes e que Y_k é dada pela equação (2.22), temos que

$$h(Y_k|X_k) = h(N_k) \ . \tag{2.26}$$

Sendo N_k uma v.a. gaussiana de variância σ^2, sua entropia diferencial é definida pela equação (2.20) e resulta em

$$h(N_k) = \frac{1}{2}\log\left(2\pi e\sigma^2\right) \ . \tag{2.27}$$

Como $h(N_k)$ independe de X_k, maximizar $I(X_k;Y_k)$ significa maximizar $h(Y_k)$. Para que $h(Y_k)$ seja máxima, Y_k deve ser uma v.a. gaussiana. Como N_k e Y_k são v.a. gaussianas, consequentemente X_k também será gaussiana para que $I(X_k;Y_k)$ seja maximizada. Portanto, podemos reformular a capacidade de canal como

$$C = I(X_k;Y_k) : X_k \ v.a. \ gaussiana, E[X_k^2] = P \ , \tag{2.28}$$

em que $I(X_k;Y_k)$ é definida pela equação (2.25).

Segundo a equação (2.27), a entropia diferencial de uma v.a. gaussiana depende apenas da sua variância. Logo, temos que a entropia de Y_k é dada por

$$h(Y_k) = \frac{1}{2}\log\left[2\pi e(P+\sigma^2)\right] \ . \tag{2.29}$$

De posse das equações (2.27) e (2.29), temos que a capacidade de canal vale

$$C = h(Y_k) - h(N_k) = \frac{1}{2}\log\left(1+\frac{P}{\sigma^2}\right) \ (bits/uso\,do\,canal) \ . \tag{2.30}$$

Usando o canal para transmitir K_p amostras em T segundos, podemos escrever C', a capacidade de canal por unidade de tempo, como

$$C' = \left(\frac{K_p}{T}\right)C \quad (bits/s) \ . \tag{2.31}$$

Substituindo as equações (2.21) e (2.30) na equação (2.31) e sabendo ainda que $\sigma^2 = N_0 W$, obtemos que

$$C = W \log\left(1 + \frac{P}{N_0 W}\right) \ (bits/s) \ . \tag{2.32}$$

A equação (2.32) representa um dos resultados mais importantes da Teoria da Informação pelo fato de relacionar três parâmetros-chave do sistema: taxa de transmissão, largura de faixa do canal e relação sinal-ruído. O Teorema da Capacidade de Canal define o limite de taxa de transmissão confiável para um canal AWGN limitado em faixa e potência. Para que a taxa máxima seja utilizada, o sinal de entrada deve possuir distribuição de probabilidade semelhante à do ruído AWGN.

Uma maneira de fazer com que o sinal de entrada tenha a distribuição de probabilidade apropriada para se atingir capacidade é utilizar-se de códigos corretores de erros para se obter tal distribuição.

2.4 Teoria da Codificação

Todos os códigos corretores de erros se baseiam em um mesmo princípio básico: a adição de redundância à informação com o objetivo de corrigir erros que possam ocorrer no processo de gravação ou transmissão de dados. No contexto das comunicações, a informação transmitida pode ser degenerada pelo ruído ao longo do meio de transmissão. Daí, o receptor deve verificar o quanto esta informação foi afetada e tentar recuperar a informação original. Em geral, quanto mais redundância se adiciona à informação, mais proteção se obtém contra o ruído, porém uma largura de faixa maior é necessária para se transmitir este sinal. Desta forma, um dos desafios da teoria da codificação é conseguir o máximo de proteção possível, adicionando-se a menor quantidade de redundância à informação.

Antes de prosseguirmos, vamos estabelecer algumas definições básicas para um sistema de comunicações. Na Figura 2.1, $\mathbf{u}$ é a saída do codificador de fonte, que pode ser considerada a informação a ser transmitida; $\mathbf{x}$ é a saída do codificador de canal ou a palavra-código e $\mathbf{s}$, o

símbolo da constelação a ser transmitido. O ruído do canal é denominado $\mathbf{n}$, enquanto $\mathbf{r}$ e $\mathbf{y}$ são a saída do canal e a saída do demodulador, respectivamente. Por fim, $\widehat{\mathbf{u}}$ é a estimação da informação transmitida.

2.4.1 Códigos de Bloco e Códigos Convolucionais

De acordo com a forma de geração das palavras-código, os códigos corretores de erros são divididos em duas classes: códigos de bloco e códigos convolucionais. Historicamente, os códigos convolucionais foram preferidos pela comunidade científica por se acreditar que os códigos de bloco não seriam decodificados de maneira eficiente por meio de decisão suave. Entretanto, estudos sobre algoritmos de decodificação por decisão suave para códigos de bloco têm sido realizados com o objetivo de se elucidar tal fato. Como resultado, pode-se afirmar que um dos melhores códigos corretores de erros da atualidade são códigos de bloco: os códigos LDPC irregulares.

A codificação de bloco pode ser definida como uma operação sem memória pelo fato de as palavras-código serem independentes umas das outras. Enquanto isso, os codificadores convolucionais têm sua saída dependente da entrada atual e da entrada e saída anteriores, ou seja, eles dependem de uma memória. Embora possuam diferenças significativas, os códigos de bloco e convolucionais apresentam propriedades estruturais comuns. Um exemplo disso é a possibilidade de descrevê-los por meio de uma estrutura de treliça. Os códigos convolucionais são descritos como códigos que possuem uma estrutura de treliça regular ou invariante no tempo, enquanto os códigos de bloco possuem uma estrutura de treliça variante no tempo. No escopo deste trabalho, iremos nos concentrar apenas nos códigos de bloco e suas propriedades.

2.4.2 Códigos de Bloco Lineares

Os códigos de bloco constituem uma das principais técnicas de codificação e controle de erro usadas em comunicações. Eles podem ser definidos como uma classe de códigos na qual o codificador transforma um bloco de informação de k_c dígitos em uma palavra-código de n_c dígitos, construída a partir de um certo alfabeto [5]. Se estes elementos pertencerem

ao conjunto $\{0,1\}$, o código é denominado um código de bloco $C(n_c, k_c)$ *binário* de taxa $r_c = k_c/n_c$. No contexto da álgebra, um código de bloco binário pode ser definido como um subconjunto de 2^{k_c} palavras-código do espaço vetorial $V_2 = \{0,1\}^{n_c}$ ou um subespaço vetorial de V_2. Cada uma das 2^{k_c} sequências de bits de informação são mapeadas em uma das 2^{n_c} possíveis palavras-código de V_2, formando um subconjunto com 2^{k_c} palavras-código. Se a soma de duas palavras-código válidas resultar em outra palavra-código válida, o código de bloco é denominado *linear*. Iremos nos restringir aos códigos de bloco lineares binários, uma vez que os demais não fazem parte do propósito desta tese.

Matriz Geradora e Matriz de Verificação de Paridade

Os códigos de bloco lineares binários também podem ser descritos como o conjunto de todos os vetores de n_c bits formados pela combinação linear sobre $GF(2)$ (do inglês, *Galois Field (Binary)*) de k_c vetores de base linearmente independentes $\mathbf{g}_1, \mathbf{g}_2, ..., \mathbf{g}_{k_c}$. Um vetor $\mathbf{x}$ é considerado uma palavra-código em C se, e somente se, ele se encontrar no espaço de vetores k_c-dimensional formado por $\{\mathbf{g}_i\}$. Alternativamente, se $\{\mathbf{g}_i\}$ for arranjado como as linhas de uma matriz de ordem $k_c \times n_c$, denotada por $\mathbf{G}_{k_c \times n_c}$, uma palavra-código $\mathbf{x}$ pode ser expressa como

$$\mathbf{x} = (u_1, u_2, ..., u_{k_c}) \begin{bmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \\ . \\ . \\ . \\ \mathbf{g}_{k_c} \end{bmatrix} = \mathbf{uG}\,, \tag{2.33}$$

em que $\mathbf{u}$ é um vetor de k_c bits de informação e $\mathbf{G}$ é chamada *matriz geradora* do código.

A matriz geradora do código $C(n_c, k_c)$ pode ser representada da seguinte forma:

$$\mathbf{G} = [\mathbf{I}_{k_c}; \mathbf{P}]\,, \tag{2.34}$$

em que $\mathbf{I}_{k_c}$ é a matriz identidade de ordem k_c e $\mathbf{P}_{k_c \times (n_c - k_c)}$ é a matriz de paridade cujos elementos determinam os bits de paridade obtidos a partir dos bits de informação. Os códigos cuja matriz geradora é representada segundo a equação (2.34) são chamados *códigos sistemáticos* e a palavra-código $\mathbf{x}$ gerada pode ser representada como mostrado na Figura 2.3.

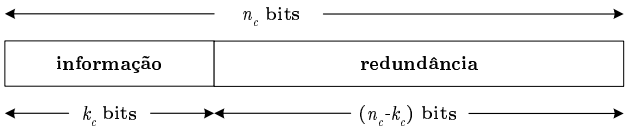


Figura 2.3: Estrutura de uma palavra-código de um código sistemático.

Os códigos de bloco que possuem a matriz $\mathbf{G}$ com estrutura diferente de (2.34) são denominados *não-sistemáticos*. Eles são equivalentes aos sistemáticos, a menos de permutações de colunas e operações elementares sobre as linhas de $\mathbf{G}$, que permitem a conversão de uma forma para outra. Percebe-se a preferência pelos códigos sistemáticos pela facilidade das operações de codificação e decodificação.

Além da matriz geradora $\mathbf{G}$, existe outra forma de se relacionar os bits de informação e os bits de paridade em um código de bloco linear. Seja a matriz $\mathbf{H}_{(n_c-k_c)\times n_c}$ dada por

$$\mathbf{H} = \left[\mathbf{P}^T; \mathbf{I}_{(n_c-k_c)}\right], \tag{2.35}$$

em que $\mathbf{P}^T$ é a transposta da matriz de paridade $\mathbf{P}$ e $\mathbf{I}_{(n_c-k_c)}$ é a matriz identidade de ordem $(n_c - k_c)$. A matriz $\mathbf{H}$ é chamada *matriz de verificação de paridade* e toda palavra-código $\mathbf{x}$ deve satisfazer a equação

$$\mathbf{x}\mathbf{H}^T = 0\,. \tag{2.36}$$

Existe uma conexão entre a matriz geradora e a matriz de verificação de paridade de um código de bloco linear, que se traduz na relação $\mathbf{GH^T} = 0$. Esta relação pode ser interpretada como uma necessidade que cada linha de $\mathbf{G}$, por ser uma palavra-código, tem de satisfazer todas as equações de verificação de paridade.

Vamos exemplificar os conceitos de matriz geradora e matriz de verificação de paridade por meio do código de bloco $C_1(6,3)$. Para este código, temos $2^{k_c} = 2^3 = 8$ vetores de informação e portanto, oito palavras-código. A matriz geradora do código C_1 pode ser dada sistematicamente por

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}, \tag{2.37}$$

em que $\mathbf{g}_1, \mathbf{g}_2$ e $\mathbf{g}_3$ são três vetores linearmente independentes pelos quais podemos gerar todas as palavras-código de C_1. Seja por exemplo, o vetor de informação $\mathbf{u}_1 = [1 \;\; 0 \;\; 1]$. Usando a matriz $\mathbf{G}$ dada em (2.37), podemos gerar a palavra-código $\mathbf{x}_1$ a partir da equação (2.33) da seguinte maneira:

$$\mathbf{x}_1 = [1 \;\; 0 \;\; 1] \begin{bmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \end{bmatrix} = 1 \cdot \mathbf{g}_1 \oplus_2 0 \cdot \mathbf{g}_2 \oplus_2 1 \cdot \mathbf{g}_3 = [1 \;\; 0 \;\; 1 \;\; 0 \;\; 1 \;\; 1] \;\; ,$$

em que o operador $\oplus_2$ representa soma módulo 2. De posse da matriz $\mathbf{G}$, podemos aplicar a equação (2.35) e obter a matriz $\mathbf{H}$ da seguinte maneira:

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \;\; . \tag{2.38}$$

Na próxima seção, veremos que a matriz $\mathbf{H}$ está diretamente relacionada com um parâmetro fundamental de um código corretor de erros chamado *distância mínima*.

2.4.3 Distância Mínima e Capacidade de Correção de um Código

Sejam $\mathbf{x}_1$ e $\mathbf{x}_2$, duas palavras-código de comprimento n_c do código C. A distância de Hamming entre $\mathbf{x}_1$ e $\mathbf{x}_2$, $d_H(\mathbf{x}_1, \mathbf{x}_2)$, é definida como o número de posições em que as palavras-código diferem. Outro conceito importante é o peso de Hamming de uma palavra-código, $w_H(\mathbf{x}_1)$, que é o número de posições não-nulas do vetor $\mathbf{x}_1$. De maneira alternativa, o peso de Hamming também pode ser definido como a distância de Hamming entre o vetor $\mathbf{x}_1$ e o vetor nulo.

Conhecidos os conceitos de distância e peso de Hamming, podemos definir um dos principais parâmetros de um código corretor de erros. A distância mínima d_{min} de um código C é a menor distância de Hamming entre qualquer par de palavras-código. Uma vez que os códigos que estamos considerando aqui são lineares, sabemos que a soma de duas palavras-código resulta em outra palavra-código devido à propriedade de fechamento. Daí, podemos redefinir a distância mínima como o menor peso de Hamming de uma palavra não-nula pertencente ao código C.

A distância mínima de um código C pode ser relacionada com a matriz de verificação de paridade $\mathbf{H}$. Para entender como isto acontece, vamos reescrever a matriz $\mathbf{H}$ como

$$\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_i, \ldots, \mathbf{h}_{n_c}], \tag{2.39}$$

em que $\mathbf{h}_i$ é a i-ésima coluna de $\mathbf{H}$. Sabendo-se que uma palavra-código que contém w bits 1 tem peso w, o peso mínimo de qualquer palavra-código em C é o número mínimo de colunas distintas de $\mathbf{H}$ que são linearmente dependentes. Visto que o peso mínimo de uma palavra-código é igual à distância mínima de Hamming entre qualquer par de palavras-código distintas, o número mínimo de colunas de $\mathbf{H}$ linearmente dependentes define a distância mínima do código.

A distância mínima d_{min} é a propriedade fundamental de um código que determina sua capacidade de detectar e corrigir erros. Um código é capaz de corrigir até t erros em qualquer palavra recebida, se a distância mínima entre as palavras for, pelo menos, $2t+1$. A capacidade de correção t de um código de bloco linear C com distância mínima d_{min} é dada por

$$t = \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor , \tag{2.40}$$

em que $\lfloor \cdot \rfloor$ é a função piso, i.e., ela determina o menor inteiro menor que o valor do seu argumento. Se o código possuir d_{min} ímpar, t assumirá $(d_{min} - 1)/2$ e o código é denominado corretor de t erros. Caso possua d_{min} par, t assumirá $(d_{min} - 2)/2$ e o código é denominado corretor de t erros e detector de $(t + 1)$ erros.

Para consolidar os conceitos definidos nesta seção, vamos ilustrá-los com alguns exemplos de códigos. Em 1950, Hamming forneceu a primeira classe de códigos lineares para correção de erros bastante usada em comunicações digitais [6]. Os chamados *códigos de Hamming* são uma família de códigos de bloco lineares $C(n_c, k_c)$ com $n_c = (2^{m_p} - 1)$ bits de comprimento, $k_c = (2^{m_p} - m_p - 1)$ bits de informação e m_p bits de paridade, tal que $m_p \geq 3$. O exemplo mais simples desta família de códigos é o código de Hamming $C_2(7, 4)$, no qual a matriz $\mathbf{H}$ é composta por $2^{(n_c - k_c)} - 1 = 2^3 - 1 = 7$ colunas não-nulas de $(n_c - k_c) = 3$ bits, conforme

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix} . \tag{2.41}$$

Podemos perceber pela matriz dada em (2.41) que o número de colunas linearmente dependentes de **H** é igual a 3, ou seja, que a distância mínima deste código vale 3. De fato, os códigos de Hamming têm a propriedade de possuir $d_{min} = 3$, independentemente do número de bits de paridade m_p. Daí, o fato da capacidade de correção t dos códigos de Hamming, determinada pela equação (2.40), ser igual a 1. Em outras palavras, os códigos de Hamming são chamados códigos corretores de erros simples.

Dado um código linear $C(n_c, k_c)$ com distância mínima d_{min}, é possível construir um código linear $C_e(n_c+1, k_c)$, chamado *código estendido*, com a adição de um bit de paridade a todas as palavras-código. A estrutura da matriz de verificação de paridade $\mathbf{H}_e$ dos códigos estendidos é ilustrada pela Figura 2.4, em que **H** é a matriz de verificação de paridade do código $C(n_c, k_c)$.

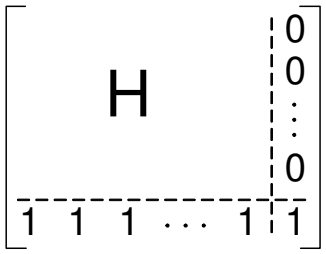


Figura 2.4: `Matriz de verificação de paridade dos códigos estendidos.`

Tomando como exemplo a matriz **H** do código de Hamming $C_2(7, 4)$, podemos obter a matriz de verificação de paridade do código de Hamming estendido $C_e(8, 4)$ conforme a Figura 2.4. Após algumas operações sobre as linhas e permutações sobre as colunas, chegamos à seguinte matriz $\mathbf{H}_e$ equivalente

$$\mathbf{H}_e = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} . \tag{2.42}$$

Como consequência, a distância mínima do código C_e aumenta para 4 e assim, o código de Hamming estendido $C_e(8, 4)$ é denominado corretor de erros simples e detetor de erros duplos.

2.4.4 Representação de Códigos de Bloco por Treliça

A representação de códigos por meio de treliças é mais comum para os códigos convolucionais, porém também é possível fazê-la para os códigos de bloco lineares. A principal diferença entre as treliças que representam os códigos convolucionais e os códigos de bloco é que os últimos possuem estrutura de treliça irregular, visto que um código de bloco é equivalente a uma fonte de Markov variante no tempo [7].

Para representarmos um código de bloco por uma treliça, primeiramente é necessário definir os estados da treliça, assim como em [7]. Considerando S_t , o estado da treliça em um instante de tempo t, podemos descrever a treliça por meio da equação

$$S_t = S_{t-1} + x_t h_t = \sum_{i=1}^{t} x_i h_i \ , \tag{2.43}$$

com $1 \leq t \leq n_c$ e $S_0 = S_{n_c} = 0$. Na equação (2.43), S_0 e S_{n_c} são os estados inicial e final da treliça, respectivamente, h_t é a t-ésima coluna de $\mathbf{H}$ e x_t é o t-ésimo símbolo da palavra-código $\mathbf{x} = [x_1, x_2, \ldots, x_t, \ldots, x_{n_c}]$. A Figura 2.5 ilustra a treliça do código $C_1(6, 3)$, cuja matriz $\mathbf{H}$ é dada por (2.38). Na treliça mencionada, os ramos horizontais representam $x_t = 0$, ao passo que os ramos inclinados representam $x_t = 1$. Por exemplo, para calcularmos S_1 usando a equação (2.43), partimos do estado inicial $S_0 = 0$ e consideramos que x_1 pode assumir 0 ou 1. Para $x_1 = 0$, temos que $S_1 = S_0 = 000$, e para $x_1 = 1$, temos que $S_1 = 000 + 110 = 110$. A partir daí, basta seguir o mesmo raciocínio, considerando todos os valores dos estados anteriores S_{t-1} . É importante ressaltar que o fato de S_{n_c} ser igual a zero implica que certos ramos são proibidos na treliça, pois não terminam em S_{n_c} . Sendo assim, apenas os caminhos que partem de $S_0 = 0$ e chegam a $S_{n_c} = 0$ são palavras-código válidas.

Contudo, à medida que a complexidade do código aumenta, a sua representação convencional por treliça se torna mais complicada. Isto porque as seções intermediárias ficam mais complexas do que as seções das extremidades, deixando a treliça desbalanceada. Para solucionar este problema, podemos representar os códigos de bloco lineares por estruturas de treliças mais simples, chamadas treliças *tail-biting* [8],[9]. Nas treliças *tail-biting*, o estado final está conectado ao estado inicial, ou seja, uma palavra-código válida é representada por

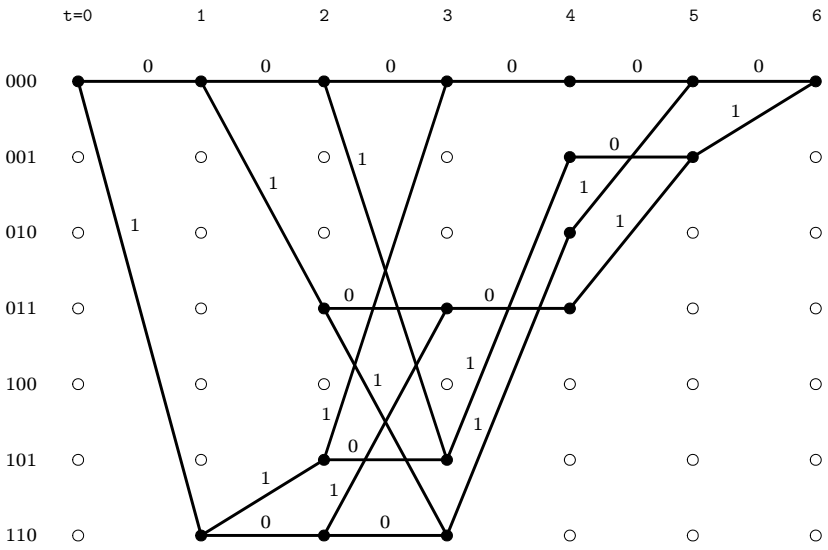


Figura 2.5: Treliça convencional do código $C_1(6,3)$.

um caminho na treliça que começa e termina no mesmo estado. A Figura 2.6 ilustra a treliça *tail-biting* do código $C_1(6,3)$. Comparando as Figuras 2.5 e 2.6, podemos perceber que a complexidade da treliça *tail-biting* é menor, pois o seu número de estados é menor.

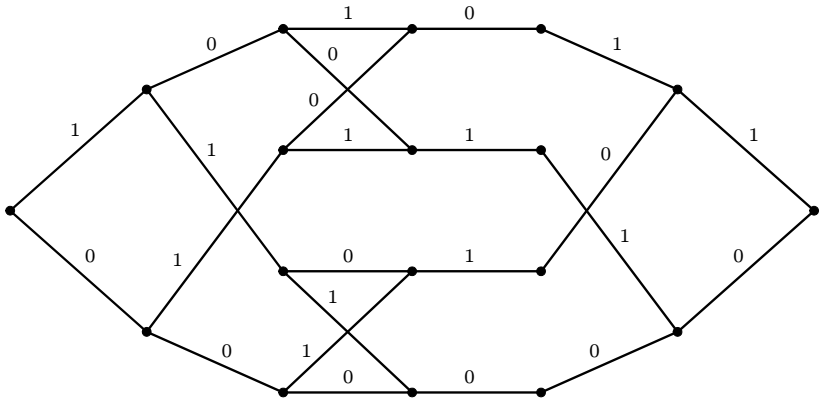


Figura 2.6: Treliça *tail-biting* do código $C_1(6,3)$.

As treliças tail-biting também podem ser construídas de maneira compacta, conforme a treliça do código de Hamming estendido $C_e(8,4)$ ilustrada na Figura 2.7. Nesta treliça, os estados são rotulados de cima para baixo a partir do estado 0. Note que se o estado for rotulado com um índice par, os ramos que chegam neste estado tem rótulo de entrada $u_i = 0$. Porém,

se o estado for rotulado com um índice ímpar, os ramos que chegam nele possuem rótulo de entrada $u_i = 1$. Desta forma, a treliça da Figura 2.7 pode ser obtida pela combinação das seções adjacentes da treliça *tail-biting* tradicional do código $C_e(8, 4)$.

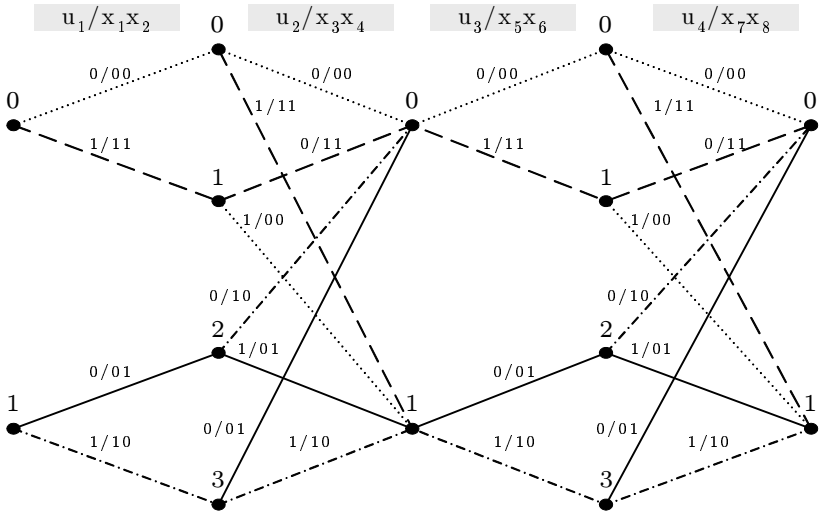


Figura 2.7: Treliça *tail-biting* compacta do código $C_e(8, 4)$.

2.4.5 Regras de Decisão

Na seção anterior, procuramos descrever um pouco da estrutura dos códigos. Foi mencionado que os códigos, além de úteis na detecção de erros, também são fundamentais na correção destes. Entretanto, para poder corrigir os erros introduzidos pelo canal, é preciso estabelecer um mecanismo pelo qual o receptor possa observar o sinal recebido e decidir qual bit ou vetor de informação foi gerado pela fonte. Este mecanismo nada mais é do que um processo de tomada de decisões e pode ser chamado simplesmente de *regras de decisão*. A medida da qualidade da decisão é avaliada pela probabilidade do receptor tomar uma decisão correta, P_C , e é definida por

$$\begin{aligned} P_C &\triangleq P\left[\widehat{\mathbf{u}}(\mathbf{y}) = \mathbf{u}\right] \\ &= \sum_{\mathbf{y}} P\left(\widehat{\mathbf{u}}\left(\mathbf{y}\right)|\mathbf{y}\right) P\left(\mathbf{y}\right) \ , \end{aligned} \tag{2.44}$$

O objetivo do processo de tomada de decisão é maximizar P_C e para tal, existem duas regras mais conhecidas: a regra de decisão MAP (*Maximum-A-Posteriori*) e a regra de decisão MV (*Máxima Verossimilhança*) .

Regras de Decisão MAP e MV

Visto que o objetivo da tomada de decisões pelo receptor é maximizar a probabilidade de acertar a sequência de informações enviada pela fonte, podemos ver pela equação (2.44) que isto corresponde a maximizar $P\left(\widehat{\mathbf{u}}(\mathbf{y})\,|\mathbf{y}\right)$. Sendo assim, *a regra de decisão MAP* é definida como

$$\widehat{\mathbf{u}}_{MAP}(\mathbf{y}) = \arg\max_{\mathbf{u}} P(\mathbf{u}\,|\mathbf{y}) \ . \tag{2.45}$$

Esta regra de decisão é denominada MAP porque maximiza a probabilidade a posteriori $P(\mathbf{u}|\mathbf{y})$ para uma dada sequência $\mathbf{y}$. A regra MAP dá origem a um dos algoritmos de decodificação mais conhecidos da literatura, o algoritmo BCJR [7].

Uma vez que a maximização de P_C independe de $P(\mathbf{y})$, podemos reescrever a equação (2.45) tal que

$$\begin{aligned} \widehat{\mathbf{u}}_{MAP}(\mathbf{y}) &= \arg\max_{\mathbf{u}} \frac{P(\mathbf{u},\mathbf{y})}{P(\mathbf{y})} \\ &= \arg\max_{\mathbf{u}} P(\mathbf{u},\mathbf{y}) \\ &= \arg\max_{\mathbf{u}} P(\mathbf{y}\,|\mathbf{u})P(\mathbf{u}) \ . \end{aligned} \tag{2.46}$$

Considerando que na maioria dos casos $\mathbf{u}$ é um vetor uniformemente distribuído, podemos definir uma nova regra de decisão

$$\widehat{\mathbf{u}}_{MV}(\mathbf{y}) = \arg\max_{\mathbf{u}} P(\mathbf{y}\,|\mathbf{u}) \ . \tag{2.47}$$

Esta regra de decisão é conhecida como *regra de decisão MV*, na qual se busca maximizar a probabilidade de transição do canal $P(\mathbf{y}|\mathbf{u})$. Em geral, a regra MV é mais facilmente implementada. Um dos exemplos mais tradicionais de uso desta regra é o algoritmo de Viterbi [10]. Outro ponto importante é que, quando os símbolos da fonte são equiprováveis, as regras MAP e MV são equivalentes. Geralmente, a regra MV não maximiza P_C pelo fato de não

haver garantia de que $\mathbf{u}$ seja uniforme. Em aplicações práticas, nas quais $P(\mathbf{u})$ é desconhecida, a regra MV é usada mesmo não sendo uma regra ótima.

Considerando que há um mapeamento um-para-um entre a sequência de informação $\mathbf{u}$ e a palavra-código $\mathbf{x}$, e que o canal é sem memória, podemos reescrever as regras MAP e MV tais que

$$\widehat{\mathbf{u}}_{MAP}(\mathbf{y}) = \arg\max_{\mathbf{x}} P(\mathbf{x}\,|\mathbf{y}\,) = \arg\max_{\mathbf{x}} \left[\prod_{i=1}^{n_c} P\left(x_i\,|y_i\right)\right] \tag{2.48}$$

$$\widehat{\mathbf{u}}_{MV}(\mathbf{y}) = \arg\max_{\mathbf{x}} P(\mathbf{y}\,|\mathbf{x}) = \arg\max_{\mathbf{x}} \left[\prod_{i=1}^{n_c} P\left(y_i\,|x_i\right)\right] \ . \tag{2.49}$$

Regras de Decisão de Bloco e de Bit

Na subseção anterior, as regras de decisão MAP e MV foram definidas no contexto da decisão de bloco. A *regra de decisão de bloco* é usada para estimar a probabilidade da sequência recebida ser uma sequência do código. Em suma, a regra de decisão de bloco, seja ela MAP ou MV, tem como objetivo maximizar $P_C = P(\widehat{\mathbf{u}} = \mathbf{u})$, ou equivalentemente, minimizar a probabilidade de erro de bloco $P_B = P(\widehat{\mathbf{u}} \neq \mathbf{u})$. Esta regra de decisão é representada pelas equações (2.48) e (2.49).

Seja agora u_i, o i-ésimo bit de informação da sequência $\mathbf{u}$. Se o receptor for utilizado para determinar cada bit u_i, então a regra de decisão é denominada *regra de decisão de bit*. A regra de decisão de bit tem por finalidade maximizar a probabilidade de acerto de bit $P_c = P(\widehat{u}_i = u_i)$, ou minimizar a probabilidade de erro de bit $P_b = P(\widehat{u}_i \neq u_i)$. Considerando novamente que o canal é sem memória, as regras de decisão de bit MAP e MV são dadas, respectivamente, pelas equações

$$\widehat{u}_{i\,MAP}(\mathbf{y}) = \arg\max_{u} \left[\sum_{\mathbf{u}:u_i=u} \prod_{i=1}^{n_c} P\left(x_i\,|y_i\right)\right] \tag{2.50}$$

e

$$\widehat{u}_{i\,MV}(\mathbf{y}) = \arg\max_{u} \left[\sum_{\mathbf{u}:u_i=u} \prod_{i=1}^{n_c} P\left(y_i\,|x_i\right)\right] \ . \tag{2.51}$$

2.5 Síntese do Capítulo

Este capítulo ofereceu conceitos básicos e definições importantes com o objetivo de facilitar o entendimento dos estudos feitos neste trabalho. Ele abordou inicialmente, de maneira sucinta, a principal função do projetista de um sistema de comunicações. Em seguida, mencionou a questão da inserção do ruído como um dos principais problemas a serem resolvidos no projeto do sistema. Foram expostas algumas definições básicas da Teoria da Informação e da Codificação, ferramentas matemáticas importantes na busca por soluções sistêmicas. Sobre a Teoria da Informação, definimos conceitos sobre entropia, informação mútua e capacidade de canal. Por fim, sobre a Teoria de Codificação, definimos conceitos importantes sobre códigos de bloco lineares, a representação destes por meio de treliças e as regras de decisão utilizadas pelo receptor.

3

Sobre a Capacidade do Canal M-APSK/AWGN Não-Coerente de Bloco

O cálculo da capacidade de canal em forma fechada é possível para o caso de canais simples, como o canal AWGN. Entretanto, para outros tipos de canais como, por exemplo, os canais de comunicação sem fio, a capacidade só pode ser obtida numericamente ou por meio de limitantes. Este Capítulo trata de um estudo sobre a capacidade de canal AWGN não-coerente de bloco utilizando um esquema de modulação com número finito de sinais. Inicialmente, são apresentadas as constelações utilizadas no transmissor, bem como considerações sobre seus parâmetros e configurações. Em seguida, o Capítulo aborda o modelo matemático do canal estudado, algumas definições e considerações importantes a respeito da capacidade de canal. Uma proposta para cálculo de limitantes de capacidade do canal não-coerente é apresentada. Posteriormente, apresentamos a extensão de um algoritmo existente na literatura, com o objetivo de calcular simultaneamente os parâmetros da constelação escolhida e a distribuição de entrada que levam à capacidade do canal coerente. Finalmente, resultados numéricos relativos ao cálculo dos limitantes são expostos e conclusões são apresentadas.

3.1 Introdução

Em sistemas de transmissão sem fio, os sinais transmitidos normalmente chegam ao receptor com um deslocamento na fase da sua portadora. Este deslocamento pode ser provocado por atrasos no sinal devido ao meio de propagação, desvanecimento, mobilidade entre transmissor e receptor, ou ainda, instabilidades nos osciladores utilizados nos circuitos. Quando estes deslocamentos de fase permanecem estáveis por um intervalo de tempo longo, é possível estimá-los no receptor e aplicar a detecção coerente de maneira eficiente. Entretanto, em muitos sistemas de transmissão, uma recepção coerente satisfatória é difícil de ser alcançada. Daí, é necessário que o receptor utilize outras técnicas de detecção, como a detecção não-coerente ou diferencial, na qual uma referência de fase não é exigida. Para isso, precisamos adotar um modelo que descreva como se comporta a rotação de fase introduzida pelo canal no sinal transmitido.

Um modelo de canal bem comum é aquele no qual a rotação de fase desconhecida é considerada constante sobre um bloco de L símbolos e independente de bloco a bloco. O parâmetro L, que determina o número de símbolos de um bloco, é mais comumente conhecido na literatura como *intervalo de coerência do canal*. Este modelo é bem apropriado para sistemas de salto em frequência, nos quais a fase é considerada constante durante o período de um salto e tem o seu valor alterado de salto para salto de maneira arbitrária. A justificativa para o uso deste modelo de canal é que, além da dinâmica da fase ser caracterizada em função do intervalo de coerência do canal, o modelo também é bastante simples.

Apesar de sua simplicidade, existem algumas questões importantes a respeito deste modelo que continuam sendo investigadas na literatura, como por exemplo, a capacidade de canal. A comunidade científica tem dado muitas contribuições a respeito dos canais AWGN não-coerentes. A capacidade do canal AWGN não-coerente de bloco para o caso em que os símbolos de entrada estão restritos à modulação M-PSK (do inglês, *M-ary Phase Shift Keying*) foi investigada por Peleg e Shamai [11]. Eles demonstraram que a capacidade é atingida por uma distribuição de símbolos de entrada u.i.i.d.. Para o caso de sobreposição

de um símbolo, limitantes superiores e inferiores para a capacidade também foram obtidos. Colavolpe e Raheli fizeram uma caracterização parcial da distribuição de entrada que leva à capacidade para modulação contínua [12]. Foi mostrado que o sinal de entrada que maximiza a informação mútua é composto de L variáveis complexas, cuja distribuição de fases é u.i.i.d. e independente da distribuição de amplitudes.

Nuriyev e Anastasopoulos provaram que a densidade de entrada que leva à capacidade de um canal AWGN não-coerente de bloco, considerando entrada contínua, possui simetria circular [13]. A densidade de entrada é também discreta em amplitude com um número infinito de pontos, sendo um deles na origem. Também foi demonstrado em [13] que a capacidade do canal, em algumas faixas de valores, é bem aproximada pela capacidade que considera apenas dois pontos de amplitude, com um deles na origem. Entretanto, para taxas acima de 1,4 bits por uso do canal, usar um esquema de sinalização sem um ponto na origem não afeta significativamente a capacidade.

Como modulações com um número finito de sinais são utilizadas em aplicações práticas, um estudo comparativo de capacidade de algumas modulações foi também realizado em [13]. Alguns desses esquemas de modulação discreta buscam se aproximar da estrutura circularmente simétrica sugerida, como por exemplo, as constelações PSK. Entretanto, para se trabalhar com sistemas de eficiência espectral alta, costuma-se utilizar modulações cujos sinais variam em fase e amplitude, como as constelações QAM (do inglês, *Quadrature Amplitude Modulation*). Por esta razão, iremos utilizar uma classe especial de constelações QAM composta por anéis PSK concêntricos e alinhados denominada M-APSK [14]. A segunda razão que justifica a utilização de tais constelações é o fato de serem apropriadas à aplicação de codificação diferencial na transmissão.

3.2 Constelações M-APSK

Consideraremos constelações de sinais APSK compostas de N anéis de amplitudes distintas, cada um contendo P valores de fase alinhados. Os raios dos anéis diferem por um fator constante r denominado *razão de raio*. Tais constelações são denotadas M-APSK (N, P), com

$M = NP$. A Figura 3.1 ilustra dois exemplos de constelações para $N = 2$, com $P = 4$ e $P = 8$, respectivamente.

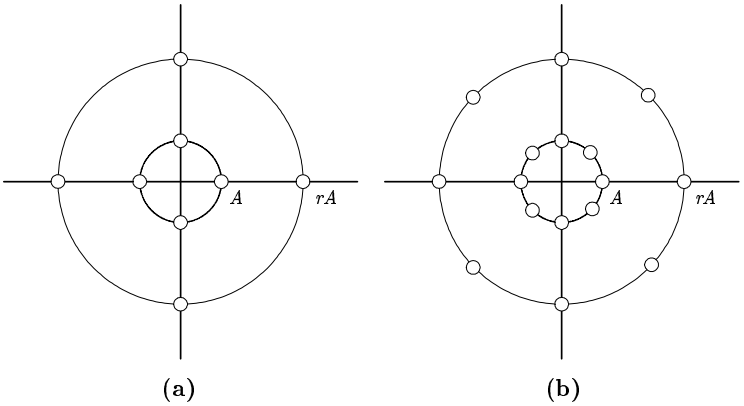


Figura 3.1: Diagramas das constelações APSK com dois níveis de amplitude A e rA. (a) 8-APSK $(2, 4)$. (b) 16-APSK $(2, 8)$.

Observa-se que também é possível representar a constelação 16-APSK numa outra configuração com quatro níveis de amplitude e quatro valores de fase por cada nível. Tal configuração é denominada 16-APSK(4,4) segundo a definição exposta anteriormente. Desta forma, surge uma questão natural a respeito de qual configuração seria mais apropriada para o sistema.

Para responder esta pergunta, vamos utilizar como parâmetro a capacidade efetiva do canal M-APSK/AWGN coerente, denotada por C^*. A capacidade efetiva é obtida quando assumimos que a distribuição de entrada é uniforme. A energia média dos símbolos APSK vale E_s, ao passo que a Relação Sinal-Ruído (RSR) é definida como E_s/N_0. Para um valor fixo de RSR, a capacidade efetiva do canal APSK depende da razão de raio. Tal capacidade pode ser calculada experimentalmente pelos métodos de Monte Carlo de maneira eficiente [15]. Logo, podemos obter os valores de r que maximizam C^* para cada valor de RSR.

Inicialmente, verificamos os cálculos de C^* para o canal AWGN usando modulação 8-APSK$(2, 4)$. Fixando-se o valor de E_s/N_0, foi calculada a capacidade efetiva para diversos valores de r, variando no intervalo de $1,1$ a $4,0$. A Figura 3.2 ilustra as curvas de capacidade C^* em função de r. Os valores de r que maximizam C^* para $E_s/N_0 = 5\,dB$ e $E_s/N_0 = 10\,dB$ são $2,4$ e $2,425$, respectivamente. Repetindo estes cálculos para outros valores de E_s/N_0,

podemos otimizar a razão de raio r em função da RSR, conforme ilustrado na Figura 3.3.

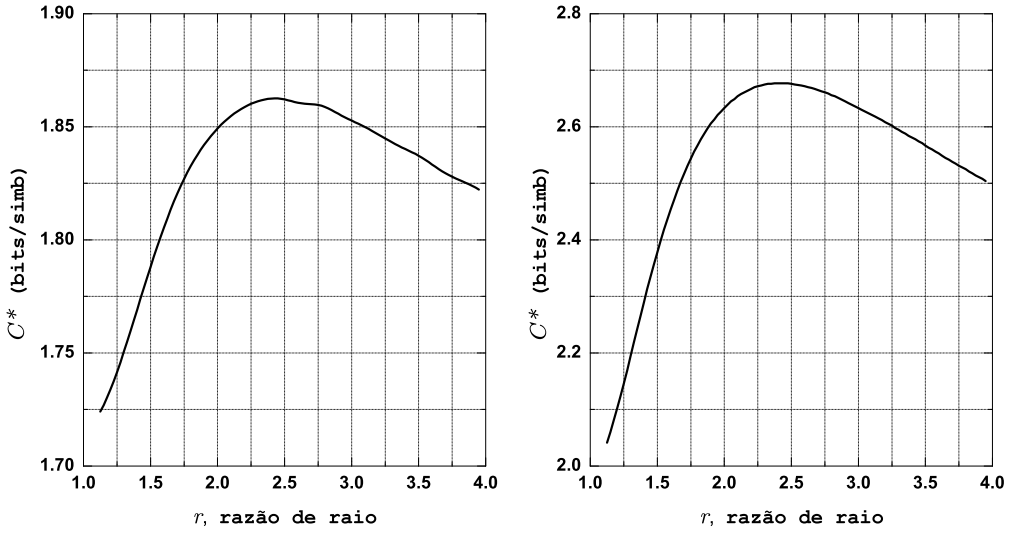


Figura 3.2: Capacidade C^* da modulação 8-APSK$(2,4)$ em função da razão de raio r . (a) $E_s/N_0 = 5\,dB$. (b) $E_s/N_0 = 10\,dB$.

Uma vez que sabemos os melhores valores de r para cada E_s/N_0 específica, podemos encontrar os valores de C^* e assim, obter a curva otimizada de capacidade efetiva de canal. A Figura 3.4 mostra resultados para as constelações 8-APSK$(2,4)$, 16-APSK$(2,8)$ e 16-APSK$(4,4)$. Em relação às constelações de 16 sinais, os resultados indicam que a constelação 16-APSK$(2,8)$ possui capacidade efetiva maior que a 16-APSK$(4,4)$. Portanto, como a configuração dos sinais de uma constelação APSK influencia em sua capacidade efetiva coerente, é de se esperar que influencie também na capacidade de canal não-coerente.

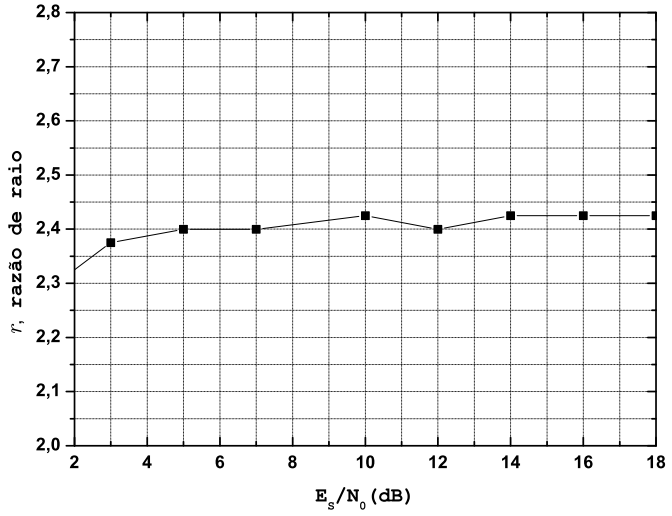


Figura 3.3: Otimização da razão de raio(r) da constelação 8-APSK$(2,4)$ em função de E_s/N_0.

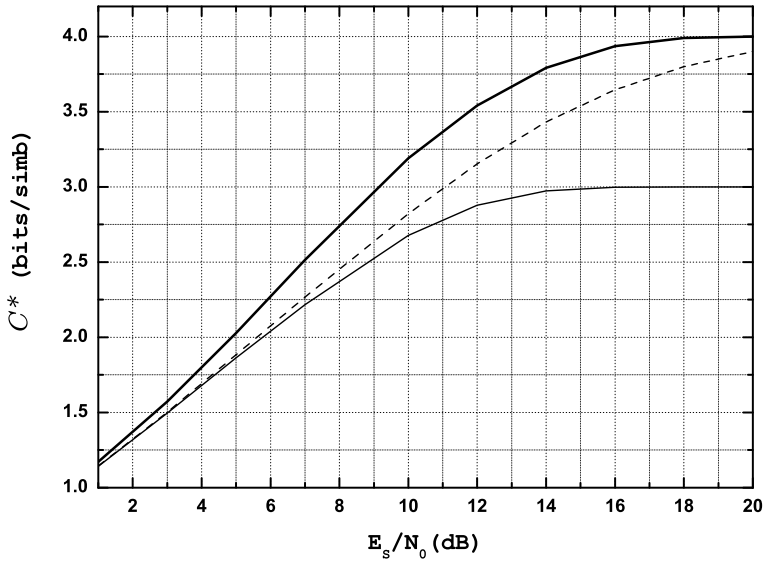


Figura 3.4: Capacidade efetiva coerente para constelações APSK com razão de raio ótima. Linha cheia fina: 8-APSK$(2,4)$, linha tracejada: 16-APSK$(4,4)$, linha cheia grossa: 16-APSK$(2,8)$.

3.3 Modelo do Canal e Capacidade

Vamos apresentar o modelo do canal AWGN não-coerente de bloco e algumas definições importantes. Vale ressaltar que neste Capítulo, abordaremos o sistema não-codificado, sem codificação diferencial e com variáveis definidas no intervalo de coerência do canal (L). Para um melhor entendimento, representaremos tais variáveis por letras maiúsculas em negrito.

A entrada do canal é um vetor de comprimento L, $\mathbf{S} = [s_1, s_2, ..., s_L]$, cujos elementos $s_l = a_l e^{j\phi_l}$ representam os símbolos APSK. A RSR E_s/N_0 determina os valores dos níveis de amplitude da constelação. Estes níveis de amplitude a_l podem assumir um de N valores discretos possíveis e ϕ_l pode assumir uma de P fases discretas, de acordo com o que foi definido anteriormente para uma constelação M-APSK (N, P). A saída também é um vetor de comprimento L, $\mathbf{R} = [r_1, r_2, ..., r_L]$, cujos elementos podem ser expressos como

$$r_l = s_l \exp(j\theta) + n_l\,,\ l = 1, 2, \ldots, L \tag{3.1}$$

em que θ, a fase aleatória introduzida pelo canal, é representada por uma v.a. contínua com distribuição uniforme no intervalo $[0, 2\pi)$. A fase é considerada constante durante cada bloco de L símbolos e varia de maneira independente de bloco a bloco. O ruído AWGN complexo é expresso por n_l , cujas componentes real e imaginária têm média zero e variância $\sigma^2 = N_0/2$ cada uma. Uma vez que cada componente do vetor $\mathbf{S}$ pode ser representada em coordenadas polares, também podemos definir os vetores $\mathbf{A} = [a_1, a_2, ..., a_L]$ e $\mathbf{\Phi} = [\phi_1, \phi_2, ..., \phi_L]$ como vetores componentes de $\mathbf{S}$.

A *Informação Mútua Média* (IMM), I_{nc}, do canal não-coerente descrito anteriormente é definida pela equação

$$I_{nc} = I(\mathbf{S}; \mathbf{R}) = E_{\mathbf{S},\mathbf{R}} \log\left(\frac{p(\mathbf{R}|\mathbf{S})}{p(\mathbf{R})}\right), \tag{3.2}$$

em que $E_{\mathbf{S},\mathbf{R}}$ representa o valor esperado com relação às variáveis $\mathbf{S}$ e $\mathbf{R}$. As densidades de probabilidade de transição $p(\mathbf{R}|\mathbf{S})$ são dadas por [16]

$$p(\mathbf{R}|\mathbf{S}) = \frac{1}{(2\pi\sigma^2)^L} \exp\left[-\frac{1}{2\sigma^2}\sum_{l=1}^{L}\left(|r_l|^2 + |s_l|^2\right)\right] \cdot I_0\left(\frac{1}{\sigma^2}\left|\sum_{l=1}^{L} r_l s_l^*\right|\right), \tag{3.3}$$

em que $I_0(\cdot)$ é a função de Bessel modificada de tipo um e ordem zero. A densidade de probabilidade $p(\mathbf{R})$ pode ser obtida pela equação

$$p\left(\mathbf{R}\right) = \sum_{\mathbf{S}} p\left(\mathbf{R}|\mathbf{S}\right) P\left(\mathbf{S}\right), \tag{3.4}$$

na qual $P(\mathbf{S})$ é a distribuição da entrada do canal.

A capacidade de canal é atingida pela distribuição de $\mathbf{S}$ que maximiza I_{nc}. Para caracterizar essa distribuição que leva à capacidade, vamos considerar a variável de entrada

$$\mathbf{S}^{'} = \left[a_1^{'} e^{j\phi_1^{'}}, a_2^{'} e^{j\phi_2^{'}}, \ldots, a_L^{'} e^{j\phi_L^{'}}\right] ,$$

um vetor aleatório que atinge a IMM máxima denotada por $I_{nc}^{'}$. Seja agora

$$\mathbf{S}^{''} = \mathbf{S}^{'} e^{j\mathbf{\Theta}} = \left[a_1^{'} e^{j(\phi_1^{'} \oplus_{2\pi} \theta_1)}, a_2^{'} e^{j(\phi_2^{'} \oplus_{2\pi} \theta_2)}, \ldots, a_L^{'} e^{j(\phi_L^{'} \oplus_{2\pi} \theta_L)}\right] ,$$

um novo vetor aleatório de entrada contendo o vetor $\mathbf{\Theta}$, cujas componentes θ_l, $l = 1, 2, \ldots, L$ assumem valores discretos do mesmo conjunto associado às componentes $\phi_l^{'}$ e além disso, são u.i.i.d.. O operador $\oplus_{2\pi}$ representa a soma módulo 2π. Sabemos que a IMM entre $\mathbf{S}^{''}$ e $\mathbf{R}$, dado que a variável $\mathbf{\Theta}$ é conhecida, é idêntica à IMM $I(\mathbf{S}^{'}; \mathbf{R})$, ou seja, que $I(\mathbf{S}^{''}; \mathbf{R}|\mathbf{\Theta}) = I_{nc}^{'}$. O descondicionamento de $I(\mathbf{S}^{''}; \mathbf{R}|\mathbf{\Theta})$ em relação à variável $\mathbf{\Theta}$ implica em $I(\mathbf{S}^{''}; \mathbf{R}) \geq I_{nc}^{'}$. A demonstração desta afirmação vem a seguir.

Por definição, sabemos que $I(\mathbf{S}^{''}; \mathbf{R}) = H(\mathbf{S}^{''}) - H(\mathbf{S}^{''}|\mathbf{R})$. Além disso,

$$H(\mathbf{S}^{''}) = H(\mathbf{S}^{'}) + H(\mathbf{\Theta}|\mathbf{S}^{'}) = H(\mathbf{S}^{'}) + H(\mathbf{\Theta}) \tag{3.5}$$

e

$$H(\mathbf{S}^{''}|\mathbf{R}) = H(\mathbf{S}^{'}|\mathbf{R}) + H(\mathbf{\Theta}|\mathbf{S}^{'}, \mathbf{R}) \ . \tag{3.6}$$

Como $H(\mathbf{\Theta}|\mathbf{S}^{'}, \mathbf{R}) \leq H(\mathbf{\Theta})$, temos que

$$H(\mathbf{S}^{''}|\mathbf{R}) \leq H(\mathbf{S}^{'}|\mathbf{R}) + H(\mathbf{\Theta}) \ . \tag{3.7}$$

Assim, substituindo as equações (3.5) e (3.7) na definição de $I(\mathbf{S}^{''}; \mathbf{R})$, temos que

$$\begin{aligned} I(\mathbf{S}^{''}; \mathbf{R}) &\geq H(\mathbf{S}^{'}) - H(\mathbf{S}^{'}|\mathbf{R}) \\ &\geq I(\mathbf{S}^{'}; \mathbf{R}) = I_{nc}^{'} \ . \end{aligned}$$

Portanto, como $I_{nc}^{'}$ é a IMM máxima, a distribuição de $\mathbf{S}^{''}$ também atinge esta IMM. Adicionalmente, podemos dizer que independente da distribuição que a fase da variável de entrada $(\phi_l^{'})$ possua, o fato de somarmos uma nova fase u.i.i.d.(θ_l), que pode assumir os mesmos valores de $\phi_l^{'}$, faz com que a fase resultante também seja u.i.i.d.. Logo, podemos dizer que a capacidade de canal é atingida por uma variável de entrada cuja distribuição de fases $\{\phi_l, l = 1, 2, \ldots, L\}$ é u.i.i.d. e independente da distribuição conjunta de amplitudes $P(a_1, ..., a_L)$. Portanto,

$$P(\mathbf{S}) = P(a_1, \ldots, a_L, \phi_1, \ldots, \phi_L) = P(a_1, \ldots, a_L) \left(\frac{1}{2\pi}\right)^L . \tag{3.8}$$

A prova deste fato é similar à que foi dada em [17] para canais sem memória e com desvanecimento.

Uma vez caracterizada a distribuição de entrada que leva à capacidade de canal, podemos perceber que é necessário obtermos a distribuição conjunta de amplitudes $P(a_1, ..., a_L)$ em N^L dimensões. Para encontrar tal distribuição, são necessários algoritmos numéricos de otimização. Acontece que, para L e M grandes, tais algoritmos computam repetidamente valores de I_{nc} e cada valor obtido depende do cálculo de integrais multidimensionais. Portanto, a utilização de tais algoritmos torna-se bastante complicada e o uso de limitantes de capacidade é bem apropriado.

3.4 Limitantes de Capacidade

3.4.1 Proposta de Limitantes

Foi mencionado que a capacidade de canal é atingida por uma distribuição de fases $\{\phi_l, l = 1, 2, \ldots, L\}$ u.i.i.d.. Desta maneira, os passos para a obtenção dos limitantes são semelhantes àqueles feitos em [11] para sinais M-PSK. A rotação de fase θ é vista como uma entrada adicional do canal com IMM I_v dada por

$$I_v = I(\theta, \mathbf{S}; \mathbf{R}) . \tag{3.9}$$

A partir daí, a regra da cadeia para IMM [4], definida pela equação (2.13), é aplicada a I_v, resultando em

$$I_v = I(\mathbf{S};\mathbf{R}) + I(\theta;\mathbf{R}\,|\mathbf{S}) \ . \tag{3.10}$$

Como $I_{nc} = I(\mathbf{S};\mathbf{R})$, temos que

$$I_{nc} = I_v - I(\mathbf{R};\theta|\mathbf{S}) . \tag{3.11}$$

Aplicando novamente a regra da cadeia para IMM no termo I_v da equação anterior, obtemos que

$$I_{nc} = I(\mathbf{S};\mathbf{R}|\theta) + I(\theta;\mathbf{R}) - I(\theta;\mathbf{R}|\mathbf{S}) . \tag{3.12}$$

Semelhante aos sinais M-PSK, o primeiro termo é a IMM sobre o canal APSK/AWGN coerente, ao passo que $[I(\theta;\mathbf{R}|\mathbf{S}) - I(\theta;\mathbf{R})]$ representa a perda devido à fase desconhecida θ. O limitante superior é obtido pela equação (3.12), considerando que a fase θ é distribuída discreta e uniformemente sobre o mesmo número de fases de entrada, i.e., θ tem a mesma distribuição de ϕ_l.

Para calcularmos o segundo termo, $I(\theta;\mathbf{R})$, iremos considerar o modelo de canal indicado na Figura 3.5. Este é o modelo de um canal de entrada simples e saídas múltiplas (SIMO, do inglês, *Single Input Multiple Output*) [18], no qual θ é a entrada simples.

Seja $\phi_l^* = (\theta \oplus_{2\pi} \phi_l)$, tal que $l = 1, 2, \ldots, L$. Considerando que ϕ_l é u.i.i.d., ϕ_l^* será independente de θ, implicando que $p(r_l|\theta)$ seja independente de θ também. Consequentemente, teremos que

$$I(\theta;\mathbf{R}) = 0 \ . \tag{3.13}$$

Entretanto, se a codificação diferencial, na qual existe um símbolo de referência em cada bloco, for usada no transmissor, podemos escrever que

$$I(\theta;\mathbf{R}) = I(\theta;r_1) . \tag{3.14}$$

O símbolo APSK de referência $s_1 = a_1 e^{j\phi_1}$ pode assumir qualquer um dos $M(= NP)$ valores. Daí, podemos dizer que

$$I(\theta;\mathbf{R}) = I(\theta;r_1|s_1) = \sum_{i=1}^{M} P(s_1 = i) I(\theta;r_1\,|s_1 = i)$$

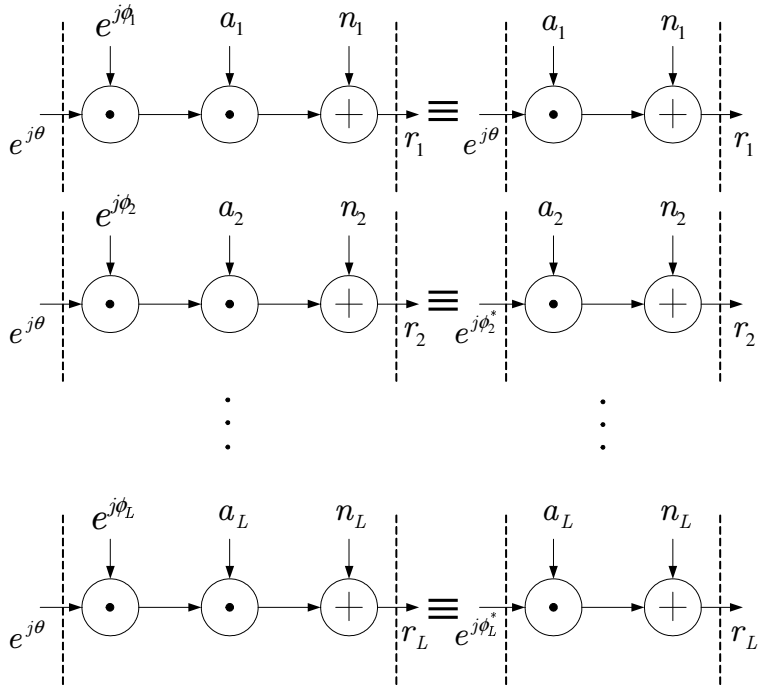


Figura 3.5: Modelo de canal SIMO: uma entrada e L saídas.

ou

$$I\left(\theta; r_1\right) = \sum_{k=1}^{N} P\left(a_1 = r^{(k-1)}A\right) I\left(\theta; r_1 \left| a_1 = r^{(k-1)}A\right.\right) \ , \tag{3.15}$$

já que rotações de fase não alteram a informação mútua.

A partir da equação (3.15), concluímos que $I(\theta; \mathbf{R})$ é calculada como uma média de capacidades de modulações PSK sobre um canal AWGN coerente. Por exemplo, considerando a modulação 8-APSK$(2, 4)$, temos que

$$I\left(\theta; \mathbf{R}\right) = P(a_1 = A)\ C_{c-4PSK(A)} + P(a_1 = rA)\ C_{c-4PSK(rA)} \ ,$$

em que $C_{c-4PSK(A)}$ e $C_{c-4PSK(rA)}$ são capacidades 4-PSK para dois níveis de amplitude, A e rA, respectivamente. Os termos $P(a_1 = A)$ e $P(a_1 = rA)$ são as probabilidades *a priori* dos níveis de amplitude da constelação.

Finalmente, o termo $I(\theta; \mathbf{R}|\mathbf{S})$ é dado pela equação [19]

$$I\left(\theta; \mathbf{R}|\mathbf{S}\right) = \sum_{\boldsymbol{\alpha}} P_{\mathbf{S}}\left(\boldsymbol{\alpha}\right) I\left(\theta; \mathbf{R}|\mathbf{S} = \boldsymbol{\alpha}\right) \ . \tag{3.16}$$

Usando novamente o conceito de canal de entrada simples e saídas múltiplas, $I(\theta;\mathbf{R}|\mathbf{S})$ é obtida pelo cálculo de $I(\theta;r_l|s_l)$, a IMM da l-ésima componente, com RSR aumentada de um fator L. Sendo assim, temos que

$$I\left(\theta;\mathbf{R}|\mathbf{S}\right) = I\left(\theta;r_l|s_l\right)|_{(L\cdot RSR)} = \sum_{i=1}^{M} P\left(s_l = i\right) I\left(\theta;r_l|s_l = i\right). \tag{3.17}$$

Pelo mesmo raciocínio aplicado na obtenção da equação (3.15), temos que

$$I\left(\theta;\mathbf{R}|\mathbf{S}\right) = I\left(\theta;r_l|a_l\right)|_{(L\cdot RSR)} = \sum_{k=1}^{N} P\left(a_l = r^{(k-1)}A\right) I\left(\theta;r_l|a_l = r^{(k-1)}A\right). \tag{3.18}$$

O limitante inferior também é obtido a partir da equação (3.12), porém sabendo-se que a fase desconhecida θ tem distribuição uniforme e contínua. Semelhante à [11], incorporamos a inequação

$$I\left(\theta;\mathbf{R}\right) \geq I\left(\theta;r_1\right)$$

à equação (3.12), resultando em

$$I_{nc} \geq I\left(\mathbf{S};\mathbf{R}|\theta\right) + I\left(\theta;r_1\right) - I\left(\theta;\mathbf{R}|\mathbf{S}\right). \tag{3.19}$$

O primeiro termo do lado direito da equação (3.19) é equivalente ao primeiro termo do limitante superior. O segundo termo, $I(\theta;r_1)$, também é dado pela equação (3.15), porém com a fase θ contínua. Cada IMM $I(\theta;r_1|a_1 = r^{(k-1)}A)$ se iguala à capacidade de um canal coerente com fase de entrada contínua [20]. O terceiro termo da equação (3.19) também é equivalente ao segundo termo do limitante superior, exceto pela necessidade do cálculo de capacidades para um canal com entrada simples contínua. As capacidades de canal com entrada contínua, assim como as com entrada discreta, também foram determinadas experimentalmente por métodos de Monte Carlo.

Segundo o que foi visto até agora, para calcularmos os limitantes superiores e inferiores de capacidade, precisamos obter as probabilidades *a priori* $P(a_l)$. Tais probabilidades constituem, na verdade, a distribuição marginal de amplitudes, obtida a partir da distribuição conjunta $P(a_1,...,a_L)$. Encontrar esta distribuição marginal é uma tarefa árdua de se realizar, visto a dificuldade de se obter a distribuição $P(a_1,...,a_L)$, conforme explicado anteriormente.

Considerando o caso da capacidade efetiva, ou seja, a capacidade obtida partindo da premissa que a distribuição de entrada é uniforme, podemos dizer que $P(a_l) = 1/N$. Para este caso, limitantes de capacidade foram exibidos em [21]. Adicionalmente, resultados para distribuição de entrada não-uniforme foram relatados em [22]. Nos dois casos, os resultados indicaram que para valores de L grande, a capacidade do canal não-coerente converge para a capacidade do coerente. Portanto, iremos utilizar a distribuição de amplitudes $P(a_l)$ que atinge a capacidade do canal M-APSK/AWGN coerente. Esta distribuição pode ser obtida por meio da extensão do algoritmo proposto em [23], aplicando-o às constelações M-APSK com parâmetros ótimos A e r.

3.4.2 Algoritmo de Otimização para Canais Coerentes

O problema de otimização da distribuição de entrada que maximiza a informação mútua de um determinado canal é um problema de otimização convexa desde que a distribuição de entrada seja a única incógnita a ser determinada. Para o nosso caso, temos que o canal depende também dos parâmetros A e r da constelação M-APSK. Logo, encontrar a distribuição de entrada que leva à capacidade de canal juntamente com A e r ótimos não caracteriza um problema de otimização convexa, independente do canal ser coerente ou não-coerente. Entretanto, uma vez fixados os valores de A e r, nos deparamos com o problema tradicional de otimização mencionado anteriormente.

Consideraremos um canal AWGN sem memória e um alfabeto de entrada limitado a um conjunto finito de símbolos $\Omega = \{x_1, x_2, \ldots, x_{kl}, \ldots, x_M\} \subset \mathbb{R}^2$, tal que $x_{kl} = b_k e^{j\phi_l}$, com $k = 1, 2, \ldots, N$ e $l = 1, 2, \ldots, P$, conforme a constelação M-APSK definida na Seção 3.2. A distribuição de fase dos símbolos de entrada é considerada independente da distribuição de amplitude e uniformemente distribuída, ou seja, $p(x_i) = p(b_k)\, p(\phi_l)$, com $p(\phi_l) = 1/P$. Para simplificar a notação, iremos denotar as probabilidades $p(b_k)$ e $p(\phi_l)$ como p_k e p_l, respectivamente. Desta maneira, podemos definir a função de distribuição de probabilidade das amplitudes de entrada como $\mathbf{p} = \{p_1, ..., p_k, ..., p_N\}$. A potência do kl-ésimo símbolo da constelação é dada por b_k^2. Assumiremos que a variância σ^2 do ruído por dimensão e que a

potência média máxima P_m da constelação são conhecidas. A fase aleatória introduzida pelo canal é conhecida pelo receptor. Na saída do canal, será observada uma variável aleatória contínua Y. Uma vez definidas nossas variáveis, estamos interessados em calcular

$$C = \max_{\mathbf{p},A,r} I\left(B, \Phi; Y\right) \tag{3.20}$$

sob as restrições

$$\sum_{k=1}^{N} p_k = 1 \tag{3.21}$$

e

$$\sum_{k=1}^{N} p_k b_k^2 \leq P_m \ , \tag{3.22}$$

em que A e r são os parâmetros da constelação APSK, que juntamente com as fases ϕ_l definem a localização dos sinais de entrada. A restrição (3.22) é equivalente à restrição $\sum_{k=1}^{N} p_k b_k^2 = P_m$ [23].

Seja a informação mútua $I(B, \Phi; Y)$ dada pela equação

$$\begin{aligned} I\left(B, \Phi; Y\right) &= H\left(B, \Phi\right) - H\left(B, \Phi | Y\right) \\ &= \sum_{k=1}^{N} p_k \log\left(\frac{1}{p_k}\right) + \log P + \sum_{k=1}^{N} p_k T_k \ , \end{aligned} \tag{3.23}$$

na qual $T_k \triangleq \dfrac{1}{P}\sum_{l=1}^{P} T_{kl}$. T_{kl} é definido como o elemento de uma matriz $\mathbb{T}_{N\times P}$ e dado pela integral

$$T_{kl} = \int f\left(y | b_k, \phi_l\right) \log p\left(b_k, \phi_l | y\right) \, dy \ , \tag{3.24}$$

na qual

$$f\left(y | b_k, \phi_l\right) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{\left\| y - b_k e^{j\phi_l} \right\|^2}{2\sigma^2}\right)$$

e $p\left(b_k, \phi_l | y\right)$ é a probabilidade *a posteriori* $Pr\left(B = b_k, \Phi = \phi_l |\ Y = y\right)$.

Sendo assim, a primeira parte do algoritmo é descrita da seguinte forma:

Passo 1: `Fixar o valor de` A `e` r. `Escolher` $p_k \geq 0$ $(k = 1, 2, \ldots, N)$, `tal que` $\sum_{k=1}^{N} p_k = 1$.

Passo 2: Para um valor fixo de p_k $(k = 1, 2, \ldots, N)$, calcular $T_k = \dfrac{1}{P}\sum_{l=1}^{P} T_{kl}$, em que T_{kl} é dado pela equação (3.24).

Como é difícil obtermos uma expressão fechada para a integral dada pela equação (3.24), podemos calcular T_{kl} experimentalmente por meio de métodos de Monte Carlo, segundo a equação

$$T_{kl} = E_Y \left[\frac{p\left(b_k, \phi_l | y\right)}{p_k p_l} \log(p\left(b_k, \phi_l | y\right)) \right] \ , \tag{3.25}$$

na qual E_Y é o valor esperado.

Passo 3: Para cada valor fixo de T_k $(k = 1, 2, \ldots, N)$, encontrar

$$\mathbf{p} = \arg\max_{\mathbf{q}} \left(\sum_{k=1}^{N} q_k \left[\log\left(\frac{1}{q_k}\right) + T_k \right] + \log P \right)$$

sob as restrições $\sum_{k=1}^{N} q_k = 1$ e $\sum_{k=1}^{N} q_k b_k^2 = P_m$.

Para encontrarmos o valor máximo de $\mathbf{p}$ no Passo 3 da primeira parte do algoritmo, aplicaremos o método dos Multiplicadores de Lagrange. Para isso, definimos a função $F(\mathbf{p}, \mathbf{T}, \lambda_1, \lambda_2)$ por meio da expressão

$$\begin{aligned} F\left(\mathbf{p}, \mathbf{T}, \lambda_1, \lambda_2\right) = & \left(\sum_{k=1}^{N} p_k \left[\log_2\left(\frac{1}{p_k}\right) + T_k \right] \right) + \log_2 P + \\ & \lambda_1 \left[\left(\sum_{k=1}^{N} p_k \right) - 1 \right] + \lambda_2 \left[\left(\sum_{k=1}^{N} p_k b_k^2 \right) - P_m \right] \end{aligned} \tag{3.26}$$

na qual λ_1 e λ_2 representam os multiplicadores de Lagrange. Calculando as derivadas de $F(\mathbf{p}, \mathbf{T}, \lambda_1, \lambda_2)$ com relação às probabilidades p_k e igualando-as a zero para $k = 1, 2, \ldots, N$, obtemos que

$$p_k = \frac{2^{(T_k + \lambda_2 b_k^2)}}{\sum_{k=1}^{N} 2^{(T_k + \lambda_2 b_k^2)}} \tag{3.27}$$

e

$$\sum_{k=1}^{N} \left(P_m - b_k^2\right) \cdot 2^{(T_k + \lambda_2 b_k^2)} = 0 \quad . \tag{3.28}$$

Assim como em [23], a solução das equações (3.27) e (3.28) atingem o máximo global de $G(\mathbf{p},\mathbf{T}) = \sum_k p_k \left[\log\left(\frac{1}{p_k}\right) + T_k\right]$ sob as restrições (3.21) e (3.22) no intervalo $0 \leq p_k \leq 1$ ($k = 1, 2, \ldots, N$). A solução da equação (3.28) (λ_2) pode ser determinada por métodos numéricos, como o método da bisseção por exemplo.

Passo 4: `Repetir os passos 2 e 3 até a convergência do algoritmo e ao final, utilizar a distribuição de probabilidades obtida para calcular a informação mútua candidata à capacidade de canal.`

Na parte 1 deste algoritmo, o valor da informação mútua foi calculado para valores fixos de A e r. Na segunda parte, precisamos calcular a informação mútua para uma faixa de valores dos parâmetros A e r, que será definida para o caso em que $N = 2$ e pode ser estendida para N arbitrário.

A partir deste momento, vamos nos referir à distribuição de probabilidades de amplitude como $\{P(a_l = A), P(a_l = rA)\}$ para facilitar o entendimento, pois consideraremos o caso em que $N = 2$. Considerando a restrição de potência média máxima, temos que

$$\sum_{k=1}^{N} P(a_l = r^{(k-1)}A)(r^{(k-1)}A)^2 = A^2[P(a_l = A) + P(a_l = rA)r^2] = P_m \ . \tag{3.29}$$

Por definição, a razão de raio r assume valor mínimo unitário ($r_{min} = 1$). Para obtermos A_{max}, podemos aplicar r_{min} na equação (3.29) e considerar a distribuição de probabilidades de amplitude $\{P(a_l = A) = 1, P(a_l = rA) = 0\}$, ou seja, existe apenas a probabilidade de serem usados os sinais do nível mais interno da constelação APSK. Por outro lado, para calcularmos A_{min}, iremos considerar que apenas os sinais do nível externo serão usados ($\{P(a_l = A) = 0, P(a_l = rA) = 1\}$) e que $r_{max} = r$, ficando a razão de raio máxima como um parâmetro livre do sistema. Desta forma, os valores limites para A são dados por

$$A_{max} = \sqrt{P_m}$$

e

$$A_{min} = \sqrt{\frac{P_m}{r^2}} \ .$$

Primeiramente, aplicamos o algoritmo de otimização mencionado para o canal AWGN utilizando modulação 8-APSK$(2, 4)$. Assumimos o valor máximo de r igual a $4,0$. A Tabela 3.1 indica os resultados otimizados para o primeiro nível de amplitude (A), a razão de raio r e a distribuição de amplitudes $\{P(a_l = A), P(a_l = rA)\}$ para uma ampla faixa de valores de RSR.

RSR(dB)	A	r	$\{P(a_l = A), P(a_l = rA)\}$
-2	1,054	2,6	{0,976 0,024}
0	1,183	2,2	{0,888 0,112}
2	1,392	2,2	{0,834 0,166}
4	1,718	2,4	{0,852 0,148}
6	1,834	2,4	{0,712 0,288}
8	2,309	2,4	{0,712 0,288}
10	2,646	2,4	{0,609 0,391}
13	3,207	2,6	{0,499 0,501}
15	4,037	2,6	{0,499 0,501}
18	5,703	2,6	{0,497 0,503}

Tabela 3.1: Parâmetros A e r otimizados para a constelação 8-APSK$(2, 4)$.

O segundo caso analisado diz respeito à constelação 16-APSK$(2, 8)$. Assim como para a constelação 8-APSK$(2, 4)$, assumimos o valor máximo de r igual a $4,0$. A Tabela 3.2 indica os parâmetros otimizados obtidos pelo algoritmo para a constelação 16-APSK$(2, 8)$.

3.4.3 Limitantes: Resultados Numéricos

Uma vez calculados os parâmetros A, r e a distribuição de amplitudes $\{P(a_l = A), P(a_l = rA)\}$ que levam à capacidade do canal APSK/AWGN coerente, estes serão utilizados para a obtenção dos limitantes superiores e inferiores da capacidade de canal não-coerente. Para compararmos os limitantes propostos com a capacidade de canal coerente, é necessário fazermos a normalização da IMM I_{nc}. Sendo assim, vamos definir C_{nc}, a capacidade de

RSR(dB)	A	r	$\{P(a_l = A), P(a_l = rA)\}$
-2	0,842	2,0	{0,740 0,260}
0	1,001	2,4	{0,791 0,209}
2	1,157	2,4	{0,713 0,287}
4	1,457	2,4	{0,713 0,287}
6	1,834	2,4	{0,712 0,288}
8	2,196	2,2	{0,578 0,422}
10	2,764	2,2	{0,578 0,422}
13	4,106	2,0	{0,544 0,456}
15	5,170	2,0	{0,543 0,457}
18	7,738	1,8	{0,503 0,497}

Tabela 3.2: Parâmetros A e r otimizados para a constelação 16-APSK$(2, 8)$.

canal não-coerente normalizada, como

$$C_{nc} = \frac{\max\{I_{nc}\}}{L_e} \tag{3.30}$$

em que L_e é o número de símbolos da variável de entrada $\mathbf{S}$ que carregam informação.

A proposta apresentada para os limitantes na Seção 3.4 considera a utilização de codificação diferencial na transmissão, ou seja, $L_e = L - 1$. Portanto, tanto o limitante superior quanto o inferior são calculados a partir da normalização da equação (3.12), que é dada por

$$C_{nc} = C_c - \frac{I(\theta; \mathbf{R}\,|\mathbf{S})}{L_e} + \frac{I(\theta; \mathbf{R})}{L_e}\,. \tag{3.31}$$

A Figura 3.6 indica as curvas dos limitantes de capacidade (em bits por símbolo da modulação) versus E_s/N_0 (em dB) para a constelação 16-APSK$(2, 8)$, considerando $L_e = L-1$. As linhas cheias representam os resultados para os limitantes superiores ao passo que as linhas tracejadas representam os limitantes inferiores. Podemos perceber como se dá o comportamento dos limitantes em relação à capacidade de canal coerente em função do intervalo de coerência do canal L. Percebe-se que à medida que L cresce, os limitantes se aproximam da capacidade coerente. Além disso, para $L = 8, 16, 32$, os limitantes coincidem

sobre uma larga faixa de RSR (a diferença entre o superior e o inferior é menor do que 0,2 bit por símbolo).

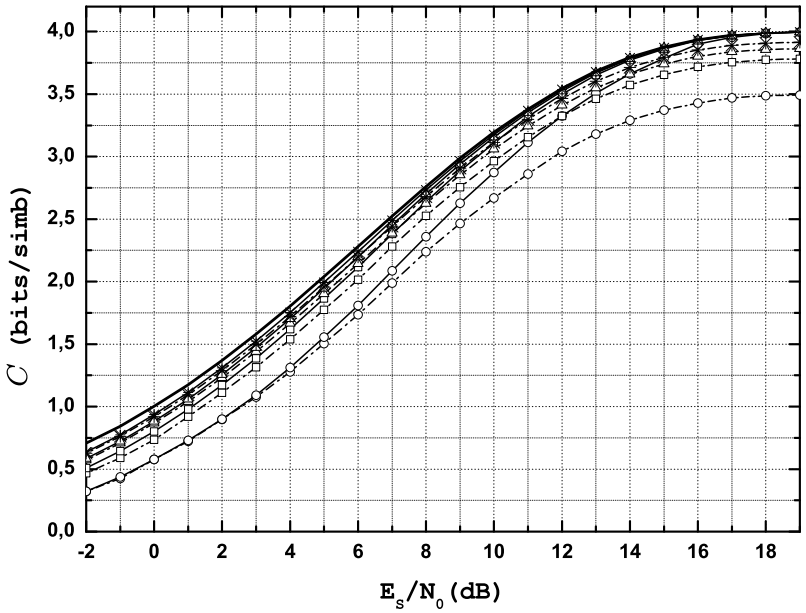


Figura 3.6: Limitantes de capacidade para o canal 16-APSK$(2,8)$/AWGN não-coerente de bloco, considerando $L_e = L-1$. ○ : $L=2$, □ : $L=8$, △ : $L=16$, ∗ : $L=32$, − : capacidade 16-APSK$(2,8)$/AWGN coerente.

Considerando agora que o transmissor não utiliza codificação diferencial, ou seja, $L_e = L$, vamos investigar o que acontece com os limitantes. Neste caso, sabe-se que $I(\theta;\mathbf{R}) = 0$ conforme foi definido na equação (3.13), ao passo que os primeiros termos da equação (3.31) são obtidos da mesma maneira. Portanto,

$$C_{nc} = C_c - \frac{I\left(\theta;\mathbf{R}\,|\mathbf{S}\right)}{L_e}\ . \tag{3.32}$$

A Figura 3.7 ilustra os limitantes para a constelação 16-APSK$(2,8)$, porém assumindo agora que $L_e = L$. Podemos observar que os limitantes superiores e inferiores estão mais próximos um do outro para RSRs mais baixas. Isto pode ser constatado facilmente por meio dos limitantes obtidos para $L = 2$ na região de RSR que varia de $-2\,dB$ a $6\,dB$. Por outro lado, verificou-se que a convergência dos limitantes para a capacidade coerente ocorre de maneira

mais lenta. Este fato é rapidamente evidenciado observando-se os limitantes para $L = 2$ e $L = 8$ na região de RSR acima de $12\,dB$. Sendo assim, podemos concluir que a existência de um símbolo de referência no bloco que caracteriza o canal leva os limitantes a se aproximarem mais rapidamente da capacidade de canal coerente.

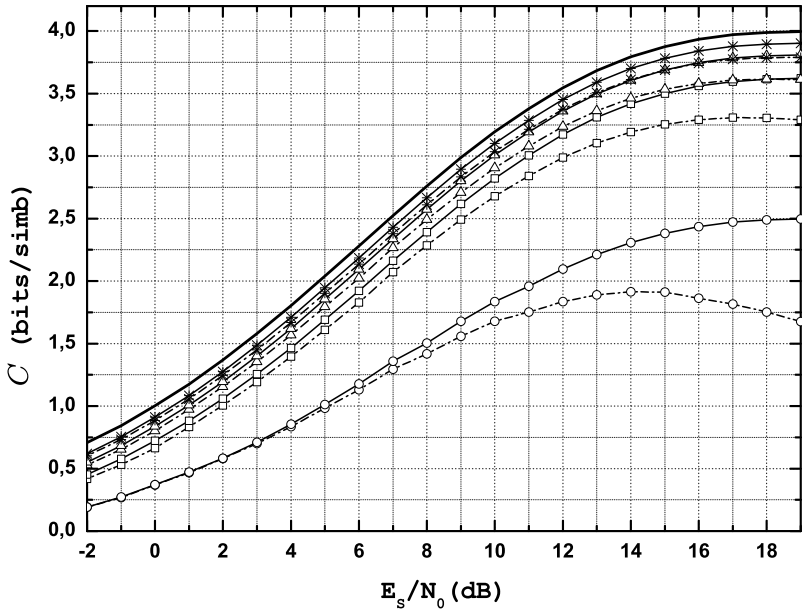


Figura 3.7: Limitantes de capacidade para o canal 16-APSK$(2,8)$/AWGN não-coerente de bloco, considerando $L_e = L$. $\circ$: $L = 2$, $\square$: $L = 8$, $\triangle$: $L = 16$, $*$: $L = 32$, $-$: capacidade 16-APSK$(2,8)$/AWGN coerente.

E finalmente, com o objetivo de tornar a obtenção dos limitantes mais simples, verificamos como eles se comportam para o caso em que a capacidade efetiva, C^*, é usada como referência. Primeiramente, vale relembrar a consideração de que a distribuição de amplitudes dos símbolos de entrada que leva à capacidade do canal M-APSK/AWGN coerente foi usada como distribuição marginal de amplitudes para o cálculo dos limitantes. Portanto, desejamos descobrir o que acontece se esta distribuição for substituída pela distribuição uniforme, assim como em [21].

As Figuras 3.8 e 3.9 ilustram as curvas de capacidade de canal coerente para as constelações

8-APSK$(2, 4)$ e 16-APSK$(2, 8)$, respectivamente. Elas se referem aos casos de distribuição de amplitudes e razão de raio r obtidas pelo algoritmo mencionado na Seção 3.4.2, distribuição de entrada uniforme e r fixo, e por fim, distribuição de entrada Gaussiana (capacidade de Shannon). A capacidade de Shannon é utilizada apenas como referência ilustrativa. Os valores de r foram fixados em $2,42$ para 8-APSK$(2, 4)$ e em $2,0$ para 16-APSK$(2, 8)$. Este último valor foi sugerido também em [24].

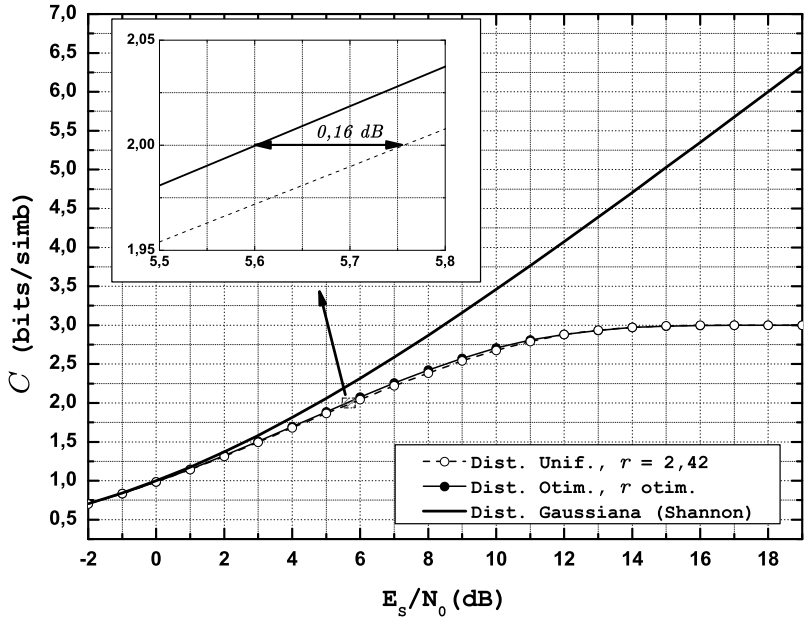


Figura 3.8: Comparação das curvas de capacidade do canal 8-APSK$(2, 4)$ com a capacidade de Shannon.

Para a constelação 8-APSK$(2, 4)$ e considerando uma taxa de 2 bits por símbolo, a distribuição de entrada obtida com o algoritmo de otimização fornece um ganho de RSR de apenas $0,16$ dB nas curvas de capacidade, quando comparada à distribuição de entrada uniforme. De maneira semelhante na constelação 16-APSK$(2, 8)$, para uma taxa de 3 bits por símbolo, existe apenas um pequeno ganho de $0,12$ dB favorável à capacidade obtida com a distribuição de entrada otimizada. É possível perceber por meio das Tabelas 3.1 e 3.2, que para uma RSR acima de 10 dB, a distribuição $\{P(a_l = A), P(a_l = rA)\}$ tende a ser uniforme.

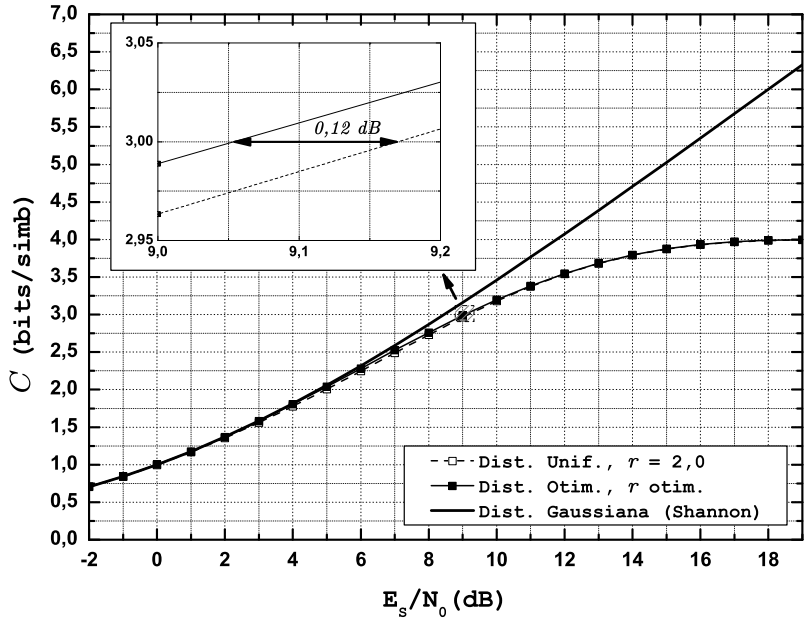


Figura 3.9: Comparação das curvas de capacidade do canal 16-APSK$(2, 8)$ com a capacidade de Shannon.

Sendo assim, comparando os resultados obtidos para as duas constelações APSK mencionadas, concluímos que a otimização da distribuição de entrada para a obtenção da capacidade de canal não fornece um ganho considerável em RSR. Além disso, podemos utilizar um valor fixo de r juntamente com a distribuição uniforme de entrada, sem a necessidade de se otimizar a razão de raio para cada valor de RSR.

A Figura 3.10 ilustra os limitantes de capacidade para a constelação 16-APSK$(2, 8)$, considerando $L_e = L - 1$, $r = 2,0$ e distribuição de entrada uniforme. Comparando tais limitantes com a Figura 3.6, temos que a perda em capacidade (bits por símbolo) não ultrapassa o valor de $0,01$ bit por símbolo para $L = 16$ e 32. Logo, o uso da distribuição uniforme e razão de raio fixa é uma simplificação aceitável no cálculo dos limitantes de capacidade.

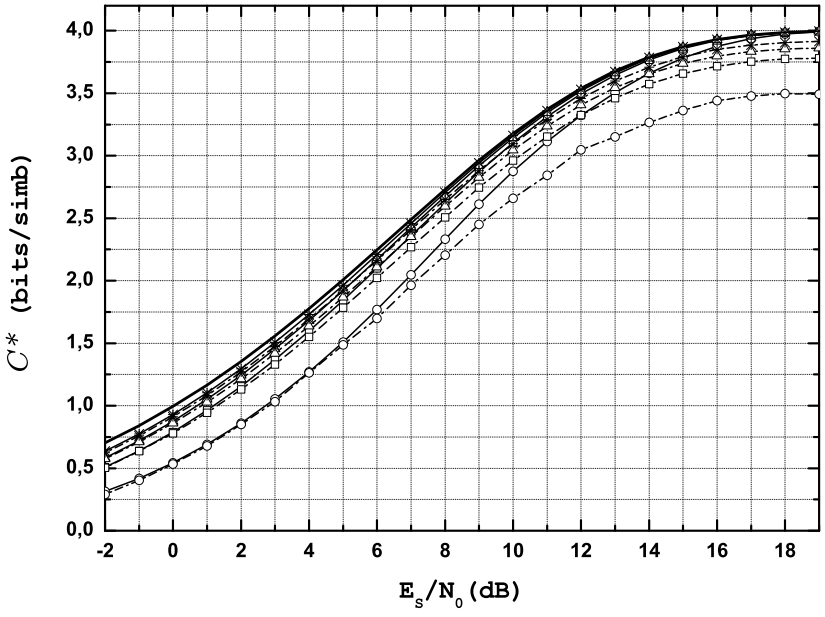


Figura 3.10: Limitantes de capacidade para o canal 16-APSK(2,8)/AWGN não-coerente de bloco com distribuição de entrada uniforme e $L_e = L - 1$. ○ : $L = 2$, □ : $L = 8$, △ : $L = 16$, ∗ : $L = 32$, **-** : capacidade 16-APSK$(2, 8)$/AWGN coerente efetiva.

3.5 Comentários Finais

A capacidade de canais não-coerentes é um tema bastante atrativo que desperta interesse na comunidade científica. O caso particular da capacidade de canal AWGN não-coerente de bloco foi o alvo dos estudos descritos neste Capítulo. Neste caso, assumimos que a amplitude do desvanecimento provocado pelo canal é conhecida no receptor, ao passo que a fase é desconhecida, porém caracterizada por um modelo matemático bem definido. Tal modelo descreve a variação da fase em função do intervalo de coerência do canal, que é considerado conhecido tanto no transmissor quanto no receptor.

Trabalhos encontrados na literatura investigaram a distribuição de entrada que leva à capacidade de canal não-coerente para o caso de não haver restrições nos sinais de entrada [12],[13]. Os símbolos de entrada que maximizam a informação mútua do canal se caracterizam

por possuírem distribuição de fase u.i.i.d. e independente da distribuição de amplitudes. Além disso, a distribuição de amplitudes é discreta, com um número infinitos de pontos, sendo um deles na origem. Para certas regiões de taxa, também foram consideradas distribuições de amplitudes nas quais o ponto da origem foi suprimido, resultando em boas aproximações para a capacidade.

Diante disso, um dos nossos objetivos foi escolher esquemas de sinalização com número finito de pontos e sem ponto na origem, para utilizar em um sistema de transmissão aplicado ao canal estudado. Um dos critérios de escolha das constelações utilizadas foi a caracterização da distribuição de entrada que atinge capacidade para o caso de entrada contínua. A justificativa para tal foi uma analogia ao fato de que no caso da capacidade de canal AWGN coerente, busca-se atingir ao máximo a distribuição Gaussiana por meio de uma constelação finita como, por exemplo, a constelação M-QAM. Sendo assim, procuramos escolher a constelação finita que se aproximasse de certa forma da distribuição de entrada que leva à capacidade do canal AWGN não-coerente. Logo, a constelação M-APSK foi assumida com símbolos cujas fases são u.i.i.d. e independentes das amplitudes. A disposição dos sinais em níveis de amplitude e com fases alinhadas em cada nível foi motivada pela facilidade da utilização de codificação diferencial na transmissão. Por fim, não menos importante, o desejo de propor sistemas espectralmente eficientes também motivou a escolha de tais constelações.

A dificuldade de se caracterizar a distribuição de entrada que atinge capacidade, ou particularmente a distribuição conjunta de amplitudes dos sinais de entrada nos motivou à sugestão do cálculo de limitantes de capacidade. Na verdade, a obtenção dos limitantes também depende da distribuição conjunta, visto que a distribuição marginal de amplitudes depende da conjunta. Entretanto, o fato de a capacidade de canal não-coerente se aproximar da capacidade coerente, para altos valores de intervalo de coerência do canal e RSRs, nos levou a considerar o uso da distribuição de amplitude que atinge a capacidade coerente na proposta dos limitantes, proporcionando resultados interessantes.

Outro fato importante foi a questão de se considerar a utilização ou não da codificação diferencial no transmissor no estudo dos limitantes de capacidade. Os resultados indicaram

que o uso da codificação diferencial leva os limitantes a se aproximarem mais rapidamente da capacidade de canal coerente em função do intervalo de coerência do canal e da RSR.

3.6 Síntese do Capítulo

Este Capítulo abordou um estudo sobre a capacidade de canal AWGN não-coerente de bloco com o intuito de complementar as contribuições já existentes na literatura. Inicialmente, foram apresentados esquemas práticos de sinalização, as constelações M-APSK, a serem usados no sistema de transmissão proposto. Considerações a respeito de seus parâmetros e configurações também foram feitas. Em seguida, o Capítulo apresentou o modelo matemático do canal estudado, algumas definições e considerações importantes a respeito da capacidade de canal. Uma proposta para o cálculo de limitantes de capacidade foi efetuada, seguida da extensão de um algoritmo que calcula simultaneamente os parâmetros ótimos da constelação e a distribuição de entrada que atinge a capacidade do canal M-APSK/AWGN coerente. Por fim, resultados numéricos relativos ao cálculo dos limitantes foram apresentados com as devidas conclusões.

4 Grafos-Fatores e Códigos LDPC

A modelagem por grafos sempre foi uma ferramenta indispensável nas atividades cotidianas dos engenheiros. Como exemplos de tais modelos, temos os diagramas de circuitos, os fluxogramas e os diagramas de treliça. Em áreas como Inteligência Artificial, Processamento de Sinais e Teoria de Códigos, a modelagem por grafos é normalmente associada a certos tipos de algoritmos. Na Teoria de Códigos, por exemplo, a decodificação iterativa de códigos turbo pode ser interpretada em função de um modelo de um código por um grafo. No contexto dos códigos corretores de erros, os grafos-fatores e o algoritmo Soma-Produto constituem uma técnica interessante para a representação de diversos tipos de códigos. Este Capítulo trata inicialmente das definições básicas dos grafos-fatores e a descrição do algoritmo Soma-Produto. Em seguida, o Capítulo aborda como os grafos-fatores podem ser utilizados para modelar sistemas codificados por meio das modelagens comportamental e probabilística. O algoritmo de decodificação aplicado a estes sistemas é tratado a seguir, destacando tanto os grafos sem ciclos quanto os que possuem ciclos. Por fim, o Capítulo apresenta de maneira resumida os códigos LDPC, enfatizando suas formas de representação e seus algoritmos de decodificação iterativa nos domínios das probabilidades e dos logaritmos.

4.1 Grafos-Fatores: Definição

Seja $g(x_1, x_2, \ldots, x_n)$ uma função com n argumentos $x_1, x_2, \ldots, x_n$. Define-se como grafo-fator o modelo gráfico biparticionado [1] que descreve a função g. A Figura 4.1 ilustra o grafo-fator da função $g(x_1, x_2, \ldots, x_n)$. Os círculos representam os nós de variável x_1, x_2, ..., x_n, ao passo que o quadrado preto representa o nó de função f. Cada linha que conecta um círculo ao quadrado indica que a variável representada pelo círculo é argumento da função f.

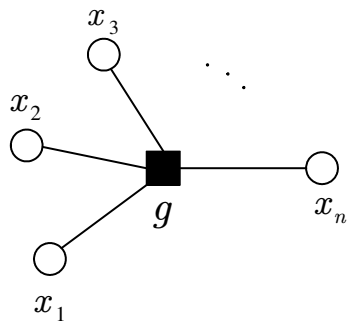


Figura 4.1: Grafo-fator da função $g(x_1, x_2, x_3, \ldots, x_n)$.

Considere que a função g pode ser fatorada em um produto de várias funções f_j conforme a equação

$$g\left(x_1, x_2, \ldots, x_n\right) = \prod_{j \in J_a} f_j\left(X_j\right) \ , \tag{4.1}$$

na qual J_a é um conjunto de índices discretos e X_j é um subconjunto de $\{x_1, x_2, \ldots, x_n\}$ [25]. A função $g(x_1, x_2, \ldots, x_n)$ é denominada *função global* e as funções $f_j(X_j)$ são chamadas de *funções locais*. Para uma função g que pode ser fatorada, um grafo-fator pode ser entendido como um modelo gráfico biparticionado que representa a fatoração da função global g no produto de funções locais f_j [25]. Desta maneira, os grafos-fatores [2] possuem um nó de variável associado a cada variável x_k, um nó de função para cada função f_j e uma conexão (ramo) entre o nó de variável e o nó de função caso x_k seja argumento de f_j.

[1] O termo biparticionado se deve ao grafo possuir apenas dois tipos de nós denominados nós do tipo variável e nós do tipo função. Tais nós serão chamados de nós de variável e nós de função apenas.

[2] Para fins de simplificação, o termo grafo-fator será chamado apenas de grafo.

Seja por exemplo, a função global $g(x_1, x_2, x_3, x_4)$, cuja fatoração é dada por

$$g\left(x_1, x_2, x_3, x_4\right) = f_1\left(x_1\right) f_2\left(x_1, x_2, x_3\right) f_3\left(x_3, x_4\right) \ . \tag{4.2}$$

A Figura 4.2 ilustra o grafo que representa a fatoração da função $g(x_1, x_2, x_3, x_4)$. Um nó de variável é denominado de grau α caso possua α nós de função conectados a ele. Esta denominação também é válida para os nós de função caso possua α nós de variável conectados. Além disso, é importante ressaltar que em grafos biparticionados, nós do mesmo tipo não se conectam diretamente.

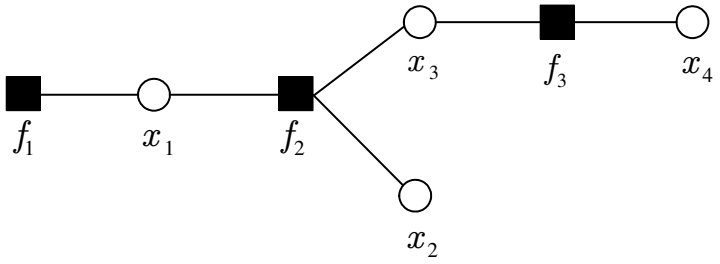


Figura 4.2: `Grafo da função` $g(x_1, x_2, x_3, x_4) = f_1(x_1) f_2(x_1, x_2, x_3) f_3(x_3, x_4)$.

Suponha agora que desejamos calcular as funções marginais $g_k(x_k), k \in \{1, 2, 3, 4\}$. Por exemplo, seja a função marginal $g_1(x_1)$ definida por

$$\begin{aligned} g_1\left(x_1\right) &\triangleq \sum_{x_2, x_3, x_4} g\left(x_1, x_2, x_3, x_4\right) \\ &= \sum_{x_2, x_3, x_4} f_1\left(x_1\right) f_2\left(x_1, x_2, x_3\right) f_3\left(x_3, x_4\right) \\ &= \sum_{\sim\{x_1\}} f_1\left(x_1\right) f_2\left(x_1, x_2, x_3\right) f_3\left(x_3, x_4\right) \end{aligned} \tag{4.3}$$

em que $\sim \{x_1\}$ indica que o somatório é realizado em todas as variáveis da função exceto x_1. O grafo ilustrado na Figura 4.2 nos fornece uma maneira simples e direta de calcularmos $g_1(x_1)$, assim como as demais funções marginais $g_k(x_k), k \neq 1$.

4.2 Algoritmo Soma-Produto

Imaginemos que o grafo ilustrado na Figura 4.2 seja visto como uma rede de processadores. Cada nó do grafo é considerado um processador e as conexões entre os nós são os canais de comunicação pelos quais os processadores podem enviar mensagens entre si. A mensagem enviada por um ramo que conecta um nó de variável x_k a um nó de função f_j, em qualquer dos sentidos, é uma função do argumento x_k. Sendo assim, a mensagem não é apenas um valor e sim, um conjunto de valores que depende das variáveis do grafo.

O algoritmo responsável pelo cálculo da função $g_k(x_k)$ inicia nas extremidades do grafo nos chamados *nós de extremidade*. Se o nó de extremidade for um nó de variável, ele envia uma mensagem unitária ao seu nó vizinho. Caso o nó de extremidade seja um nó de função, a mensagem enviada será uma descrição da função local. A Figura 4.3 ilustra os tipos de nós de extremidade e as mensagens geradas por eles. Em geral, as mensagens num grafo são representadas por $\mu_{a\to b}(x)$, em que a e b indicam o nó de origem e o nó de destino, respectivamente.

Figura 4.3: Nós de extremidade.

Cada nó espera pelas mensagens provenientes dos demais nós conectados a ele para calcular a mensagem de saída. Um nó de variável x_k envia para o nó de função f_j (conectado a ele) o produto das mensagens vindas dos nós conectados a ele, exceto o nó f_j para o qual a mensagem de saída será enviada. Já para um nó de função, a mensagem de saída é obtida em duas etapas:

1. Calcula-se o produto da função que o nó representa pelas mensagens recebidas dos demais nós conectados, exceto o nó de destino.
2. Em seguida, o produto gerado na etapa anterior é submetido ao somatório $\sum_{\sim\{x\}}$ para a obtenção da mensagem final de saída.

Uma vez que o algoritmo opera efetuando diversas somas e produtos, ele é conhecido como *algoritmo Soma-Produto* (SP) . Considerando a analogia mencionada dos grafos com uma rede de processadores, este algoritmo também é denominado de *algoritmo de passagem de mensagens*.

Para reforçar o entendimento de como as mensagens são geradas e passadas no grafo, vamos calcular a função $g_1(x_1)$ segundo a equação (4.3). A Figura 4.4 ilustra como ocorre a passagem de mensagens pelo grafo da função g. O nó de variável x_1 está em destaque para representar que a função marginal que se deseja calcular é $g_1(x_1)$. As setas representam o fluxo das mensagens para se obter $g_1(x_1)$. Os índices numéricos em cada seta representam a ordem temporal do cálculo das mensagens.

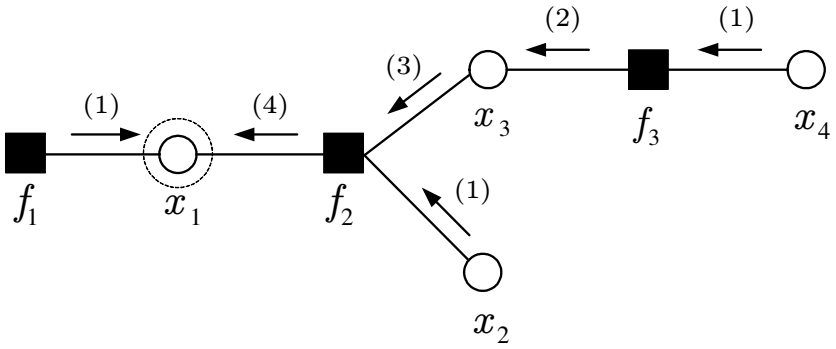


Figura 4.4: Mensagens geradas em cada passo do algoritmo SP para o cálculo de $g_1(x_1)$.

Como foi mencionado anteriormente, o algoritmo SP tem início pelas extremidades do grafo, com o cálculo das mensagens $\mu_{f_1 \to x_1}(x_1) = f_1(x_1)$, $\mu_{x_2 \to f_2}(x_2) = 1$ e $\mu_{x_4 \to f_3}(x_4) = 1$. Em seguida, calcula-se a mensagem $\mu_{f_3 \to x_3}(x_3)$ como

$$\begin{aligned} \mu_{f_3 \to x_3}(x_3) &= \sum_{\sim\{x_3\}} \mu_{x_4 \to f_3}(x_4) \cdot f_3(x_3, x_4) \\ &= \sum_{\sim\{x_3\}} 1 \cdot f_3(x_3, x_4) = \sum_{\{x_4\}} f_3(x_3, x_4) \ . \end{aligned} \tag{4.4}$$

No passo seguinte, representado pelo instante de tempo (3), temos o cálculo da mensagem

$\mu_{x_3 \to f_2}(x_3)$ dada por

$$\mu_{x_3 \to f_2}(x_3) = \mu_{f_3 \to x_3}(x_3) = \sum_{x_4} f_3(x_3, x_4) \ . \tag{4.5}$$

E finalmente, calcula-se a mensagem $\mu_{f_2 \to x_1}(x_1)$ tal que

$$\begin{aligned} \mu_{f_2 \to x_1}(x_1) &= \sum_{\sim\{x_1\}} \left(\mu_{x_2 \to f_2}(x_2) \cdot \mu_{x_3 \to f_2}(x_3)\right) \cdot f_2(x_1, x_2, x_3) \\ &= \sum_{\sim\{x_1\}} \left(1 \cdot \sum_{x_4} f_3(x_3, x_4)\right) \cdot f_2(x_1, x_2, x_3) \\ &= \sum_{x_2, x_3, x_4} f_2(x_1, x_2, x_3) f_3(x_3, x_4) \ . \end{aligned} \tag{4.6}$$

O cálculo do algoritmo SP termina no nó x_1 e para obtermos $g_1(x_1)$, basta multiplicarmos as mensagens que chegam a este nó. Logo,

$$\begin{aligned} g_1(x_1) &= \mu_{f_1 \to x_1}(x_1) \cdot \mu_{f_2 \to x_1}(x_1) \\ &= \sum_{x_2, x_3, x_4} f_1(x_1) f_2(x_1, x_2, x_3) f_3(x_3, x_4) \ . \end{aligned} \tag{4.7}$$

Assim como calculamos $g_1(x_1)$, podemos estar interessados em obter as demais funções marginais $g_k(x_k), k \neq 1$. Seria bastante ineficiente repetir sequencialmente todos os passos descritos no cálculo de $g_1(x_1)$. De fato, a solução encontrada é calcular todas as funções $g_k(x_k)$ utilizando o algoritmo SP simultaneamente no grafo. As mensagens relativas a cada função $g_k(x_k)$ são geradas e passadas da mesma maneira que descrevemos para $g_1(x_1)$. O fim dos cálculos ocorre quando duas mensagens passam pelo mesmo ramo, uma em cada direção. Se o algoritmo for usado de maneira iterativa, o processo de envio de mensagens descrito anteriormente se repete.

De maneira geral, a Figura 4.5 descreve como se dá a regra de atualização das mensagens em um grafo [25]. Sejam $n(x) = \{f, h_1, h_2, ...\}$ e $n(f) = \{x, y_1, y_2, ...\}$, os conjuntos de nós conectados a um nó de variável x e a um nó de função f, respectivamente. O cálculo das mensagens executadas pelo algoritmo SP pode ser resumido pelas equações

$$\mu_{x \to f}(x) = \prod_{h \in n(x) | \{f\}} \mu_{h \to x}(x) \tag{4.8}$$

e

$$\mu_{f\to x}(x) = \sum_{\sim\{x\}} \left(\prod_{y\in n(f)|\{x\}} \mu_{y\to f}(y)\, f(x, y_1, y_2, ...) \right). \tag{4.9}$$

Na equação (4.8), $n(x)|\{f\}$ representa o conjunto de nós de função conectados ao nó x, exceto o nó f. Analogamente na equação (4.9), $n(f)|\{x\}$ representa o conjunto de nós de variável conectados ao nó f, exceto o nó x.

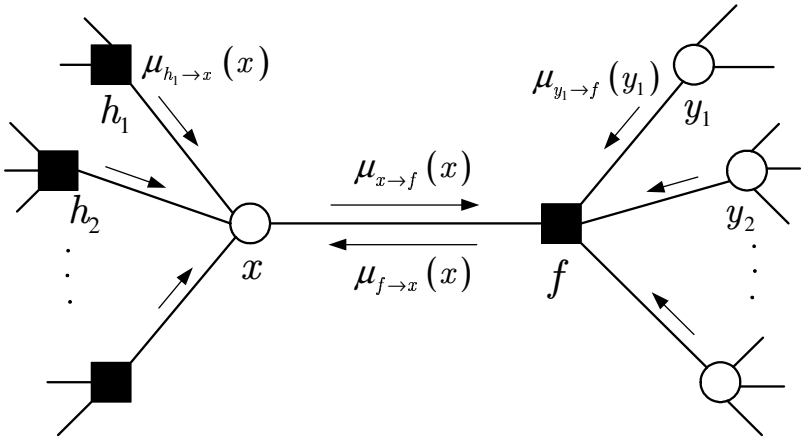


Figura 4.5: Regra geral de atualização de mensagens do algoritmo SP em um fragmento de grafo.

A partir da regra de atualização das mensagens ilustrada na Figura 4.5, podemos ainda destacar duas importantes observações a respeito do algoritmo SP. A primeira refere-se apenas ao caso particular de um nó de variável que possui dois nós vizinhos. Percebe-se pela equação (4.8) ou então graficamente, que não há cálculo de uma nova mensagem neste caso. O nó de variável apenas encaminha a mensagem que chega até ele ao nó de destino. A segunda observação trata de uma maneira equivalente pela qual podemos obter a função marginal $g_k(x_k)$. Para isso, basta efetuarmos o produto de duas mensagens que passam em direções opostas sobre qualquer ramo incidente em x_k. Isto pode ser constatado tanto no exemplo dado para o cálculo de $g_1(x_1)$ quanto na Figura 4.5.

4.3 Modelagem de Sistemas Codificados por meio de Grafos

4.3.1 Introdução

A utilização de grafos para representar códigos corretores de erros teve início com o trabalho de Tanner [26] no início da década de 80. Tais modelos continham dois tipos de nós: o primeiro representava os símbolos da palavra-código e o segundo tipo representava as funções verificação de paridade do código. Estes grafos ficaram conhecidos como *grafos de Tanner*. Tanner também formulou algoritmos para decodificar palavras-código aplicadas ao grafo. Pode-se considerar que o exemplo pioneiro da representação desenvolvida por Tanner são os códigos LDPC propostos por Gallager na década de 60 [27].

Inicialmente, os grafos de Tanner só se referiam a códigos baseados em matrizes de verificação de paridade. Porém, na década de 90, Wiberg, Loeliger e Kötter estenderam a representação dos grafos de Tanner [28]. Um dos tipos de nó passou a representar não apenas símbolos da palavra-código, mas também estados. As variáveis que representam estados também são conhecidas por variáveis ocultas [25]. Desta maneira, os códigos descritos por treliças também puderam ser representados por grafos.

No presente trabalho, iremos nos referir tanto aos grafos de Tanner quanto aos grafos que contém estados em sua representação. O termo grafo será utilizado para se referir tanto a um tipo quanto ao outro. A seguir, veremos como os grafos podem ser utilizados para modelar sistemas codificados.

4.3.2 Modelagem Comportamental

Em algumas aplicações, modelar um sistema em função do seu comportamento é uma técnica interessante. O comportamento de um sistema é especificado em termos das configurações válidas das variáveis que o compõem [29]. Esta modelagem é chamada modelagem comportamental e pode ser representada por meio de grafos.

Sejam por exemplo, um código binário linear $C(n_c, k_c)$ e o espaço do código, S_c,

determinado pelas 2^{k_c} palavras-código que o compõe. Qualquer subconjunto de palavras-código que pertencem a S_c é denominado *comportamento válido* e é denotado por B_c. As palavras-código pertencentes a B_c são chamadas de *configurações válidas*.

Sendo assim, considere a função $I_{B_c}(x_1, x_2, \ldots, x_{n_c})$ em que $(x_1, x_2, \ldots, x_{n_c})$ são os bits de uma palavra-código. A função $I_{B_c}(\cdot)$ é definida como sendo a função característica, isto é,

$$I_{B_c}(x_1, x_2, \ldots, x_{n_c}) = \begin{cases} 1, & se\ (x_1, x_2, ..., x_{n_c}) \in B_c \\ 0, & caso\ contrário \end{cases} . \tag{4.10}$$

Como a função $I_{B_c}(\cdot)$ indica se uma determinada configuração é válida ou não, ela é denominada *função indicadora*.

Funções indicadoras têm uma grande importância na teoria de grafos aplicada à codificação. A função indicadora $I_{B_c}(\cdot)$ é interpretada como uma função global que pode ser fatorada no produto de várias funções locais. Logo, a função global $I_{B_c}(\cdot)$ será igual a 1 quando todas as funções locais forem iguais a 1, caracterizando assim uma operação lógica E (AND).

Sabe-se que o código $C(n_c, k_c)$ é definido pelas $(n_c - k_c)$ equações de verificação de paridade da matriz H. Portanto, a função indicadora global de C pode ser fatorada em $(n_c - k_c)$ funções indicadoras locais, que são as equações de verificação de paridade. Caso as $(n_c - k_c)$ equações de paridade sejam satisfeitas para uma determinada palavra-código, todas as funções indicadoras locais assumem valor 1 e consequentemente, a função indicadora global também vale 1. Logo, a palavra-código é considerada uma configuração válida.

A função indicadora de um código binário linear definido por uma matriz $\mathbf{H}_{(n_c-k_c)\times n_c}$ pode ser representada por um grafo com n_c nós de variável e $(n_c - k_c)$ nós de função. Neste caso, os nós de função denominam-se nós de verificação de paridade, ou simplesmente, nós de verificação. Cada linha da matriz $\mathbf{H}$ está relacionada a um nó de verificação ao passo que cada coluna se refere a um nó de variável. O grafo associado à matriz $\mathbf{H}$ de um código é construído da seguinte maneira: para cada elemento $h_{ij} \neq 0$ existente na matriz, faz-se a conexão do i-ésimo nó de variável ao j-ésimo nó de verificação.

Considere a matriz $\mathbf{H}$ do código $C_1(6, 3)$ dada pela equação (2.38). O grafo do código $C_1(6, 3)$ representado por tal matriz está ilustrado na Figura 4.6. Neste grafo, os nós de

verificação são representados por quadrados com um sinal "+", que simboliza a operação OU-exclusivo (XOR). As funções indicadoras locais são funções verificação de paridade par (número par de elementos 1) das palavras-código. Por exemplo, $f_1(x_1, x_3, x_4) = x_1 \oplus_2 x_3 \oplus_2 x_4$ será igual a 0 se os bits x_1, x_3 e x_4 formarem uma configuração de paridade par. Sendo assim, a função indicadora global $I_{C_1}(x_1, x_2, \ldots, x_6)$ será igual a 1 quando a palavra-código $(x_1, x_2, \ldots, x_6)$ satisfizer todas as equações de paridade de C_1.

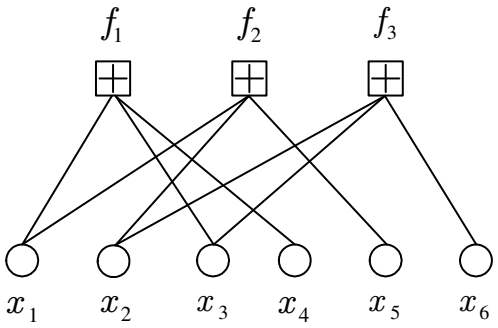


Figura 4.6: `Grafo (Tanner) do código` $C_1(6, 3)$.

Outro caso importante de modelagem comportamental é a representação de códigos definidos por treliças. O comportamento do sistema codificado definido por uma treliça é especificado em função da configuração $\{s_0, \ldots, s_{n_c}, x_1, \ldots, x_{n_c}\}$. As variáveis $\{s_0, \ldots, s_{n_c}\}$ são os estados da treliça ao passo que $\{x_1, \ldots, x_{n_c}\}$ representam a palavra-código. A treliça do código se divide em n_c seções e cada seção T_i possui uma função cujos argumentos são o estado anterior s_{i-1}, o bit x_i e o estado atual s_i.

Na representação de treliças por meio de grafos, também lidamos com o conceito de função indicadora. Sendo assim, a função indicadora global do grafo terá configuração válida quando cada seção T_i da treliça tiver suas restrições locais satisfeitas. Cada comportamento local é denotado por $T_i(s_{i-1}, x_i, s_i)$ e a função indicadora global é fatorada em n_c funções indicadoras locais que compõem uma operação E (AND), assim como no exemplo anterior.

Considere a treliça do código $C_1(6, 3)$ reproduzida na Figura 2.5. O grafo correspondente a esta treliça está ilustrado na Figura 4.7. Este tipo de grafo foi concebido por Wiberg [28]. As variáveis ocultas, ou estados, são representadas por círculos duplos. As funções indicadoras

locais f_i são representadas por quadrados pretos e correspondem a cada seção da treliça.

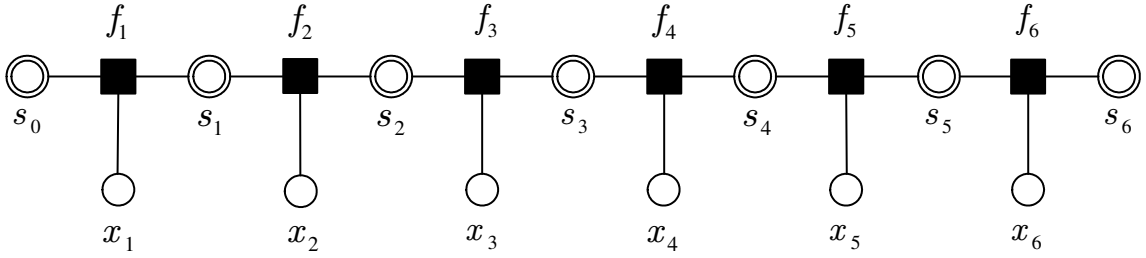


Figura 4.7: `Grafo (Wiberg) do código` $C_1(6,3)$.

Por exemplo, seja o comportamento local T_5 correspondente à quinta seção, definido como

$$T_5(s_4, x_5, s_5) = \{(000, 0, 000)\,; (001, 0, 001)\,; (010, 1, 000)\,; (011, 1, 001)\}\ , \tag{4.11}$$

em que s_4 e s_5 pertencem aos subconjuntos de estados $\{000, 001, 010, 011\}$ e $\{000, 001\}$, respectivamente. A função indicadora local f_5 é dada por $f_5(s_4, x_5, s_5) = [(s_4, x_5, s_5) \in T_5]$ e cada ramo da seção corresponde a um elemento de T_5 .

4.3.3 Modelagem Probabilística

Em um sistema codificado, obter o melhor decodificador não significa alcançar apenas o melhor desempenho. Tão importante quanto otimizar o desempenho do sistema por meio de um bom decodificador, é implementá-lo com baixa complexidade. Os decodificadores algébricos utilizam a saída quantizada do canal e aproveitam a estrutura especial construída no código para tornar a decodificação mais fácil. Por outro lado, os decodificadores probabilísticos não quantizam a saída do canal e procuram obter a sequência de informação transmitida usando critérios estatísticos. Para que os decodificadores probabilísticos possam usufruir de alguma estrutura assim como os decodificadores algébricos, é possível utilizar grafos para modelar a estrutura probabilística dos códigos.

A modelagem probabilística tem a finalidade de representar a distribuição de probabilidade conjunta das variáveis que descrevem o sistema. As mensagens que são passadas entre os nós do grafo são valores de probabilidades. Sendo assim, neste tipo de modelagem os grafos podem ser interpretados como uma rede de propagação de probabilidades.

Seja $\mathbf{x} = (x_1, x_2, \ldots, x_{n_c})$ uma palavra-código transmitida por um canal sem memória e $\mathbf{y} = (y_1, y_2, \ldots, y_{n_c})$ a sequência observada na saída do canal DMC. Considerando que o decodificador utiliza a regra MAP, temos que a probabilidade $p(\mathbf{x}|\mathbf{y})$ é utilizada como critério de decisão para cada observação $\mathbf{y}$. Assume-se que $p(\mathbf{x})$ é uniforme, portanto $p(\mathbf{x}) = I_C(x_1, x_2, \ldots, x_{n_c})/|C|$, em que $I_C(\cdot)$ e $|C|$ são a função indicadora e a cardinalidade do código, respectivamente. Como $p(\mathbf{x}|\mathbf{y})$ é diretamente proporcional ao produto $p(\mathbf{y}|\mathbf{x})p(\mathbf{x})$ e cada observação y_i é constante, $p(\mathbf{y}|\mathbf{x})$ pode ser considerada apenas função de $\mathbf{x}$. Desta maneira, a função modelada pelo grafo é dada por

$$g\left(x_1, x_2, \ldots, x_{n_c}\right) = I_C\left(x_1, x_2, \ldots, x_{n_c}\right) \prod_{i=1}^{n_c} p\left(y_i|x_i\right) . \tag{4.12}$$

A equação (4.12) indica a junção da modelagem comportamental com a modelagem probabilística para representar sistemas codificados. A modelagem comportamental serve para decidir se a palavra-código é válida ou não. Já a modelagem probabilística ajuda a decidir, segundo alguma regra de decisão, qual das palavras-código válidas foi transmitida. A Figura 4.8 ilustra o grafo do código $C_1(6, 3)$ com a adição da modelagem probabilística. Os nós de variável y_i não aparecem porque são incorporados aos nós de função $p(y_i|x_i)$ para fins de simplificação.

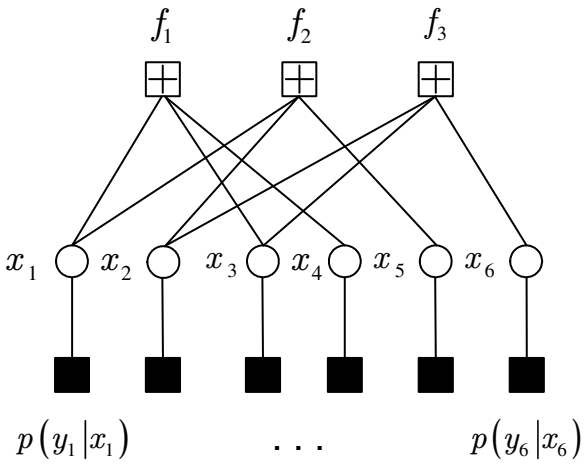


Figura 4.8: Grafo do código $C_1(6, 3)$ com a inclusão das probabilidades de transição do canal.

4.3.4 Algoritmo de Decodificação em um Grafo sem Ciclos

A modelagem de sistemas codificados por grafos é bastante útil para solucionar o problema da decodificação. Basicamente, o problema da decodificação consiste em escolher qual palavra-código foi transmitida dado uma sequência observada na saída do canal. O critério de escolha pode ser relativo ao bloco como um todo, ou então a cada símbolo do bloco, de acordo com o que definimos na Seção 2.4.5. Considere que o decodificador utilize a regra de decisão MAP de bit definida pela equação (2.50). Esta regra pode ser reescrita como

$$\widehat{u}_i\left(\mathbf{y}\right) = \arg\max_{u_i}\left[\sum_{\sim\{u_i\}} I_C(\mathbf{x}) p\left(x_i\left|y_i\right.\right)\right] \ , \tag{4.13}$$

em que $I_C(\mathbf{x})$ representa a função indicadora que designa se $\mathbf{x}$ é uma palavra-código válida ou não.

Para um código descrito por uma treliça, podemos novamente redefinir a regra de decisão MAP de bit segundo a equação

$$\widehat{u}_i\left(\mathbf{y}\right) = \arg\max_{u_i}\left[\sum_{\sim\{u_i\}} I_T(\mathbf{x}) p\left(x_i\left|y_i\right.\right)\right] \ , \tag{4.14}$$

na qual $I_T(\mathbf{x})$ é a função que indica se um determinado caminho da treliça é válido. Em outras palavras, esta função também indica se $\mathbf{x}$ é uma palavra-código válida. Os caminhos na treliça são fornecidos a partir de suas seções. Cada seção é composta por quatro variáveis: os estados s_{i-1} e s_i, o bit de informação u_i e o bit codificado x_i. As Figuras 4.9(a) e 4.9(b) representam o grafo da i-ésima seção de uma treliça convencional.

A omissão dos nós de variável y_i e dos nós de função do canal $p(y_i|x_i)$ ilustrada na Figura 4.9(b) ocorre com o intuito de simplificar o grafo. Desta forma, a junção dos grafos das n_c seções da treliça dão origem ao grafo que representa uma combinação das modelagens comportamental e probabilística do código. Como o grafo final não possui ciclos, as probabilidades marginais $p(u_i|\mathbf{y})$ podem ser obtidas por meio do algoritmo SP de maneira ótima.

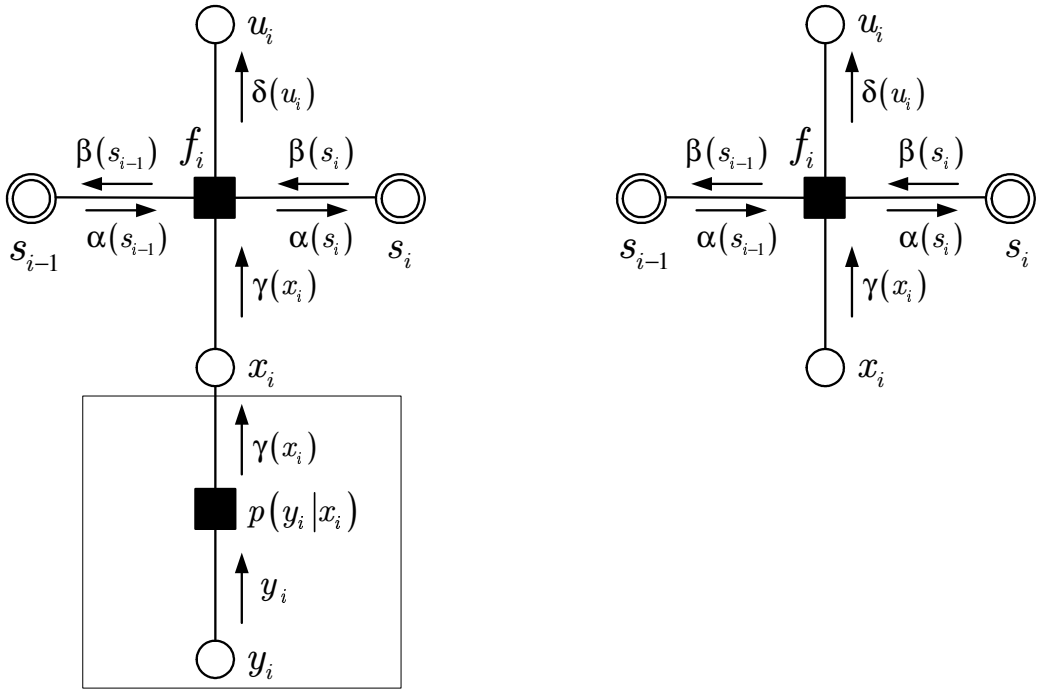


Figura 4.9: (a) Grafo da i-ésima seção de uma treliça convencional.(b) Grafo simplificado.

Em virtude de os grafos serem vistos como uma rede de propagação de probabilidades, o algoritmo SP pode ser denominado de *algoritmo de propagação de probabilidades*. Para o caso de grafos que representam códigos descritos por treliças, o algoritmo de propagação de probabilidades recebe a denominação de algoritmo Progressivo-Regressivo (do inglês, *Forward-Backward*). Na Teoria de Códigos, este algoritmo também é conhecido por algoritmo MAP ou ainda, algoritmo BCJR [7].

Como em todo grafo que não possui ciclos, o algoritmo de decodificação inicia a partir dos nós de extremidade. No nosso caso, tanto os nós de variável quanto os nós de estado de extremidade enviam mensagens conhecidas para os seus nós vizinhos. Considerando a simplificação ilustrada na Figura 4.9(b), os nós de variável x_i passam a ser nós de extremidade que emitem mensagens conhecidas. Na realidade, os nós x_i assim como os nós de estado que não são de extremidade, pertencem à categoria de nós que possuem dois vizinhos, portanto

apenas encaminham a mensagem recebida de um dos vizinhos para o outro. De acordo com [7], as mensagens $\mu_{x_i \to f_i}(x_i)$, $\mu_{f_i \to s_i}(s_i)$, $\mu_{s_i \to f_i}(s_i)$ e $\mu_{f_i \to u_i}(u_i)$ são denominadas de $\gamma(x_i)$, $\alpha(s_i)$, $\beta(s_i)$ e $\delta(u_i)$ respectivamente, e são definidas pelas equações

$$\alpha(s_i) = \sum_{s_{i-1}} \sum_{u_i} \sum_{x_i} I_{T_i}(s_{i-1}, x_i, u_i, s_i) \alpha(s_{i-1}) \gamma(x_i) \ , \tag{4.15}$$

$$\beta(s_{i-1}) = \sum_{s_i} \sum_{u_i} \sum_{x_i} I_{T_i}(s_{i-1}, x_i, u_i, s_i) \beta(s_i) \gamma(x_i) \tag{4.16}$$

e

$$\delta(u_i) = \sum_{x_i} \sum_{s_{i-1}} \sum_{s_i} I_{T_i}(s_{i-1}, x_i, u_i, s_i) \alpha(s_{i-1}) \beta(s_i) \gamma(x_i) \ , \tag{4.17}$$

nas quais $s_0(0) = 1$ e $s_{n_c}(0) = 1$. Como as mensagens enviadas pelos nós de variável u_i são unitárias e as mensagens recebidas pelos nós x_i são irrelevantes, elas não são consideradas nos diagramas das Figuras 4.9(a) e 4.9(b).

Na etapa progressiva do algoritmo, as mensagens $\alpha(s_i)$ são calculadas em função de $\alpha(s_{i-1})$ e $\gamma(x_i)$ segundo a equação (4.15). Na etapa regressiva, as mensagens $\beta(s_{i-1})$ são obtidas em função de $\beta(s_i)$ e $\gamma(x_i)$ conforme a equação (4.16). Como as recursões efetuadas nas duas etapas não interagem, as etapas progressiva e regressiva podem ser executadas simultaneamente. O fim do algoritmo ocorre com o cálculo das mensagens $\delta(u_i)$ por meio da equação (4.17). As mensagens $\delta(u_i)$ representam as probabilidades *a posteriori* pelas quais estimaremos que sequência de informação $\mathbf{u}$ foi enviada pelo transmissor.

Considere o grafo ilustrado na Figura 4.10, que representa o código de Hamming estendido $C_e(8,4)$ descrito pela treliça *tail-biting* compacta mostrada na Figura 2.7. Os nós de função do canal e os nós de variável y_i são omitidos para simplificar o grafo. Ainda para simplificar, os nós de variável x_{2i-1} e x_{2i} são agrupados e representados pelo nó γ_i. Os nós de função f_A e f_B representam as seções da treliça. As mensagens enviadas pelos nós γ_i são calculadas pela equação

$$\gamma_i(ab) = p(x_{2i-1} = a \,|y_{2i-1}) \, p(x_{2i} = b \,|y_{2i-1}) \ , \tag{4.18}$$

na qual $a, b \in \{0,1\}$, $i \in \{1,2,3,4\}$ e as probabilidades $p(x_k = 1|y_k)$ e $p(x_k = 0|y_k)$ são dadas

por

$$p\left(x_k = 1\,|y_k\right) = \frac{1}{1 + \exp(-\frac{4y_k\sqrt{E}}{2\sigma^2})} \tag{4.19}$$

e

$$p\left(x_k = 0\,|y_k\right) = 1 - p\left(x_k = 1\,|y_k\right)\ , \tag{4.20}$$

em que E é a energia média do símbolo e σ^2 é a variância do ruído AWGN. Portanto, partindo de valores iniciais conhecidos para $\alpha(\cdot)$ e $\beta(\cdot)$, e considerando as mensagens $\gamma(\cdot)$ descritas anteriormente, podemos calcular as novas mensagens $\alpha(\cdot)$ e $\beta(\cdot)$ durante as etapas progressiva e regressiva do algoritmo de decodificação. Quando $\alpha(\cdot)$ e $\beta(\cdot)$ convergirem, podemos calcular $\delta(u_i)$ e tomar a decisão sobre o bit de informação u_i.

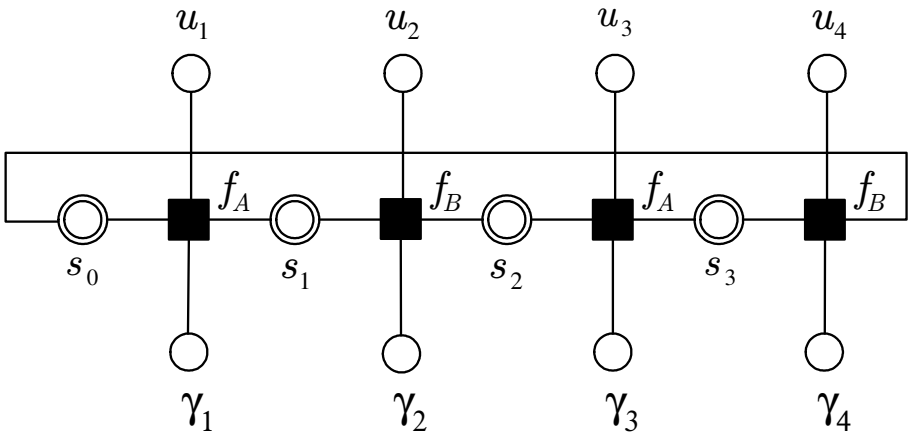


Figura 4.10: Grafo do código de Hamming estendido $C_e(8,4)$.

Seja por exemplo, a primeira seção da treliça representada pelo nó f_A que tem como argumentos os estados s_0 e s_1, o bit de informação u_1 e o nó γ_1. As mensagens $\alpha(s_1)$ e $\beta(s_0)$ calculadas nas etapas progressiva e regressiva, respectivamente, são dadas por

$$\begin{aligned}
\alpha\left(s_1 = 0\right) &= \alpha\left(s_0 = 0\right)\gamma_1\left(00\right)\\
\alpha\left(s_1 = 1\right) &= \alpha\left(s_0 = 0\right)\gamma_1\left(11\right)\\
\alpha\left(s_1 = 2\right) &= \alpha\left(s_0 = 1\right)\gamma_1\left(01\right)\\
\alpha\left(s_1 = 3\right) &= \alpha\left(s_0 = 1\right)\gamma_1\left(10\right)
\end{aligned} \tag{4.21}$$

e

$$\begin{aligned}\beta\left(s_0=0\right)&=\beta\left(s_1=0\right)\gamma_1\left(00\right)+\beta\left(s_1=1\right)\gamma_1\left(11\right)\\ \beta\left(s_0=1\right)&=\beta\left(s_1=2\right)\gamma_1\left(01\right)+\beta\left(s_1=3\right)\gamma_1\left(10\right)\ .\end{aligned}\tag{4.22}$$

As mensagens $\delta(u_1)$ são calculadas tais que

$$\begin{aligned}\delta\left(u_1=0\right)&=\alpha\left(s_0=0\right)\gamma_1\left(00\right)\beta\left(s_1=0\right)+\alpha\left(s_0=1\right)\gamma_1\left(01\right)\beta\left(s_1=2\right)\\ \delta\left(u_1=1\right)&=\alpha\left(s_0=0\right)\gamma_1\left(11\right)\beta\left(s_1=1\right)+\alpha\left(s_0=1\right)\gamma_1\left(10\right)\beta\left(s_1=3\right)\ .\end{aligned}\tag{4.23}$$

Finalmente, o algoritmo de decodificação compara as probabilidades $\delta(u_1 = 0)$ e $\delta(u_1 = 1)$. Se $\delta(u_1 = 0) > \delta(u_1 = 1)$, o bit de informação u_1 assume valor 0 ; caso contrário, assume valor 1. Os demais bits da sequência de informação são obtidos de maneira similar.

4.3.5 Algoritmo de Decodificação em um Grafo com Ciclos

Vimos que o algoritmo SP pode ser aplicado a um grafo sem ciclos de maneira ótima. As regras de atualização das mensagens definidas pelas equações (4.8) e (4.9) são executadas com o objetivo de fornecer, ao final do algoritmo, as probabilidades marginais dos bits de informação transmitidos. Acontece que as regras de atualização das mensagens independem do fato de o grafo não possuir ciclos. Sendo assim, tais regras também podem ser aplicadas a grafos que possuem ciclos, mesmo com a possibilidade de não fornecerem resultados exatos.

Um ciclo de comprimento ε é definido como o caminho composto de ε ramos que inicia e termina em um mesmo nó do grafo. Na Figura 4.6, temos um ciclo de comprimento 6 que segue o seguinte percurso: $x_1 - f_1 - x_3 - f_3 - x_2 - f_2 - x_1$. A presença de ciclos no grafo causa uma propagação indefinida de mensagens, deixando o algoritmo de decodificação sem um critério natural de parada. As mensagens deixam de ser interpretadas como funções de distribuição de probabilidade e, de uma maneira geral, o resultado final não pode mais ser considerado funções marginais de probabilidade. Entretanto, a aplicação do algoritmo SP na decodificação de códigos definidos por grafos com ciclos tem se mostrado uma abordagem excelente [30].

Neste caso, o algoritmo é iniciado com a passagem de mensagens unitárias virtuais em todos os ramos do grafo. Em seguida, as regras de atualização utilizadas nos grafos sem ciclos são aplicadas de maneira semelhante. Assim como nos grafos sem ciclos, o algortimo SP também precisa ter um *cronograma*[3] de execução. Um exemplo tradicional de cronograma do algoritmo é o cronograma *por inundação* [4], no qual cada ramo do grafo é ativado em ambas as direções em cada passo do algoritmo. Finalmente, o algoritmo de decodificação em um grafo com ciclos termina sob condições pré-determinadas. Por exemplo, o algoritmo de decodificação de códigos turbo geralmente termina após um número determinado de iterações.

4.4 Códigos LDPC

4.4.1 Introdução

Os códigos LDPC são uma classe de códigos de bloco lineares que fornece desempenho próximo ao limitante teórico (de desempenho) estabelecido por Shannon para uma variedade de canais de comunicações [31]. Além disso, são códigos que admitem decodificadores de implementação com complexidade razoavelmente baixa. Tais códigos foram inicialmente propostos em [27] juntamente com um algoritmo iterativo de decodificação probabilística. Passaram mais de 30 anos esquecidos até que ressurgiram nos anos 90 com os trabalhos de MacKay [32] e Luby [33]. Neste ressurgimento, verificou-se que o algoritmo de decodificação proposto por Gallager era um caso particular do algoritmo de Pearl de propagação de probabilidades sobre grafos [34]. Além destes fatos, esta classe de códigos possui desempenho que se aproxima da capacidade do canal AWGN para comprimentos longos [35].

[3] O termo cronograma corresponde ao termo *schedule* em inglês.
[4] O termo cronograma por inundação corresponde ao termo *flooding schedule* em inglês.

4.4.2 Representação de Códigos LDPC

Representação Matricial

Como qualquer código de bloco linear binário, os códigos LDPC binários podem ser definidos como um subespaço do espaço vetorial V_2. Desta maneira, eles podem ser representados por uma matriz geradora $\mathbf{G}$ ou uma matriz de verificação de paridade $\mathbf{H}$. As matrizes de verificação de paridade dos códigos LDPC possuem uma característica bem peculiar. Elas possuem uma densidade baixa de elementos 1 e por esta razão, são chamadas de matrizes esparsas. De acordo com a estrutura da matriz $\mathbf{H}$, os códigos LDPC podem ser classificados em regulares e irregulares.

Os códigos LDPC regulares são aqueles em que cada coluna da matriz $\mathbf{H}$ possui o mesmo peso w_c e cada linha tem o mesmo peso w_l. Sendo assim, podemos representá-los por meio da notação $C^{w_c,w_l}(n_c, k_c)$. Para estes códigos, a taxa r_c pode ser definida como [31]

$$r_c = \frac{k_c}{n_c} = 1 - \frac{w_c}{w_l} \ . \tag{4.24}$$

Seja, por exemplo, o código LDPC regular $C_3^{2,4}(10, 5)$ cuja matriz $\mathbf{H}_3$ é dada por

$$\mathbf{H}_3 = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \ . \tag{4.25}$$

Neste caso, temos que $w_c = 2$ e $w_l = 4$, portanto, segundo a equação (4.24), a taxa r_c vale $0,5$. A principal característica dos códigos LDPC regulares é dada por

$$m_c w_l = n_c w_c \ , \tag{4.26}$$

em que $m_c = (n_c - k_c)$ é o número de equações de paridade do código.

Já nos códigos LDPC irregulares, o número de 1's em cada linha ou coluna da matriz $\mathbf{H}$

não é constante, conforme ilustra a matriz $\mathbf{H}_4$ do código LDPC irregular $C_4(9,3)$ [36]

$$\mathbf{H}_4 = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}. \tag{4.27}$$

Representação usando Grafos

Na década de 80, Tanner introduziu uma representação gráfica para os códigos LDPC [26]. Conforme vimos na Seção 4.3.1, os grafos de Tanner possuem dois tipos de nós: os nós de variável e os nós de verificação de paridade, ou simplesmente nós de verificação. Cada nó de verificação representa cada uma das m_c equações de paridade do código LDPC.

Para obter o grafo de um código LDPC, basta conectar o j-ésimo nó de verificação ao i-ésimo nó de variável caso o elemento h_{ji} da matriz $\mathbf{H}$ seja igual a 1. Como a matriz $\mathbf{H}$ possui m_c linhas e n_c colunas, o grafo correspondente possui m_c nós de verificação e n_c nós de variável.

Seja novamente o código $C_3^{2,4}(10,5)$ definido pela matriz $\mathbf{H}_3$ dada por (4.25). A Figura 4.11 ilustra o grafo construído segundo a estrutura da matriz $\mathbf{H}_3$. Por exemplo, o nó x_1 conecta-se com o nó f_1, pois o elemento h_{11} vale 1. As demais conexões seguem a mesma regra.

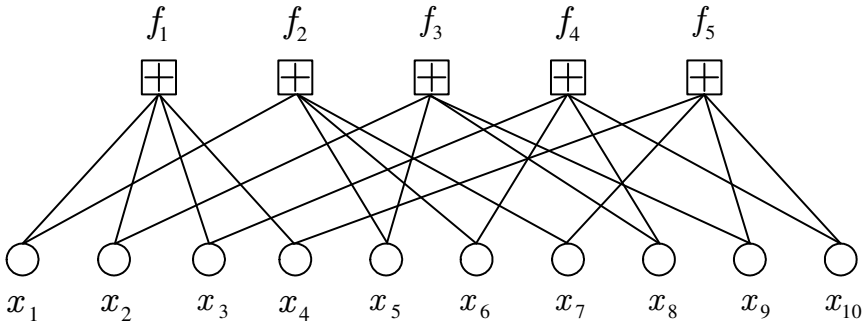


Figura 4.11: `Grafo do código LDPC regular` $C_2(10,5)$.

Considere agora o código $C_4(9,3)$ definido pela matriz $\mathbf{H}_4$ dada por (4.27). De maneira semelhante ao código C_3, as conexões do grafo são estabelecidas com base nos elementos

não-nulos de $\mathbf{H}_3$. A Figura 4.12 ilustra o grafo para o código LDPC irregular $C_4(9,3)$.

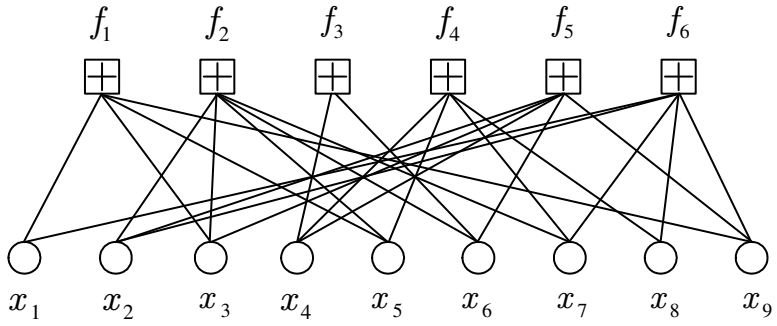


Figura 4.12: Grafo do código LDPC irregular $C_3(9,3)$.

Uma vez que os códigos LDPC irregulares não possuem os parâmetros w_c e w_l constantes, é bastante comum se caracterizar tais códigos por *polinômios de distribuição de grau* [37]. Seja $\lambda(x)$, o polinômio de distribuição de grau dos nós de variável, definido pela equação

$$\lambda\left(x\right) = \sum_{d=1}^{d_v} \lambda_d x^{d-1} \ . \tag{4.28}$$

O parâmetro λ_d representa a fração de todos os ramos conectados aos nós de variável de grau d e d_v denota o grau máximo dos nós de variável. O outro polinômio de distribuição de grau, referente aos nós de verificação, é denotado por $\rho(x)$ e definido por

$$\rho\left(x\right) = \sum_{d=1}^{d_f} \rho_d x^{d-1} \ . \tag{4.29}$$

Neste polinômio, ρ_d representa a fração de todos os ramos conectados aos nós de verificação de grau d e d_f denota o grau máximo dos nós de verificação. Para o código LDPC irregular $C_4(9,3)$, temos que

$$\lambda(x) = 0,16\,x + 0,84\,x^2 \tag{4.30}$$

e

$$\rho(x) = 0,08\,x + 0,32\,x^3 + 0,6\,x^4 \ . \tag{4.31}$$

A taxa de um código LDPC irregular com polinômios de distribuição de grau $\lambda(x)$ e $\rho(x)$

é dada pela equação [37]

$$r_c = 1 - \frac{\int_0^1 \rho(x)dx}{\int_0^1 \lambda(x)dx} \ . \tag{4.32}$$

Desta maneira, a taxa do código $C_4(9,3)$, definido pelas equações (4.30) e (4.31), vale

$$r_c = 1 - \frac{\int_0^1 (0,08x + 0,32x^3 + 0,6x^4)dx}{\int_0^1 (0,16x + 0,84x^2)dx} \approx 0,333 \ .$$

Apesar de não ser tão habitual, os códigos LDPC regulares também podem ser caracterizados pelos polinômios $\lambda(x)$ e $\rho(x)$. Para o código $C_3^{2,4}(10,5)$, $\lambda(x) = x$ e $\rho(x) = x^3$.

4.4.3 Algoritmo de Decodificação Iterativa (Domínio das Probabilidades)

O algoritmo de decodificação baseado em grafos para códigos é classificado como um algoritmo MAP somente se o grafo não possuir ciclos. Como os grafos que representam os códigos LDPC possuem ciclos, o algoritmo de passagem de mensagens não calcula probabilidades a posteriori. Porém, o seu desempenho é considerado bom e assim, o algoritmo mencionado é uma ótima aproximação.

Considere primeiramente que o decodificador conhece a matriz $\mathbf{H}$ do código, o grafo associado à $\mathbf{H}$ e a saída do canal DMC $\mathbf{y} = [y_1, y_2, \ldots, y_{n_c}]$. O objetivo do algoritmo é calcular um valor numérico que represente a probabilidade *a posteriori* de que um dado bit na palavra-código transmitida $\mathbf{x} = [x_1, x_2, \ldots, x_{n_c}]$ seja igual a 1 , dada a sequência observada $\mathbf{y}$ na saída do canal. De uma maneira mais simples, o algoritmo busca obter um valor para $p(x_i = 1|\mathbf{y})$ e por esta razão, dizemos que o decodificador que o emprega atua no domínio das probabilidades.

Antes de iniciar a descrição do algoritmo, vamos definir algumas notações de maneira semelhante à que foi realizada em [31]. Os dois tipos de mensagens trocadas pelos nós do grafo são denotados por $q_{ij}(b), b \in \{0,1\}$, mensagens de nós de variável para nós de verificação, e

$r_{ji}(b)$, mensagens de nós de verificação para nós de variável. O conjunto de nós de variável conectados ao nó de verificação f_j é denotado por V_j, ao passo que o conjunto de nós de verificação conectados ao nó de variável x_i é chamado de C_i. Uma vez que excluamos o nó x_i do conjunto V_j, teremos um novo conjunto denotado por $V_{j|i}$. De maneira análoga, o conjunto $C_{i|j}$ se refere aos nós de verificação conectados a x_i exceto o nó f_j. Finalizando as definições, temos a probabilidade *a posteriori* $p_i = \Pr(x_i = 1 \,|y_i)$.

As regras do algoritmo SP no domínio das probabilidades para um código LDPC são descritas a seguir.

1. Inicialização:

O algoritmo é inicializado com o cálculo das mensagens $q_{ij}(b) = Pr(x_i = b|y_i)$ para todo elemento $h_{ij} = 1$ na matriz $\mathbf{H}$. Os nós de variável processam suas mensagens $q_{ij}(1) = p_i$ e $q_{ij}(0) = 1 - p_i$ de acordo com as equações (4.19) e (4.20). Tais probabilidades devem-se ao modelo do canal, que é AWGN. Em seguida, estas mensagens são passadas aos seus nós de verificação vizinhos.

2. Atualização para os nós de verificação:

Neste momento, tem início a primeira etapa de iteração, em que os nós de verificação recebem as mensagens vindas dos nós de variável e processam suas mensagens $r_{ji}(b)$. Para o cálculo de $r_{ji}(b)$, primeiramente precisamos mencionar o resultado exposto em [27], p. 4:

"*Considere uma sequência de n dígitos binários independentes tal que* $Pr(a_l = 1) = p_l$. *A probabilidade de que ocorra um número par de 1's é*

$$\frac{1}{2} + \frac{1}{2}\prod_{l=1}^{n}(1 - 2p_l)\ . \text{"} \tag{4.33}$$

Portanto, substituindo p_l por $q_{i'j}(1)$ temos que:

$$r_{ji}\,(0) = \frac{1}{2} + \frac{1}{2}\prod_{i' \in V_j|i}\left(1 - 2q_{i'j}\,(1)\right) \tag{4.34}$$

e

$$r_{ji}(1) = 1 - r_{ji}(0) \ . \tag{4.35}$$

As mensagens $r_{ji}(b)$ são então passadas aos nós de variável vizinhos.

3. Atualização para os nós de variável:

Na segunda etapa de iteração, os nós de variável calculam novamente suas mensagens $q_{ij}(b)$ segundo as equações

$$q_{ij}\,(0) = K_{ij}\,(1 - p_i) \prod_{j' \in C_i|j} r_{j'i}\,(0) \tag{4.36}$$

e

$$q_{ij}\,(1) = K_{ij}\ p_i \prod_{j' \in C_i|j} r_{j'i}\,(1)\ , \tag{4.37}$$

nas quais K_{ij} é uma constante escolhida tal que $q_{ij}(0) + q_{ij}(1) = 1$.

4. Término:

Completada uma iteração, o algoritmo calcula as pseudo probabilidades *a posteriori* $Q_i(b)$ dadas por

$$Q_i\,(0) = K_i\ (1 - p_i) \prod_{j \in C_i} r_{ji}\,(0) \tag{4.38}$$

e

$$Q_i\,(1) = K_i\ p_i \prod_{j \in C_i} r_{ji}\,(1)\ , \tag{4.39}$$

em que K_i é uma constante tal que $Q_i(0) + Q_i(1) = 1$. Em seguida, o decodificador calcula a estimativa do valor mais provável para cada bit codificado, pela regra

$$\widehat{x}_i = \begin{cases} 0, & se \ \ Q_i(0) \geq Q_i(1) \\ 1, & caso\ contrário \end{cases} \ . \tag{4.40}$$

Após obter a sequência $\widehat{\mathbf{x}}$, o algoritmo checa se $\widehat{\mathbf{x}}\mathbf{H}^T = \mathbf{0}$, ou seja, se $\widehat{\mathbf{x}}$ é uma palavra-código válida. Caso $\widehat{\mathbf{x}}\mathbf{H}^T \neq \mathbf{0}$, o algoritmo retorna ao passo 2, prosseguindo até o passo 4 onde checará novamente se $\widehat{\mathbf{x}}$ é uma palavra-código válida. Um erro é declarado se um número fixo de iterações for executado sem que uma palavra-código válida seja encontrada. Por outro

lado, se $\widehat{\mathbf{x}} \in C$, o algoritmo se encerra. Entretanto, encontrar uma palavra-código válida não significa encontrar a palavra-código transmitida. Uma vez que isto aconteça, temos o que se chama de erro *indetectável*. Caso contrário, o erro é *detectável*.

O critério de parada descrito anteriormente difere daquele comumente usado em decodificadores de códigos turbo, que adotam um número fixo de iterações, já mencionado na Seção 4.3.5. Neste caso, o critério de parada aumenta o tempo computacional e principalmente, pode inserir erros indevidos se o algoritmo prosseguir mesmo após ter achado uma estimativa $\widehat{\mathbf{x}}$ válida.

Nesta Seção, procuramos descrever de uma maneira geral o algoritmo SP no domínio das probabilidades. O estudo da implementação deste algoritmo em hardware tornou-se uma interessante área da pesquisa científica. Um dos objetivos deste estudo consiste em evitar a complexidade e a instabilidade numérica do algoritmo em sua forma original, ou seja, no domínio das probabilidades. Para isto, podemos utilizar a versão do algoritmo SP no domínio dos logaritmos [38], que será abordada na Seção a seguir.

4.4.4 Algoritmo de Decodificação Iterativa (Domínio dos Logaritmos)

O algoritmo SP no domínio dos logaritmos ou algoritmo Log-SP é uma simplificação que visa reduzir a complexidade de execução ao preço de uma pequena perda de desempenho. Neste caso, as mensagens entre os nós de variável e os nós de verificação do grafo são razões de log-verossimilhança [5]. O uso de LLRs é mais vantajoso para a implementação prática do algoritmo, pois as multiplicações são substituídas por somas, evitando assim problemas de precisão numérica.

O algoritmo Log-SP possui diversas aproximações, como o algoritmo de Soma Mínima (SM) [39] e o algoritmo SM com fator de correção. O objetivo do algoritmo SM é evitar o manuseio da função tangente hiperbólica usada na implementação força-bruta do algoritmo Log-SP. Como é de se esperar, a simplificação proporcionada pelo algoritmo SM acarreta uma

[5] O termo razão de log-verossimilhança corresponde ao termo em inglês *Log-Likelihood Ratio* (LLR). Daqui em diante, utilizaremos a sigla LLR por praticidade.

perda de desempenho. Para compensar esta perda, podemos utilizar o algoritmo SM com fator de correção. Desta forma, evitamos a utilização de uma função complexa de se implementar e ainda minimizamos a perda associada ao algoritmo SM. A partir de agora, sempre que nos referirmos ao algoritmo de decodificação Log-SP, estaremos implicitamente nos reportando ao algoritmo SM com fator de correção.

Antes de explicarmos como funcionam as regras de atualização de mensagens no algoritmo Log-SP, precisamos definir alguns conceitos importantes da álgebra de verossimilhança [40]. Seja U uma v. a. binária, vamos definir a LLR $L(U)$ como

$$L(U) \triangleq \ln\left(\frac{P(U=0)}{P(U=1)}\right) \ . \tag{4.41}$$

Considere agora as v. a. binárias independentes U_1 e U_2, com probabilidades $P(U_i = b)$, $b \in 0, 1$ e LLRs L_i . A LLR da soma $S_2 = (U_1 \oplus_2 U_2)$ é definida por

$$L(S_2) = L(U_1 \oplus_2 U_2) = \ln\left(\frac{1 + e^{L_1+L_2}}{e^{L_1} + e^{L_2}}\right) \ . \tag{4.42}$$

Se a soma envolver mais uma variável como, por exemplo, $S_3 = (U_1 \oplus_2 U_2 \oplus_2 U_3)$, temos que

$$L(S_3) = \ln\left(\frac{1 + e^{L(S_2)+L_3}}{e^{L(S_2)} + e^{L_3}}\right) \ . \tag{4.43}$$

Para representar esta soma da álgebra de verossimilhança, utilizaremos o operador $\boxplus$. Portanto, podemos escrever que

$$L(S_2) = L(U_1 \oplus_2 U_2) = L(U_1) \boxplus L(U_2)$$

e

$$L(S_3) = L(U_1 \oplus_2 U_2 \oplus_2 U_3) = L_1 \boxplus L_2 \boxplus L_3 \ .$$

Este conceito se estende para a soma de n variáveis.

Considere agora x e y, dois números reais, tal que o logaritmo Jacobiano [41] seja definido como

$$J(x, y) \triangleq ln(e^x + e^y) \tag{4.44}$$

e aproximado por

$$J(x,y) \approx max(x,y) + ln(1 + e^{-|x-y|}) \ . \tag{4.45}$$

A partir das equações (4.42) e (4.45), temos que

$$\begin{aligned} L_1 \boxplus L_2 &= J(0, L_1 + L_2) - J(L_1, L_2) \\ &= max(0, L_1 + L_2) - max(L_1, L_2) + s(L_1, L_2) \ , \end{aligned}$$

em que $s(L_1, L_2)$ é denominado *fator de correção* e é dado por

$$s(L_1, L_2) = ln(1 + e^{-|L_1+L_2|}) - ln(1 + e^{-|L_1-L_2|}) \ . \tag{4.46}$$

O fator de correção $s(L_1, L_2)$ pode ser implementado por tabela de quantização ou aproximação por pedaços de funções lineares [38]. De acordo com [40], sabemos que

$$max(0, L_1 + L_2) - max(L_1, L_2) = sinal(L_1) \cdot sinal(L_2) \cdot min(|L_1|, |L_2|) \ .$$

em que

$$sinal(x) = \begin{cases} +1, & x > 0 \\ -1, & x < 0 \end{cases} \ .$$

Sendo assim, concluímos que

$$L_1 \boxplus L_2 = sinal(L_1) \cdot sinal(L_2) \cdot min(|L_1|, |L_2|) + s(L_1, L_2) \ . \tag{4.47}$$

Definidos os conceitos da álgebra de verossimilhança necessários ao entendimento do algoritmo Log-SP, vamos explicar as regras de atualização de mensagens a seguir.

1. Inicialização:

O algoritmo é inicializado com o cálculo das LLRs iniciais $L_c(y_i|x_i)$, dadas pela equação

$$L_c(y_i|x_i) = \ln\left(\frac{p\left(x_i = 0\,|y_i\right)}{p\left(x_i = 1\,|y_i\right)}\right) = -\frac{2y_i\sqrt{E}}{\sigma^2} \ . \tag{4.48}$$

Os nós de variável processam suas mensagens $L(q_{ij}) = L_c(y_i|x_i)$ e as enviam para os seus nós de verificação vizinhos.

2. Atualização para os nós de verificação:

Na primeira etapa de iteração, os nós de verificação recebem as mensagens $L(q_{ij})$ e processam suas mensagens $L(r_{ji})$ por meio da equação

$$L(r_{ji}) = \boxplus_{i' \in V_{j|i}} L\left(q_{i'j}\right) . \tag{4.49}$$

As mensagens $L(r_{ji})$ são então passadas aos nós de variável vizinhos.

3. Atualização para os nós de variável:

Neste momento, os nós de variável calculam novamente suas mensagens $L(q_{ij})$ tal que

$$L(q_{ij}) = L_c\left(y_i|x_i\right) + \sum_{j' \in C_{i|j}} L\left(r_{j'i}\right) . \tag{4.50}$$

4. Término:

Completada uma iteração, o algoritmo calcula as LLRs $L(Q_i)$, dadas por

$$L(Q_i) = L_c\left(y_i|x_i\right) + \sum_{j \in C_i} L\left(r_{ji}\right) . \tag{4.51}$$

Em seguida, o decodificador calcula a estimativa do valor mais provável para cada bit codificado, pela regra

$$\widehat{x}_i = \begin{cases} 0, & se \;\; L(Q_i) < 0 \\ 1, & caso\ contrário \end{cases} . \tag{4.52}$$

Assim como no algoritmo SP no domínio das probabilidades, o algoritmo Log-SP checa se $\widehat{\mathbf{x}}\mathbf{H}^T = \mathbf{0}$ e caso não seja, executa novamente os passos 2, 3 e 4, checando ao final a condição de parada novamente. Os conceitos de erros detectáveis e indetectáveis definidos para o domínio das probabilidades também é válido no domínio dos logaritmos.

A Figura 4.13 ilustra o desempenho de códigos regulares $C^{3,6}$ de MacKay[6] para diversos comprimentos, considerando sinalização antipodal e canal AWGN. As curvas com símbolos vazados representam o desempenho do algoritmo SP ao passo que as curvas com símbolos cheios

[6]Códigos 96.3.963, 204.33.486 e 252.252.3.252 assim referenciados, disponíveis em [42].

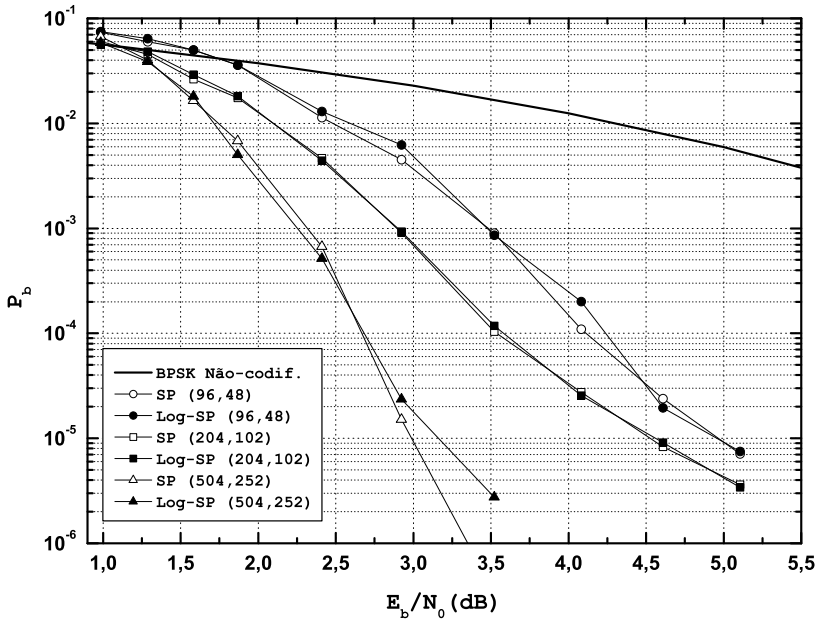


Figura 4.13: Desempenho dos algoritmos SP e Log-SP para os códigos regulares $C^{3,6}$ de Mackay com comprimento $n_c = 96, 204$ e 504 .

representam o desempenho do algoritmo Log-SP. Pode-se perceber que o desempenho dos dois algoritmos estão bem próximos, principalmente no que se refere ao código de comprimento $n_c = 204$. Observa-se ainda um princípio de patamar de erro (do inglês, *error floor*) na curva de desempenho do código de comprimento $n_c = 204$ a partir de uma probabilidade de erro de bit $P_b = 3 \cdot 10^{-5}$.

4.5 Síntese do Capítulo

Este Capítulo apresentou inicialmente as definições básicas sobre grafos-fatores e a descrição do algoritmo Soma-Produto. Em seguida, ele abordou como os grafos-fatores podem ser utilizados para modelar sistemas codificados por meio das modelagens comportamental e probabilística. O algoritmo de decodificação aplicado a estes sistemas foi tratado posteriormente, com destaque tanto para os grafos sem ciclos quanto para os grafos com ciclos. Por fim, o Capítulo apresentou de maneira resumida os códigos LDPC, enfatizando

suas formas de representação e seus algoritmos de decodificação iterativa, nas versões domínio das probabilidades e domínio dos logaritmos.

5 Proposta de Esquema de Codificação Eficiente em Faixa

O principal objetivo das técnicas de codifcação eficientes em faixa é maximizar a eficiência espectral do sistema. Uma das técnicas utilizadas para atingir este objetivo é a combinação de esquemas de modulação e codificação. Este Capítulo aborda uma proposta de esquema de codificação eficiente em faixa para canais com ruído de fase. Inicialmente, os componentes do transmissor são apresentados. Em seguida, um receptor iterativo descrito por um grafo, inspirado em [43], é proposto. Conceitos, como função objetivo e distribuições canônicas, são definidos. Posteriormente, as mensagens que compõem o algoritmo e o seu cronograma de execução são apresentados. Por fim, resultados de simulação são apresentados com as respectivas conclusões.

5.1 Introdução

O crescimento rápido dos sistemas de comunicações pessoais e celulares tem motivado a procura por soluções eficientes tanto em potência quanto em largura de banda para canais de comunicação sem fio variantes no tempo. Em tais canais, as variações de fase na portadora do sinal transmitido ocorre, dentre outras causas, pelo movimento relativo entre transmissor e receptor, conforme mencionado no Capítulo 3. Uma vez que obter sincronismo de fase nestes canais é algo extremamente difícil, existem basicamente duas linhas de ação para lidar com este problema.

A primeira delas consiste em se adicionar informação redundante à sequência transmitida e usar tal informação para obter uma referência de fase que será utilizada na detecção coerente efetuada pelo receptor. Por exemplo, esta informação redundante pode se apresentar na forma de símbolos-piloto que podem ser inseridos no início de cada bloco de símbolos transmitidos. O uso de símbolos-piloto é uma solução amplamente utilizada na maioria dos sistemas celulares atuais. Entretanto, a presença de símbolos-piloto acarreta uma perda na potência transmitida, gerando um compromisso entre a potência alocada entre os símbolos-piloto e os símbolos codificados. A outra técnica utilizada consiste na detecção não-coerente, que elimina a necessidade de se estimar a fase introduzida pelo canal.

Devido à sua complexidade exponencial, a implementação de receptores MAP ótimos para sistemas de comunicações que utilizam canais não-coerentes é uma tarefa impraticável. Desta forma, a busca por soluções sub-ótimas tem sido o alvo dos esforços voltados a este ramo da pesquisa. Algoritmos poderosos de estimação de fase e decodificação conjuntas têm sido estudados na literatura. Nuriyev e Anastasopoulos propuseram receptores iterativos que executam estimação da fase da portadora separadamente da detecção e estimação da fase conjuntamente com a detecção de uma maneira iterativa [44]. Em ambos os casos, um esquema de transmissão com código baseado em símbolos-piloto foi considerado. Peleg et al. investigaram sistemas codificados concatenados para comunicação em canais AWGN com ruído de fase [45]. O transmissor foi formado pela concatenação serial de códigos turbo

ou convolucionais com entrelaçadores e modulações PSK com codificação diferencial. Já o receptor utilizou demodulação de canal e decodificação conjuntas de forma iterativa.

O sistema de comunicações proposto nesta tese está representado pelo diagrama em blocos da Figura 5.1 . O sistema de transmissão é baseado na linha de ação que descarta o uso de referência de fase na recepção. A seguir, vamos descrever os elementos que compõem o sistema proposto, iniciando pelo bloco transmissor.

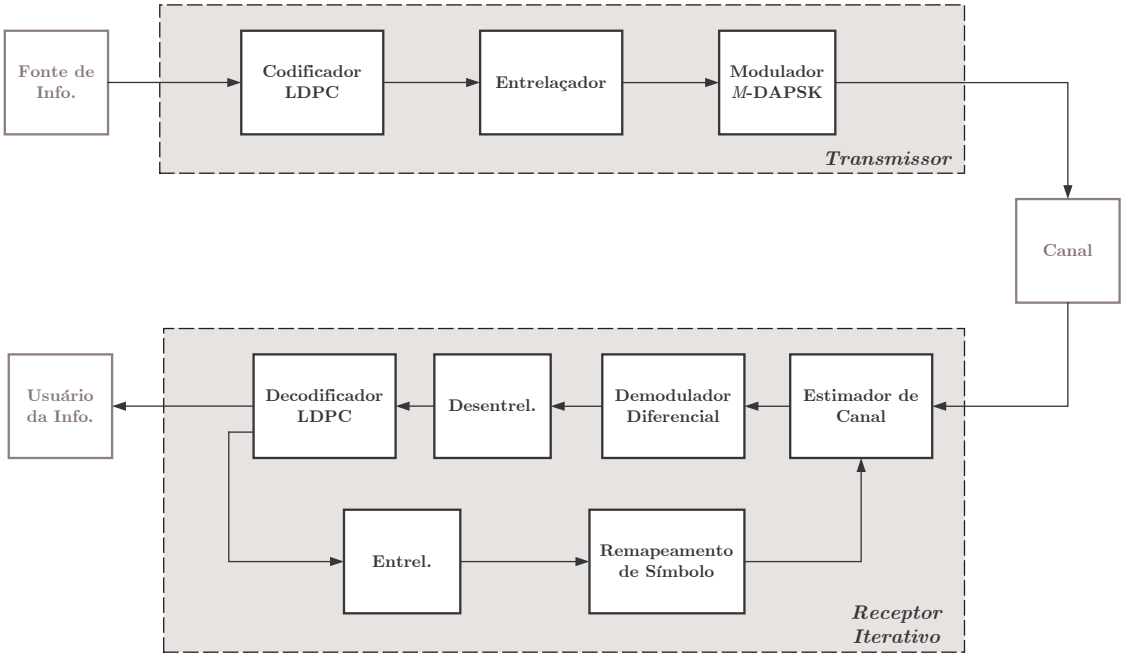


Figura 5.1: Diagrama em blocos do sistema de comunicações proposto.

5.2 Proposta de Transmissor

A Figura 5.2 ilustra o diagrama em blocos do transmissor do sistema de comunicações sugerido. Ele é formado por três componentes: o codificador de canal, mais especificamente um codificador LDPC; o entrelaçador e o modulador M-DAPSK (do inglês, *M-ary Differential Amplitude Phase Shift Keying*) . O codificador LDPC, como o próprio nome já diz, utiliza códigos LDPC regulares, que foram definidos e explicados previamente no Capítulo 4. Os códigos LDPC utilizados no transmissor proposto neste trabalho foram extraídos de [42].

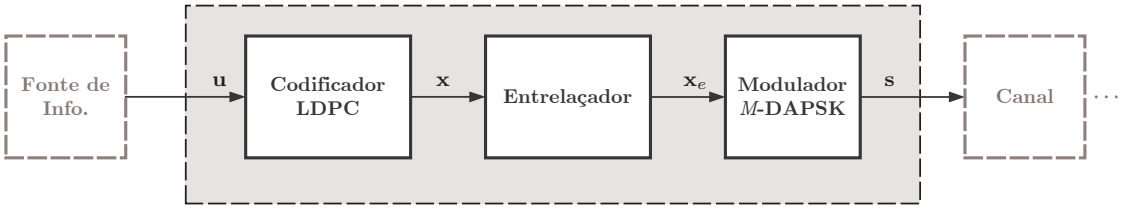


Figura 5.2: Diagrama em blocos do transmissor.

O entrelaçador utilizado no transmissor proposto é o entrelaçador de bloco clássico [46]. A palavra-código (entrada) é dividida em N_r subsequências de bits, de tamanho N_c cada uma delas. Estas subsequências preenchem, linha a linha, uma matriz de dimensões N_r x N_c. Em seguida, os bits entrelaçados (saída) são obtidos pela leitura coluna a coluna dessa matriz. A Figura 5.3 representa uma matriz de entrelaçamento hipotética de dimensões $N_r = 4$ x $N_c = 7$ que manipula palavras-código de comprimento $n_c = 28$ e ilustra graficamente o que acabamos de explicar. Na recepção, o processo inverso é executado. Como apenas um código é utilizado na concatenação com o entrelaçador e o modulador, fica caracterizado o uso do esquema de modulação codificada denominado BICM (do inglês, *Bit-Interleaved Coded Modulation*) [47].

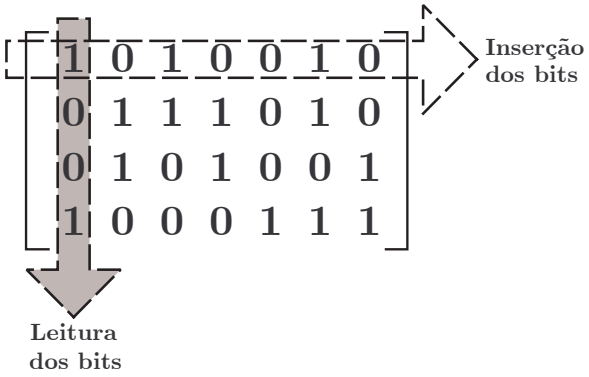


Figura 5.3: Representação do processo de entrelaçamento linha-coluna.

O modulador M-DAPSK é composto por um bloco mapeador M-APSK e um codificador diferencial, conforme ilustra a Figura 5.4. A função do bloco mapeador M-APSK é receber os bits codificados entrelaçados e gerar os símbolos do alfabeto M-APSK. Cada símbolo M-APSK é mapeado por uma sequência de m bits tal que $M = 2^m$. Logo, a sequência entrelaçada $\mathbf{x}_e$

é dividida em subsequências $\mathbf{b}_i = (b_i^1, \ldots, b_i^\alpha, \ldots, b_i^m)$ que são mapeadas nos símbolos da constelação. Vamos utilizar a constelação 8-APSK$(2, 4)$ para ilustrar estes conceitos. Cada símbolo 8-APSK é rotulado por sequência de $m = 3$ bits, que iremos denotar por $\mathbf{b}_i = [b_i^1, b_i^2, b_i^3]$. O bit mais significativo, b_i^1, é utilizado para representar o nível de amplitude do símbolo 8-APSK ao passo que os bits b_i^2 e b_i^3 têm a função de indicar o valor da fase.

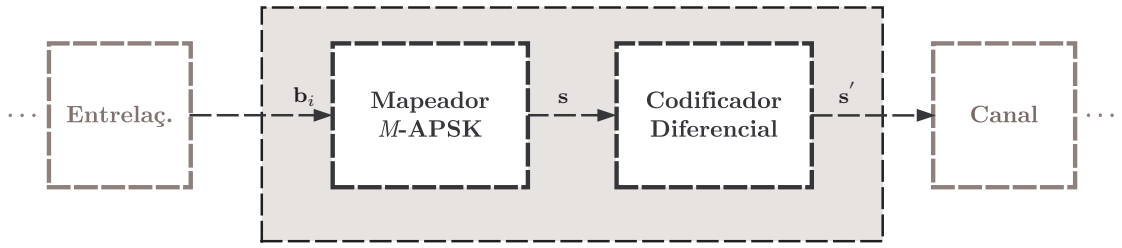


Figura 5.4: Diagrama em blocos do modulador M-DAPSK.

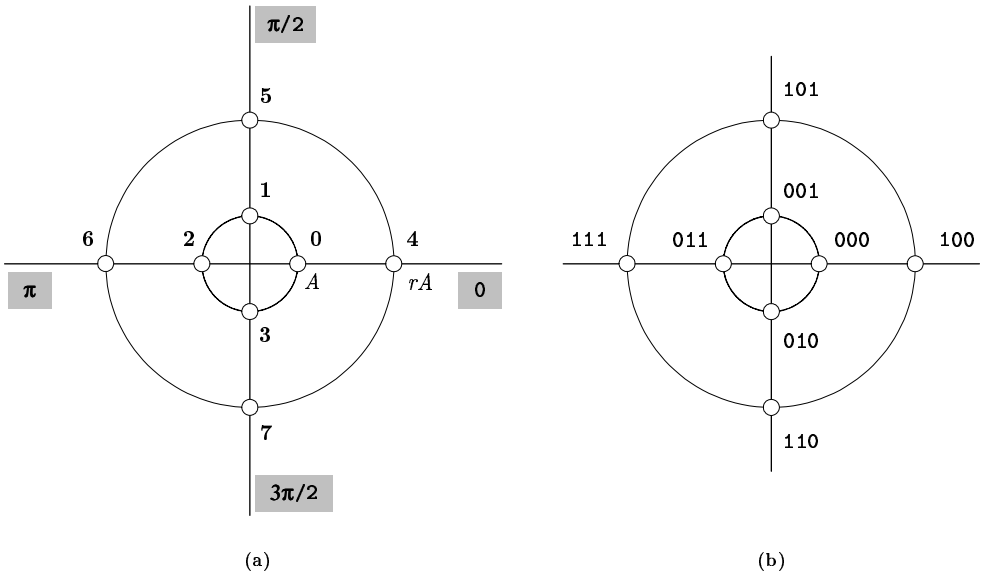
As Figuras 5.5(a) e (b) ilustram a constelação 8-APSK$(2, 4)$ e o rotulamento dos símbolos da constelação. O bit b_i^1 afeta a transição entre os níveis de amplitude A e rA. Já as transições de fase são dadas pelo mapeamento dos bits remanescentes b_i^2 e b_i^3. O mapeamento de fase é tal que, em um mesmo nível de amplitude, de uma fase para outra apenas um bit varia (Mapeamento Gray), conforme ilustra a Figura 5.5(b).

Completando o bloco modulador temos o codificador diferencial, que segundo o próprio nome, tem a função de codificar diferencialmente os símbolos 8-APSK provenientes do mapeador. Os resultados obtidos no Capítulo 3 relativos à convergência da capacidade de canal não-coerente para a capacidade de canal coerente motivaram o uso da codificação diferencial. Vamos definir como funciona esta codificação a seguir.

Seja $s_i^{'}$ o sinal banda básica do i-ésimo símbolo codificado diferencialmente, obtido pela equação

$$s_i^{'} = s_{i-1} \oplus_M s_{i-1}^{'} \ , \tag{5.1}$$

na qual $s_{i-1}^{'}$ é o símbolo diferencial anterior, s_{i-1} é o símbolo proveniente do mapeador M-APSK e o operador $\oplus_M$ representa soma módulo M. Podemos escrever $s_i^{'}$ como função da



(a) (b)

Figura 5.5: (a)Representação dos sinais da constelação 8-APSK$(2, 4)$. (b)Rotulamento dos símbolos.

amplitude e da fase, tal que

$$s_i^{'} = a_i^{'} \exp(j\phi_i^{'}) \ . \tag{5.2}$$

Os símbolos $s_{i-1}^{'}$ e s_{i-1} podem ser representados de maneira semelhante.

O nível de amplitude $a_{i-1}^{'}$ é definido por

$$a_{i-1}^{'} = r^{D_{i-1}^{'}} A \ . \tag{5.3}$$

Analogamente, temos que $a_{i-1} = r^{D_{i-1}} A$. As variáveis $D_{i-1}^{'}$ e D_{i-1} dependem do valor de N. No caso da constelação 8-APSK$(2, 4)$, temos que $N = 2$ e portanto, $D_{i-1}^{'}$ e D_{i-1} são variáveis binárias e $D_{i-1} = b_{i-1}^{1}$. Assim, podemos definir a variável binária Δ_i, que representa a codificação diferencial das amplitudes, como

$$\Delta_i = D_{i-1} \oplus_2 D_{i-1}^{'} \ . \tag{5.4}$$

A Tabela 5.1 indica a relação entre Δ_i e o nível de amplitude $a_i^{'}$.

Conforme mencionado anteriormente, os bits b_{i-1}^{2} e b_{i-1}^{3} servem para mapear o valor da fase, mais especificamente a fase ϕ_{i-1}. A Tabela 5.2 indica como se dá este mapeamento de

Δ_i	$a_i^{'}$
0	A
1	rA

Tabela 5.1: Relação entre Δ_i e o nível de amplitude $a_i^{'}$ para a constelação 8-APSK$(2,4)$.

$b_{i-1}^2 b_{i-1}^3$	ϕ_{i-1}
00	0
01	$\pi/2$
11	π
10	$3\pi/2$

Tabela 5.2: Mapeamento diferencial para a fase na constelação 8-APSK$(2,4)$.

fase. A partir daí, a fase $\phi_l^{'}$ é obtida pelas expressões

$$\phi_i^{''} = \phi_{i-1}^{'} + \phi_{i-1} \tag{5.5}$$

e

$$\phi_i^{'} = (\phi_i^{''})_{mod\ 2\pi}\ . \tag{5.6}$$

Se $\phi_i^{''} \geq 2\pi$, precisamos inverter o valor de Δ_i tal que

$$\Delta_i^n = \bar{\Delta}_i^a \tag{5.7}$$

antes de obtermos o nível de amplitude $a_i^{'}$. As variáveis Δ_i^n e Δ_i^a correspondem ao novo valor e ao antigo valor de Δ_i, respectivamente.

Para exemplificar o que acabamos de definir para o codificador diferencial, considere o exemplo a seguir. Seja o sinal $s_{i-1}^{'} = rAe^{j\pi}$, identificado na Figura 5.5(b) pelo rótulo 111. Considere agora que o mapeador recebeu em sua entrada a sequência $\mathbf{b}_{i-1} = [110]$. Assim, temos que $b_{i-1}^1 = 1$ e consequentemente, $a_{i-1} = rA$. Pela Tabela 5.2, vemos que os bits $b_{i-1}^2 b_{i-1}^3 = 10$ correspondem a uma fase $\phi_{i-1} = 3\pi/2$. Estas informações são passadas ao codificador diferencial para a obtenção do sinal $s_i^{'}$, o que é feito pela determinação da amplitude

$a_i^{'}$ e da fase $\phi_i^{'}$. Primeiramente, temos que

$$\Delta_i = D_{i-1} \oplus_2 D_{i-1}^{'} = 1 \oplus_2 1 = 0 \ .$$

Em seguida, usando a equação (5.5) obtemos que $\phi_i^{''} = 5\pi/2$. Logo, um novo valor de Δ_i é obtido tal que

$$\Delta_i = \Delta_i^n = 0 \oplus_2 1 = 1 \ .$$

De posse da variável $\Delta_i = 1$, concluímos que $a_i^{'} = rA$ observando a Tabela 5.1. Já a fase $\phi_l^{'}$ é obtida por meio da equação (5.6), resultando em $\phi_l^{'} = \pi/2$. Desta maneira, $s_i^{'} = rAe^{j\pi/2}$ o que pode ser comprovado por meio da Figura 5.5(a) pela soma $(6 \oplus_8 7) = 5$.

Uma vez definida a codificação diferencial, o modulador envia blocos de L símbolos codificados pelo canal. Cada bloco possui um símbolo de referência em seu início, denotado por $s_b^{'}$, tal que $b = (k-1)L+1, k = 1, 2, \ldots, N_b$, em que N_b é o número de blocos transmitidos. Esta estratégia de modulação foi utilizada em [48] para canais com desvanecimento Rayleigh de bloco. Sem perda de generalidade, vamos nos referir ao primeiro bloco e portanto, considerar $s_1^{'}$ tal que $a_1^{'} = A$ e $\phi_1^{'} = 0$. A partir daí, cada símbolo $s_i^{'}$ é codificado diferencialmente tendo como referência o símbolo anterior $s_{i-1}^{'}$. Cada bloco de símbolos independe do bloco anterior, ou seja, o primeiro símbolo do bloco não é codificado a partir do último símbolo do bloco anterior, ratificando o fato de que cada bloco possui seu símbolo de referência fixo e independente.

Por fim, para obtermos a taxa de transmissão do sistema denotada por R_s, reconsidere a Figura 5.2. O codificador LDPC recebe k_c bits em sua entrada e fornece n_c bits codificados em sua saída. Ao dividirmos estes n_c bits por m, obtemos o número de símbolos gerados pelo mapeador M-APSK. Como o codificador diferencial recebe $(L-1)$ símbolos de informação codificada, temos que

$$N_b = \frac{n_c}{m\,(L-1)} \ . \tag{5.8}$$

Considerando todo o bloco transmissor, podemos definir a taxa R_s como

$$R_s = \frac{k_c}{N_b\,L} \ . \tag{5.9}$$

Substituindo a equação (5.8) na (5.9), temos que

$$R_s = r_c\, m\, \frac{(L-1)}{L} \quad (bits/simb)\ . \tag{5.10}$$

5.3 Proposta de Receptor Iterativo

Um dos grandes desafios em se trabalhar com comunicações sem fio é projetar sistemas que sejam robustos à variação do canal. Para tentarmos nos aproximar do desempenho ótimo em um canal AWGN afetado por ruído de fase, podemos utilizar *receptores iterativos*. Estes receptores são assim chamados por utilizarem algoritmos que executam conjuntamente estimação de canal, detecção e decodificação. Algoritmos iterativos para canais com fase desconhecida utilizando grafos têm despertado um grande interesse na literatura recente [43], [49], [50]. Portanto, nosso objetivo nesta Seção é desenvolver um grafo no qual algoritmos iterativos que executam conjuntamente estimação de canal e decodificação possam ser aplicados.

5.3.1 Função Objetivo

Reconsidere o sistema de comunicações apresentado na Figura 2.1. Para facilitar o entendimento, suponha que o modulador que compõe o transmissor utiliza sinalização antipodal. A sequência de informação $\mathbf{u} \in \{0,1\}^{K_c}$ é codificada em uma palavra-código $\mathbf{x} \in C \subset \{0,1\}^{N_c}$. O canal, que introduz uma fase aleatória, é completamente descrito por sua probabilidade de transição $p(\mathbf{r}|\mathbf{x},\boldsymbol{\Theta})$ e pela probabilidade *a priori* do estado do canal $p(\boldsymbol{\Theta})$, que é considerada independente do sinal de entrada. A palavra recebida $\mathbf{r}$, corrompida pelo canal, é então processada pelo receptor iterativo, que desconhece o vetor $\boldsymbol{\Theta}$, para gerar estimativas da informação transmitida $\hat{\mathbf{u}}$.

A regra de decisão ótima utilizada pelo receptor iterativo para estimar o bit de informação $\widehat{u}_l$ é dada pela equação

$$\widehat{u}_l \triangleq \arg\max_{u_l} p\left(u_l \mid \mathbf{r}\right)\ . \tag{5.11}$$

Assumiremos que o codificador é sistemático, logo a estimativa da mensagem é um subconjunto

das estimativas dos bits da palavra-código [51]. Sendo assim, a equação (5.11) pode ser reescrita como

$$\begin{aligned} \widehat{x}_l &= \arg\max_{x_l} p\left(x_l \mid \mathbf{r}\right) \\ &= \arg\max_{\varsigma} \sum_{\mathbf{x}:\, x_l=\varsigma} p\left(\mathbf{x}\,|\mathbf{r}\right) . \end{aligned} \tag{5.12}$$

Aplicando a Regra de Bayes e considerando o estado do canal, temos que

$$\widehat{x}_l = \arg\max_{\varsigma} \sum_{\mathbf{x}:\, x_l=\varsigma} \int p\left(\mathbf{r}\,|\mathbf{x}, \boldsymbol{\Theta}\right) p\left(\boldsymbol{\Theta}\right) p\left(\mathbf{x}\right) d\boldsymbol{\Theta} . \tag{5.13}$$

Assumindo que as palavras-código são equiprováveis,

$$p\left(\mathbf{x}\right) = \frac{1}{|C|} I_C(\mathbf{x}) . \tag{5.14}$$

Como $1/|C|$ independe de $\mathbf{x}$, chegamos à regra de decisão

$$\widehat{x}_l = \arg\max_{\varsigma} \sum_{\mathbf{x}:\, x_l=\varsigma} \int p\left(\mathbf{r}\,|\mathbf{x}, \boldsymbol{\Theta}\right) p\left(\boldsymbol{\Theta}\right) I_C(\mathbf{x}) d\boldsymbol{\Theta} . \tag{5.15}$$

A função global representada pelo grafo denomina-se *função objetivo* e é definida por

$$G\left(\mathbf{x}, \boldsymbol{\Theta}\right) = p\left(\mathbf{r}\,|\,\mathbf{x}, \boldsymbol{\Theta}\right) p\left(\boldsymbol{\Theta}\right) I_C(\mathbf{x}) . \tag{5.16}$$

Para obter algoritmos passíveis de implementação, é necessário que a função objetivo seja fatorada de forma que as operações de atualização do algoritmo SP possam ser realizadas. Além disso, a fatoração da função objetivo permite que o receptor seja representado por meio de um grafo mais simples, em que os graus dos nós são pequenos.

Para o nosso caso, a função objetivo G_s pode ser escrita por meio da equação

$$G_s(\mathbf{s}^{'}, \boldsymbol{\Theta}) = I_C(\mathbf{x}) I\{s_i = V(\mathbf{b}_i)\} I\{s_i^{'} = s_{i-1} \oplus_M s_{i-1}^{'}\} I\{s_b^{'} = 0\} p(\mathbf{r}|\,\mathbf{s}^{'}, \boldsymbol{\Theta}) p(\boldsymbol{\Theta}) . \tag{5.17}$$

A função indicadora $I_C(\mathbf{x})$ pode ser representada como

$$\prod_{t=1}^{m_c} I\left\{\mathbf{h}_t \cdot \mathbf{x} = 0\right\} , \tag{5.18}$$

s_i	$b_i^1 b_i^2 b_i^3$
0	0 0 0
1	0 0 1
2	0 1 1
3	0 1 0
4	1 0 0
5	1 0 1
6	1 1 1
7	1 1 0

Tabela 5.3: Descrição da função de mapeamento $V(\mathbf{b}_i)$.

em que $\mathbf{h}_t$ é a t-ésima linha da matriz de verificação de paridade do código C e $m_c = (n_c - k_c)$ é o número de linhas da matriz. A função $I\{s_i = V(\mathbf{b}_i)\}$ se refere ao mapeamento de cada subconjunto de m bits da palavra-código $\mathbf{x}$ em um símbolo s_i da modulação. O mapeamento para a constelação 8-APSK$(2, 4)$ está ilustrado na Tabela 5.3 .

A codificação diferencial usada na transmissão é representada pelas funções $I\{s_i^{'} = s_{i-1} \oplus_M s_{i-1}^{'}\}$ e $I\{s_b^{'} = 0\}$. Dessa forma, o modulador codifica diferencialmente o símbolo s_{i-1} de acordo com a equação $s_i^{'} = s_{i-1} \oplus_M s_{i-1}^{'}$, na qual $s_i^{'}$ é o símbolo codificado de saída. Já a função $I\{s_b^{'} = 0\}$ se refere ao símbolo de referência de cada bloco de símbolos.

Para finalizar a fatoração da função G_s , resta-nos analisar as distribuições de probabilidade $p(\mathbf{r}|\,\mathbf{s}^{'}, \mathbf{\Theta})$ e $p(\mathbf{\Theta})$. Considere que o canal aqui estudado não possui interferência intersimbólica, logo, a sua saída é condicionalmente dependente apenas da entrada atual dado o estado do canal. Daí,

$$p(\mathbf{r}|\,\mathbf{s}^{'}, \mathbf{\Theta}) = \prod_{i=1}^{N_s} p(r_i|s_i^{'}, \mathbf{\Theta}) \ , \tag{5.19}$$

em que N_s é o número de símbolos a serem transmitidos.

A fase introduzida pelo canal é constante para cada bloco de L símbolos, assim o canal possui um efeito de memória de bloco. Entretanto, o valor da fase é independente de bloco a bloco, o que nos leva a considerar o canal sem memória se cada bloco de L símbolos for visto

como um sinal a ser transmitido. Portanto, a probabilidade de estado do canal $p(\mathbf{\Theta})$ pode ser expressa como

$$p(\mathbf{\Theta}) = \prod_{k=1}^{N_b} p(\mathbf{\Theta}_k) \ , \tag{5.20}$$

em que $N_b = N_s/L$. Além disso, podemos concluir que

$$p(\mathbf{r}|\, \mathbf{s}^{'}, \mathbf{\Theta}) = \prod_{i=1}^{N_s} p(r_i|s_i^{'}, \mathbf{\Theta}_{\lceil i/L \rceil}) \ , \tag{5.21}$$

em que a função $\lceil \cdot \rceil$ retorna o menor inteiro maior ou igual ao seu argumento.

Dessa forma, a fatoração completa da função objetivo G_s pode ser escrita pela equação

$$\begin{aligned} G_s(\mathbf{s}^{'}, \mathbf{\Theta}) &= \prod_{t=1}^{m_c} I\left\{\mathbf{h}_t \cdot \mathbf{x} = 0\right\} \cdot \prod_{i=1}^{N_s} p(r_i|\, s_i^{'}, \mathbf{\Theta}_{\lceil i/L \rceil}) \cdot I\{s_i = V(\mathbf{b}_i)\} \cdot I\{s_i^{'} = s_{i-1} \oplus_M s_{i-1}^{'}\} \\ &\cdot \ I\{s_b^{'} = 0\} \cdot \prod_{k=1}^{N_b} p(\mathbf{\Theta}_k) \ . \end{aligned} \tag{5.22}$$

5.3.2 Distribuições Canônicas

Em algumas situações, é possível que as mensagens geradas pelo algoritmo SP dependam de funções que possuem argumentos com valores reais. As mensagens formadas a partir destas funções podem se tornar gradativamente complicadas à medida que o algoritmo SP vai sendo executado. Exemplos de tais funções são as distribuições de probabilidade de variáveis contínuas. A manipulação destas funções, assim como as suas simplificações, não é uma tarefa fácil. Logo, para evitar o aumento de complexidade do algoritmo, vamos representar essas mensagens por meio de *distribuições canônicas* paramétricas [51].

A utilização de distribuições canônicas consiste na substituição das mensagens nominais, obtidas a partir de funções dependentes de valores reais, por mensagens calculadas a partir de funções parametrizadas. Essa aproximação deve ser efetuada tanto para as mensagens oriundas de nós de variável quanto para as mensagens provenientes de nós de função. Todavia, não é trivial encontrar bons parâmetros para que o algoritmo resultante tenha complexidade baixa e bom desempenho.

Consideraremos a aproximação que utiliza distribuições canônicas quantizadas. Neste contexto, iremos assumir que a distribuição parametrizada é uma série de funções Delta ponderadas. Sejam X e Y um nó de função e um nó de variável, respectivamente. Denotaremos por $\mu'_{X\to Y}(y)$ a mensagem parametrizada enviada do nó X para o nó Y. Esta mensagem pode ser descrita pela equação

$$\mu'_{X\to Y}(y) = \sum_{z=1}^{Z} a_z \delta(y - \widehat{y}_z) \ , \tag{5.23}$$

na qual y pode assumir Z valores possíveis associados a seus respectivos pesos a_z. Os coeficientes a_z são definidos como amostras das mensagens nominais $\mu_{X\to Y}(y)$ para $y = \widehat{y}_z$. Esta parametrização também é válida para as mensagens $\mu'_{Y\to X}(y)$. Estes conceitos se tornarão mais claros na próxima Seção ao apresentarmos as mensagens envolvidas na execução do algoritmo SP.

5.3.3 Representação Gráfica do Receptor Iterativo

A Figura 5.6 ilustra o grafo que representa o receptor proposto. Iniciando pela parte inferior, temos o sub-grafo 1 que representa o decodificador LDPC. Ele é formado pela barra horizontal C, que representa o conjunto de nós verificação de paridade do código, e pelos nós de variável $(x_1, \ldots, x_{n_c})$ situados logo acima, representando a palavra-código. O bloco Π_e representa o entrelaçador do sistema. Finalizando, temos o sub-grafo 2, que reúne o mapeador de símbolos, o codificador diferencial e o estimador de canal.

As mensagens trocadas pelos nós de variável e nós verificação de paridade do decodificador LDPC seguem as regras definidas na Seção 4.4.4, portanto uma nova descrição torna-se desnecessária. Sendo assim, vamos iniciar a descrição das mensagens pelo nó de variável θ_k.

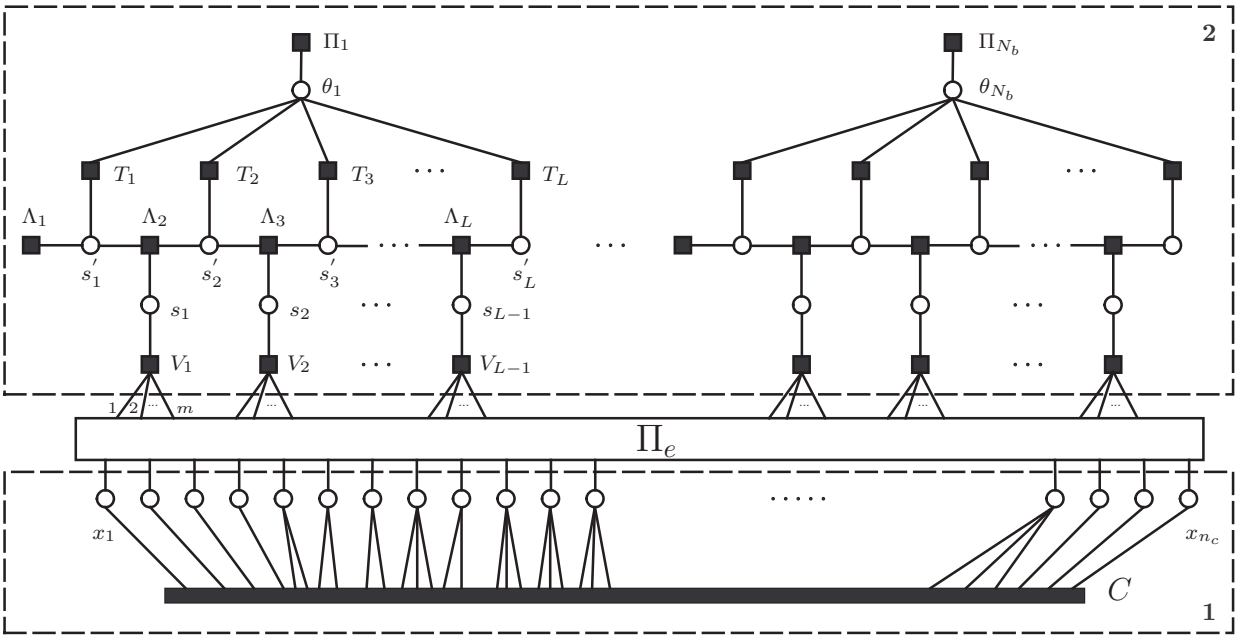


Figura 5.6: Grafo que representa o receptor iterativo para o canal AWGN não-coerente de bloco considerando o uso de codificação diferencial. $C \triangleq I_C(\mathbf{x})\,; T_i \triangleq p(r_i|s_i^{'},\theta_k)\,; \Pi_k \triangleq p(\theta_k)\,; \Lambda_i \triangleq I\{s_i^{'} = s_{i-1} \oplus_M s_{i-1}^{'}\}\,; \Lambda_1 \triangleq I\{s_1^{'} = 0\}; V_i \triangleq I\{s_i = V(\mathbf{b}_i)\}\,.$

Processamento nos nós θ_k

O nó de variável θ_k envia a mensagem $\mu_{\theta_k \to T_i}(\theta_k)$ que, conforme a regra de atualização definida pela equação (4.8), tem seu valor definido por

$$\mu_{\theta_k \to T_i}(\theta_k) = \mu_{\Pi_k \to \theta_k}(\theta_k) \cdot \prod_{\substack{j=(k-1)L+1 \\ j \neq i}}^{kL} \mu_{T_j \to \theta_k}(\theta_k)\ , \tag{5.24}$$

em que $\mu_{T_j \to \theta_k}(\theta_k)$ é a mensagem enviada do nó T_j para o nó θ_k e $\mu_{\Pi_k \to \theta_k}(\theta_k)$ é a mensagem enviada do nó Π_k para o nó θ_k. A mensagem $\mu_{\Pi_k \to \theta_k}(\theta_k)$ descreve a distribuição da fase introduzida pelo canal. Ambas as mensagens serão definidas posteriormente.

Processamento nos nós T_i

O nó de função T_i é caracterizado pela probabilidade de transição do canal e é representado por

$$T_i(r_i, s_i^{'}, \theta_k) \triangleq p(r_i|\, s_i^{'}, \theta_k) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\left\| r_i - s_i^{'} \exp(j\theta_k) \right\|^2}{2\sigma^2} \right) . \tag{5.25}$$

Segundo a equação (4.9), a mensagem $\mu_{T_i \to \theta_k}(\theta_k)$ pode ser escrita como

$$\begin{aligned} \mu_{T_i \to \theta_k}(\theta_k) &= \sum_{\sim\{\theta_k\}} T_i(r_i, s_i^{'}, \theta_k) \cdot \mu_{s_i^{'} \to T_i}(s_i^{'}) \\ &= \sum_{s_i^{'}} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\left\| r_i - s_i^{'} \exp(j\theta_k) \right\|^2}{2\sigma^2} \right) \cdot \mu_{s_i^{'} \to T_i}(s_i^{'}) \ . \end{aligned} \tag{5.26}$$

Prosseguindo a definição das mensagens, seja a mensagem $\mu_{T_i \to s_i^{'}}(s_i^{'})$ tal que

$$\mu_{T_i \to s_i^{'}}(s_i^{'}) = \int_0^{2\pi} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\left\| r_i - s_i^{'} \exp(j\theta_k) \right\|^2}{2\sigma^2} \right) \mu_{\theta_k \to T_i}(\theta_k) d\theta_k \ . \tag{5.27}$$

A operação de integração substitui o somatório pelo fato da variável θ_k ser contínua. Dependendo da complexidade da mensagem $\mu_{\theta_k \to T_i}(\theta_k)$, o cálculo da equação (5.27) pode se complicar bastante. Logo, faremos uso de uma distribuição canônica para a mensagem $\mu_{\theta_k \to T_i}(\theta_k)$.

Conforme foi mencionado anteriormente, utilizaremos uma distribuição canônica quantizada. Vamos assumir que a fase θ_k é discretizada em Z valores e portanto, que a mensagem parametrizada $\mu_{\theta_k \to T_i}^{'}(\theta_k)$ é dada por

$$\mu_{\theta_k \to T_i}^{'}(\theta_k) = \sum_{z=1}^{Z} a_z \delta(\theta_k - \widehat{\theta}_{kz}) \ . \tag{5.28}$$

Outras boas aproximações, utilizando parametrizações de Fourier e Tikhonov, foram utilizadas em propostas de algoritmos iterativos para outros modelos de canais com ruído de fase em [50] e [52].

Para determinar os coeficientes a_z, podemos calcular a mensagem nominal $\mu_{\theta_k \to T_i}(\theta_k)$ dada pela equação (5.24), considerando as fases discretas $\widehat{\theta}_{kz}$. Matematicamente, os coeficientes a_z

são dados por

$$a_z = \mu_{\theta_k \to T_i}(\theta_k = \widehat{\theta}_{kz}) = \mu_{\Pi_k \to \theta_k}(\theta_k = \widehat{\theta}_{kz}) \cdot \prod_{\substack{j=(k-1)L+1 \\ j \neq i}}^{kL} \mu_{T_j \to \theta_k}(\theta_k = \widehat{\theta}_{kz}) \ . \tag{5.29}$$

Desta maneira, os coeficientes a_z se tornam os parâmetros a serem passados durante esta etapa do algoritmo e a mensagem $\mu_{T_i \to s_i^{'}}(s_i^{'})$ pode ser escrita como

$$\begin{aligned} \mu_{T_i \to s_i^{'}}(s_i^{'}) &= \sum_{\sim\{s_i^{'}\}} T_i(r_i, s_i, \theta_k) \mu_{\theta_k \to T_i}(\theta_k) \\ &= \sum_{z=1}^{Z} a_z \exp\left(-\frac{\left\| y_i - s_i^{'} \exp(j\widehat{\theta}_{kz}) \right\|^2}{2\sigma^2} \right) . \end{aligned} \tag{5.30}$$

Processamento nos nós Π_k

A quantização da fase define os valores da mensagem $\mu_{\Pi_k \to \theta_k}(\theta_k)$, tal que

$$\mu_{\Pi_k \to \theta_k}(\theta_k) = \frac{1}{Z} \ , \tag{5.31}$$

pois a fase do canal possui distribuição uniforme em cada bloco de L símbolos.

Após o cálculo da mensagem $\mu_{T_i \to s_i^{'}}(s_i^{'})$, a parte do grafo relativa à estimação de canal se encerra e tem início o estágio de demodulação diferencial. Este estágio se caracteriza pela execução do algoritmo Progressivo-Regressivo definido na Seção 4.3.4. A etapa progressiva é responsável pelo cálculo das mensagens $\mu_{s_i^{'} \to \Lambda_{i+1}}(s_i^{'})$ e $\mu_{\Lambda_i \to s_i^{'}}(s_i^{'})$.

Processamento nos nós $s_i^{'}$ e Λ_i

A mensagem $\mu_{s_i^{'} \to \Lambda_{i+1}}(s_i^{'})$ é dada pela equação

$$\mu_{s_i^{'} \to \Lambda_{i+1}}(s_i^{'}) = \mu_{\Lambda_i \to s_i^{'}}(s_i^{'}) \cdot \mu_{T_i \to s_i^{'}}(s_i^{'}) \ . \tag{5.32}$$

Especificamente no início de cada k-ésimo bloco de símbolos, temos que a mensagem $\mu_{\Lambda_i \to s_i^{'}}(s_i^{'})$ é dada por

$$\mu_{\Lambda_i \to s_i^{'}}(s_i^{'}) = \begin{cases} 1, & s_i^{'} = 0 \\ 0, & s_i^{'} = 1, 2, \ldots M-1 \end{cases} \ , \tag{5.33}$$

em que $i = (k-1)L+1, k = 1, 2, \ldots, N_b$. A justificativa para os valores da mensagem $\mu_{\Lambda_i \to s_i'}(s_i')$ na equação (5.33) deve-se ao conhecimento do primeiro símbolo de cada bloco, que é o símbolo de referência, pelo receptor.

Para os demais símbolos do bloco, a mensagem $\mu_{\Lambda_i \to s_i'}(s_i')$ é calculada segundo a equação

$$\mu_{\Lambda_i \to s_i'}(s_i') = \sum_{\sim\{s_i'\}} \Lambda_i(s_{i-1}', s_{i-1}, s_i') \cdot \mu_{s_{i-1}' \to \Lambda_i}(s_{i-1}') \cdot \mu_{s_{i-1} \to \Lambda_i}(s_{i-1}) \ , \tag{5.34}$$

na qual $\Lambda_i(s_{i-1}', s_{i-1}, s_i')$ é a função indicadora $I\{s_i' = s_{i-1} \oplus_M s_{i-1}'\}$. Esta função pode ser representada por uma treliça associada à codificação diferencial, conforme ilustra a Figura 5.7. Vamos exemplificar o que foi definido com o cálculo da mensagem $\mu_{\Lambda_2 \to s_2'}(s_2' = 0)$ para o caso

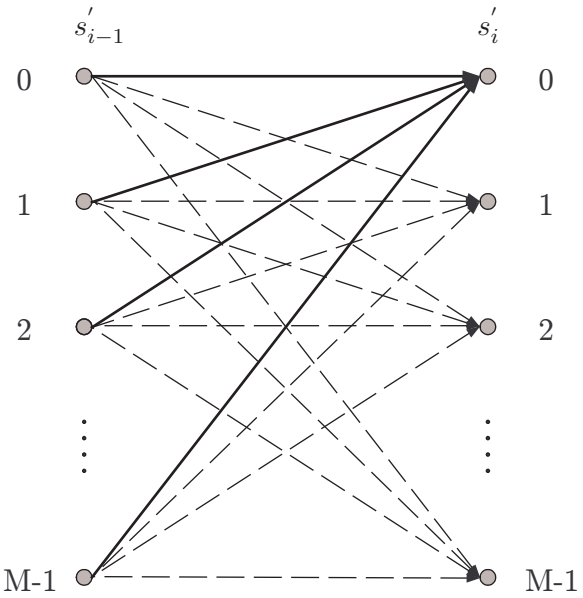


Figura 5.7: Diagrama de treliça de um codificador diferencial de M símbolos.

em que a modulação 8-APSK$(2, 4)$ é utilizada.

De acordo com a equação (5.34), a mensagem $\mu_{\Lambda_2 \to s_2'}(s_2' = 0)$ é dada por uma soma de termos que são obtidos em todas as possibilidades das variáveis s_1' e s_1. Estes termos são o produto das mensagens que chegam ao nó Λ_2, porém a função indicadora $\Lambda_2(s_1', s_1, s_2')$ seleciona apenas aqueles termos que resultam em $s_2' = 0$. Assim, a mensagem $\mu_{\Lambda_2 \to s_2'}(s_2' = 0)$

resulta em

$$\begin{aligned}\mu_{\Lambda_2\to s_2'}(s_2'=0) &= \mu_{s_1'\to\Lambda_2}(s_1'=0)\cdot\mu_{s_1\to\Lambda_2}(s_1=0)+\mu_{s_1'\to\Lambda_2}(s_1'=1)\cdot\mu_{s_1\to\Lambda_2}(s_1=7)\\ &+ \mu_{s_1'\to\Lambda_2}(s_1'=2)\cdot\mu_{s_1\to\Lambda_2}(s_1=6)+\mu_{s_1'\to\Lambda_2}(s_1'=3)\cdot\mu_{s_1\to\Lambda_2}(s_1=5)\\ &+ \mu_{s_1'\to\Lambda_2}(s_1'=4)\cdot\mu_{s_1\to\Lambda_2}(s_1=4)+\mu_{s_1'\to\Lambda_2}(s_1'=5)\cdot\mu_{s_1\to\Lambda_2}(s_1=3)\\ &+ \mu_{s_1'\to\Lambda_2}(s_1'=6)\cdot\mu_{s_1\to\Lambda_2}(s_1=2)+\mu_{s_1'\to\Lambda_2}(s_1'=7)\cdot\mu_{s_1\to\Lambda_2}(s_1=1).\end{aligned} \tag{5.35}$$

Particularizando a Figura 5.7 para o caso em que $M = 8$ e $i = 2$, temos que cada termo da soma da equação (5.35) é representado por um ramo em linha cheia na treliça.

Finalizada a etapa progressiva, inicia-se a etapa regressiva do algoritmo, na qual cada nó de variável s_i' e cada nó de função Λ_i são responsáveis por processar duas mensagens, totalizando quatro tipos diferentes. Em primeiro lugar, os nós de variável s_i' atualizam as mensagens $\mu_{s_i'\to T_i}(s_i')$ e calculam $\mu_{s_i'\to\Lambda_i}(s_i')$. A atualização das mensagens $\mu_{s_i'\to T_i}(s_i')$ se dá por meio da equação

$$\mu_{s_i'\to T_i}(s_i') = \mu_{\Lambda_{i+1}\to s_i'}(s_i')\cdot\mu_{\Lambda_i\to s_i'}(s_i')\ , \tag{5.36}$$

com exceção para o último nó de cada bloco, em que $\mu_{s_i'\to T_i}(s_i') = \mu_{\Lambda_i'\to s_i'}(s_i')$ pois se trata de um nó de variável com apenas dois vizinhos. A outra mensagem calculada pelos nós de variável s_i', $\mu_{s_i'\to\Lambda_i}(s_i')$, é dada por

$$\mu_{s_i'\to\Lambda_i}(s_i') = \mu_{T_i\to s_i'}(s_i')\cdot\mu_{\Lambda_{i+1}\to s_i'}(s_i')\ . \tag{5.37}$$

Posteriormente, os nós de função Λ_i calculam as mensagens $\mu_{\Lambda_i\to s_{i-1}'}(s_{i-1}')$ e $\mu_{\Lambda_i\to s_{i-1}}(s_{i-1})$, tal que

$$\mu_{\Lambda_i\to s_{i-1}'}(s_{i-1}') = \sum_{\sim\{s_{i-1}'\}}\Lambda_i(s_{i-1}', s_{i-1}, s_i')\cdot\mu_{s_i'\to\Lambda_i}(s_i')\cdot\mu_{s_{i-1}\to\Lambda_i}(s_{i-1}) \tag{5.38}$$

e

$$\mu_{\Lambda_i\to s_{i-1}}(s_{i-1}) = \sum_{\sim\{s_{i-1}\}}\Lambda_i(s_{i-1}', s_{i-1}, s_i')\cdot\mu_{s_{i-1}'\to\Lambda_i}(s_{i-1}')\cdot\mu_{s_i'\to\Lambda_i}(s_i')\ . \tag{5.39}$$

Semelhante à equação (5.34), a função indicadora $\Lambda_i(s_{i-1}', s_{i-1}, s_i')$ filtra os termos do somatório de acordo com o valor da variável que é argumento da mensagem de interesse. Desta forma,

o cálculo das mensagens $\mu_{\Lambda_i \to s'_{i-1}}(s'_{i-1})$ e $\mu_{\Lambda_i \to s_{i-1}}(s_{i-1})$ segue o mesmo raciocínio empregado no cálculo de $\mu_{\Lambda_i \to s_i'}(s'_i)$.

Processamento nos nós s_i

Após a etapa regressiva, as mensagens $\mu_{\Lambda_i \to s_{i-1}}(s_{i-1})$ fluem para o trecho inferior do grafo em direção ao decodificador LDPC. Por serem nós de variável com dois vizinhos, os nós s_i apenas encaminham as mensagens $\mu_{\Lambda_i \to s_{i-1}}(s_{i-1})$, agora nomeadas $\mu_{s_i \to V_i}(s_i)$, para os nós de função V_i.

Outra mensagem calculada pelos nós s_i é a mensagem $\mu_{s_i \to \Lambda_{i+1}}(s_i)$. Esta mensagem é simplesmente igual à mensagem $\mu_{V_i \to s_i}(s_i)$, por se tratar de um nó de variável com dois vizinhos. A mensagem $\mu_{V_i \to s_i}(s_i)$ será definida na Subseção a seguir.

Processamento nos nós V_i

A mensagem $\mu_{V_i \to s_i}(s_i)$ é dada por

$$\mu_{V_i \to s_i}(s_i) = \sum_{\sim\{s_i\}} I\{s_i = V(\mathbf{b}_i)\} \prod_{n=1}^{m} \mu_{b_i^n \to V_i}(b_i^n) \ . \tag{5.40}$$

Assim como (5.42), a equação (5.40) depende da função indicadora $I\{s_i = V(\mathbf{b}_i)\}$ referente ao mapeamento de símbolos da Tabela 5.3. Para exemplificar, vejamos como a mensagem $\mu_{V_1 \to s_1}(s_1)$ é calculada para $s_1 = 0$. Neste caso, apenas a primeira linha da Tabela 5.3 é considerada, consequentemente

$$\mu_{V_1 \to s_1}(s_1 = 0) = \mu_{b_1^1 \to V_1}(b_1^1 = 0) \cdot \mu_{b_1^2 \to V_1}(b_1^2 = 0) \cdot \mu_{b_1^3 \to V_1}(b_1^3 = 0) \ . \tag{5.41}$$

Continuando a definição das mensagens, sabemos que cada símbolo s_i é mapeado a partir de uma sequência $\mathbf{b}_i = (b_i^1, \ldots, b_i^\alpha, \ldots, b_i^m)$. Daí, temos que os nós V_i calculam as mensagens $\mu_{V_i \to b_i^\alpha}(b_i^\alpha)$ usando a equação

$$\mu_{V_i \to b_i^\alpha}(b_i^\alpha) = \sum_{\sim\{b_i^\alpha\}} I\{s_i = V(\mathbf{b}_i)\} \mu_{s_i \to V_i}(s_i) \prod_{\substack{n=1 \\ n \neq \alpha}}^{m} \mu_{b_i^n \to V_i}(b_i^n) \ , \tag{5.42}$$

na qual a função $I\{s_i = V(\mathbf{b}_i)\}$ se refere a um dos mapeamentos indicados na Tabela 5.3. Vale ressaltar que a sequência $\mathbf{b}_i$ é um subconjunto da palavra-código $\mathbf{x}$ para evitar confusão na descrição das mensagens de entrada e de saída do decodificador LDPC.

Para esclarecer o cálculo definido pela equação (5.42) vamos considerar novamente a modulação 8-APSK$(2, 4)$, ou seja, $m = 3$, e que desejamos obter a mensagem $\mu_{V_1 \to b_i^1}(b_1^1 = 0)$. Assim como no cálculo de $\mu_{\Lambda_i \to s_i'}(s_i')$, a função indicadora $I\{s_i = V(\mathbf{b}_i)\}$ seleciona os termos do somatório segundo a condição estabelecida pelo valor do argumento da mensagem, que neste caso é a variável b_1^1. Para $b_1^1 = 0$, a função $I\{s_i = V(\mathbf{b}_i)\}$ seleciona as quatro primeiras linhas da Tabela 5.3 e a equação (5.42) pode ser expressa como

$$\begin{aligned}
\mu_{V_1 \to b_1^1}(b_1^1 = 0) &= \mu_{s_1 \to V_1}(s_1 = 0) \cdot \mu_{b_1^2 \to V_1}(b_1^2 = 0) \cdot \mu_{b_1^3 \to V_1}(b_1^3 = 0) \\
&+ \mu_{s_1 \to V_1}(s_1 = 1) \cdot \mu_{b_1^2 \to V_1}(b_1^2 = 0) \cdot \mu_{b_1^3 \to V_1}(b_1^3 = 1) \\
&+ \mu_{s_1 \to V_1}(s_1 = 2) \cdot \mu_{b_1^2 \to V_1}(b_1^2 = 1) \cdot \mu_{b_1^3 \to V_1}(b_1^3 = 1) \\
&+ \mu_{s_1 \to V_1}(s_1 = 3) \cdot \mu_{b_1^2 \to V_1}(b_1^2 = 1) \cdot \mu_{b_1^3 \to V_1}(b_1^3 = 0) \ .
\end{aligned} \tag{5.43}$$

As demais mensagens $\mu_{V_1 \to b_1^\alpha}(b_1^\alpha)$ podem ser deduzidas de maneira análoga.

Processamento nos nós b_i^α

Prosseguindo a descrição, temos as mensagens $\mu_{b_i^\alpha \to V_i}(b_i^\alpha)$, que são as mensagens enviadas pelos nós de variável que representam a palavra-código para os nós de mapeamento da modulação. Tais mensagens correspondem às probabilidades *a priori* dos bits codificados e são dadas por

$$\mu_{b_i^\alpha \to V_i}(b_i^\alpha) = P(b_i^\alpha) \ , \tag{5.44}$$

em que

$$\mu_{b_i^\alpha \to V_i}(b_i^\alpha = 0) = \frac{\exp(LLR(b_i^\alpha))}{1 + \exp(LLR(b_i^\alpha))} \tag{5.45}$$

e

$$\mu_{b_i^\alpha \to V_i}(b_i^\alpha = 1) = \frac{1}{1 + \exp(LLR(b_i^\alpha))} \ . \tag{5.46}$$

Tal consideração é semelhante àquela realizada por Niu et al. em [53].

Uma vez definidas as mensagens que são processadas pelos nós do grafo, vamos estabelecer o cronograma de execução do algoritmo SP na próxima Seção.

5.3.4 Cronograma de Execução do Algoritmo SP

O cronograma de execução do algoritmo SP tem início nos nós x_i e $s_i^{'}$, que estão indicados na cor cinza no grafo da Figura 5.8(a). As mensagens $\mu_{s_i^{'}\rightarrow T_i}(s_i^{'})$ são calculadas por meio da equação

$$\mu_{s_i^{'}\rightarrow T_i}(s_i^{'}) = \begin{cases} 1, & s_i^{'} = 0 \\ 0, & s_i^{'} = 1, 2, \ldots M-1 \end{cases}, \tag{5.47}$$

na qual $i = (k-1)L+1, k = 1, 2, \ldots, N_b$. Os demais símbolos $s_i^{'}$ pertencentes a cada bloco emitem mensagens iniciais $\mu_{s_i^{'}\rightarrow T_i}(s_i^{'}) = 1/M$. Enquanto isso, as mensagens $\mu_{b_i^{\alpha}\rightarrow V_i}(b_i^{\alpha})$ são calculadas pelas equações (5.45) e (5.46), considerando um valor de $LLR(b_i^{\alpha})$ inicial igual a zero. As setas da Figura 5.8(a) indicam o sentido do fluxo das mensagens.

Em seguida, os nós T_i, Π_k e V_i calculam $\mu_{T_i\rightarrow\theta_k}(\theta_k)$, $\mu_{\Pi_k\rightarrow\theta_k}(\theta_k)$ e $\mu_{V_i\rightarrow s_i}(s_i)$, por meio das equações (5.26), (5.31) e (5.40), respectivamente. Esta etapa do cronograma está indicada na Figura 5.8(b) e assim como na etapa anterior, as setas indicam o fluxo das mensagens.

A próxima etapa do cronograma de execução ocorre com o cálculo das mensagens $\mu_{\theta_k\rightarrow T_i}(\theta_k)$ e $\mu_{s_i\rightarrow\Lambda_{i+1}}(s_i)$. As mensagens $\mu_{\theta_k\rightarrow T_i}(\theta_k)$ são calculadas segundo a equação (5.24). As setas em cor cinza no grafo da Figura 5.8(c) representam $\mu_{\theta_k\rightarrow T_i}(\theta_k)$ e $\mu_{s_i\rightarrow\Lambda_{i+1}}(s_i)$. Após isso, os nós T_i geram as mensagens $\mu_{T_i\rightarrow s_i^{'}}(s_i^{'})$, indicadas pelas setas em cor branca na Figura 5.8(c). De posse das mensagens $\mu_{T_i\rightarrow s_i^{'}}(s_i^{'})$ e $\mu_{s_i\rightarrow\Lambda_{i+1}}(s_i)$, tem início a etapa Progressiva-Regressiva do algoritmo SP proposto. As setas em cor preta na Figura 5.8(c) representam as mensagens $\mu_{\Lambda_i\rightarrow s_i^{'}}(s_i^{'})$ e $\mu_{s_i^{'}\rightarrow\Lambda_{i+1}}(s_i^{'})$ calculadas na fase progressiva.

A fase regressiva, responsável pela obtenção das mensagens $\mu_{s_i^{'}\rightarrow\Lambda_i}(s_i^{'})$, $\mu_{s_i^{'}\rightarrow T_i}(s_i^{'})$, $\mu_{\Lambda_i\rightarrow s_{i-1}}(s_{i-1})$ e $\mu_{\Lambda_i\rightarrow s_{i-1}^{'}}(s_{i-1}^{'})$, vem logo a seguir e está ilustrada pelas setas em cor cinza da Figura 5.8(d). A partir deste ponto, o fluxo de mensagens começa a descer com o cálculo de $\mu_{s_i\rightarrow V_i}(s_i)$ (setas em cor branca na Figura 5.8(d)). Dando sequência ao algoritmo, os nós V_i processam as mensagens $\mu_{V_i\rightarrow b_i^{\alpha}}(b_i^{\alpha})$ via entrelaçador. As setas em cor preta da Figura 5.8(d)

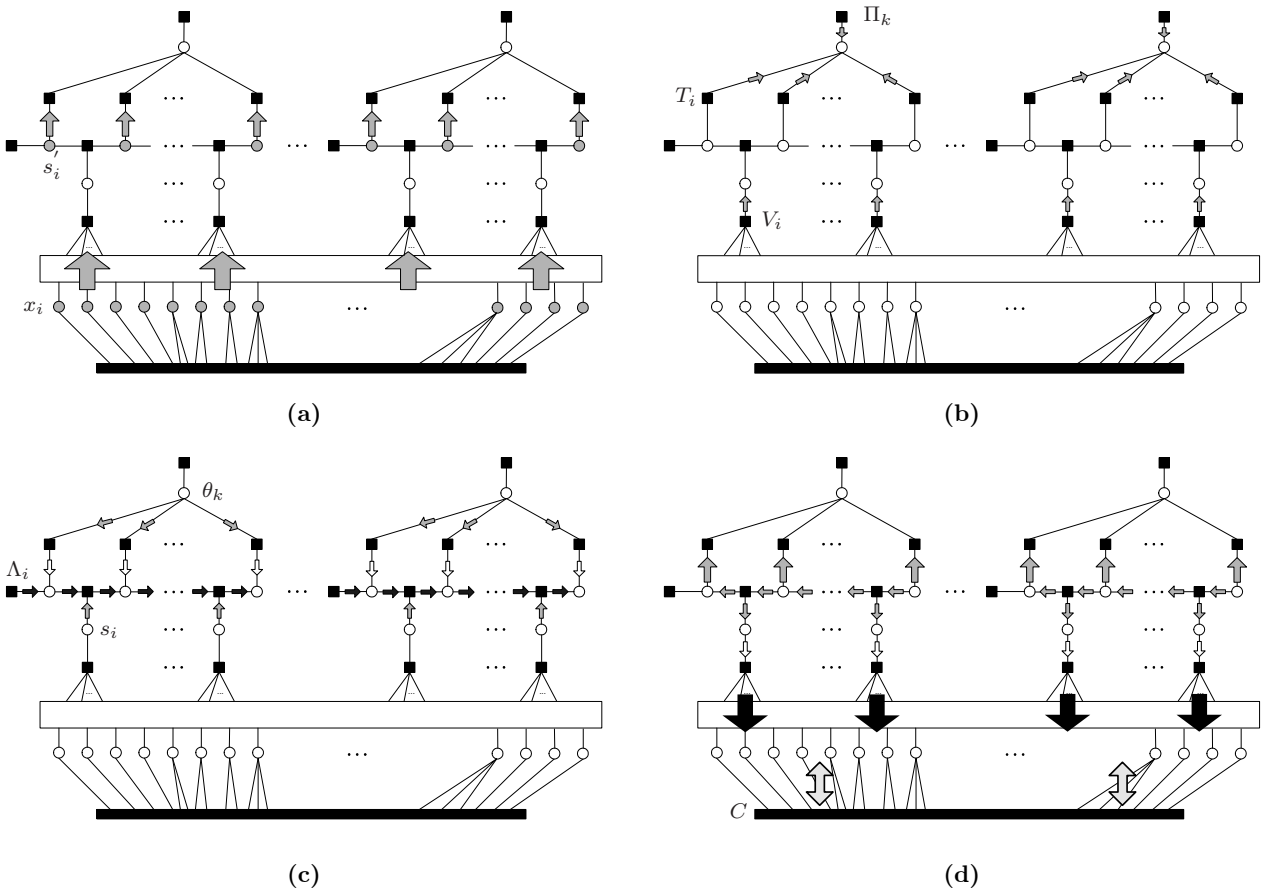


Figura 5.8: Cronograma de execução do algoritmo SP no grafo do receptor iterativo.

representam esta etapa.

Antes de serem passadas ao subgrafo que corresponde ao decodificador LDPC, as mensagens que chegam aos nós de variável x_i são convertidas para o domínio dos logaritmos por meio da equação

$$L_c(x_i) = \ln\left(\frac{\mu_{V_i \to x_i}(x_i = 0)}{\mu_{V_i \to x_i}(x_i = 1)}\right) , \tag{5.48}$$

originando as LLRs iniciais do algoritmo Log-SP, definidas na Seção 4.4.4. A partir daí, as mensagens $L(q_{ij})$ e $L(r_{ji})$ são calculadas de acordo com as regras de atualização do algoritmo Log-SP. Estas mensagens estão representadas na Figura 5.8(d) pelas setas duplas situadas na parte inferior do grafo.

Finalmente, o algoritmo checa a condição de parada, ou seja, se $\widehat{\mathbf{x}}\mathbf{H}^T = \mathbf{0}$. Caso esta

condição não se verifique, a iteração se encerra com o cálculo das mensagens $\mu_{x_i \to V_i}(x_i)$ ou $\mu_{b_i^\alpha \to V_i}(b_i^\alpha)$, se considerarmos as sequências $\mathbf{b}_i$, segundo as equações (5.45) e (5.46) com $LLR(b_i^\alpha) = L(Q_i)$. Em seguida, uma nova iteração do algoritmo é iniciada.

5.4 Análise dos Resultados

Nesta Seção, iremos apresentar o desempenho do sistema considerando a utilização do algoritmo de demodulação e decodificação conjuntas proposto. Este algoritmo foi avaliado por meio de simulações computacionais em termos da probabilidade de erro de bit (P_b) versus $E_b/N_0(dB)$, em que E_b é a energia por bit de informação. A probabilidade de erro de bit analisada nesta Seção considera que o sistema transmitiu a palavra toda nula ao invés de considerar a probabilidade de erro de bit média, o que é mais comum. Espera-se que a P_b real esteja bem próxima dos resultados apresentados.

Inicialmente, consideramos a utilização do código LDPC regular de taxa $r_c = 1/2$ com comprimento $n_c = 96$. A modulação M-APSK utilizada foi a 8-APSK$(2, 4)$ com distribuição de entrada uniforme e razão de raio r=2,42. A fase introduzida pelo canal foi considerada constante em um bloco de $L = 9$ símbolos. A Figura 5.9 ilustra o desempenho do sistema para três níveis diferentes de quantização da fase (Z) introduzida pelo canal. Além disso, também é mostrado o desempenho quanto ao número de iterações (N_{it}) do algoritmo. Utilizamos um entrelaçador de bloco com $N_r = 4$ linhas e $N_c = 24$ colunas. A estimativa da capacidade de canal ilustrada na Figura 5.9 foi obtida como uma média dos limitantes superior e inferior para $L = 9$. Com isso, obtivemos a RSR E_s/N_0 (em dB) a partir da curva $C \,\mathrm{X}\, E_s/N_0(dB)$. Com a devida correção pela taxa R_s chegamos à RSR E_b/N_0 indicada na Figura 5.9.

Fixando-se o valor de Z, podemos perceber pelas curvas da Figura 5.9 que o aumento do número de iterações ocasionou um melhor desempenho do sistema. Dado que $Z = 8, 16$ e 32, o ganho de codificação obtido com o aumento de 10 vezes no valor de N_{it}, considerando $P_b = 10^{-3}$, vale aproximadamente $2,55\ dB$, $1,5\ dB$ e $2,3\ dB$, respectivamente.

A variação do nível de quantização da fase introduzida pelo canal, dado um número fixo de iterações do algoritmo, também alterou o desempenho do sistema. Considerando $N_{it} = 500$,

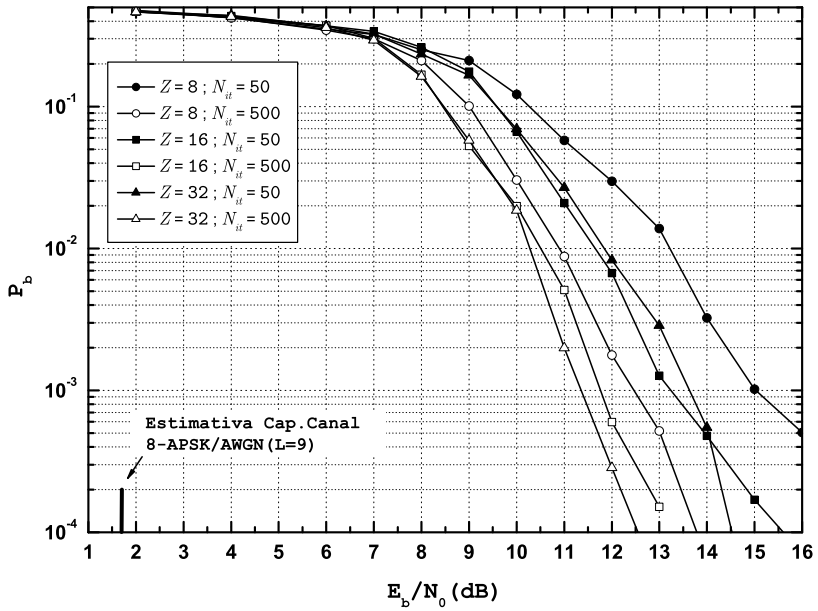


Figura 5.9: Desempenho do sistema considerando o uso do código LDPC(96,48). Níveis de quantização de fase utilizados: $Z = 8, 16, 32$. Número de iterações utilizadas: $N_{it} = 50$ e 500 . $R_s = 1,33$ $bits/simb$.

o ganho obtido com o aumento de Z de 8 para 16 é de $0,7$ dB, e de 16 para 32, é de $0,4$ dB . Apesar de ser um ganho decrescente com o aumento de Z, ele reflete a melhora de desempenho diante de uma estimação mais apurada da fase do canal. Por outro lado, o aumento de Z de 16 para 32, considerando $N_{it} = 50$, provocou uma pequena perda em desempenho para E_b/N_0 abaixo de 14 dB .

Outro ponto importante avaliado foi a influência do entrelaçador de bloco no desempenho do sistema. Como foi definido anteriormente, o entrelaçador Π_e permuta os bits de uma palavra-código que são enviadas ao modulador M-DAPSK. A mudança dos parâmetros N_r e N_c da matriz de entrelaçamento resulta em diferentes entrelaçadores. Na realidade, a mudança do entrelaçador no esquema de codificação corresponde a uma permutação das colunas da matriz **H** do código LDPC. Uma tentativa de se justificar a influência do entrelaçamento em esquemas LDPC-BICM foi apresentada em [54].

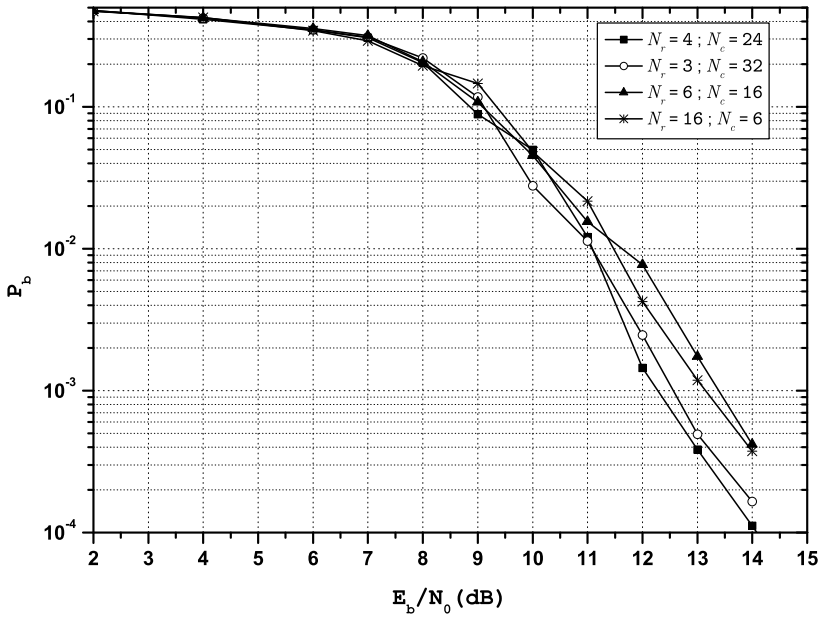


Figura 5.10: Desempenho do sistema para diversas configurações da matriz de entrelaçamento de bloco, considerando a utilização do código LDPC(96,48). Níveis de quantização de fase utilizados: $Z = 16$. Número de iterações utilizadas: $N_{it} = 200$.

A Figura 5.10 ilustra o desempenho do sistema para alguns entrelaçadores. Novamente o ruído de fase foi considerado constante num bloco de $L = 9$ símbolos. Podemos notar que a partir de $E_b/N_0 = 11\ dB$, o desempenho do sistema que utiliza uma matriz de entrelaçamento com $N_r = 4$ linhas e $N_c = 24$ colunas se destacou em relação aos demais. Para uma $P_b = 10^{-3}$, o uso deste entrelaçador forneceu um ganho aproximado de 1 dB quando comparado ao que possui $N_r = 6$ e $N_c = 16$.

Em seguida, consideramos a utilização do código LDPC regular de taxa $r_c = 1/2$ com comprimento $n_c = 1008$. A modulação continuou sendo a 8-APSK$(2, 4)$ com $r = 2,42$, todavia neste caso, assumiu-se que o ruído de fase foi constante num bloco de $L = 29$ símbolos. A Figura 5.11 ilustra o desempenho do sistema para dois níveis de quantização de fase e dois valores de iterações. Para o código LDPC(1008,504), utilizamos um entrelaçador de bloco com $N_r = 12$ linhas e $N_c = 84$ colunas. A estimativa da capacidade de canal e a RSR

E_b/N_0 associada foram obtidas de maneira similar ao caso anterior (código LDPC(96,48)). O desempenho do sistema para o código LDPC(96,48) é mostrado apenas como referência. Para uma probabilidade de erro de bit $P_b = 10^{-3}$, o aumento do nível de quantização Z e do número de iterações N_{it} ocasionou um ganho de codificação em torno de $1,8\ dB$.

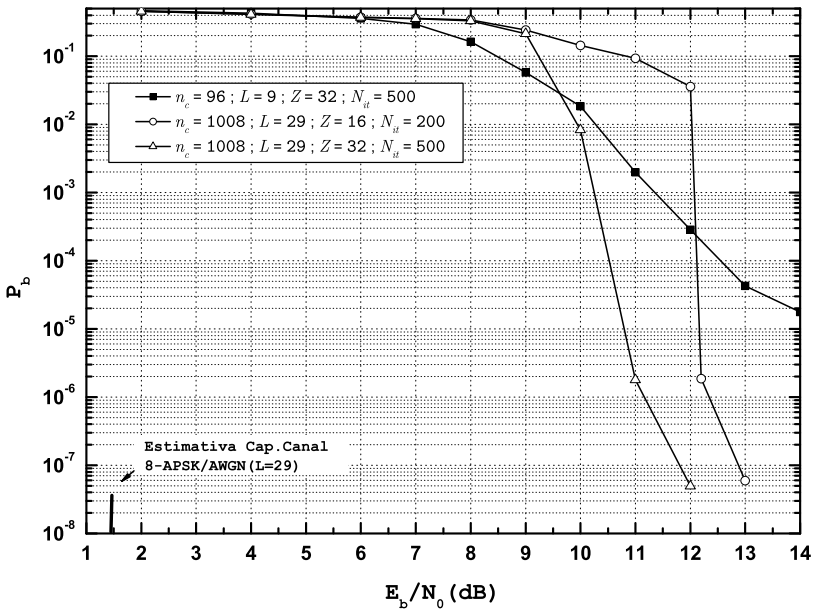


Figura 5.11: Desempenho do sistema considerando o uso do código LDPC(1008,504). Níveis de quantização de fase utilizados: $Z = 16$ e 32. Número de iterações utilizadas: $N_{it} = 200$ e 500. $R_s = 1,45\ bits/simb$.

Para RSRs abaixo de $10\ dB$, o desempenho do sistema utilizando o código LDPC(1008,504) foi pior do que o desempenho considerando o uso do código LDPC(96,48), independente dos valores de Z e N_{it}. Entretanto, podemos perceber que a partir de $E_b/N_0 = 10\ dB$, o valor da P_b decresceu em torno de uma ordem de grandeza para um aumento de $0,3\ dB$ em E_b/N_0. Este decréscimo aumentou para algo em torno de cinco ordens de grandeza quando $E_b/N_0 = 11\ dB$. Isto nos indica que apesar do desempenho do algoritmo ter iniciado de maneira ruim, ele passou a fornecer ganhos significativos em P_b para RSRs altas.

O efeito do entrelaçador de bloco também foi verificado para o código LDPC(1008,504).

A Figura 5.12 compara os desempenhos do sistema para dois entrelaçadores distintos. No primeiro deles, a matriz de entrelaçamento possui $N_r = 12$ linhas e $N_c = 84$ colunas, ao passo que no segundo temos $N_r = 4$ e $N_c = 252$.

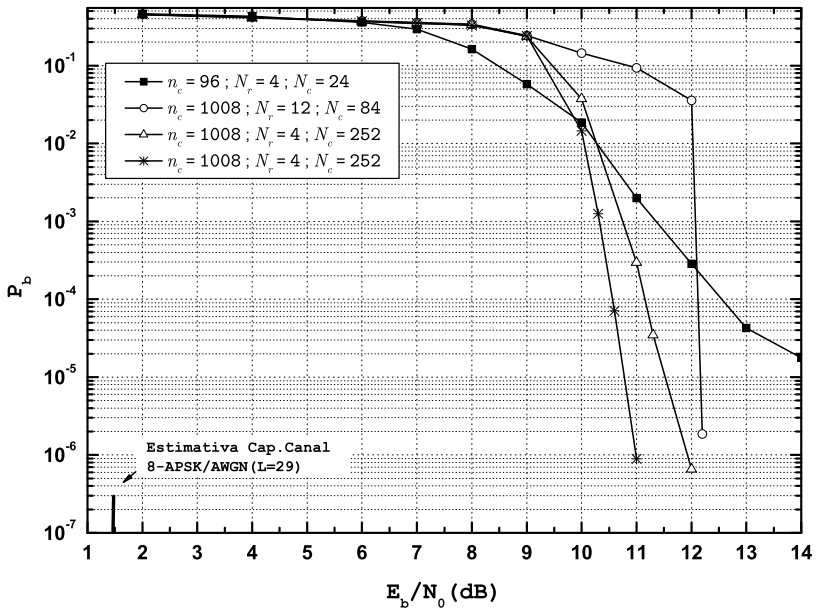


Figura 5.12: Desempenho do sistema para algumas configurações da matriz de entrelaçamento de bloco, considerando a utilização do código LDPC(1008,504). $\circ, \triangle : Z = 16,\ N_{it} = 200$. $\blacksquare, * : Z = 32,\ N_{it} = 500$.

Podemos observar que o entrelaçador cujos parâmetros valem $N_r = 4$ e $N_c = 252$ forneceu um pequeno ganho de codificação. Para uma $P_b = 10^{-3}$, o ganho obtido foi de aproximadamente $1,3\ dB$. Além disso, aumentando o valor de Z e N_{it} para o esquema com o entrelaçador de $N_r = 4$ linhas e $N_c = 252$ colunas, tivemos um ganho adicional de $0,5\ dB$. Isto nos mostra que a permutação das colunas de H gerou pequenos ganhos de codificação tanto para o código LDPC(96,48) quanto para o de comprimento $n_c = 1008$. Para comprimentos maiores, espera-se que este efeito permaneça.

Os resultados expostos nesta Seção mostraram que o desempenho do sistema considerando a utilização de códigos LDPC de comprimentos $n_c = 96$ e $n_c = 1008$ estão bem distantes

da estimativa de capacidade de canal obtida. Para o primeiro código, tais resultados são esperados, uma vez que seu comprimento é curto. Para o segundo código, apesar de um aumento de cerca de 10 vezes no seu comprimento, este também pode ser considerado curto diante dos códigos LDPC normalmente empregados em canais não-coerentes. Franceschini et al. em [55] utilizaram códigos LDPC com comprimento de 12000 bits concatenados serialmente com modulações M-PSK diferenciais ao passo que Colavolpe et al. fizeram uso de códigos com comprimento 64800 em [50]. Sendo assim, diante dos resultados apresentados nesta tese espera-se que, com o aumento do comprimento dos códigos LDPC, o desempenho do sistema proposto se aproxime da estimativa da capacidade de canal. Além disso, consideramos interessante a investigação de como o entrelaçador de bloco altera o desempenho do algoritmo, conforme pudemos constatar.

Vale salientar a dificuldade encontrada na implementação das rotinas computacionais do algoritmo SP para o receptor proposto, uma vez que esta questão praticamente não é abordada na literatura. O algoritmo executado no grafo esteve sujeito à instabilidade numérica em virtude das multiplicações realizadas durante a passagem de mensagens. Para lidar com isso, normalizações e limitações de valores foram necessárias a ponto de permitir a implementação do algoritmo sem *bugs*. Em nosso caso, o sub-grafo 2, responsável pela demodulação diferencial e estimação de canal (ver Figura 5.6), necessitou de limitação nas mensagens $\mu_{\theta_k \to T_i}(\theta_k)$ por conta do *overflow* causado pelas multiplicações. Pelo mesmo motivo, uma normalização foi realizada nas mensagens geradas na parte central do grafo, local da etapa Progressiva-Regressiva do algoritmo.

O aumento do comprimento do código acarretará no aumento da complexidade do algoritmo e, consequentemente, no aumento da latência do receptor. Entretanto, acreditamos que isto pode ser resolvido com mudanças no cronograma de execução. O algoritmo poderia limitar a participação do sub-grafo 2, uma vez que este é responsável pelos problemas numéricos de execução. Desta forma, o canal seria estimado nas iterações iniciais, dispensando novas atualizações e minimizando a complexidade do algoritmo. Niu et al. consideraram esta abordagem em uma proposta de algoritmo iterativo para canais com desvanecimento Rayleigh

em [53].

5.5 Síntese do Capítulo

Este Capítulo apresentou inicialmente um esquema de transmissão baseado em concatenação serial de um codificador LDPC e um codificador diferencial M-APSK. Em seguida, foi apresentada a proposta do receptor iterativo, começando pela definição de conceitos importantes, como função objetivo e distribuições canônicas. Após isso, o grafo que representa o receptor foi ilustrado e as mensagens que compõem o algoritmo foram definidas. Adicionalmente, o cronograma de execução do algoritmo foi exposto. Por fim, o Capítulo relatou os resultados de simulação e as respectivas conclusões.

6 Conclusões

Esta tese apresentou um estudo dividido em duas partes. Na primeira parte deste trabalho, apresentamos um estudo sobre a capacidade de canal AWGN não-coerente de bloco considerando a utilização de um esquema de modulação com um número finito de sinais denominado M-APSK. A justificativa para a escolha de sinais modulados em amplitude e fase foi o desejo de propor sistemas com alta eficiência espectral para canais com ruído de fase, uma vez que não há resultados para sistemas com taxa acima de 1 bit por uso do canal na literatura. As constelações M-APSK aqui consideradas são compostas de anéis PSK concêntricos com as fases alinhadas, dada a sua facilidade de se implementar codificação diferencial.

A distribuição de entrada que atinge a capacidade do canal M-APSK/AWGN não-coerente de bloco se caracteriza por possuir distribuição de fases u.i.i.d. e independente da distribuição conjunta de amplitudes. Uma vez que é difícil o cálculo da distribuição conjunta de amplitudes para N^L dimensões, sugerimos uma proposta de cálculo de limitantes de capacidade. Nossa proposta partiu do cálculo de limitantes de capacidade para canais AWGN não-coerentes com modulações M-PSK feito em [11], e de maneira semelhante, indicou que os limitantes se

aproximam da capacidade do canal coerente para intervalos de coerência grandes [21].

Estendemos o algoritmo de otimização proposto em [23] com o intuito de obtermos os parâmetros da constelação M-APSK (A e r) e a distribuição de amplitudes de entrada que atingem a capacidade do canal coerente. Com isso, estabelecemos uma referência para analisar os limitantes propostos.

Adicionalmente, comparamos a capacidade coerente obtida com o algoritmo de otimização mencionado e a capacidade efetiva de canal. Por serem muito próximas, concluímos que os limitantes para a capacidade efetiva são uma boa referência para a capacidade do canal não-coerente. Além disso, o uso da distribuição uniforme de entrada e de uma razão de raio r fixa é uma simplificação aceitável no cálculo dos limitantes de capacidade.

Finalizando a primeira etapa, analisamos a proposta de cálculo dos limitantes considerando a presença e a ausência de codificação diferencial na transmissão. Verificamos que a existência de um símbolo de referência no bloco que caracteriza o canal levou os limitantes a se aproximarem mais rapidamente da capacidade coerente. Sendo assim, os resultados de informação mútua indicaram o uso de codificação diferencial para L crescente.

Para verificar esta afirmação teórica, propusemos um esquema de transmissão utilizando codificação diferencial para $L > 10$. Devido às dificuldades de simulação do esquema para códigos LDPC com comprimentos longos, consideramos o uso de códigos curtos. Apesar de estarem distantes da capacidade para os comprimentos de código considerados, os resultados mostraram que o desempenho do algoritmo pode evoluir com o uso de códigos de comprimentos maiores. O nível de quantização da fase, o número de iterações e o entrelaçamento empregado são parâmetros a se investigar na busca por melhores desempenhos. Além do mais, é importante destacar o caráter inovador da proposta, já que não há na literatura uma descrição de receptor iterativo baseada em grafos utilizando codificação diferencial e modulações não-binárias.

Sugerimos como perspectiva de trabalhos futuros, a investigação do desempenho do algoritmo para códigos de comprimentos longos. Diante das dificuldades de simulação, uma idéia inicial seria a simplificação do cronograma de execução do algoritmo. Esta simplificação

poderia iniciar com uma verificação da estimação da fase introduzida pelo canal. Uma estimação mais rápida da fase poderia limitar o cronograma de execução do algoritmo ao sub-grafo do código LDPC, diminuindo de forma considerável a complexidade do algoritmo. Finalizando, recomendamos também a busca por uma estratégia de entrelaçamento que permita que o desempenho do sistema proposto se aproxime dos limites teóricos de capacidade.

Lista de Acrônimos

Nesta seção são listadas as abreviações que ocorrem com freqüência no decorrer do trabalho. Na tabela, a primeira coluna refere-se ao acrônimo. Na segunda coluna, temos a sua definição em inglês ou em português. Por fim, a terceira coluna contém a página do texto na qual o acrônimo aparece pela primeira vez.

Referências Bibliográficas

[1] C.E.Shannon, "A Mathematical Theory of Communication," *Bell Syst. Tech. J.* , vol. 27, pp. 379-423, 623-657, Out 1948.

[2] S.Haykin, **Communication Systems**. 4 ed., New York: J.Wiley, 2001.

[3] A.M.Michelson & A.H.Levesque, **Error-control Techniques for Digital Communication**. New York: J.Wiley, 1985.

[4] T.Cover, **Elements of Information Theory**. New York: J.Wiley, 1991.

[5] B.Sklar, **Digital Communications : Fundamentals and Applications**. 2 ed., Prentice-Hall Inc., 2001.

[6] R.W.Hamming, "Error Detecting and Error Correcting Codes," *Bell Syst. Tech. J.* , vol. 29, pp. 147-160, Abr 1950.

[7] L.R.Bahl, J.Cocke, F.Jelinek & J.Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate," *IEEE Trans. on Information Theory* , vol. 20, n. 2, pp. 284-287, Mar 1974.

[8] A.R.Calderbank, G.D.Forney Jr. & A.Vardy, "Minimal Tail-Biting Trellises: The Golay Code and more," *IEEE Trans. on Information Theory* , vol. 45, n. 5, pp. 1435-1455, Jul 1999.

[9] R.Kötter & A.Vardy, "Construction of Minimal Tail-Biting Trellises," In Proceedings of *IEEE Information Theory Workshop* , pp. 72-74, Killarney, Irlanda, Jun 1998.

[10] A.J.Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," *IEEE Trans. on Information Theory* , vol. 13, n. 2, pp. 260-269, Abr 1967.

[11] M.Peleg & S.Shamai(Shitz), "On the Capacity of the Blockwise Incoherent MPSK Channel," *IEEE Trans. on Communications*, vol. 46, n. 5, pp. 603-609, Mai 1998.

[12] G.Colavolpe & R.Raheli, "The Capacity of the Noncoherent Channel," *European Trans. on Telecommunications*, vol. 12, n. 4, pp. 289-296, Jul/Ago 2001.

[13] R.Nuriyev & A.Anastasopoulos, "Capacity and Coding for the Block-Independent Noncoherent AWGN Channel," *IEEE Trans. on Information Theory*, vol. 51, n. 3, pp. 866-883, Mar 2005.

[14] N.Jacobsen & U.Madhow, "Code and Constellation Optimization for Efficient Noncoherent Communication," In Proceedings of *Conference Record of the Thirty-Eighth Asilomar Conf. on Signals, Systems and Computers*, vol. 1, pp. 198-202, Pacific Grove, Califórnia, EUA, Nov 2004.

[15] G.Ungerboeck, "Channel Coding with Multilevel/Phase Signals," *IEEE Trans. on Information Theory*, vol. IT-28, n.1, pp. 55-67, Jan 1982.

[16] D.Divsalar & M.Simon, "Multiple-Symbol Differential Detection of MPSK," *IEEE Trans. on Communications*, vol. 38, n. 3, pp. 300-308, Mar 1990.

[17] R.Palanki, Iterative Decoding for Wireless Networks, Tese de Doutorado, California Institute of Technology, Pasadena, Califórnia, EUA, Mai 2004.

[18] D.Tse & P.Viswanath, **Fundamentals of Wireless Communications**. Cambridge University Press, Hardcover, 2005.

[19] R.Blahut, **Principles and Practice of Information Theory**. New York: Addison-Wesley, 1987.

[20] A.D.Wyner, "Bounds on Communication with Polyphase Coding," *Bell Syst. Tech. J.*, vol. XLV, pp. 523-559, Abr 1966.

[21] D.C.Cunha & J.Portugheis, "Bounds on the Capacity of the Blockwise Noncoherent APSK-AWGN Channels," In Proceedings of *IEEE International Symposium on Information Theory* (ISIT'05), pp. 627-630, Adelaide, Austrália, Set 2005.

[22] D.C.Cunha & J.Portugheis, "Sobre a Capacidade do Canal M-APSK/AWGN Não-Coerente de Bloco," Em Anais do *XXII Simpósio Brasileiro de Telecomunicações* (SBrT'05), pp. 520-524, Campinas-SP, Brasil, Set 2005.

[23] N.Varnica, X.Ma & A.Kavcic, "Capacity of Power Constrained Memoryless AWGN Channels with Fixed Input Constellations," In Proceedings of *Global Telecommunications Conference* (GLOBECOM'02), vol.2, pp. 1339-1343, Taipei, Taiwan, Nov 2002.

[24] L.Lampe & R.Fischer, "Comparison and Optimization of Differentially Encoded Transmission on Fading Channels," In Proceedings of 3^{rd} *International Symposium on Power-Line Communications and its Applications* (ISPLC'99), pp. 107-113, Lancaster, Reino Unido, Mar 1999.

[25] F.R.Kschischang, B.J.Frey & H.-A.Loeliger, "Factor Graphs and the Sum-Product Algorithm," *IEEE Trans. on Information Theory* , vol. 47, n. 2, pp. 498-519, Fev 2001.

[26] R.M.Tanner, "A recursive approach to low complexity codes," *IEEE Trans. on Information Theory* , vol. 27, pp. 533-547, Set 1981.

[27] R.G.Gallager, "Low-Density Parity Check Codes," *IRE Trans. on Information Theory* , pp. 21-28, Jan 1962.

[28] N.Wiberg, H.-A.Loeliger & R. Kötter, "Codes and Iterative Decoding on General Graphs," *European Trans. on Telecommunications* , vol. 6, pp. 513-525, Set/Out 1995.

[29] J.C.Willems, "Models for Dynamics," In *Dynamics Reported, Volume 2* , U.Kirchgraber and H.O.Walther, pp. 171-269, New York: J.Wiley, 1989.

[30] D.J.C.MacKay, **Information Theory, Inference, and Learning Algorithms**. Cambridge University Press, 2003.

[31] W.E.Ryan, **CRC Handbook for Coding and Signal Processing for Recording Systems**. CRC Press, 2004, cap. An Introduction to LDPC Codes.

[32] D.J.C.MacKay, "Good Error-Correcting Codes Based on Very Sparse Matrices," *IEEE Trans. on Information Theory* , vol. 45, n. 2, pp. 399-431, Mar 1999.

[33] N.Alon & M.Luby, "A Linear Time Erasure-Resilient Code with Nearly Optimal Recovery," *IEEE Trans. on Information Theory* , vol. 42, n. 6, pp. 1732-1736, Nov 1996.

[34] R.McEliece, D.J.C.Mackay & J.Cheng, "Turbo Decoding as an Instance of Pearl's 'Belief Propagation' Algorithm," *IEEE Journal on Selected Areas in Communications,* vol. 16, n. 2, pp. 140-152, Fev 1998.

[35] D.J.C.MacKay & R.M.Neal, "Near Shannon Limit Performance of Low-Density Parity-Check Codes," *Electronics Letters,* vol. 33, pp. 457-458, Mar 1997.

[36] T.Tian, C.Jones, J.D.Villasenor & R.D.Wesel, "Construction of Irregular LDPC Codes with Low Error Floors," In Proceedings of *IEEE International Conference on Communications*(ICC'03), vol. 5, pp. 3125-3129, Anchorage, Alaska, EUA, Mai 2003.

[37] T.J.Richardson, A.Shokrollahi & R.Urbanke, "Design of Capacity-Approaching Irregular Low-Density Parity Check Codes," *IEEE Trans. on Information Theory,* vol. 47, n. 2, pp. 619-637, Fev 2001.

[38] X.-Y.Hu, E.Eleftheriou, D.-M.Arnold & A.Dholakia, "Efficient Implementations of the Sum-Product Algorithm for Decoding LDPC Codes," In Proceedings of *IEEE Global Conference on Telecommunications*(GLOBECOM'01), vol. 2, pp. 1036-1036E, San Antonio, Texas, EUA, Nov 2001.

[39] N.Wiberg, Codes and Decoding on General Graphs, Tese de Doutorado, U. Linkoping, Suécia, 1996.

[40] J.Hagenauer, E.Offer & L.Papke, "Iterative Decoding of Binary Block and Convolutional Codes," *IEEE Trans. on Information Theory*, vol. 42, n. 2, pp. 429-445, Mar 1996.

[41] P.Robertson, E.Villebrun & P.Hoeher, "A Comparison of Optimal and Sub-Optimal MAP Decoding Algorithms Operation in the Log Domain," In Proceedings of *IEEE Conference on Communications*(ICC'95), vol. 2, pp. 1009-1013, Seattle, EUA, Jun 1995.

[42] D.J.C.MacKay, Encyclopedia of Sparse Graph Codes, Home-page of the Inference Group, Cavendish Lab., Cambridge Univ. Disponível em: http://www.inference.cam.ac.uk/mackay/codes/data.html. Acesso: 01 Dez 2005.

[43] A.P.Worthen & W.E.Stark, "Unified Design of Iterative Receivers using Factor Graphs," *IEEE Trans. on Information Theory*, vol. 47, n.2, pp. 843-849, Fev 2001.

[44] R.Nuriyev & A.Anastasopoulos, "Pilot-Symbol-Assisted Coded Transmission Over the Block-Noncoherent AWGN Channel," *IEEE Trans. on Communications*, vol. 51, n. 6, pp. 953-963, Jun 2003.

[45] M.Peleg, S.Shamai(Shitz) & S.Galán, "Iterative Decoding for Coded Noncoherent MPSK Communications Over Phase-Noisy AWGN Channel," *IEE Proceedings on Communications*, vol. 147, pp. 87-95, Abr 2000.

[46] C.Heegard & S.B.Wicker, **Turbo Coding**. Kluwer Academic Publishers, 1999.

[47] G.Caire, G.Taricco & E.Biglieri, "Bit-Interleaved Coded Modulation," *IEEE Trans. on Information Theory*, vol. 44, n. 3, pp. 927-946, Mai 1998.

[48] R.Chen, R.Koetter, U.Madhow & D.Agrawal, "Joint Noncoherent Demodulation and Decoding for the Block Fading Channel: A Practical Framework for Approaching Shannon Capacity," *IEEE Trans. on Communications*, vol. 51, n. 10, pp. 1676-1689, Out 2003.

[49] J.Dauwels & H.-A.Loeliger, "Phase Estimation by Message Passing," In Proceedings of *IEEE International Conference on Communications 2004* (ICC'04), pp. 523-527, Paris, França, Jun 2004.

[50] G.Colavolpe, A.Barbieri & G.Caire, "Algorithms for Iterative Decoding in the Presence of Strong Noise," *IEEE Journal on Selected Areas in Communications*, vol.23, n.9, pp. 1748-1757, Set 2005.

[51] A.P.Worthen, Codes and Iterative Receivers for Wireless Communication Systems, Tese de Doutorado, University of Michigan, EUA, 2001.

[52] G.Ferrari, G.Colavolpe & R.Raheli, **Detection Algorithms for Wireless Communications with Applications to Wired and Storage Systems**. New York: J.Wiley, Hardcover, 2004.

[53] H.Niu, M.Shen, J.A.Ritcey & H.Liu, "A Factor Graph Approach to Iterative Channel Estimation and LDPC Decoding over Fading Channels," *IEEE Trans. on Wireless Communications*, vol.4, n.4, pp. 1345-1350, Jul 2005.

[54] R.D.Maddock & A.H.Banihashemi, "Reliability-Based Coded Modulation With Low-Density Parity-Check Codes," *IEEE Trans. on Communications*, vol.54, n.3, pp. 403-406, Mar 2006.

[55] M.Franceschini, G.Ferrari, R.Raheli & A.Curtoni, "Serial Concatenation of LDPC Codes and Differential Modulations," *IEEE Journal on Selected Areas in Communications*, vol.23, n.9, pp. 1758-1768, Set 2005.