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

CORREÇÃO DE APAGAMENTOS EM RAJADAS UTILIZANDO CÓDIGOS LDPC GERADOS PELA COMPOSIÇÃO DE MATRIZES BASES E PELOS MOVIMENTOS DE MATRIZES CIRCULANTES

CASSIO ANDRÉ SOUSA DA SILVA

TD 17/2016

UFPA/ITEC/PPGEE

Campus Universitário do Guamá

Belém-Pará-Brasil

2016

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

CASSIO ANDRÉ SOUSA DA SILVA

CORREÇÃO DE APAGAMENTOS EM RAJADAS UTILIZANDO CÓDIGOS LDPC GERADOS PELA COMPOSIÇÃO DE MATRIZES BASES E PELOS MOVIMENTOS DE MATRIZES CIRCULANTES

TD 17/2016

UFPA/ITEC/PPGEE

Campus Universitário do Guamá

Belém-Pará-Brasil

2016

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

CASSIO ANDRÉ SOUSA DA SILVA

CORREÇÃO DE APAGAMENTOS EM RAJADAS UTILIZANDO CÓDIGOS LDPC GERADOS PELA COMPOSIÇÃO DE MATRIZES BASES E PELOS MOVIMENTOS DE MATRIZES CIRCULANTES

.

Tese de Doutorado submetida à Banca Examinadora do Programa de Pós-graduação em Engenharia Elétrica da UFPA para obtenção do Grau de Doutor em Engenharia Elétrica na área de Telecomunicação

UFPA/ITEC/PPGEE

Campus Universitário do Guamá

Belém-Pará-Brasil

2016

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

Silva, Cassio André Sousa da, 1971-

Correção de apagamentos em rajadas utilizando códigos LDPC gerados pela composição de matrizes bases e pelos movimentos de matrizes circulantes / Cássio André Sousa da Silva.- 2016.

Orientador : Evaldo Gonçalves Pelaes.

Tese (Doutorado) - Universidade Federal do Pará, Instituto de Tecnologia, Programa de Pós-Graduação em Engenharia Elétrica, Belém, 2016.

1. Códigos corretores de erro. 2. Matrizes. 3. Tecnologia da informação – modelos matemáticos, 3. Métodos de simulação. I. Título.

CDD 23. ed. 005.717

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

CORREÇÃO DE APAGAMENTOS EM RAJADAS UTILIZANDO CÓDIGOS LDPC GERADOS PELA COMPOSIÇÃO DE MATRIZES BASES E PELOS MOVIMENTOS DE MATRIZES CIRCULANTES

AUTOR: CASSIO ANDRÉ SOUSA DA SILVA

TESE DE DOUTORADO SUBMETIDA À AVALIAÇÃO DA BANCA EXAMINADORA APROVADA PELO COLEGIADO DO PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA DA UNIVERSIDADE FEDERAL DO PARÁ E JULGADDA ADEQUADA PARA OBTENÇÃO DO GRAU DE DOUTOR EM ENGENHARIA ELÉTRICA NA ÁREA DE TELECOMUNIÇÕES

APROVADA EM: 21/10/2016

BANCA EXAMINADORA

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

Prof. Dr. . Aldebaro Barreto da Rocha Klautau (UFPA)

Prof. Dr. Johelden Campos Bezerra (IFPA)

Prof. Dr. Francisco Marcos de Assis (UFCG)

Profa. Dra. Valquíria Gusmão Macedo (UFPA)

VISTO:

Prof. Dr. Evaldo Gonçalves Pelaes
(COORDENADOR DO PPGEE/ITEC/UFPA)

Dedicatória

José Luiz Moura da Silva, meu pai *im memoriun*

Agradecimentos

Ao meu orientador pelo direcionamento e motivação.

À minha família pela compreensão e suporte essenciais.

À UFPA pela experiência.

Sumário

Lista de abreviaturas

AWGN	*Additive White Gaussian Noise* - Ruído Branco Aditivo Gaussiano
BEC	*Binary Erasure Channels*
BuEC	*Burst Erasure Channels*
BER	*Bit Error Rate* - Taxa de Erro de Bit
BP	*Belief-propagation*
BPSK	*Binary Phase Shift Keying* - Chaveamento Binário por Deslocamento de Fase
CH-LDPC	*Cicle-Hold LDPC Codes*
LDPC	*Low-Density Parity Check*
LI	Vetores linearmente independentes
LD	Vetores linearmente dependentes
LLR	*Likelihood Ratio*- Razão de Log-Verossimilhança
MDS	*Maximum Distance Serparable*
QC-LDPC	*Quasi-Cyclic codes Low-Density Parity Check*
SPA	Algoritmo Soma e Produto (*Sum-Product Algorithm*)
SNR	Relação Sinal Ruido

Lista de símbolos

$G \times H$	Produto direto de G por H
$H \leq G$	H subgrupo de G
$M_n(\mathbb{R})$	Conjunto de todas as matrizes $n \times n$ sobre $\mathbb{R}$
D_4	Grupo diedral D_4
D_3	Grupo diedral D3
$\mathbb{Z}_m$	Conjunto das classes resto ou residuais módulo m
aH	Classe lateral à esquerda
$\frac{G}{H}$	Grupo Quociente de G por H
$\lvert G \rvert$	Cardinalidade do grupo G
$\det(A)$	Determinante da matriz A
$H \triangleright G$	H subgrupo normal de G
$G \simeq H$	G isomorfo a H
$W \subseteq V$	W subespaço de V
$\mathrm{Aut}(G)$	Automorfismo em G
$\mathcal{L}(V,V)$	Conjunto de todos os operadores lineares de V.
$N(x) = \|x\|^2$	Norma quadrática ou peso Euclidiano
Isom($\mathbb{R}^n$)	Conjunto de todas as isometrias de $\mathbb{R}^n$.
$S^{\perp}$	Complemento Ortogonal de S
$U \oplus W$	Soma direta de U e W
$[T]^{\beta}$	Matriz da transformação T em relação a base β
$d_H(x, y)$	Distância de Hamming entre os vetores x e y
$w_H(x)$	Peso de Hamming
C(n, k, d)	Código linear de comprimento n, dimensão k e distância d
(d_j)	Nó de variável
(h_j)	Nó de cheque

Δ	Bit apagado
Q_{ij}^{x}	Estimativa de probabilidade de Nó de variável para Nó de cheque
R_{ij}^{x}	Estimativa de probabilidade de Nó de cheque para Nó de variável
f_j^{x}	Coeficiente de probabilidade a priori
$\widehat{d_j}$	Estimativa do vetor recebido
$I_{\max}$	Máxima iteração do algoritmo
$E_{j,i}$	Soma binária dos valores conhecidos de uma palavra código
$L_{\max}$	Máximo comprimento do apagamento recuperado
$\mathrm{E_f}$	Eficiência do Código
$\mathrm{I}_{v}^{(m)}$	Matriz identidade circulante de movimento *m* e dimensão *v*
$S_{\min}$	Número mínimo de bits apagados que podem causar uma falha na decodificação.

Lista de Figuras

Lista de Tabelas

Resumo

Nesta tese são propostos procedimentos para a construção de matrizes de verificação de paridade para codificação e decodificação de códigos LDPC (*low-density parity-check*) na recuperação de bits apagados no canal com apagamentos em rajada. As matrizes de verificação de paridade são produzidas por concatenação das matrizes bases binárias justapostas por matrizes circulantes sendo de fácil implementação e de menor aleatoriedade. As matrizes bases são desenvolvidas a partir de fundamentos da álgebra e da geometria. Para demonstrar o potencial da técnica foi elaborado um conjunto de simulações que usa codificação de baixa complexidade, bem como o uso dos algoritmos soma e produto para recuperar os apagamentos. Foram gerados vários códigos LDPC, a partir das matrizes, e os resultados obtidos foram comparados com outros códigos LDPC obtidos da literatura. São ainda apresentados os resultados da simulação da recuperação de apagamentos resultantes da transmissão de uma imagem através de um canal ruidoso.

Palavras chaves- Códigos LDPC, Códigos corretores de erros em rajada, Canais com apagamentos em rajadas, códigos corretores de erro, Matriz por superposição, Matriz circulante.

Abstract

This thesis proposed procedures for the construction of parity check matrices for encoding and decoding of LDPC codes in the recovery of deleted bits in Burst Erasure Channel. The parity check matrices are produced by concatenation of binary bases matrices juxtaposed by circulating matrices are easy to implement and lower randomness. The base arrays are developed from the foundations of algebra and geometry. To demonstrate the potential of the technique, we developed a number of simulations using low complexity encoding as well as the sum-product algorithm. Several LDPC codes (matrices) were generated and the results were compared with other approaches. We also present the outcomes of erasure recovery simulations that result from the transmission of an image through a noisy channel.

Keywords- - Low-density parity-check codes, burst erasure correcting codes, burst erasure channels, erasure-correcting codes, matrix by superposition

Capítulo 1

Introdução

Atualmente as Tecnologias de informação e comunicação (TIC's) são atuantes em todas as manifestações sociais que se utilizam de uma mídia como forma de comunicação. Desde grandes negócios empresarias [1-3], até mesmo em análise de ecossistemas hidrográficos [4], as TIC's se apresentam como uma grande maneira de comunicação e de apresentação de conclusões científicas. O grande uso destas mídias eletrônicas vem aumentando em grande parte pela popularização da Televisão de Alta Definição (HDTV) [5] e pela transmissão de dados via internet. Nos últimos anos uma nova modalidade vem ganhando espaço em todos os sentidos nessas transmissões e vem contribuído para uma verdadeira revolução na forma de fazer comunicação: As transmissões em tempo real. Vários centros de pesquisa vêm estudando novos processos que melhoram o sistema de codificação e decodificação, de modo que se tenham informações mais confiáveis, livre de erros ou perda de informações.

O objetivo de um sistema de comunicação é transmitir uma mensagem através de um canal de comunicação, de modo que o receptor seja capaz recuperar a mensagem com um dado critério de fidelidade, diante das diversas variáveis físicas impostas pelo canal [1]. Um sistema de comunicação real depara-se com diferentes problemas, em especial, certas perturbações introduzidas pelo meio de comunicação, gerando apagamento durante a transmissão através do canal. Em grande parte os problemas de perda de informações nas transmissões são ocasionados principalmente por ruídos. O ruído é sem duvida um dos fatores básicos que limitam a taxa de comunicação. A internet é um exemplo simples e clássico onde a comunicação é feita através da troca de informação sob a forma de pacotes. Quando se envia a informação através de um meio modelado como um canal com apagamento, a informação recebida pode ser, reconhecidamente, diferente daquela que foi transmitida. A estratégia utilizada para garantir uma transmissão fidedigna de informação através de canais não ideais é codificar a

informação a ser transmitida, inserindo redundância, para reduzir os efeitos do ruído sobre a mensagem original.

A Figura 1.1 mostra o modelo clássico de um sistema de comunicações. Estamos interessados em sistemas em que a informação produzida pela fonte seja binária e agrupada em blocos, formando um bloco de bits de mensagem de tamanho *k* a ser enviada. No transmissor, a mensagem *s* é enviada a um codificador que adiciona redundâncias e transforma em uma mensagem codificada x de comprimento *n* que é transmitida pelo canal.

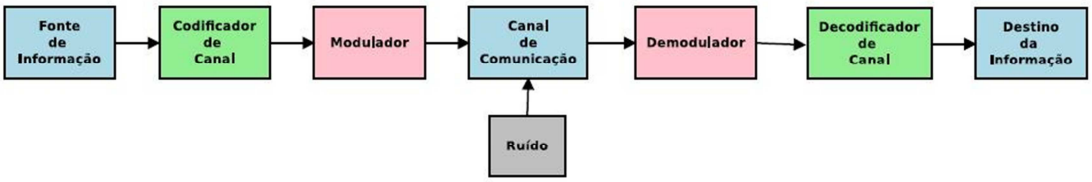


Figura 1.1: Sistema de comunicação

Se y é a mensagem recebida na saída do canal, esta pode não ser uma cópia fiel da mensagem codificada x. O decodificador então usa a redundância introduzida pelo codificador para recuperar a mensagem sem erro. No entanto, pode ser que neste processo alguns bits de informação sejam perdidos (ou pacotes apagados) e desta forma o sistema adiciona um símbolo a mais à saída do canal. Canais com esta características são chamados *canais com apagamentos*.

Introduzidos por Elias [6], o canal binário com apagamento, o BEC (*Binary Erasure Channel*), é um canal em que a mensagem é recebida sem erro ou é convertida em um apagamento. Esse canal é discreto sem memória tendo dois símbolos de entrada {0,1} e três símbolos de saída {0, 1, Δ}, em que o símbolo Δ representa um apagamento [7]. A probabilidade dos *bits recebidos* transmitidos corretamente por um canal BEC é calculada por $P_{Y|X}(0|0) = P_{Y|X}(1|1) = 1-\varepsilon$ e a probabilidade dos bits recebidos com símbolos apagados é dada por $P_{Y|X}(\Delta|0) = P_{Y|X}(\Delta|1) = \varepsilon$ onde ε é a probabilidade de apagamento do BEC e seu valor está compreendido entre 0 e 1, isto é $0 \leq \varepsilon \leq 1$. Se, por exemplo, o pacote de 12 símbolos binários “001 101 100 111” é enviado através do BEC é possível que o pacote recebido seja “001 Δ01 10 Δ 111”. Neste caso dois símbolos binários foram apagados ao atravessar o canal.

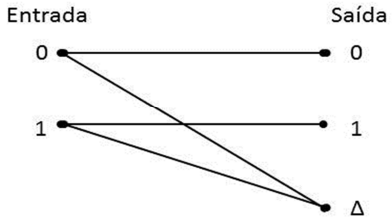


Figura 1.1: Representação do Canal BEC

Para os canais sem memória, o BEC tem proporcionado um quadro útil para compreender o desempenho de códigos LDPC (*low-density parity-check*) [8], e muitas das observações feitas utilizando o BEC pode ser utilmente aplicada a mais canais generalistas sem memória. Neste trabalho emprega-se semelhante idéia para canais com apagamento binários em rajadas como ponto de partida natural para considerar códigos LDPC para canais com memória. Um canal que o receptor seja capaz de distinguir uma sequência de bits apagados será considerado um canal com apagamentos em rajada (*Burst Erasure Channel - BuEC*), cuja definição foi apresentada em [9]. As transmissões via internet são bons exemplos de perdas de dados em rajadas ou em pacotes, onde cada pacote pode conter toda ou parte da mensagem original [11].

Um canal ruidoso com um bom código detector de erro pode ser comparado a um canal com apagamento. Caso um pacote de informação recebido contenha erros não detectáveis, pelo código detector de erro em uso, o pacote é descartado e interpretado como apagado.

Os códigos de bloco são tradicionalmente utilizados como códigos detectores de erros para detectar símbolos não reconhecíveis produzidos por canais com apagamento. Os códigos usados em correções de apagamentos têm despertado grande interesse em diversas áreas, no entanto, é na codificação de pacotes em redes as mais recentes pesquisas [12-13]. Recentemente surgiram os códigos LDPC que têm sido bastante estudados para correção e detecção de erros bem como para detectar e corrigir símbolos não reconhecíveis.

Código LDPC é um código de bloco definido por uma matriz de verificação de paridade esparsa H, isto é, são gerados através de uma matriz com muitos zeros e poucos números uns. Foram descobertos por Galager [14] em 1962 mas ficaram esquecidos por muito tempo, isto por que o esforço computacional requerido era demasiado “avançado” para a tecnologia da época. Gallager também introduziu o seu algoritmo iterativo de decodificação, conhecido hoje como algoritmo Soma e Produto

(SPA) ou *belief propagation*. Com o aparecimento dos Códigos Turbo [15] em 1993 e a redescoberta por MacKay em 1999 [16], a codificação LDPC readquiriu impulso. A forma mais simples de projetar um código LDPC consiste na construção da matriz de verificação de paridade que cumpra um conjunto de requisitos para definir um código regular (caracterizado por um dado peso para cada coluna e linha) ou irregular (podendo ser especificado o grau de cada nó de variável e nó de paridade). A construção da matriz **H** é feita de forma quase aleatória, mas seguindo algumas regras que maximizam a probabilidade do código obtido possuir um bom desempenho. No entanto, essas regras não nos dão qualquer garantia de que tal aconteça. Poderão inclusive suceder situações em que a matriz **H** obtida não será de característica máxima, ou seja, algumas linhas ou colunas não serão linearmente independentes.

Muitos autores vêm estudando formas de compor a matriz de verificação de paridade de maneira que a aleatoriedade seja a menor possível.

1.1 Motivação

A decodificação LDPC está diretamente ligada à construção da matriz de verificação de paridade. O processo do desenvolvimento desta matriz passa por um conjunto de regras e técnicas matemáticas bem diversas.

No estudo de códigos LDPC o desenvolvimento da matriz de verificação de paridade é um dos temas mais interessantes para aqueles que gostam de aplicações práticas em matemática, pois na construção da matriz de verificação de paridade podem-se extrair relacionamentos interessantes entre vários ramos da matemática, como álgebra e geometria, por exemplo. A construção da matriz esparsa é um tópico de grande interesse e que desafia engenheiros e matemáticos por todo o mundo.

O primeiro método utilizado para obter matrizes de códigos LDPC regulares foi originalmente proposto por Gallager [14]. Para construir a matriz de verificação de paridade é preciso primeiro construir uma submatriz de dimensões k x (k **.** dc), na qual as colunas apresentam peso 1 e as linhas apresentam peso dc. A partir desta submatriz registram-se varias operações que geram outras submatrizes que juntas constituem a matriz de verificação de paridade **H**.

Em [17] Makay e Neal apresentam um conjunto de estratégias para gerar códigos LDPC. Estas estratégias são apresentadas de forma numerada, sendo convicção dos autores que as de ordem superior maximizam a probabilidade do código obtido possuir

um melhor desempenho. No entanto, eles próprios reconhecem não possuir qualquer prova deste fato.

Em [18] são utilizados matrizes circulantes para determinar a matriz geradora de códigos QC-LDPC e apresentados vários tipos de circuitos usando os métodos propostos.

Em [19] é apresentado um método algébrico para a construção de códigos LDPC e construído uma classe de códigos LDPC chamados códigos CH-LDPC. É usado matrizes de permutação circulantes para mostra que o número de ciclos de um dado comprimento no gráfico Tanner de códigos longos podem ser igual ou menor do que os códigos de comprimentos curtos.

Em [20] são desenvolvidos vários procedimentos para construção de matrizes de verificação de paridade baseado nas classes resíduas $\mathbb{Z}_m$. São usados movimentos cíclicos da matriz identidade circulantes para a construção do código QC-LDPC. Este procedimento também é extensivo aos códigos Convolucionais.

Em [21] é apresentado uma classe algébrica de QC-LDPC códigos que têm menor complexidade de codificação. A matriz de verificação de paridade destes códigos é desenvolvida a partir da permutação de matrizes circulantes.

Em [22] a matriz cheque paridade é desenvolvia por uma composição de uma família de matrizes circulantes. Esta decomposição reduz a complexidade da codificação e melhora o desempenho do código.

Em [23] são apresentados dois métodos algébricos baseados em Campos de Galois para a construção de códigos LDPC QC-regulares para três tipos diferentes de canais: O AWGN, o BEC e o BEC (canal com apagamentos em rajada). É utilizada uma geometria finita para a composição da matriz de verificação de paridade na correção de apagamentos em rajadas.

Em [24] é proposto um método para correção de apagamentos em rajadas baseadas na permutação das colunas de uma dada matriz de verificação de paridade original. Este método muda a capacidade de correção dos apagamentos do código original em canais aleatórios.

Em [25] é apresentada uma ferramenta simples e eficaz para o projeto de códigos LDPC usando algorítmo interativo para a correção de apagamentos em rajadas. O

método consiste em desenvolver uma matriz de verificação de paridade otimizada a partir de uma dada matriz que cheque a paridade de um código LDPC. A nova matriz é caracterizada por ter colunas permutadas da matriz original. O código desenvolvido tem bom desempenho na correção de apagamentos em rajadas.

Em [26] Sarah desenvolve, a partir de matrizes bases, um bom código corretor de apagamentos em rajadas em uma palavra código. É utilizada superposição de matrizes circulantes nas matrizes bases para se chegar à matriz de verificação de paridade.

A qualidade de um código LDPC e seu algoritmo de decodificação dependem crucialmente da matriz de verificação de paridade que o define. A construção da matriz esparsa é tópico relevante na problemática dos códigos LDPC; está é a motivação central do trabalho.

Especificamente a motivação central é desenvolver um processo matemático (menos dependente de aleatoriedade) para construção das matrizes de verificação de paridade.

1.2 Objetivos e contribuições

Os principais objetivos e contribuições desta tese são:

- Desenvolvimento de matrizes bases a partir da Álgebra e Geometria
- Geração de matriz de verificação de paridade para códigos LDPC construídos a partir das matrizes bases e circulante de superposição
- Implementação de um método eficiente para corrigir apagamentos em rajadas utilizando os códigos LDPC proposto.

1.3 Organização da tese

Os demais capítulos estão organizados da seguinte forma: No Capítulo 2 é apresentada uma introdução sobre códigos LDPC necessários para os capítulos seguintes. No Capítulo 3 são apresentados dois métodos que geram matrizes de verificação de paridade usadas na decodificação de códigos LDPC. No Capítulo 4 são apresentadas as simulações, resultados obtidos e encontrados na análise e desempenho de códigos LDPC implementados a partir das matrizes propostas e no Capítulo 5 as conclusões do trabalho.

Capítulo 2

Códigos LDPC

Os códigos LDPC são códigos de blocos lineares e descritos por matrizes de paridades esparsas. Neste capítulo é apresentada uma breve introdução à teoria de código de Bloco linear necessária para o entendimento de códigos LDPC. O leitor interessado em mais detalhes pode consultar [31- 35].

2. 1 Códigos de bloco lineares

Uma classe muito importante de códigos de detecção e correção de erros são os códigos de bloco lineares. Algebricamente falando, um Código de Bloco Linear é um subespaço Vetorial do corpo de Galois. Dito ou escrito de outra forma, pode-se definir um código de bloco como um bloco de mensagem de uma sequência binária de *k* dígitos de informações em um alfabeto *X*. Caso o alfabeto seja binário, $X = \{0,1\}$, o código diz-se binário. Para o total de *k* bits existem 2^k mensagens e, assim sendo, existem 2^k palavras-código de comprimento *n*. Logo, um código de blocos (*n, k*) é composto basicamente por mensagens de *k* bits, em blocos codificados de *n* bits, onde $n > k$, isto é, a palavra código é composta por *n* bits codificados [31].

Uma característica desejada para códigos de blocos lineares é que as palavra-códigos sejam apresentadas de forma sistemática, isto é, é possível separá-las em duas partes, a parte de mensagem e a parte de redundância [31]. Assim, os bits de redundância consistem em $n - k$ bits. A taxa de codificação é dada por *k/n*

Nesta seção se fará uma breve revisão algébrica a respeito dos conceitos básicos da teoria de códigos de blocos lineares e dos princípios da decodificação.

Definição 2.1 *Um código linear C de comprimento n e dimensão k sobre um corpo finito K com a elementos é um subespaço vetorial de dimensão k do espaço vetorial* K^n.

Aos vetores de *C* dá-se o nome de **palavra código**, **vetor código** ou simplesmente palavras. Portanto, *C* possui q^k palavras. Diz-se, nestas condições, que *C* é um (*n*, *k*) código linear sobre *K*.

Como *C* é um subespaço vetorial de K^n de dimensão *k*, então existem *k* vetores em *C*, $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_k$ linearmente independentes, e que geram *C*. Seja agora, G a matriz $k \times n$, cujas linhas são os vetores $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_k$. Tal matriz **G** é denominada a matriz geradora do código linear *C*. Note que *C* é na verdade o espaço linha de **G**.

Considere em K^n o produto interno definido por:

$$\langle \mathbf{u}, \mathbf{v} \rangle = \mathbf{u}_1\mathbf{v}_1 + \mathbf{u}_2\mathbf{v}_2 + \cdots + \mathbf{u}_n\mathbf{v}_n \tag{2.1}$$

onde $\mathbf{u} = \{\mathbf{u}_1, ..., \mathbf{u}_n\}$ e $\mathbf{v} = \{\mathbf{v}_1, ..., \mathbf{v}_n\}$ são elementos de K^n e seja $C \subset K^n$ um código linear de dimensão k. Considere o conjunto

$$C^{\perp} = \{\mathbf{v} \in K^n : \langle \mathbf{u}, \mathbf{v} \rangle = 0, \forall \mathbf{u} \in K^n\}. \tag{2.2}$$

Sabe-se da álgebra linear que $C^{\perp}$ é um subespaço de K^n de dimensão $n - k$, chamado de espaço ortogonal de *C*. Portanto, $C^{\perp}$ é também um código linear, chamado código dual de *C*.

Definição 2.2 *Dado um código linear C, a matriz* ***H*** *geradora de* $C^{\perp}$ *é chamada de matriz verificação de paridade ou teste paridade de C [27].*

Definição 2.3 *A distância de Hamming* $d_H(\mathbf{x}, \mathbf{y})$ *entre dois vetores* $\mathbf{x} = (x_1, x_2, ..., x_n)$ *e* $\mathbf{y} = (y_1, y_2, ..., y_n)$ *de* K^n *é o número de posições onde eles diferem.*

Definição 2.4 *A distância mínima de Hamming, d, de um código C é dada por*

$$d = \min\{d_H(\mathbf{x}, \mathbf{y}), \quad \mathbf{x}, \mathbf{y} \in \mathrm{C}, \ \mathbf{x} \neq \mathbf{y}\} \tag{2.3}$$

Um código linear de comprimento *n*, dimensão *k* e distância *d* será chamado um (*n, k, d*) código linear.

Definição 2.5 O peso de Hamming $w_H(\mathbf{x})$ de um vetor $\mathbf{x} = (x_1, x_2, ..., x_n) \in K^n$ é dado por $w_H(\mathbf{x}) = d_H(\mathbf{x}, \mathbf{0})$, onde **0** é o vetor nulo. Como consequência, tem-se que

$$d_H(\mathbf{x}, \mathbf{y}) = d_H(\mathbf{x} - \mathbf{y}; 0) = w_H(\mathbf{x} - \mathbf{y}). \tag{2.4}$$

Seja C um código linear, e $\mathbf{x}, \mathbf{y} \in C$. Então $\mathbf{x} - \mathbf{y} \in C$ e, portanto, o teorema a seguir é válido.

Teorema 2.*1* *Em um código linear C a distância mínima de Hamming é igual ao peso mínimo de Hamming de suas palavras código, isto é,*

$$d = \min\{w_H(\mathbf{x}), \quad \mathbf{x} \in C\} \tag{2.5}$$

Suponha que seja transmitida uma palavra código $\mathbf{v} = (v_1, ..., v_n)$ através de um canal ruidoso. Devido ao ruído introduzido pelo canal, o vetor recebido $\mathbf{r} = (r_1, r_2, ..., r_n)$ pode ser diferente de **v**. Define-se então o *vetor erro* por

$$\mathbf{e} = \mathbf{r} - \mathbf{v} \tag{2.6}$$

.

Diz-se que um código *C* detecta erros quando ele é capaz de decidir se o vetor recebido é ou não uma palavra código. Se, além de detectar erros, ele ainda é capaz de determinar a palavra transmitida, então se diz que *C* corrige erros.

Teorema 2.2 *Seja C um código com distância mínima d. Se C for usado somente para detecção, então C é capaz de detectar a presença de erros se ocorrerem até* $d - 1$ *erros. Se C for usado somente para correção, ele é capaz de corrigir até* $\left\lfloor \frac{d-1}{2} \right\rfloor$ *erros, onde* $\lfloor m \rfloor$ *denota o maior inteiro menor ou igual ao número m.*

Teorema 2.3 *Um código C, com distância mínima d, é capaz de detectar até* λ *erros e corrigir até t erros, simultaneamente, se* $\lambda + t + 1 \leq d$ e $t < \lambda$.

Dados um código C, com uma matriz verificação de paridade **H**, e um vetor $\mathbf{v} \in K^n$, chama-se o vetor $\mathbf{s} = \mathbf{H}\mathbf{v}^t$ de *síndrome* de **v**.

Quando **v** é uma palavra de *C*, tem-se $\mathbf{H}\mathbf{v}^{t} = 0$. Logo, $\mathrm{H}\mathrm{G}^{t} = 0$. Portanto, a síndrome $\mathbf{s} = \mathbf{H}\mathbf{r}^{t} = 0$ se, e somente se, **r** é uma palavra código.

Seja $\mathbf{v} \in C$ a palavra transmitida, **e** o padrão de erros introduzido pelo canal, e **r** a palavra recebida. Logo, a síndrome de **r** é:

$$\mathbf{s} = \mathbf{H}\mathbf{r}^{t} = \mathbf{H}(\mathbf{v}+\mathbf{e})^{t} = \mathbf{H}\mathbf{v}^{t} + \mathbf{H}\mathbf{e}^{t} = \mathbf{H}\mathbf{e}^{t}, \tag{2.7}$$

isto é, a síndrome de **r** é igual à síndrome de **e**.

Assim, a síndrome do vetor recebido pode ser útil na detecção de erros, no seguinte sentido: Se $\mathbf{s} \neq \mathbf{0}$, então certamente o canal introduziu um padrão de erro. Agora, se o canal não introduziu erros, isto é, **e** = 0, ou se o padrão de erro é uma palavra código, então **s** = 0. Entretanto, sempre que a síndrome do vetor r for nula, assume-se que não ocorram erros.

Teorema 2.4 *Sejam **H** a matriz de verificação de paridade de um código C. Tem-se que o peso de C é d se, e somente se, quaisquer $d-1$ colunas de **H** são linearmente independentes, e existem d colunas de **H** que são linearmente dependentes.*

Corolário 2.1 *(Cota de Singleton) Os parâmetros (n,k,d) de um código linear C satisfazem a desigualdade*

$$d \leq n - k + 1 \tag{2.8}$$

Definição 2.6 *Códigos com $d = n - k + 1$ são chamados de códigos com máxima distância de separação, ou códigos MDS (maximum distance serparable)*

Isto significa que a distância mínima de Hamming de um código MDS é a maior possível.

Teorema 2.5 *Um código C, com matriz de verificação de paridade **H**, é MDS se, e somente se, quaisquer $n-k$ colunas de **H** são linearmente independentes.*

2.1.1 Códigos cíclicos

Dentre os códigos lineares, a classe dos códigos cíclicos constitui a mais importante, não somente do ponto de vista prático, como também do ponto de vista teórico, pelas suas estreitas relações com a teoria dos anéis e ideais [35].

Definição 2.7 *Um código C é cíclico se todo deslocamento cíclico de uma palavra de C é ainda uma palavra de C, isto é, se* $\mathbf{c} = (c_0, c_1, ..., c_{n-1}) \in C$, *então* $\mathbf{c}' = (c_{n-1}, c_0, ..., c_{n-2})$ *também pertence a C.*

Descrevem-se os códigos cíclicos em uma linguagem algébrica, associando ao vetor código $\mathbf{c} = (c_0, c_1, ..., c_{n-1}) \in K^n$ o polinômio $c(x) = c_0 + c_1 x + \cdots + c_{n-1} x^{n-1}$, o qual é denominado o polinômio código associado a **c**.

Dados um corpo finito *K* e *n* um inteiro positivo, (*n* representa o comprimento de C), define-se A_n como sendo o anel quociente $K[x]/(x^n - 1)$. Em outras palavras, A_n é o anel formado pelas classes residuais de $K[x]$ módulo $x^n - 1$. Cada polinômio de grau menor ou igual a $n-1$ pertence a uma classe residual distinta, e consequentemente, pode-se tornar este polinômio como representante de sua classe residual. Portanto, pode-se dizer que $c(x) = c_0 + c_1 x + \cdots + c_{n-1} x^{n-1}$ pertence a A_n.

Teorema 2.6 *Seja C um código cíclico de comprimento n, isto é, um ideal nulo de* A_n. *Então:*

a) Existe um único polinômio Mônico g(x) de grau mínimo em C

b) $C = (g(x))$, *isto é,* $g(x)$ *é um polinômio gerador de C;*

c) $g(x)$ *é um divisor de* $x^n - 1$*;*

d) Qualquer $c(x) \in C$ *pode ser escrito de modo único como* $c(x) = f(x)g(x)$ *em* $K[x]$, *onde* $f(x) \in K[x], \partial(f) \le n - r, r = \partial(g)$. *A dimensão de C é n-r;*

e) Se $g(x) = g_0 + g_1 x + \cdots + g_r x^r$ *então C é gerado (como um subespaço de* K^n*) pelas linhas da matriz;*

$$\mathbf{G} = \begin{pmatrix} g_0 & g_1 & g_2 & \dots & g_r & 0 & \dots & 0 \\ 0 & g_0 & g_1 & \dots & g_{r-1} & g_r & \dots & 0 \\ & & & & & & & \\ & & & & & & & \\ 0 & \dots & 0 & g_0 & g_1 & \dots & \dots & g_r \end{pmatrix} \qquad (2.9)$$

Definição 2.8 *A matriz G (2.9), é dita matriz geradora do código cíclico C.*

Cada palavra código c é obtida multiplicando o vetor mensagem *m* por uma matriz geradora G, isto é, $\mathbf{c} = \mathbf{mG}$.

Seja C um código cíclico com polinômio gerador *g*(*x*). Pelo item c) do Teorema 2.6, tem-se que *g*(*x*) divide $x^n - 1$, isto é, existe *h*(*x*)∈ *K*[*x*] tal que $h(x)g(x) = x^n - 1$.

Supondo que $h(x) = \sum_{i=0}^{k} h_i x^i$. Seja $c(x) \in C$, tal que $c(x) = \sum_{i=0}^{k} c_i x^i = f(x)g(x)$. Então em A_n tem-se,

$$c(x)h(x) = \sum_{i=0}^{k} c_i x^i \sum_{j=0}^{k} h_j x^j = f(x)g(x)h(x) = f(x)(x^n - 1) = 0 \qquad (2.10)$$

O coeficiente de x^j no produto $c(x)h(x)$ é dado por $\sum_{i=1}^{n-1} c_i h_{j-1}$, $j = 0,1,...,n-1$. Logo, os coeficientes do produto $c(x)h(x)$ são todos nulos. Portanto, as equações em (2.10) são todas equações de verificação de paridade. Isto justifica a seguinte definição:

Definição 2.9 O *polinômio h*(*x*) *diz-se o polinômio verificação de paridade de C.*

Seja

$$\mathbf{H} = \begin{pmatrix} 0 & ... & ... & h_k & ... & h_2 & h_1 & h_0 \\ 0 & ... & h_k & h_{k-1} & ... & h_1 & h_0 & 0 \\ & & & & & & & \\ & & & & & & & \\ h_k & ... & ... & h_1 & h_0 & 0 & ... & 0 \end{pmatrix} \qquad (2.11)$$

As equações (2.10) mostram que $\mathrm{H}\mathbf{c}^t = \mathbf{0}$.

Definição 2.10 *A matriz* **H** *diz-se a matriz verificação ou teste paridade do código C*

Um código de bloco linear sistemático *C*(*n*, *k*) é especificado unicamente por sua matriz geradora **G**, que tem sua forma sistemática dada por [31]:

$$\mathbf{G} = \underbrace{\begin{bmatrix} g_0 \\ g_1 \\ \vdots \\ g_{k-1} \end{bmatrix}}_{\text{Submatriz } P_{k\times(n-k)}} = \underbrace{\begin{bmatrix} p_{00} & p_{01} & \cdots & p_{0,n-k-1} \\ p_{10} & p_{11} & \cdots & p_{1,n-k-1} \\ \vdots & \vdots & \vdots & \vdots \\ p_{k-1,0} & p_{k-1,1} & \cdots & p_{k,n-k-1} \end{bmatrix}}_{} \underbrace{\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}}_{\text{Submatriz } \mathrm{I}_{k\times k}} \qquad (2.12)$$

ou em notação abreviada escrita como $\mathbf{G} = [\mathbf{P} \mid \mathrm{I}_k]$ onde **P** é uma submatriz de paridade e $\mathbf{I}_k$ é a submatriz identidade de dimensão $k \times k$. Nesta forma sistemática de codificação, os bits da mensagem aparecem no final da palavra código.

A forma sistemática da matriz **H** do código $C(n, k)$ gerado pela matriz é dada por:

$$\mathbf{H} = \underbrace{\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}}_{\text{Submatriz } \mathbf{I}_{(n-k)\times(n-k)}} \underbrace{\begin{bmatrix} p_{00} & p_{10} & \cdots & p_{k-1,0} \\ p_{01} & p_{11} & \cdots & p_{k-1,1} \\ \vdots & \vdots & & \vdots \\ p_{0,n-k-1} & p_{1,n-k-1} & \cdots & p_{k-1,n-k-1} \end{bmatrix}}_{\text{Submatriz } \mathbf{P}^T{}_{(n-k)\times k}} = [\mathbf{I}_{n-k} \mathbf{P}^T] \tag{2.13}$$

onde $\mathbf{P}^T$ é a transporta da submatriz de paridade **P**.

2.1.2 Princípios da decodificação

Seja o sistema de comunicações digitais como mostrado na Figura 2.1.

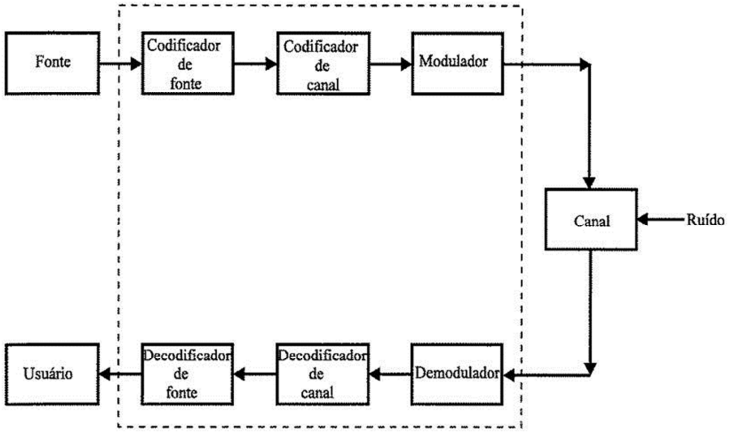


Figura 2.1: Sistema de Comunicações Digitais.

Suponha que a palavra código a ser enviada através do canal seja **v**. Supõe-se também, que o canal introduza um padrão de erro **e**. Assim, a palavra recebida será $\mathbf{r} = \mathbf{v} + \mathbf{e}$. O decodificador deve decidir a partir de **r** qual foi a palavra transmitida. Devido à natureza aleatória do ruído, o decodificador não é capaz de determinar com certeza absoluta qual foi o erro que realmente ocorreu. Diante disso, ele escolherá o vetor erro que tenha ocorrido com maior probabilidade, dado que **r** foi recebido.

Suponha que todas as palavras sejam igualmente prováveis, de forma a minimizar a probabilidade de erro da palavra, técnica (ou estratégia) conhecida como *decodificação de máxima verossimilhança*. Em poucas linhas pode-se resumir esta decodificação assim:

1. A partir dos $2^{n-k}-1$ padrões de erro mais prováveis calcula-se uma tabela de $2\,2^{n-k}-1$ síndromes $\mathbf{s}=\mathbf{eH}$ possíveis, onde **e** é a matriz de erros e **H** a matriz de verificação de paridade

2. Tendo recebido uma palavra **z** de *n* bits calcula-se a síndrome respectiva, $\mathbf{s}=\mathbf{zH}$.

3. Consulta-se a tabela de síndromes para determinar o padrão de erro mais provável, **ê**, que corresponde a síndrome calculada.

4. A palavra transmitida mais provável, $\hat{\mathbf{y}}$, obtém-se adicionando a palavra recebida, **z**, ao padrão de erro estimado.

Diz-se que não ocorrem erros durante a transmissão, em uma determinada palavra código, quando o vetor erro for todo nulo.

Seja, agora, X uma variável aleatória que representa o número de erros ocorridos em uma transmissão, isto é, X é o número de componentes não nulas de e. Em geral, se os erros forem independentes, tem-se que:

$$P(X=0)>P(X=1)>\cdots>P(X=n). \tag{2.14}$$

Portanto, a estratégia de decodificação de máxima verossimilhança é tornar o vetor erro de menor peso de Hamming, isto é, a palavra recebida será decodificada como a palavra código mais próxima, no espaço de Hamming [42]. Em particular, este tipo de decodificação é conhecido por decodificação pelo vizinho mais próximo. Uma maneira de implementá-la é a seguinte: recebida uma palavra, compare-a com todas as palavras código e tome a mais próxima. Entretanto, em geral, o número de palavras código é muito grande e este método torna-se altamente complexo e impraticável.

2.2 Códigos LDPC

Códigos baseados em matrizes esparsas têm desempenhado um importante papel na história da teoria da codificação, [29]. Muitos estudos em codificação nos últimos 50 anos foram direcionados à construção de códigos com estrutura robusta e grande distância mínima. Um código com grande distância mínima garante um bom desempenho do sistema, porém, a estrutura robusta torna o processo de decodificação mais complexo. Os códigos LDPC constituem uma família de códigos definidos a partir de matrizes de verificação de paridade esparsas (muitos zeros e um número pequeno de uns) que apresentam desempenho excelente em uma grande variedade de canais [37]. O sucesso desses códigos se deve, principalmente, porque eles podem ser representados eficientemente através de grafos que permite uma decodificação eficiente através de um algoritmo iterativo baseado em algoritmo soma e produto cuja complexidade cresce linearmente com o comprimento da palavra-código e também pela existência de uma técnica analítica conhecida como *density evolution* [41] para analisar e projetar códigos com desempenho próximo da capacidade de diversos canais.

Nesta seção são apresentadas as informações importantes sobre códigos LDPC necessárias para o capítulo posterior e são características que levam este código a ser considerado uma das mais poderosas famílias de códigos corretores de erro.

2.2.1 Códigos de Gallager

Os códigos LDPC foram primeiramente descobertos por Gallager [14] no começo dos anos sessenta e tem sido recentemente redescoberto e estudado [16]. O que levou o seu esquecimento por tantos anos foi a elevada complexidade computacional requerida tanto para a geração de uma matriz **H,** quanto para a codificação e decodificação. Nesta época, os computadores não eram capazes de simular o desempenho de códigos com grandes comprimentos e com baixas taxas de erro que garantissem uma boa distância mínima do código, na codificação e na sua decodificação tendo por base o algoritmo Soma e Produtos (SPA) proposto pelo próprio Gallager.

Essa dificuldade computacional não provocou interesse nos pesquisadores e fez com que os códigos LDPC fossem deixados de lado por muito tempo. Somente mais tarde, em 1981, R.M.Tanner [38] generalizou o trabalho de Gallager e introduziu a

representação gráfica de códigos LDPC através de grafos bipartidos. Em meados de 90, Mackay e Neal [16], após o advento dos códigos turbo os redescobriu. Eles mostraram que os códigos LDPC longos, quando decodificados com o algoritmo Soma-Produto (SPA), são capazes de atingir um desempenho muito próximo ao limite de Shannon [37], [39-41]. Desta forma, estes códigos se tornaram os competidores naturais dos códigos turbo para a correção de erro em muitos canais de comunicação como transmissão via internet e sistemas de armazenamento magnéticos.

Embora tenha sido mostrado que os códigos LDPC alcancem excelente desempenho, nenhum método analítico (algébrico ou geométrico) foi desenvolvido para construir estes códigos. Gallager apenas estabeleceu uma classe de códigos pseudo-aleatórios LDPC [14], [15]. Bons códigos de LDPC foram basicamente gerados e determinados por processos computacionais, especialmente os códigos de longo comprimento.

Definição 2.11 Um código LDPC é por definição o espaço nulo de uma matriz de verificação de paridade **H** de dimensões m x n que possui as seguintes propriedades [35]:

1) cada linha possui peso constante ρ 1's*;*

2) cada coluna possui peso constante γ 1's;

3) O numero de 1's em comum entre duas colunas quaisquer, denotado por λ não é maior do que 1; isto é, λ =0 ou λ =1;

4) ρ e γ são ambos pequenos se comparados com o comprimento do código e o numero de linhas em **H**.

O código dado pela Definição 2.11 é chamado de código LDPC (ρ, γ) regular de comprimento n. As propriedades (1) e (2) dizem que a matriz de verificação de paridade **H** tem linhas e colunas constantes com pesos ρ e γ respectivamente. A propriedade (3) nos diz que nenhuma de duas linhas de **H** tem mais do que um elemento 1 em comum. Como ρ e γ são pequenos comparados com o comprimento n do código e o número de linhas da matriz, **H** tem portanto uma pequena densidade de 1's. Por esta razão, **H** é chamada matriz de baixa densidade e o código especificado por **H** chamado LDPC.

Define-se a densidade r da matriz de verificação de paridade H (porcentagem de 1's na matriz), como a razão entre o numero total de 1's em **H** e o número total de elementos em H. Como o total de elementos na matriz é m x n, a densidade de **H** é dada por: $r = \frac{\rho}{n} = \frac{\gamma}{m}$.

Exemplo 2.1 Considere a matriz

$$\mathbf{H} = \begin{bmatrix}
0&0&0&0&0&0&0&1&1&0&1&0&0&0&1\\
1&0&0&0&0&0&0&0&1&1&0&1&0&0&0\\
0&1&0&0&0&0&0&0&0&1&1&0&1&0&0\\
0&0&1&0&0&0&0&0&0&0&1&1&0&1&0\\
0&0&0&1&0&0&0&0&0&0&0&1&1&0&1\\
1&0&0&0&1&0&0&0&0&0&0&0&1&1&0\\
0&1&0&0&0&1&0&0&0&0&0&0&0&1&1\\
1&0&1&0&0&0&1&0&0&0&0&0&0&0&1\\
1&1&0&1&0&0&0&1&0&0&0&0&0&0&0\\
0&1&1&0&1&0&0&0&1&0&0&0&0&0&0\\
0&0&1&1&0&1&0&0&0&1&0&0&0&0&0\\
0&0&0&1&1&0&1&0&0&0&1&0&0&0&0\\
0&0&0&0&1&1&0&1&0&0&0&1&0&0&0\\
0&0&0&0&0&1&1&0&1&0&0&0&1&0&0\\
0&0&0&0&0&0&1&1&0&1&0&0&0&1&0
\end{bmatrix}_{15\times 15}$$

Cada linha e cada coluna da matriz **H** possui quatro 1's respectivamente. Observa-se que nenhuma entre duas colunas ou linhas desta matriz possui mais do que um elemento 1 em comum. A densidade da matriz é r=4/15=0,267. Portanto **H** é uma matriz de baixa densidade.

2.2.2 Construção de códigos de Gallager

O projeto de um código LDPC consiste na construção da matriz de verificação de paridade **H** que atinja os objetivos pretendidos para o código, seja ele um código LDPC regular ou irregular. Existem diversos métodos para se obter esta matriz de verificação de paridade e fazendo-se algumas restrições nesta matriz, como número de 1's por coluna, taxa do código, etc., pode-se criar várias famílias de códigos LDPC.

Seja k um inteiro maior do que 1. A construção seguinte é conhecida por *Construção de Gallager*. Forme uma $k\gamma \times k\rho$ matriz **H** a qual consiste de γ $k \times k\rho$ submatrizes, $\mathbf{H}_1, \mathbf{H}_2, \ldots, \mathbf{H}_\gamma$. Cada linha da submatriz tem ρ 1's, e cada coluna da submatriz contém um único 1. Portanto, cada submatriz tem um total de $k\rho$ 1's. Para $1 \le i \le k$, a i-ésima linha de $\mathbf{H}_1$ contém todos os ρ 1's nas colunas $(i\text{-}1)\rho+1$ à $i\,\rho$. As outras submatrizes são apenas permutações das colunas de $\mathbf{H}_1$. Então:

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_1 \\ \mathbf{H}_2 \\ \vdots \\ \mathbf{H}_\gamma \end{bmatrix}$$

Exemplo 2.2: *Seja k = 5, ρ = 4 e γ = 3. Usando a construção de Gallager forma-se a matriz H 15 x 20 como mostra a figura abaixo, a qual consiste de três submatrizes 5 x 20, $\boldsymbol{H}_1$,$\boldsymbol{H}_2$, e $\boldsymbol{H}_3$. Cada linha de $\boldsymbol{H}_1$ consiste de quatro 1's consecutivos, e nenhuma de duas linhas possuem o numero 1 como componente em comum. As submatrizes $\boldsymbol{H}_2$, e $\boldsymbol{H}_3$ são obtidas por duas permutações diferentes das colunas de $\boldsymbol{H}_1$ de tal forma que, nenhumas de duas colunas (ou duas linhas) de **H** tem mais do que um 1 em comum. A densidade de **H**, r = 0,20. Consequentemente, o espaço nulo de **H** gera um código LDPC.*

$$H = \left[\begin{array}{cccccccccccccccccccc}
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
\hdashline
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
\hdashline
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1
\end{array}\right]$$

2.2.3 O Grafo de Tanner

Um grafo $\mathcal{G}$ é uma estrutura que consiste de um conjunto de vértices (ou nós), denotados por $\mathcal{V}$={v_1,v_2,...}, e um conjunto de ramos, denotados por $\mathcal{R}$={ $\eta_{1,}$ $\eta_{2,...}$}.

Cada ramo η_k está associado a um par (ordenado ou não) (v_i,v_j) de vértices (não necessariamente distintos) do conjunto $\mathcal{V}$. Os vértices v_i e v_j são chamados de vértices terminais de η_k. O grafo é denotado por $\mathcal{G}$ = ($\mathcal{V}$,$\mathcal{R}$).

Um grafo $\mathcal{G}$ pode ser representado por um diagrama de pontos e ramos. Os pontos correspondem aos vértices e as linhas aos ramos. Um grafo $\mathcal{G}$ é dito simples quando:

1) não existe nenhum ramo conectando um nó a si mesmo;
2) existe no máximo um ramo entre quaisquer par de vértices;
3) todos os ramos são não direcionados (não ordenado).

Dois vértices x e y pertencentes ao conjunto V são adjacentes se o par (x,y) corresponde a um ramo em $\mathcal{R}$. O conjunto de todos os vértices que são adjacentes a x é chamado de vizinhança de x.

Seja S uma sequência de vértices distintos. Se $x, y \in S$, de tal forma que x é o início da sequência e y o término, então S é um caminho entre x e y se todos os vértices contidos em S forem adjacentes. Se existe um caminho entre x e y, diz-se que x e y estão conectados e pode-se determinar a distância entre eles. Denotada por $d(x, y)$ a distância entre x e y, é por definição, o número de ramos do caminho mais curto entre x e y.

Um ciclo de x é um caminho fechado que inicia e termina em x. O ciclo mínimo é o comprimento do menor ciclo em um grafo. Dado o vértice x, denota-se por g_x o comprimento do menor ciclo que passa por x, o qual é chamado de ciclo mínimo local de x. Dai pode-se concluir que o ciclo mínimo g do grafo $\mathcal{G}$ = ($\mathcal{V}$, $\mathcal{R}$) é dado por

$$g = \min_{x \in V}(g_x) \tag{2.15}$$

Um *grafo bipartido* é um grafo cujo conjunto de vértices V é dividido em dois subconjuntos distintos, denotados por V_c (nó de verificação de paridade) e V_s (nó de símbolos). Nenhum par de vértices pertencentes a um mesmo subconjunto é adjacente.

A Figura 2.2.a abaixo é um exemplo de grafo bipartido. Os vértices em branco e preto são dois subconjuntos de vértices. Observe que os vértices em branco estão conectados apenas aos vértices em preto e vice-versa. Na Figura 2.2.b, os vértices em preto estão conectados entre si e, portanto não são bipartidos.

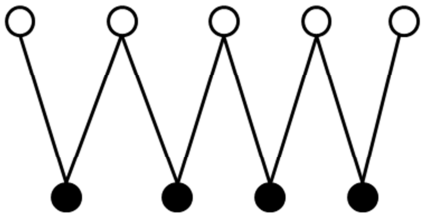


Figura 2.2.a Grafo bipartido

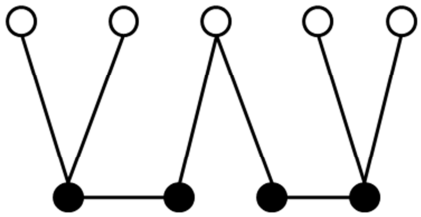


Figura 2.2.b Grafo não bipartido

Um *grafo de Tanner* para um código de bloco C é um grafo bipartido não direcionado obtido a partir de uma matriz de verificação de paridade **H** com *n* colunas e *m* linhas. As *n* colunas correspondem a N nós de símbolo e as *m* linhas correspondem a M nós de verificação de paridade. Um ramo conecta o nó de símbolo x_n ao *m-ésimo* nó de verificação c_m, se e somente se, $h_{m,n}$=1 onde $h_{m,n}$ denota o elemento na *m-ésima* linha e *n-ésima* coluna de **H**.

O grau de um nó de um vértice x_n em um grafo de Tanner corresponde ao número de ramos que incidem neste nó e é denotado por $\partial(x_n)$. Um grafo de Tanner é chamado de *regular* quando $\partial(x_n)=\gamma$, $1\le n\le N$ e $\partial(c_n)=\rho$, $1\le m\le M$, caso contrário, o grafo é chamado de irregular.

Exemplo 2.3 *Seja* ***H*** *a matriz de verificação de paridade de um código C*

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

e considere $\boldsymbol{x}=(x_1,x_2,x_3,x_4,x_5,x_6,x_7)$ *uma palavra pertencente ao código C. Portanto* $\boldsymbol{xH}^T=0$ *e daí*

$$\begin{cases}x_1+x_2+x_4+x_5=0\\x_1+x_3+x_4+x_6=0\\x_2+x_3+x_4+x_7=0\end{cases} \tag{2.16}$$

O grafo de Tanner para este código possui 3 nós de verificação (representando as equações de paridade) e 7 nós de símbolo (representando os símbolos da palavra código). Denotando os nós de símbolo por ● e por ■ os nós de verificação de paridade. O grafo de Tanner para o código C(7,3) é mostrado na Figura 2.3

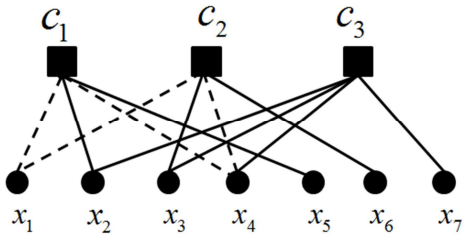


Figura 2.3 – O Grafo de Tanner com ciclo.

Um ciclo em um grafo de Tanner refere-se a um conjunto finito de conexões entre os nós, um ramo começa e termina em um mesmo nó, e satisfaz a condição que nenhum nó (exceto o inicial e o final) aparece mais do que uma vez. O número de ramos que aparecem em um ciclo determina o seu comprimento.

A Figura 2.3 mostra o grafo de Tanner. A sequência $\{x_1,c_1,x_4,c_2,x_1\}$ (linha tracejada) representa um ciclo de comprimento 4 também chamado um 4-ciclo.

No projeto de códigos LDPC é desejável que ciclos de comprimento curto sejam evitados, especificamente os de comprimento 4 [35], isto é, projetar matrizes de verificação de paridade que não tenham 4-ciclos ou seja *matrizes 4-ciclos livres*. Para projetar a matriz de 4-ciclo marque dois "1s" (α_1 e α_2) pertencentes a linha q_1 no cruzamento das colunas v_1 e v_2 respectivamente . Marque dois "1s" pertencentes a linha q_2 no cruzamento das colunas v_1 e v_2 (α_3 e α_4). O conjunto $G_4(\alpha_1, \alpha_2, \alpha_3, \alpha_4)$ denota um 4-ciclo em uma matriz de verificação de paridade, Figura 2.4 (a). A Figura 2.4 (b) mostra este 4-ciclo em arvore.

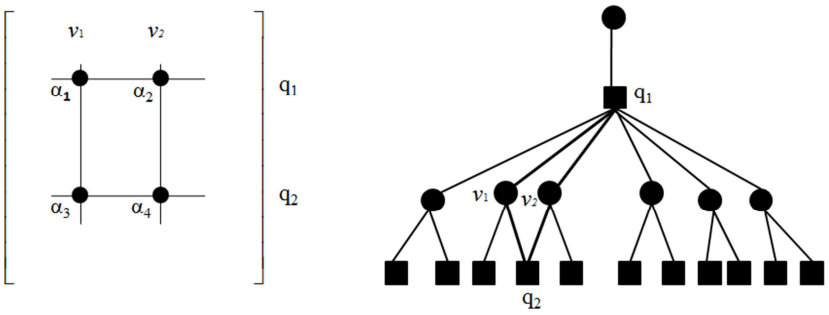


Figura 2.4: (a) 4-ciclo em matriz. **(b) 4-ciclo em arvore.**

2.2.4 Decodificação dos códigos de Gallager – O algoritmo soma e produto

Nesta seção será apresentado o algoritmo mais utilizado para a decodificação de códigos LDPC, que é uma descrição algébrica do gráfico de Tanner, conhecido como soma-produto (*sum-product algorithm-SPA).* Neste algoritmo a decodificação é um

problema de inferência estatística: dado um conjunto de observações, deseja-se deduzir qual a palavra-código mais provável de ter sido transmitida, ou qual o valor mais provável de cada símbolo transmitido [31-32]. Essencialmente este algoritmo determina a estimativa (probabilidade *posteriori*) de um vetor *d* satisfazendo a condição $\mathbf{H} \circ \mathbf{d} = \mathbf{0}$. Onde $\mathbf{H} \circ \mathbf{d}$ representa o produto da matriz **H** pelo vetor **d** em GF(2) **[**31**]**.

O algoritmo SPA opera sobre o grafo de Tanner e funciona, basicamente, como um algoritmo de transmissão de mensagens entre os nós de símbolo (d_j), isto é, os bits ou símbolos transmitidos, e o nó de verificação de paridade (h_j), que representa as equações de paridade em que os bits ou símbolos estão relacionados. De uma forma simplista, cada nó pode ser visto como um processador de mensagens recebidas dos seus vizinhos (nós ligados a ele), aos quais devolve mensagens atualizadas. As mensagens recebidas ou enviadas por um nó (de símbolo e nó de verificação de paridade), nada mais são do que "opiniões" sobre o valor lógico dos nós ligados a eles por soma binária. Essas "opiniões" são expressas em termos da distribuição probabilística dos símbolos recebidos pelo decodificador.

Daqui por diante nó de símbolo e nós de verificação de paridade serão chamados ***nó de variáveis*** e ***nó de cheque***, respectivamente.

As linhas da matriz **H** identificam os nós de variáveis, enquanto as colunas os nó de cheque, de modo que uma dada linha descreve uma equação de paridade, e as posições que contém 1's determinam a posição dos símbolos envolvidos . Desta forma, se a entrada $\{i, j\}$ de uma matriz **H** é igual a um, $\mathbf{H}_{ij}=1$, então existe um correspondente gráfico bipartido entre os nós d_j e h_j . De outra forma a conexão não está presente.

No algoritmo soma-produto, cada nó de variáveis d_j envia para cada nó "filho" h_j uma estimativa Q_{ij}^{x} de probabilidade em que este nó esta no estado *x*, baseado nas informações fornecidas pelos outros nós filhos. Na contramão, cada nó de cheque h_j envia para cada um de seus nós parentes d_j uma estimativa de probabilidade R_{ij}^{x} descrita pela equação de paridade *i* a qual relata que a paridade do nó de cheque h_j esta satisfeita se acaso o símbolo ou nó parente estiver no estado *x*, isto tomando por conta das informações provenientes dos outros nós parentes, é o que descreve a Figura 2.5 seguinte.

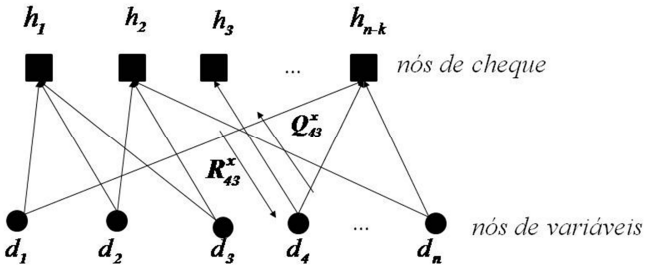


Figura 2.5 Informações entre os nós.

O algoritmo SPA, na decodificação de códigos LDPC, tem dois métodos que variam de acordo com o modelo do canal de transmissão:

1) A decodificação abrupta (*hard decoding ou hard decision*) considera que o conjunto de símbolos na entrada do decodificador é finito, ou seja, no caso de um código binário na entrada do decodificador tem-se apenas bits, a decisão se cada bit recebido foi 0 ou 1 sendo tomada antes da decodificação. Depois é escolhida a palavra-código mais próxima do vetor recebido;

2) A decodificação suave (*soft decoding ou soft decision*) considera a distribuição probabilística dos símbolos recebidos, sem a decisão prévia bit a bit, e toma sua decisão baseada na palavra inteira

Os passos seguintes deste algoritmo foram baseados em decodificação abrupta, contudo com alguns ajustes ele pode ser adaptado a decodificação suave. O leitor interessado em mais detalhes pode consultar [31], [35].

O algoritmo começa com um procedimento de inicialização que consiste de determinar os valores de Q_{ij}^x, que são um conjunto de estimativas a priori recebido pelos bits, denotadas por f_j^x, isto é, a probabilidade de que j-ésimo símbolo seja *x*. Esta informação depende do canal a ser utilizado e, neste caso será usado o canal Gaussiano AWGN, cujas probabilidades são determinadas pelo uso da função de densidade de probabilidade.

Após a inicialização começa a troca de informação entre os bits e os nós. A informação R_{ij}^x que cada nó de cheque h_i envia para o nós parentes d_j é a probabilidade que este nó de cheque tem satisfeita quando os nó parentes informarão está no estado *x*.

A equação de probabilidade na condição satisfeita é dada por:

$$\mathrm{P}\left(h_i / d_i = \mathrm{x}\right) = \sum_{\mathrm{d:d_j=x}} \mathrm{P}\left(h_i / \mathbf{d}\right) \mathrm{P}\left(\mathbf{d} / d_j = \mathrm{x}\right) \tag{2.17}$$

Esta probabilidade é calculada sobre todas as possíveis decodificações do vetor **d** para o qual é satisfeita as equações de paridade desde que os nó parentes estejam no estado *x*.

Para o nó de cheque h_i a informação enviada para o nó parente d_j é calculada por cada valor de *x* e é dada por:

$$\mathrm{R}_{\mathrm{ij}}^{\mathrm{x}} = \sum_{\mathrm{d:d_j=x}} \mathrm{P}\left(h_i / \mathbf{d}\right) \prod_{\mathrm{k}\in \mathrm{N(i)\backslash j}} \mathrm{Q}_{\mathrm{ik}}^{\mathrm{dk}} \tag{2.18}$$

Na expressão (2.18), *N(i)* representa o conjunto dos índices de todos os nó parentes conectados com os nó de cheque h_i, enquanto *N(i)\j* representa a exclusão do nó parente d_j. Para um dado vetor **d** a probabilidade *P(*h_i*/***d***)* é satisfeita quando for 1 e não satisfeita quando for 0. O nó de variável d_j envia para seus filhos nó de cheque h_i a estimativa Q_{ij}^x informando que ele esta no estado *x* de acordo com as outras informações recebidas por suas outras conexões. Então pela fórmula de Bayes tem-se:

$$\mathrm{P}\left(d_j = \mathrm{x} / \left\{h_i\right\}_{\mathrm{i}\in \mathrm{M(j)\backslash i}}\right) = \frac{\mathrm{P}\left(d_j = \mathrm{x}\right) \mathrm{P}\left(\left\{h_i\right\}_{\mathrm{i}\in \mathrm{M(j)\backslash i}} / d_j = \mathrm{x}\right)}{\mathrm{P}\left(\left\{h_i\right\}_{\mathrm{i}\in \mathrm{M(j)\backslash i}}\right)} \tag{2.19}$$

A informação que o nó de variável d_j envia para os nós de cheque (filhos) é

$$\mathrm{Q}_{\mathrm{ij}}^{\mathrm{x}} = \alpha_{\mathrm{ij}} \mathrm{f}_{\mathrm{j}}^{\mathrm{x}} \prod_{\mathrm{k}\in \mathrm{M(j)\backslash i}} \mathrm{R}_{\mathrm{kj}}^{\mathrm{x}} \tag{2.20}$$

onde *M(j)* representa o conjunto dos índices de todos os nós de cheque conectados a ele e *M(j)\i* denota a exclusão do nó de cheque h_i. O coeficiente f_j^x é a probabilidade a priori de que d_j está no estado *x*. A constante normalizada α_{ij} é o conjunto que satisfaz a condição $\sum_x Q_{ij}^x = 1$

Neste caminho, o cálculo dos coeficientes, Q_{ij}^x, nos permite determinar os valores dos coeficientes, R_{ij}^x, que pode ser usado para realizar uma estimativa para cada índice *j*. Finalmente, calcula-se a estimava $\widehat{\mathbf{d}}_j$ do vetor recebido para os dois possíveis valores de

x. Se, a síndrome de erro **S** do vetor $\widehat{\mathbf{d}}_j$, for zero então o processo termina. Se **S**≠0, calcula-se a próxima iteração até chegar-se na síndrome nula. O vetor $\widehat{\mathbf{d}}_j$ é calculado por:

$$\widehat{\mathbf{d}}_j = \arg\max \mathrm{f}_\mathrm{j}^{\mathrm{x}} \prod_{\mathrm{k}\in \mathrm{M(j)}} R_{kj}^{x} \tag{2.21}$$

Com o objetivo de facilitar o entendimento do algoritmo apresenta-se um simples exemplo baseado em decisão abrupta.

Exemplo 2.4: *Considere **H** uma matriz de verificação de paridade 8x12 correspondente ao código de bloco linear $C_b(12,4)$ de razão R_c = 1/3, o qual é um código LPDC irregular gerado pela matriz G abaixo:*

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

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

Seja *m* = (1,0,0,0) a mensagem para este código e c a palavra código determinada, isto é: ***c***=*m***G**=(1 1 1 1 1 0 0 0 1 0 0 0) de modo que *c* satisfaz a condição de síndrome ***c*** o $\mathbf{H}^T$ = *0.* Este vetor é transmitido em forma polar como vetor *t* = (+1+1+1+1+1-1-1-1+1-1-1-1). A transmissão é feita sobre um canal AWGN com desvio padrão σ=0.8 e como resultado de transmissão das amostras, o seguinte vetor recebido é obtido:

R = (+1.3129 +2.6582 +0.7413 +2.1745 +0.5981 -0.8323 -0.3962 -1.7586 +1.4905 +0.4084 -0.9290 +1.0765).

Se uma decisão abrupta na decodificação foi utilizada, então o vetor decodificado será:

(1 1 1 1 1 0 0 0 1 1 0 1)

de modo que o canal produz dois erros nas posições 10 e 12.

Como o canal nesta transmissão é o AWGN, os coeficientes f_j^x correspondentes ao vetor recebido podem ser calculados usando a função de densidade de probabilidade Gaussiana, e assim:

$$\begin{aligned} f_j^0 &= \frac{1}{\sqrt{2\pi\sigma}} e^{-(r_j+1)^2/(2\sigma^2)} \\ f_j^1 &= \frac{1}{\sqrt{2\pi\sigma}} e^{-(r_j-1)^2/(2\sigma^2)} \end{aligned} \tag{2.22}$$

Na Tabela 2.1 e nas tabelas seguintes os valores são truncados na quarta casa decimal e não modificam os resultados.

Tabela 2.1: Valores do vetor recebido e correspondentes valores de f_j^x

	1	2	3	4	5	6	7	8	9	10	11	12
r	**+1.3129**	**+2.6582**	**+0.7413**	**+2.1745**	**+0.5981**	**-0.3962**	**-0.3962**	**+1.7585**	**+1.4905**	**+0.4084**	**-0.9290**	**+1.0765**
t	**+1**	**+1**	**+1**	**+1**	**+1**	**-1**	**-1**	**-1**	**+1**	**-1**	**-1**	**-1**
f_j^0	**0.0076**	**0.0000**	**0.0467**	**0.0002**	**0.0678**	**0.4878**	**0.3751**	**0.3181**	**0.0039**	**0.1059**	**0.4967**	**0.0172**
f_j^1	**0.4619**	**0.0582**	**0.4733**	**0.1697**	**0.4396**	**0.0362**	**0.1088**	**0.0013**	**0.4132**	**0.3794**	**0.0272**	**0.4964**

Os valores de f_j^x representam as estimativas do canal de informação. Os coeficientes envolvidos nos cálculos das interações levam em conta a estrutura do código descrito pelo gráfico bipartido correspondente. Assim, a condição de síndrome, expressadas como $\mathbf{H} \circ \boldsymbol{c} = \mathbf{0}$ ou $\boldsymbol{c} \circ \mathbf{H}^T = \mathbf{0}$, significa que o produto do vetor código $\boldsymbol{c}$ pela transporta da matriz cheque deverá ser o igual ao vetor nulo. Portanto a equações teste paridade podem ser escrita como:

$$\begin{aligned}
c_2 \oplus c_4 \oplus c_6 \oplus c_7 \oplus c_8 \oplus c_{12} &= 0 \\
c_1 \oplus c_3 \oplus c_4 \oplus c_9 &= 0 \\
c_2 \oplus c_5 \oplus c_7 \oplus c_{12} &= 0 \\
c_1 \oplus c_4 \oplus c_{10} \oplus c_{11} &= 0 \\
c_3 \oplus c_5 \oplus c_6 \oplus c_{10} &= 0 \\
c_1 \oplus c_3 \oplus c_7 \oplus c_8 \oplus c_{11} &= 0 \\
c_2 \oplus c_6 \oplus c_8 \oplus c_9 \oplus c_{10} &= 0 \\
c_5 \oplus c_9 \oplus c_{11} \oplus c_{12} &= 0
\end{aligned} \tag{2.23}$$

A Figura 2.6 mostra o relacionamento entre os nós e a equação de paridade

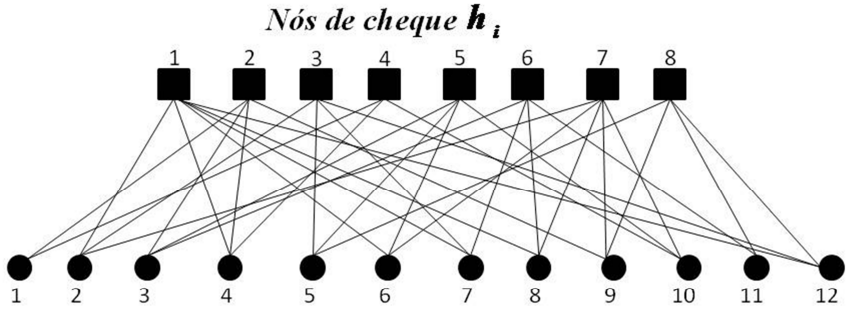


Figura 2.6: Grafo bipartido do sistema

Cada linha da matriz de verificação de paridade **H** corresponde a uma equação teste paridade, e assim a um nó de cheque. Cada bit do vetor código corresponde a um nó de variável. Assim, por exemplo, o nó de variável 2 tem como filhos os nós de cheque 1, 3, 7, enquanto o no de cheque 1 tem como parentes os nós de variáveis 2, 4, 6, 7, 8 e 12.

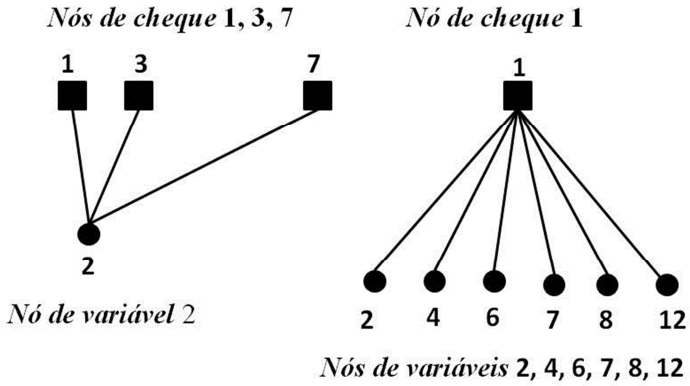


Figura 2.7: Informações entre os nós de cheque e nós de variáveis

A inicialização do algoritmo soma-produto é feita pelo ajuste dos coeficientes da informação que foi enviada do nó de variável para o nó de cheque na primeira iteração: Q_{ij}^0 e Q_{ij}^1, que são as estimativas do canal de informação f_j^0 e f_j^1 respectivamente. Assim, por exemplo, a associação entre os nós de variáveis 1, 3, 4, 9 e o nó de cheque 2 geram as estimativas $Q_{21}^x, Q_{23}^x, Q_{24}^x, Q_{29}^x$ que podem ser entendidas mais facilmente se interpreta-se como uma forma de comunicação, um envio de informação entre os seus participantes. Desta forma Q_{14}^0 é a probabilidade que o nó de variável 4 envia, no estado zero, para o nó de cheque 1. A Figura 2.8 mostra esta associação

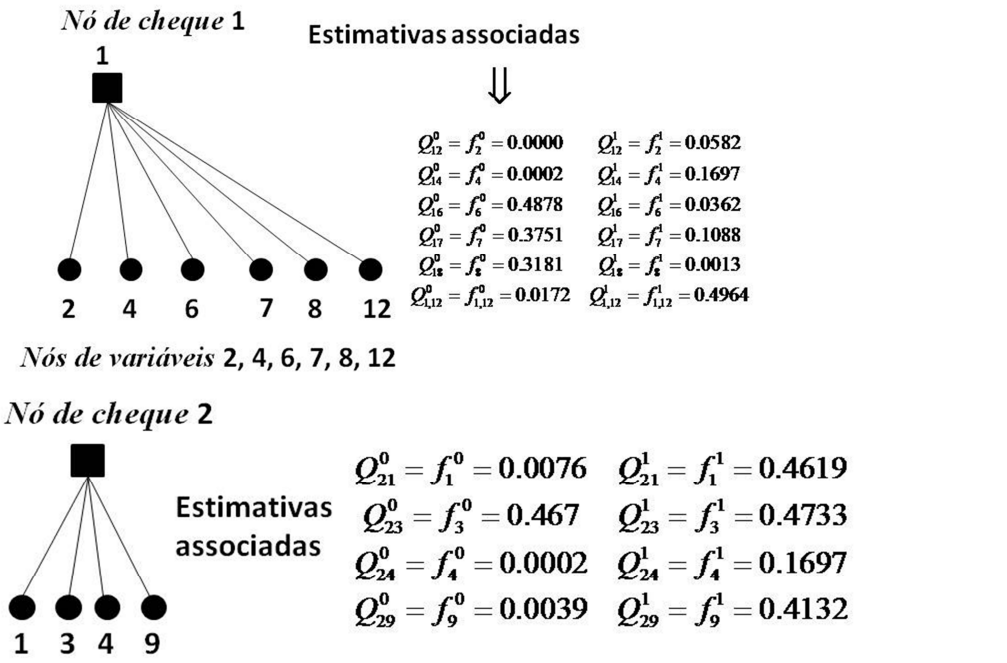


Figura 2.8: Informações entre os nós e estimativas associadas

Os valores da inicialização permitem calcular os coeficientes R_{ij}^0 e R_{ij}^1 que serão iterativamente atualizados durante a codificação. Estas estimativas são informações enviadas dos nós de cheque h_i para os nós de variáveis d_j. A notação R_{12}^0 descreve a probabilidade que o nó de cheque 1 enviou ao seu "parente" nó de variável 2 de que este (nó de cheque 1) está no estado zero. Esta probabilidade é calculada assumindo que sua equação de paridade correspondente $c_2 + c_4 + c_6 + c_7 + c_8 + c_{12} = 0$ está satisfeita quando o bit c_2 está no estado 0 (zero). Neste sentido existem 16 combinações possíveis com par de números "1s" que satisfazem a condição da equação envolvendo os bits c_4, c_6, c_7, c_8 e c_{12}. As probabilidades associadas a cada combinação são somadas para o calculo da estimativa R_{12}^0, isto é:

$$\begin{aligned} R_{12}^0 = \; & Q_{14}^0 Q_{16}^0 Q_{17}^0 Q_{18}^0 Q_{1,12}^0 + Q_{14}^0 Q_{16}^0 Q_{17}^0 Q_{18}^1 Q_{1,12}^1 + Q_{14}^0 Q_{16}^0 Q_{17}^1 Q_{18}^0 Q_{1,12}^1 + Q_{14}^0 Q_{16}^0 Q_{17}^1 Q_{18}^1 Q_{1,12}^0 + \\ & Q_{14}^0 Q_{16}^1 Q_{17}^0 Q_{18}^0 Q_{1,12}^1 + Q_{14}^0 Q_{16}^1 Q_{17}^0 Q_{18}^1 Q_{1,12}^0 + Q_{14}^0 Q_{16}^1 Q_{17}^1 Q_{18}^0 Q_{1,12}^0 + Q_{14}^0 Q_{16}^1 Q_{17}^1 Q_{18}^1 Q_{1,12}^1 + \\ & Q_{14}^1 Q_{16}^0 Q_{17}^0 Q_{18}^0 Q_{1,12}^1 + Q_{14}^1 Q_{16}^0 Q_{17}^0 Q_{18}^1 Q_{1,12}^0 + Q_{14}^1 Q_{16}^0 Q_{17}^1 Q_{18}^0 Q_{1,12}^0 + Q_{14}^1 Q_{16}^0 Q_{17}^1 Q_{18}^1 Q_{1,12}^1 + \\ & Q_{14}^1 Q_{16}^1 Q_{17}^0 Q_{18}^0 Q_{1,12}^0 + Q_{14}^1 Q_{16}^1 Q_{17}^0 Q_{18}^1 Q_{1,12}^1 + Q_{14}^1 Q_{16}^1 Q_{17}^1 Q_{18}^0 Q_{1,12}^1 + Q_{14}^1 Q_{16}^1 Q_{17}^1 Q_{18}^1 Q_{1,12}^0 \\ & = 0.0051 \end{aligned} \tag{2.24}$$

Com a notação $Q_{14}^0 Q_{16}^0 Q_{17}^0 Q_{18}^1 Q_{1,12}^1$ denota-se o produto das estimas Q_{ij}^x onde os valores de *x*: 0, 0, 0, 1, 1, são uma das 16 combinações possíveis que os bits c_4, c_6, c_7, c_8, c_{12} assumem, respectivamente, para que a equação $c_2 \oplus c_4 \oplus c_6 \oplus c_7 \oplus c_8 \oplus c_{12} = 0$ seja satisfeita.

Da mesma maneira calcula-se o valor da estimativa R_{12}^1 salientado apenas que o valor de c_2 na equação de paridade agora é 1, desta forma tem-se 16 combinações possíveis com impar números de "1s" , isto é:

$$\begin{aligned} R_{12}^1 = \; & Q_{14}^0 Q_{16}^0 Q_{17}^0 Q_{18}^0 Q_{1,12}^1 + Q_{14}^0 Q_{16}^0 Q_{17}^0 Q_{18}^1 Q_{1,12}^0 + Q_{14}^0 Q_{16}^0 Q_{17}^1 Q_{18}^0 Q_{1,12}^0 + Q_{14}^0 Q_{16}^0 Q_{17}^1 Q_{18}^1 Q_{1,12}^1 \\ & + Q_{14}^0 Q_{16}^1 Q_{17}^0 Q_{18}^0 Q_{1,12}^0 + Q_{14}^0 Q_{16}^1 Q_{17}^0 Q_{18}^1 Q_{1,12}^1 + Q_{14}^0 Q_{16}^1 Q_{17}^1 Q_{18}^0 Q_{1,12}^1 + Q_{14}^0 Q_{16}^1 Q_{17}^1 Q_{18}^1 Q_{1,12}^0 \\ & Q_{14}^1 Q_{16}^0 Q_{17}^0 Q_{18}^0 Q_{1,12}^0 + Q_{14}^1 Q_{16}^0 Q_{17}^0 Q_{18}^1 Q_{1,12}^1 + Q_{14}^1 Q_{16}^0 Q_{17}^1 Q_{18}^0 Q_{1,12}^1 + Q_{14}^1 Q_{16}^0 Q_{17}^1 Q_{18}^1 Q_{1,12}^0 \\ & Q_{14}^1 Q_{16}^1 Q_{17}^0 Q_{18}^0 Q_{1,12}^1 + Q_{14}^1 Q_{16}^1 Q_{17}^0 Q_{18}^1 Q_{1,12}^0 + Q_{14}^1 Q_{16}^1 Q_{17}^1 Q_{18}^0 Q_{1,12}^0 + Q_{14}^1 Q_{16}^1 Q_{17}^1 Q_{18}^1 Q_{1,12}^1 \\ & = 0.0020 \end{aligned} \tag{2.25}$$

O processo de troca de informação entre os nós pode ser observado na Figura 2.9

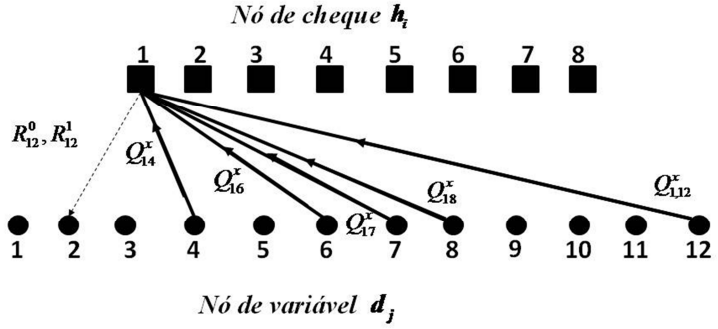


Figura 2.9 Cálculo de R_{12}^0 e R_{12}^1.

O nó que está sendo atualizado ou informado não participa da estimativa. Isto faz a iteração convergir para uma solução correta. O grande número "1"s em cada linha da matriz de verificação de paridade, implica na necessidade de grande número de combinações nos cálculos de R_{ij}^0 e R_{ij}^1. As Tabelas 2.2 e 2.3 mostram os valores destes coeficientes na forma de uma matriz, onde o índice *i* corresponde as linhas e *j* as colunas. A Tabela 2.2 representa os coeficientes R_{ij}^0 que são as estimativas para os bits quando *x = 0* e a Tabela 2.3 mostram os valores dos coeficientes de R_{ij}^1, isto é, as estimativas dos bits para quando *x = 1*.

Para tornar mais claro a notação, temos os valores de R_{ij}^{0} para os índices *i* = *4* e *j* = *10*, ou seja, $R_{4,10}^{0} = 0.0390$.

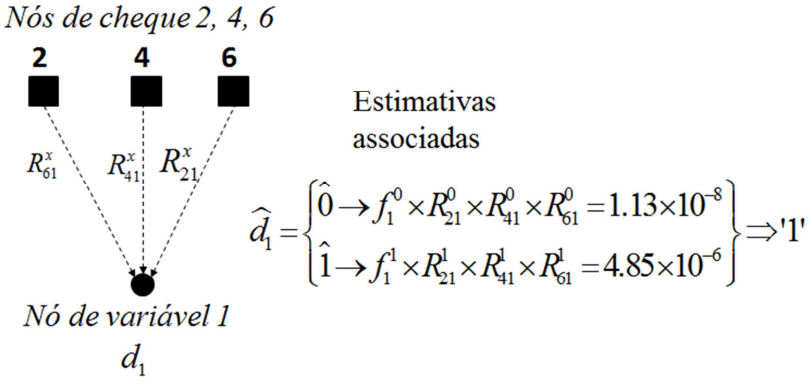
Tabela 2.2 Valores R_{ij}^{0} - primeira Iteração

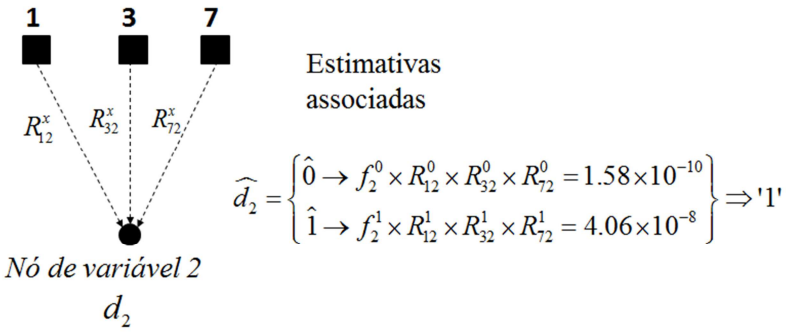
	1	2	3	4	5	6	7	8	9	10	11	12
1		**0.0051**		**0.0017**		**0.0002**	**0.0001**	**0.0004**				**0.0006**
2	**0.0360**		**0.0009**	**0.0113**					**0.0043**			
3		**0.0868**			**0.0109**		**0.0024**					**0.0100**
4	**0.0325**			**0.0889**						**0.0390**	**0.0088**	
5		**0.0875**			**0.0925**	**0.0423**				**0.1049**		
6	**0.0126**	**0.0100**					**0.0348**	**0.0431**			**0.0270**	
7		**0.0250**				**0.0008**		**0.0016**	**0.0035**	**0.0037**		
8					**0.1022**				**0.1101**		**0.0179**	**0.0912**

Tabela 2.3 Valores R_{ij}^{1}- primeira Iteração

	1	2	3	4	5	6	7	8	9	10	11	12
1		**0.0020**		**0.0070**		**0.0006**	**0.0008**	**0.0009**				**0.0002**
2	**0.0332**		**0.0324**	**0.0906**					**0.0372**			
3		**0.0393**			**0.0035**		**0.0128**					**0.0043**
4	**0.0107**			**0.0305**						**0.0028**	**0.0299**	
5			**0.0416**		**0.0398**	**0.8570**				**0.0333**		
6	**0.0295**						**0.0060**	**0.0188**			**0.0107**	
7		**0.0089**	**0.0280**			**0.0029**		**0.0046**	**0.0012**	**0.0003**		
8					**0.0101**				**0.0264**		**0.0908**	**0.0197**

Os valores dos coeficientes R_{ij}^{0} e R_{ij}^{1} nos permitem determinar a primeira estimativa do vetor código $\hat{\mathbf{d}}$. Assim, como exemplo, usando a equação (2.21) pode-se calcular os valores de $\widehat{d}_1$ e $\widehat{d}_2$





De forma análoga calculam-se as demais estimativas, ou seja:

$$\widehat{d_3} = \begin{cases} \hat{0} \to f_3^0 \times R_{23}^0 \times R_{53}^0 \times R_{63}^0 = 3.59\times10^{-8} \\ \hat{1} \to f_3^1 \times R_{23}^1 \times R_{53}^1 \times R_{63}^1 = 3.19\times10^{-7} \end{cases} \Rightarrow '1'$$

$$\widehat{d_4} = \begin{cases} \hat{0} \to f_4^0 \times R_{14}^0 \times R_{24}^0 \times R_{44}^0 = 3.31\times10^{-10} \\ \hat{1} \to f_4^1 \times R_{14}^1 \times R_{24}^1 \times R_{44}^1 = 3.19\times10^{-7} \end{cases} \Rightarrow '1'$$

$$\widehat{d_5} = \begin{cases} \hat{0} \to f_5^0 \times R_{35}^0 \times R_{55}^0 \times R_{85}^0 = 7.007\times10^{-6} \\ \hat{1} \to f_5^1 \times R_{35}^1 \times R_{55}^1 \times R_{85}^1 = 6.200\times10^{-7} \end{cases} \Rightarrow '0'$$

$$\widehat{d_6} = \begin{cases} \hat{0} \to f_6^0 \times R_{16}^0 \times R_{56}^0 \times R_{76}^0 = 3.39\times10^{-9} \\ \hat{1} \to f_6^1 \times R_{16}^1 \times R_{56}^1 \times R_{76}^1 = 5.34\times10^{-9} \end{cases} \Rightarrow '1'$$

$$\widehat{d_7} = \begin{cases} \hat{0} \to f_7^0 \times R_{17}^0 \times R_{37}^0 \times R_{67}^0 = 2.73\times10^{-9} \\ \hat{1} \to f_7^1 \times R_{17}^1 \times R_{37}^1 \times R_{67}^1 = 6.37\times10^{-9} \end{cases} \Rightarrow '1'$$

$$\widehat{d_8} = \begin{cases} \hat{0} \to f_8^0 \times R_{18}^0 \times R_{68}^0 \times R_{78}^0 = 7.96\times10^{-9} \\ \hat{1} \to f_8^1 \times R_{18}^1 \times R_{68}^1 \times R_{78}^1 = 1.03\times10^{-10} \end{cases} \Rightarrow '0'$$

$$\widehat{d_9} = \begin{cases} \hat{0} \to f_9^0 \times R_{29}^0 \times R_{79}^0 \times R_{89}^0 = 6.52\times10^{-9} \\ \hat{1} \to f_9^1 \times R_{29}^1 \times R_{79}^1 \times R_{89}^1 = 4.98\times10^{-7} \end{cases} \Rightarrow '1'$$

$$\widehat{d_{10}} = \begin{cases} \hat{0} \to f_{10}^0 \times R_{4,10}^0 \times R_{5,10}^0 \times R_{7,10}^0 = 1.62\times10^{-6} \\ \hat{1} \to f_{10}^1 \times R_{4,10}^1 \times R_{5,10}^1 \times R_{7,10}^1 = 1.17\times10^{-8} \end{cases} \Rightarrow '0'$$

$$\widehat{d_{11}} = \begin{cases} \hat{0} \to f_{11}^0 \times R_{4,11}^0 \times R_{6,11}^0 \times R_{8,11}^0 = 2.12\times10^{-6} \\ \hat{1} \to f_{11}^1 \times R_{4,11}^1 \times R_{6,11}^1 \times R_{8,11}^1 = 7.91\times10^{-7} \end{cases} \Rightarrow '0'$$

$$\widehat{d_{12}} = \begin{cases} \hat{0} \to f_{12}^0 \times R_{1,12}^0 \times R_{3,12}^0 \times R_{8,12}^0 = 9.23\times10^{-9} \\ \hat{1} \to f_{12}^1 \times R_{1,12}^1 \times R_{3,12}^1 \times R_{8,12}^1 = 8.87\times10^{-9} \end{cases} \Rightarrow '1'$$

A primeira estimativa do vetor código é $\hat{\mathbf{d}}$ = *(1 1 1 1 0 1 1 0 1 0 0 0)* que contém três erros com respeito a transmissão do vetor código ***c*** *= (1 1 1 1 1 0 0 0 1 0 0 0).* O processo de decodificação continua desde que ainda seja detectada a síndrome de erro no vetor estimado $\hat{\mathbf{d}}$.

A próxima iteração começa com o cálculo dos valores dos coeficientes Q_{ij}^0 e Q_{ij}^1. Estes valores são determinados usando a expressão (2.20), onde existe uma normalização dos coeficientes de modo que $Q_{ij}^0 + Q_{ij}^1 = 1$ é satisfeita. Assim, por exemplo, o valor do coeficiente Q_{12}^x para o qual *x = 0* ou *x = 1*, é a estimativa que o nó-símbolo 2 envia para o seu filho nó de cheque 1, calculado pelos produtos formados das estimativas R_{k2}^x de todos seus nós de cheque filhos, exceto o nó de cheque 1, o nó que esta sendo atualizado, onde *k* representa o índice dos nós filhos que participam da atualização.

Os valores dos coeficientes Q_{12}^0 e Q_{12}^1, neste exemplo, são calculados a seguir:

$$Q_{12}^{\ 0} = \alpha_{12} f_2^{\ 0} R_{32}^0 R_{72}^0 \ , \ Q_{12}^{\ 1} = \alpha_{12} f_2^{\ 1} R_{32}^1 R_{72}^1 \tag{2.26}$$

onde,

$$\alpha_{12} = \frac{1}{f_2^{\ 0} R_{32}^0 R_{72}^0 + f_2^{\ 1} R_{32}^1 R_{72}^1} \tag{2.27}$$

A Figura 2.10 mostra o fluxo de informações e o nós participantes nos cálculos dos coeficientes de Q_{12}^{0} e Q_{12}^{1} para este exemplo:

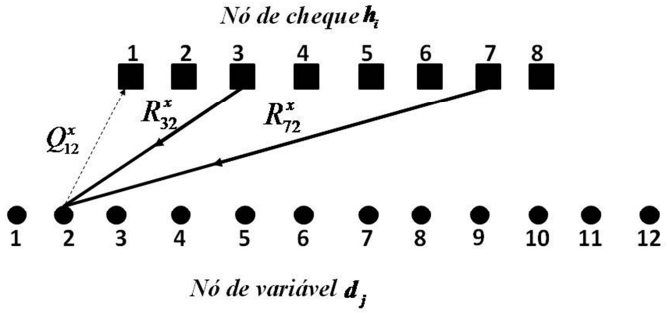


Figura 2.10 Cálculo de Q_{12}^{0} e Q_{12}^{1}.

As Tabelas 2.4 e 2.5 mostram os valores calculados dos coeficientes Q_{ij}^{0} e Q_{ij}^{1} respectivamente.

Tabela 2.4 Valores Q_{ij}^{0}- segunda Iteração

	1	2	3	4	5	6	7	8	9	10	11	12
1		**0.0015**		**0.0004**		**0.6601**	**0.7898**	**0.9950**				**0.2731**
2	**0.0209**		**0.0691**	**0.0083**					**0.1014**			
3		**0.0018**			**0.7843**		**0.6946**					**0.3068**
4	**0.0080**			**0.0004**						**0.9091**	**0.9008**	
5			**0.0010**		**0.8294**	**0.5620**				**0.9777**		
6	**0.0054**		**0.0056**				**0.0688**	**0.9710**			**0.5147**	
7		**0.0014**				**0.6847**		**0.9954**	**0.0046**	**0.9239**		
8					**0.5273**				**0.0031**		**0.9316**	**0.1838**

Tabela 2.5 Valores de Q_{ij}^{1} - segunda iteração

	1	2	3	4	5	6	7	8	9	10	11	12
1		**0.9985**		**0.9996**		**0.3399**	**0.2102**	**0.0050**				**0.7269**
2	**0.9791**		**0.9309**	**0.9917**					**0.8986**			
3		**0.0018**			**0.2157**		**0.3054**					**0.6932**
4	**0.992**			**0.9996**						**0.0909**	**0.0992**	
5			**0.9990**		**0.1706**	**0.4380**				**0.0223**		
6	**0.9946**		**0.9944**				**0.9312**	**0.0290**			**0.4853**	
7		**0.0014**				**0.3153**		**0.0046**	**0.9954**	**0.0761**		
8					**0.4727**				**0.9969**		**0.0684**	**0.8162**

Na segunda iteração, os valores atualizados de Q_{ij}^{0} e Q_{ij}^{1} nos permitem atualizar os valores de R_{ij}^{0} e R_{ij}^{1}. Esta iteração é diferente da primeira iteração no sentido que agora Q_{ij}^{0} e Q_{ij}^{1} contém informações atualizadas, e não simplesmente informação do canal. Os cálculos de R_{ij}^{0} e R_{ij}^{1} nesta segunda iteração nos conduzem a uma nova estimativa do vetor

código $\hat{\mathbf{d}}$, que novamente não satisfaz a condição de síndrome, e por isso o processo continua.

No exemplo, o decodificador é capaz de encontrar o vetor código correto após três iterações e, é capaz de corrigir também os dois erros que o vetor recebido contém por decisão difícil. Neste caso particular os erros estão na parte da mensagem do vetor código, isto é, em dois dos quatro bits de mensagem recebida, após truncamento e redundância. O algoritmo de decodificação interativo é capaz de corrigir esses dois erros. As Tabelas 2.6 e 2.7 ilustram a evolução do algoritmo de decodificação pelos valores presentes dos coeficientes envolvidos até a chegada da solução final.

Tabela 2.6 Valores R_{ij}^{0}- segunda iteração

	1	2	3	4	5	6	7	8	9	10	11	12
1		**0.5416**		**0.5415**		**0.3704**	**0.4284**	**0.4581**				**0.5915**
2	**0.1622**		**0.1244**	**0.1709**					**0.0940**			
3		**0.4572**			**0.5750**		**0.6095**					**0.3897**
4	**0.1723**			**0.1726**						**0.8999**	**0.9081**	
5			**0.5390**		**0.4409**	**0.1859**				**0.4593**		
6	**0.5115**						**0.5135**	**0.4876**			**0.1027**	
7		**0.3463**	**0.5118**			**0.9150**		**0.6547**	**0.3453**	**0.6808**		
8					**0.7713**				**0.4851**		**0.5172**	**0.4766**

Tabela 2.7 Valores R_{ij}^{1}- segunda iteração

	1	2	3	4	5	6	7	8	9	10	11	12
1		**0.4584**		**0.4585**		**0.6296**	**0.5716**	**0.5419**				**0.4085**
2	**0.8378**		**0.8756**	**0.8291**					**0.0960**			
3		**0.5428**			**0.4250**		**0.3905**					**0.6103**
4	**0.8278**			**0.8274**						**0.1001**	**0.0919**	
5			**0.4610**		**0.5591**	**0.8141**				**0.5407**		
6	**0.4882**						**0.4865**	**0.5124**			**0.8973**	
7		**0.6537**	**0.4882**			**0.0850**		**0.3453**	**0.6547**	**0.3192**		
8					**0.2287**				**0.5149**		**0.4828**	**0.5234**

A atualização dos valores de R_{ij}^{0} e R_{ij}^{1} nos permite novamente uma estimativa para o vetor código $\hat{d}$. A Tabela 2.8 nos mostra uma segunda estimativa para vetor código $\hat{d}$

Tabela 2.8 Estimativa do vetor código a segunda iteração

	1	2	3	4	5	6	7	8	9	10	11	12
r	**+1.3129**	**+2.6582**	**+0.7413**	**+2.1745**	**+0.5981**	**-0.3962**	**-0.3962**	**+1.7585**	**+1.4905**	**+0.4084**	**-0.9290**	**+1.0765**
t	**+1**	**+1**	**+1**	**+1**	**+1**	**-1**	**-1**	**-1**	**+1**	**-1**	**-1**	**-1**
d_j^0	**0.0001**	**0.0000**	**0.0016**	**0.0000**	**0.0133**	**0.0307**	**0.0503**	**0.0465**	**0.0001**	**0.0298**	**0.0240**	**0.0019**
d_j^1	**0.1564**	**0.0095**	**0.0933**	**0.0534**	**0.0239**	**0.0016**	**0.0118**	**0.0001**	**0.1262**	**0.0066**	**0.0011**	**0.0648**

O vetor código estimado é $\hat{\mathbf{d}}$ = (1 1 1 1 1 0 0 0 1 0 0 1) , que contém apenas um erro, no último bit. Este vetor estimado produz uma síndrome não-zero de modo que o processo prossegue para uma terceira iteração. Os valores de Q_{ij}^0 e Q_{ij}^1 , R_{ij}^0 e R_{ij}^1 são novamente atualizados e esta terceira iteração é apresentados nas Tabelas 2.9 e 2.10, 2.11 e 2.12

Tabela 2.9 Valores Q_{ij}^0 - terceira iteração

	1	2	3	4	5	6	7	8	9	10	11	12
1		**0.0001**		**0.0000**		**0.9707**	**0.8503**	**0.9977**				**0.0197**
2	**0.0036**		**0.1078**	**0.0003**					**0.0047**			
3		**0.0002**			**0.2909**		**0.7318**					**0.0436**
4	**0.0033**			**0.0003**						**0.3358**	**0.6910**	
5			**0.0145**		**0.4130**	**0.9884**				**0.8426**		
6	**0.0007**						**0.8013**	**0.9974**			**0.9948**	
7		**0.0002**	**0.0161**			**0.6442**		**0.9949**	**0.0009**	**0.6807**		
8					**0.1413**				**0.0005**		**0.9538**	**0.0310**

Tabela 2.10 Valores Q_{ij}^1 - terceira iteração

	1	2	3	4	5	6	7	8	9	10	11	12
1		**0.9999**		**1.0000**		**0.0293**	**0.1497**	**0.0023**				**0.9803**
2	**0.9964**		**0.8922**	**0.9997**					**0.9953**			
3		**0.9998**			**0.7091**		**0.2682**					**0.9564**
4	**0.9967**			**0.9997**						**0.6642**	**0.3090**	
5			**0.9855**		**0.5870**	**0.0116**				**0.1574**		
6	**0.9993**						**0.1987**	**0.0026**			**0.0052**	
7		**0.9998**	**0.9839**			**0.3558**		**0.0051**	**0.9991**	**0.3193**		
8					**0.3558**				**0.9995**		**0.0462**	**0.9690**

Tabela 2.11 Valores R_{ij}^0 - terceira iteração

	1	2	3	4	5	6	7	8	9	10	11	12
1		**0.8153**		**0.8153**		**0.1651**	**0.0501**	**0.1833**				**0.8282**
2	**0.1117**		**0.0085**	**0.1143**					**0.1109**			
3		**0.5885**			**0.7115**		**0.3092**					**0.5969**
4	**0.5627**			**0.5623**						**0.6896**	**0.3370**	
5			**0.4418**		**0.1750**	**0.5579**				**0.5825**		
6	**0.2129**						**0.9758**	**0.7882**			**0.7897**	
7		**0.4485**	**0.2037**			**0.6784**		**0.5520**	**0.4485**	**0.6424**		
8					**0.9252**				**0.8053**		**0.1639**	**0.8252**

Tabela 2.12 Valores R_{ij}^1 - terceira iteração

	1	2	3	4	5	6	7	8	9	10	11	12
1		**0.1847**		**0.1847**		**0.8349**	**0.9499**	**0.8167**				**0.1718**
2	**0.8883**		**0.9915**	**0.8857**					**0.8891**			
3		**0.4115**			**0.2885**		**0.6908**					**0.4031**
4	**0.4373**			**0.4377**						**0.3104**	**0.6630**	
5			**0.5582**		**0.8250**	**0.4421**				**0.4175**		
6	**0.7871**						**0.0242**	**0.2118**			**0.2103**	
7		**0.5515**	**0.7963**			**0.3216**		**0.4480**	**0.5515**	**0.3576**		
8					**0.0748**				**0.1947**		**0.8361**	**0.1748**

Com estes valores uma nova estimativa do vetor $\hat{\mathbf{d}}$ é formada. De acordo com a Tabela 2.13 o vetor código após esta terceira iteração é $\hat{\mathbf{d}} = \boldsymbol{c}$ = (1 1 1 1 1 0 0 0 1 0 0 0), cuja síndrome é o vetor nulo, de modo que o decodificador decide que este é o vetor código para a mensagem m = (1 0 0 0).

Tabela 2.13 Estimativa do vetor código após a terceira iteração

	1	2	3	4	5	6	7	8	9	10	11	12
r	**+1.3129**	**+2.6582**	**+0.7413**	**+2.1745**	**+0.5981**	**-0.3962**	**-0.3962**	**+1.7585**	**+1.4905**	**+0.4084**	**-0.9290**	**+1.0765**
t	**+1**	**+1**	**+1**	**+1**	**+1**	**-1**	**-1**	**-1**	**+1**	**-1**	**-1**	**-1**
d_j^0	**0.0001**	**0.0000**	**0.0000**	**0.0000**	**0.0077**	**0.0305**	**0.0057**	**0.0254**	**0.0002**	**0.0273**	**0.0217**	**0.0070**
d_j^1	**0.1412**	**0.0024**	**0.2086**	**0.0122**	**0.0078**	**0.0043**	**0.0017**	**0.0001**	**0.0394**	**0.0176**	**0.0032**	**0.0060**

2.3 O Algoritmo corretor de apagamento

No canal com apagamento (BEC), um bit é recebido corretamente ou completamente apagado (perdido) com a mesma probabilidade ε. Desde que os bits recebidos sejam sempre corrigidos cabe ao decodificador determinar o valor dos bits desconhecidos.

O processo de decodificação usando o algorítmo corretor de apagamento é análogo ao algoritmo SPA [33]. Na decodificação da informação recebida, cada nó de verificação pode determinar o valor de um bit apagado se ele for o único bit apagado na sua equação de paridade. Um nó de variável envia a mesma mensagem de saída para cada um dos seus nós de cheque conectados. Esta mensagem, chamadas M_i para o i-ésimo bit nó, declara o valor do bit como 1 ou 0 se o valor é conhecido ou Δ se estiver apagado. Se um nó de verificação recebe apenas um Δ na mensagem, pode-se calcular este valor escolhendo o valor que satisfaz na equação de paridade.

Os nós de cheque enviam de volta mensagens diferentes para cada um dos seus nó de variáveis conectados. A mensagem $E_{j,i}$, do j-ésimo nó de cheque para o i-ésimo nó de variáveis declara o valor do bit como 1, 0 ou Δ se o bit for desconhecido ou apagado. Se o nó de variáveis de um bit apagado recebe uma mensagem que é 1 ou 0, o nó de variáveis altera o valor para o valor da mensagem de entrada. Em uma iteração da mensagen recebida os valores dos nós de cheque e dos nós de variáveis são atualizados

e processo é repetido até que todos os valores de bit sejam conhecidos ou até que um número máximo de iterações seja atingido.

Exemplo 2.5. Considere a matriz de verificação de paridade **H** a seguir

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

Usando a notação B_j para representar o conjunto de bits na j-ésima cheque equação de paridade de **H** tem-se:

$$B_1 = \{1,2,4\}, \quad B_2 = \{2,3,5\}, \quad B_3 = \{1,5,6\}, \quad B_4 = \{3,4,6\}$$

Da mesma forma, usando notação A_i para representar as equações de paridade para o i-ésimo bit do código tem-se:

$$A_1 = \{1,3\}, \quad A_2 = \{1,2\}, \quad A_3 = \{2,3\},$$
$$A_4 = \{1,4\} \quad A_5 = \{2,3\} \quad A_6 = \{3,4\}$$

Supondo-se que a palavra código $c = [0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 1]$ seja transmitida e detectado o vetor recebido $y = [0 \quad 0 \quad 1 \quad \Delta \quad \Delta \quad \Delta]$. O algoritmo a seguir esboça o processo de reconstrução da mensagem enviada através do BEC. A Figura 2.11 mostra graficamente este processo.

.

Algoritmo: Decodificador de apagamentos (decoding Erasure)

1. procedimento decodificar $(y, I_{\max})$
2. para $i = 1:N$ faça %inicialização
3. $M_i = y_i$
4. Fim para
5. $l = 0$ % contador de iteração
6. Repita enquanto $l < I_{\max}$
7. para $j = 1:m$ faça % passo1: checar as mensagens
8. para todo $i \in B_j$ faça
9. Se todas as mensagens do nó de cheque j, diferente de M_i, são conhecidos, então

10. $E_{j,i} = \sum_{i' \in B_{j,i' \neq i}} (M_{i'} \bmod 2)$
11. Se não
12. $E_{j,i} = \Delta$
13. Fim se
14. Fim para
15. Fim para
16. Para $i = 1 : N$ faça % passo2: bit mensagens
17. Se $M_i = \Delta$ então
18. Se existe um $j \in A_i$ tal que $E_{j,i} \neq \Delta$ então
19. $M_i = E_{j,i}$
20. Fim se
21. Fim se
22. Fim Para
23. Se todos M_i são conhecidos ou $l = I_{\max}$ então % critério de parada
24. Finalizar
25. Se não
26. $l = l + 1$
27. Fim se
28. Finalizar repetição
29. Fim do procedimento

.

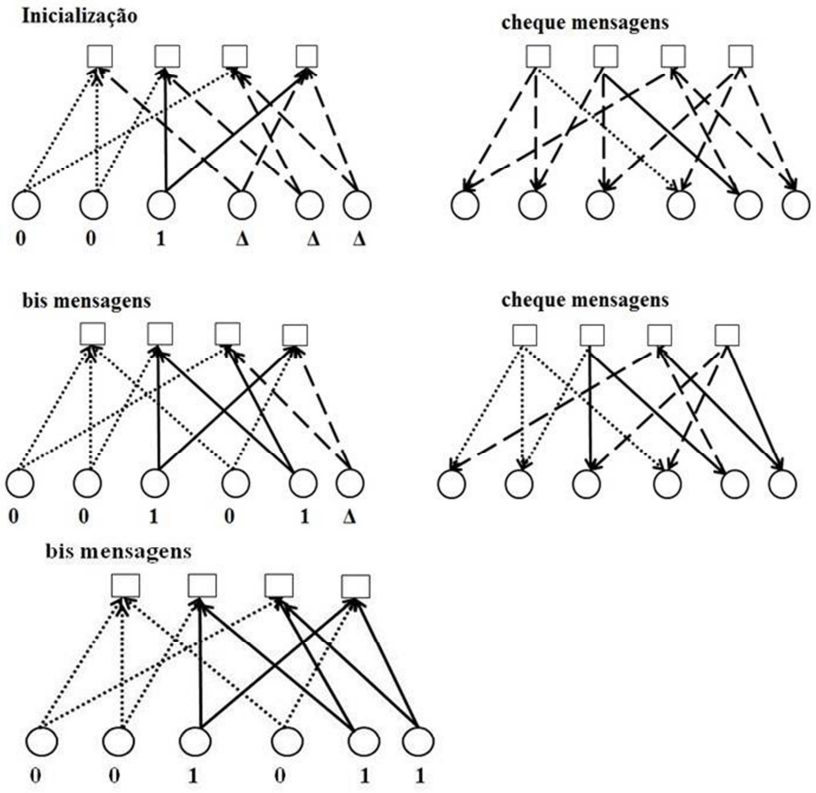


Figura 2.11: Decodificação do vetor recebido y = [0 0 1 Δ Δ Δ]. Cada sub-figura indica a decisão feita no passo do algoritmo de decodificação com base em mensagens de passos anteriores. As setas com pontos correspondem a mensagens "bit = 0", enquanto que as setas contíuas correspondem a "bit = 1" e as setas a tracejadas correspondem a "bit = Δ ".

Na inicialização tem-se $M_i = y_i$ i, então

$$M = [0 \quad 0 \quad 1 \quad \Delta \quad \Delta \quad \Delta]$$

No Passo 1 do algoritmo as mensagens do nó de cheque são calculados. O primeiro nó de cheque está unido ao primeiro, segundo e quarto nó de variável e assim tem as mensagens de entrada 0, 0 e Δ. Uma vez que este cheque nó tem um Δ na mensagem de entrada, a partir do quarto bit, pode-se calcular o valor de Δ, fazendo a soma $M_1 \oplus M_2$, e transmitir o resultado como a mensagem de saída para $E_{1,4}$, isto é,

$$\begin{aligned} E_{1,4} &= M_1 \oplus M_2 \\ &= 0 \oplus 0 \\ &= 0 \end{aligned}$$

O segundo nó de cheque está unido ao segundo, terceiro e quinto nó de variável e assim tem as mensagens de entrada 0, 1 e Δ. Uma vez que este nó de cheque tem uma entrada *e* na mensagem, a partir do quinto bit, a sua mensagem de saída é $E_{2,5}$ e pode ser calculada como:

$$\begin{aligned} E_{2,5} &= M_2 \oplus M_3 \\ &= 0 \oplus 1 \\ &= 1 \end{aligned}$$

O terceiro nó de cheque esá unido ao primeiro, quinto e sexto nó de variável e assim tem as mensagens de entrada 0, Δ e Δ. Como este nó de cheque recebe duas mensagens apagadas, ele não pode determinar o valor de qualquer bit. Neste caso, as mensagens enviadas por este nó também são Δ. Da mesma forma, esta será a decisão do quarto nó de cheque que inclui o terceiro, quarto e sexto nó de variável.

No Passo 2 cada nó de variável que tem um valor desconhecido na mensagens recebidas atualiza-o se possível. O quarto bit (nó de variável) é atualisado (subistituido) pela soma binária calculada por $E_{1,4}$, isto é, 0. O quinto bit também é desconhecido e tem sua atualização dada por $E_{2,5}$, isto é, 1. O sexto bit também é desconhecido, mas não pode ser atualiado na iteração atual, pois em sua soma binária aparecem $(0 + \Delta)$ dada por $E_{3,6}$ ou $(1 + \Delta)$ dada por $E_{4,6}$. No final do passo 2 tem-se:

$$M = [0 \quad 0 \quad 1 \quad 0 \quad 1 \quad \Delta]$$

Existe um bit restante desconhecido (o sexto bit) de modo que o algoritmo continua.

Repetindo o passo 1, o terceiro nó de cheque está unido ao primeiro, quinto e sexto nó de variável e assim este nó possui uma entrada apagada (M_6) "Δ". A mensagem enviada a partir deste nó de cheque é dada por:

$$\begin{aligned} E_{3,6} &= M_1 \oplus M_5 \\ &= 1 \oplus 0 \\ &= 1 \end{aligned}$$

O quarto nó de cheque está unido ao teceiro, quarto e sexto nó de variável de modo que ele também possui uma entrada apagada (M_6) "Δ". A mensagem enviada a partir deste nó de cheque é dada por:

$$\begin{aligned} E_{3,6} &= M_3 \oplus M_4 \\ &= 0 \oplus 1 \\ &= 1 \end{aligned}$$

Na repetição do passo 2 o sexto bit é desconhecida, mas tem entradas $E_{3,6}$, e $E_{4,6}$ com o valor 1, e, portanto, altera o seu valor para 1. Note que, uma vez que os bits recebidos do BEC são sempre corretos, as mensagens dos nós de cheque sempre concordam. Desta vez, no teste não bits da palavra código desconhecida, o algoritmo para e retorna

$$\hat{c} = M = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

como palavra código decodificada. O vetor recebido foi, portanto, determinado corretamente, embora alguns bits de palavra código tenham sido apagados.

Capítulo 3

Métodos para construção de matrizes de verificação de paridade baseados em concatenação de matrizes bases e superposição de matrizes circulantes

Neste Capítulo são apresentados dois métodos usados para geras as matrizes de verificação de paridade utilizadas na correção de apagamentos em rajadas de códigos LDPC. Os métodos geram as matrizes de verificação de paridade, 4-ciclos livre, por superposição e concatenação de matrizes bases coluna circulante peso 2. O primeiro método gera matrizes de verificação de paridade baseadas em (B.13) e o segundo método gera matrizes a partir de (B. 17).

3.1 Matrizes bases, matriz plataforma e matrizes clientes circulantes

A matriz base é uma matriz binária esparsa utilizada para a superposição de outras matrizes (as clientes) no processo de construção da matriz de verificação de paridade de um código [44]. A matriz plataforma é uma matriz auxiliar que serve para descrever de forma abreviada a superposição na matriz base. Neste trabalho, as submatrizes serão chamadas de clientes. A matriz plataforma é desenvolvida a partir da composição de duas matrizes clientes: a matriz circulante e a matriz nula ambas de ordem v. As matrizes clientes não nulas são todas matrizes circulantes.

Na correção de apagamentos em rajadas a matriz de verificação de paridade é o elemento fundamental para determinar o $L_{\max}$ [45]. Por definição, $L_{\max}$ é o número de bits apagados que podem ser recuperados independentemente de sua posição dentro da palavra código [46]. A *eficiência*, E_f, de um código de comprimento *n* e taxa *k/n* é definida por:

$$\mathrm{E_f} = \frac{L_{\max}}{n-k} \tag{3.1}$$

Yang e Ryan [36] desenvolveram um eficiente e exaustivo algoritmo para determinar o $L_{\max}$. Em [23], são apresentados dois métodos algébricos para determinar a matriz de verificação de paridade na decodificação de códigos QC-LDPC para AWGN e em canais BEC com ruído branco aditivo (BEC- AWGN). A construção da matriz de paridade é desenvolvida a partir do Campo de Galos e elementos da geometria Euclidiana.

Matrizes de verificação de paridade compostas por matrizes circulantes têm sido estudadas por vários autores para projetar códigos LDPC corretores de apagamentos em rajadas com boa eficiência [22-23]. Em [26] são utilizadas matrizes base circulantes na construção da matriz cheque que apresentam um bom desempenho na correção de apagamentos em rajadas.

Uma matriz quadrada de ordem *n* é **circulante**, se os elementos de uma coluna (ou linha) também são os elementos da coluna (ou linha) anterior, mas deslocados de uma posição à direita [43], isto é:

$$\mathbf{C} = \begin{pmatrix} c_1 & c_n & \dots & c_2 \\ c_2 & c_1 & \dots & c_3 \\ \vdots & \vdots & & \\ c_n & c_{n-1} & \cdots & c_1 \end{pmatrix} \tag{3.2}$$

A representação da matriz circulante pode ser feita pela primeira coluna, $\mathbf{C} = (c_1, c_2, \ldots, c_n)$, chamada de geradora. Quando $c_1 = 1$ e os demais valores são nulos, a circulante C é matriz identidade de ordem *n*. Assim, o vetor $\mathbf{I}_n = (1, 0, ..., 0)$ será o gerador da matriz identidade de ordem *n*. Com a notação $\mathbf{I}_v^{(m)}$, com $m \in \mathbb{N}$, descrevem-se *m* deslocamentos ou "movimentos" mod(*v*) dos elementos da matriz circulante identidade $\mathbf{I}_v$. A notação $\mathbf{I}_v^{(2)}, \mathbf{I}_v^{(3)}, \mathbf{I}_v^{(4)}, ...$ representa dois, três, quatro, etc... movimentos mod(*v*) da matriz $\mathbf{I}_v$. A matriz $\mathbf{O}_v$ representa a matriz nula de ordem *v*. A matriz $\mathbf{I}_8^{(6)}$ a seguir representa uma matriz circulante identidade com dimensão 8 e movimento *m*=6. Os elementos nulos foram desprezados.

$$\mathbf{I}_8^{(6)} = \begin{bmatrix} & & & & & & & 1 \\ & & & & & & 1 & \\ & & & & & 1 & & \\ & & & & 1 & & & \\ & & & 1 & & & & \\ & & 1 & & & & & \\ 1 & & & & & & & \\ & 1 & & & & & & \end{bmatrix}_{8\times 8}$$

Exemplo 3.1: Seja $\mathbf{H}_p$ a matriz plataforma construída a partir da superposição aleatória da matriz cliente gerada por $\mathbf{I}_8$ = [1 0 0 0 0 0 0] na matriz base $\mathbf{H}_b$ de dimensão 5 abaixo.

$$\mathbf{H}_b = \begin{bmatrix} 1 & & & & \\ 1 & 1 & & & \\ & 1 & 1 & & \\ & & 1 & 1 & \\ & & & 1 & 1 \end{bmatrix}_{5\times 5}$$

Na primeira diagonal de $\mathbf{H}_b$, os elementos não nulos "recebem' as matrizes clientes de movimentos 2,3,4,5,6 $(m \in \{2,3,4,5,6\})$, as matrizes $\mathbf{I}^{(2)}$, e $\mathbf{I}^{(6)}$, representam os movimentos 2mod8 e 6mod8 circulantes de I_8. Abaixo, temos a matriz plataforma $\mathbf{H}_p$ formada por esta superposição em $\mathbf{H}_b$. Os elementos nulos foram desprezados.

$$\mathbf{H}_p = \begin{bmatrix} \mathbf{I}^{(2)} & & & & \\ \mathbf{I} & \mathbf{I}^{(3)} & & & \\ & \mathbf{I} & \mathbf{I}^{(4)} & & \\ & & \mathbf{I} & \mathbf{I}^{(5)} & \\ & & & \mathbf{I} & \mathbf{I}^{(6)} \end{bmatrix}_{40\times 40}$$

3.2 Métodos para construção de matrizes de verificação de paridade por concatenação de matrizes bases

A decodificação iterativa usando códigos LDPC em canais com apagamentos é particularmente simples, uma vez que um bit transmitido é recebido corretamente ou apagado [26]. Se apenas um dos bits é apagado, ele pode ser recuperado pela escolha do valor que satisfaz na equação de paridade da palavra código. O algoritmo de decodificação soma e produto (SPA) de um código LDPC é um procedimento para encontrar equações teste paridade que verifica se apenas um bit foi apagado; as equações teste paridade são determinadas e os bits apagados recuperados. Após este bit

ter sido corrigido, qualquer nova equação controla outro bit apagado e o recupera na iteração subsequente. O procedimento é repetido até que todos os apagamentos sejam corrigidos ou todas as equações teste paridade restantes chequem dois ou mais bits apagados, o que ocorre se os bits apagados incluem um conjunto de bits conhecidos como os *stoppings set* .

Seja *V* o conjunto dos nós de variáveis. Um *stopping set* é um subconjunto *T* de *V*, tal que qualquer nó de cheque está conectado a ele pelo menos duas vezes [48]. O subconjunto $T=\{v_1,v_2,v_3,v_4\}$ (Figura 3.1) é um *stopping set* do conjunto $V=\{v_1,v_2,v_3,v_4,v_5,v_6,v_7,v_8,v_9,v_{10}\}$.

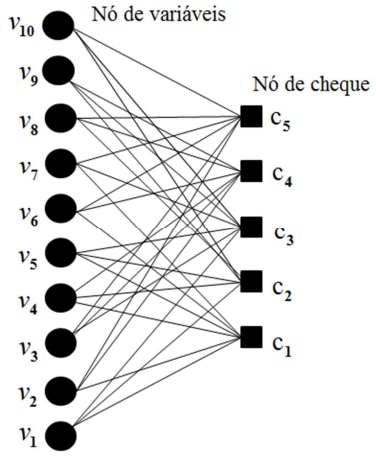


Figura 3.1 Stopping set formado pelo conjunto $\{v_1,v_2,v_3,v_4\}$

Se todos os bits em um *stopping set* são apagados nenhum deles pode ser recuperado a partir da mensagem recebida, de modo que a distribuição dos stopping set de um código LDPC determina o processo de falha no algoritmo SP [48]. O comprimento de um *stopping set* é o número de bits incluídos nele, e o menor comprimento dos *stopping set*, $S_{\min}$, de uma matriz de paridade é o número mínimo de bits apagados que podem causar uma falha na decodificação [26]. No BEC o número e comprimento dos *stopping set* podem ser usados para predizer diretamente o desempenho da decodificação da mensagem transmitida [48]. No canal BEC, a localização dos bits *stopping set* da palavra código, será um importante fator para determinar o desempenho da decodificação.

Em geral, o $S_{\min}$ de um código coluna-circulante é limitado a 2γ, onde γ é o peso da coluna das circulantes [26]. Devido a esta limitação, códigos LDPC coluna-circulante tem desempenho melhor quando o código LDPC tem comprimento grande e são aleatoriamente construídos. Para se construir códigos LDPC que tenham um bom desempenho na correção de apagamentos e sem esta limitação, serão utilizadas as matrizes de verificação de paridade compostas por concatenações de matrizes bases na forma de (B.13) e (B.17). A seguir serão apresentados dois métodos propostos para construir matrizes de verificação de paridade utilizando as matrizes bases desenvolvidas no Apêndice B.

Seja *v* um número inteiro e *m* um conjunto de índices. As matrizes 0_v e I_v^m denotam, respectivamente, as matrizes nulas e identidade circulantes de movimento *m*, cujas dimensões são *v*.

MÉTODO-1 Este método constrói matriz de verificação de paridade de dimensão *Npv* e taxa $\approx (p-1)/p$ a partir da concatenação das *p* cópias da matriz base **B**=[1 0 0...0 1;1 1 0...0 0;0 1 1...0;0 0 0...1 1] definida em (B.13) de dimensão *N*. Tomando-se uma sequência $\{\mathbf{B}_1, \mathbf{B}_2, ..., \mathbf{B}_p\}$ de *p* cópias de **B**, a primeira plataforma $\mathbf{H}_1$ é criada por superposição em $\mathbf{B}_1$, a segunda $\mathbf{H}_2$ por superposição em $\mathbf{B}_2$ e assim por diante.

ALGORITMO-1

Passo 1: Considere $\{\mathbf{B}_1, \mathbf{B}_2, ..., \mathbf{B}_p\}$ cópias da matriz **B**.

Passo 2: Construção da matriz plataforma $\mathbf{H}_1$:

A) Substitua a diagonal principal de $\mathbf{B}_1$ substitua por $\mathbf{I}_v$

B) Na diagonal abaixo da principal de $\mathbf{B}_1$ substitua por $\mathbf{I}_v$ e faça $\mathbf{I}_v^{(1)}$ em ao menos um dos elementos

Passo 3: Construção das matrizes plataformas $\mathbf{H}_2, ..., \mathbf{H}_p$:

a) Nas diagonais principais de $\mathbf{H}_2, ..., \mathbf{H}_p$ sobreponha $\mathbf{I}_v^m$ escolhendo *m* de modo que a soma dos movimentos das $\mathbf{I}_v^m$ sejam múltipla de *N*.

b) Nas diagonais abaixo da diagonal principal repita o passo 2.b

Passo 4: Sobreponha a matriz $\mathbf{O}_v$ nos elementos nulos de $\{\mathbf{B}_1, \mathbf{B}_2, ..., \mathbf{B}_p\}$

Passo 5: Nos segundos elementos não nulos da primeira linha das matrizes $\{\mathbf{B}_1, \mathbf{B}_2, ..., \mathbf{B}_p\}$ sobreponha $\mathbf{I}_v^l$ com qualquer movimento *l*.

Passo 6: Construa a matriz de verificação de paridade **H** de 4-ciclo livre, dimensão $N \times pN$ e taxa $\approx (p-1)/p$ concatenando $\{\mathbf{H}_1, \mathbf{H}_2, ..., \mathbf{H}_p\}$

NOTA: A matriz $\mathbf{I}_v^l$ do Passo 5 será chamada *matriz de movimento livre* ou *elemento livre* das matrizes $\{\mathbf{H}_1, \mathbf{H}_2, ..., \mathbf{H}_p\}$.

Exemplo 3.2: *Seja* $\boldsymbol{H}_i$ *com* $\{1 \le i \le 7\}$ *matrizes plataformas para N=5. A notação* $D_1D_2D_3D_4D_5$ *representa a diagonal principal de* $\boldsymbol{H}_i$ *enquanto* $S_1S_2S_3S_4$ *a diagonal abaixo da principal, F indica o elemento livre de* $\boldsymbol{H}_i$*. Os elementos nulos foram desprezados. A Tabela 3.3 mostra alguns movimentos propostos na matriz cliente I.*

$$\mathbf{H}_i = \begin{bmatrix} D_1 & & & & F \\ S_1 & D_2 & & & \\ & S_2 & D_3 & & \\ & & S_3 & D_4 & \\ & & & S_4 & D_5 \end{bmatrix}$$

Tabela 3.3: Movimento das matrizes clientes

Matriz Plataforma	Diagonal Principal D_1, D_2, D_3, D_4, D_5	Segunda Diagonal S_1,S_2,S_3,S_4	Elemento Livre (F)
$\mathbf{H}_1$	$\mathbf{I}, \mathbf{I}, \mathbf{I}, \mathbf{I}, \mathbf{I}$	$\mathbf{I}, \mathbf{I}^{(1)}, \mathbf{I}, \mathbf{I}$	$\mathbf{I}$
$\mathbf{H}_2$	$\mathbf{I}^{(2)}, \mathbf{I}^{(3)}, \mathbf{I}^{(4)}, \mathbf{I}^{(5)}, \mathbf{I}^{(6)}$	$\mathbf{I}, \mathbf{I}, \mathbf{I}^{(1)}, \mathbf{I}$	$\mathbf{I}$
$\mathbf{H}_3$	$\mathbf{I}^{(2)}, \mathbf{I}^{(3)}, \mathbf{I}^{(4)}, \mathbf{I}^{(5)}, \mathbf{I}^{(6)}$	$\mathbf{I}, \mathbf{I}, \mathbf{I}^{(1)}, \mathbf{I}^{(1)}$	$\mathbf{I}$
$\mathbf{H}_4$	$\mathbf{I}^{(6)}, \mathbf{I}^{(5)}, \mathbf{I}^{(4)}, \mathbf{I}^{(3)}, \mathbf{I}^{(2)}$	$\mathbf{I}, \mathbf{I}, \mathbf{I}^{(1)}, \mathbf{I}^{(1)}$	$\mathbf{I}^{(5)}$
$\mathbf{H}_5$	$\mathbf{I}^{(8)}, \mathbf{I}^{(6)}, \mathbf{I}^{(5)}, \mathbf{I}^{(4)}, \mathbf{I}^{(2)}$	$\mathbf{I}, \mathbf{I}, \mathbf{I}^{(1)}, \mathbf{I}^{(1)}$	$\mathbf{I}^{(6)}$
$\mathbf{H}_6$	$\mathbf{I}^{(9)}, \mathbf{I}^{(7)}, \mathbf{I}^{(5)}, \mathbf{I}^{(3)}, \mathbf{I}^{(1)}$	$\mathbf{I}, \mathbf{I}, \mathbf{I}, \mathbf{I}^{(1)}$	$\mathbf{I}$
$\mathbf{H}_7$	$\mathbf{I}^{(7)}, \mathbf{I}^{(5)}, \mathbf{I}^{(4)}, \mathbf{I}^{(3)}, \mathbf{I}^{(1)}$	$\mathbf{I}, \mathbf{I}^{(1)}, \mathbf{I}, \mathbf{I}$	$\mathbf{I}$

As matrizes $\mathbf{H}_1$, $\mathbf{H}_2$, $\mathbf{H}_3$ e $\mathbf{H}_4$ são mostradas a seguir.

$$\mathbf{H}_1 = \begin{bmatrix} \mathrm{I} & & & & & \mathrm{I} \\ \mathrm{I} & \mathrm{I} & & & & \\ & \mathrm{I}^{(1)} & \mathrm{I} & & & \\ & & \mathrm{I} & \mathrm{I} & & \\ & & & \mathrm{I} & \mathrm{I} & \end{bmatrix}, \qquad \mathbf{H}_2 = \begin{bmatrix} \mathrm{I}^{(2)} & & & & & \mathrm{I} \\ \mathrm{I} & \mathrm{I}^{(3)} & & & & \\ & \mathrm{I} & \mathrm{I}^{(4)} & & & \\ & & \mathrm{I}^{(1)} & \mathrm{I}^{(5)} & & \\ & & & \mathrm{I} & \mathrm{I}^{(6)} & \end{bmatrix},$$

$$\mathbf{H}_3 = \begin{bmatrix} \mathrm{I}^{(2)} & & & & & \mathrm{I} \\ \mathrm{I} & \mathrm{I}^{(3)} & & & & \\ & \mathrm{I} & \mathrm{I}^{(4)} & & & \\ & & \mathrm{I}^{(1)} & \mathrm{I}^{(5)} & & \\ & & & \mathrm{I}^{(1)} & \mathrm{I}^{(6)} & \end{bmatrix}, \quad \mathbf{H}_4 = \begin{bmatrix} \mathrm{I}^{(6)} & & & & & \mathrm{I}^{(5)} \\ \mathrm{I} & \mathrm{I}^{(5)} & & & & \\ & \mathrm{I} & \mathrm{I}^{(4)} & & & \\ & & \mathrm{I}^{(1)} & \mathrm{I}^{(3)} & & \\ & & & \mathrm{I}^{(1)} & \mathrm{I}^{(2)} & \end{bmatrix}$$

Pelo método-1, como exemplo, podem-se compor matrizes de verificação de paridade concatenando as matrizes bases $\mathbf{H}_1\mathbf{H}_2$; $\mathbf{H}_1\mathbf{H}_3$ e $\mathbf{H}_1\mathbf{H}_4$. As matrizes $\mathbf{H}_{c1}, \mathbf{H}_{c2}$, $\mathbf{H}_{c3}$ são exemplos de matrizes de verificação de paridade com taxa =1/2 e p = 2, onde, $\mathbf{H}_{c1} = [\mathbf{H}_1\mathbf{H}_2]$, $\mathbf{H}_{c2} = [\mathbf{H}_1\mathbf{H}_3]$ e $\mathbf{H}_{c3} = [\mathbf{H}_1\mathbf{H}_4]$.

Para melhor compreensão do método, é utilizada a matriz $\mathbf{I}_8$ (v=8) como cliente e m=2 e 3 para designar os movimentos circulantes dos 1's de I_8 ($\mathbf{I}^{(2)}$, $\mathbf{I}^{(3)}$), temos:.

$$\mathbf{I} = \mathbf{I}_8 = \begin{bmatrix} 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 \end{bmatrix}_{8\times 8}, \quad \mathbf{I}^{(2)} = \mathbf{I}_8^{(2)} = \begin{bmatrix} & & & & & & 1 & \\ & & & & & & & 1 \\ 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \end{bmatrix}_{8\times 8},$$

$$\mathbf{I}^{(3)} = \mathbf{I}_8^{(3)} = \begin{bmatrix} & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & 1 & \\ 1 & & & & & & & \\ & 1 & & & & & & \\ & & 1 & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \end{bmatrix}_{8\times 8}$$

Em seguida temos superposição das matrizes clientes na matriz base (B.17) para formar $\mathbf{H}_2$

$$\mathbf{H}_b = \begin{bmatrix} 1 & & & & & 1 \\ 1 & 1 & & & & \\ & 1 & 1 & & & \\ & & 1 & 1 & & \\ & & & 1 & 1 & \end{bmatrix} \xrightarrow{\text{Superposição}} \mathbf{H}_2 = \begin{bmatrix} I^{(2)} & & & & & I \\ I & I^{(3)} & & & & \\ & I & I^{(4)} & & & \\ & & I^{(1)} & I^{(5)} & & \\ & & & I & I^{(6)} & \end{bmatrix},$$

Isto é:

$$\mathbf{H}_2 = \left[\begin{array}{cccccccc|cccccccc|cccccccc|cccccccc}
0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1\\
\hline
1&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
\hline
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0\\
\hline
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1\\
\hline
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0\\
\hline
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1\\
\hline
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0
\end{array}\right]$$

A partir das matrizes propostas no Exemplo 3.2, Tabela 3.3, método-1, são gerados os códigos LDPC C1 (500, 250); C2(1800,1200) ; C3(3000,1500); C4(4158, 3465) cujas propriedades são mostradas na Tabela 3.4.

Tabela 3.4 Códigos propostos pelo méto-1

Comprimento do código	**500** **N=5**	**500** **N=5**	**1800** **N=5**	**3000** **N=5**	**4158** **N=3**
Matriz de Verificação de Paridade	**[H_1H_2], [H_1H_3], [H_1H_4]**	**[$H_1H_2H_3H_4$]**	**[$H_1H_2H_3$], [$H_1H_3H_4$]**	**[H_1H_4]**	**H_b**
Cópias (*p*)	2	4	3	2	6
Vetor das circulantes (*v*)	50	25	120	300	231
Taxa	0.5	0.75	0.666	0,5	0,833

$$\mathbf{H}_{b2} = \begin{bmatrix} I & & I^{(5)} & I^{(2)} & & I^{(5)} & I^{(2)} & & I^{(5)} & I^{(6)} & & I^{(5)} & I^{(8)} & & I^{(6)} & I^{(1)} & & I^{(6)} \\ I & I & & I & I^{(3)} & & I & I^{(4)} & & I & I^{(5)} & & I & I^{(5)} & & I & I^{(3)} & \\ & I^{(1)} & I & & I^{(1)} & I^{(4)} & & I^{(1)} & I^{(6)} & & I^{(1)} & I^{(4)} & & I^{(1)} & I^{(2)} & & I^{(1)} & I^{(5)} \end{bmatrix}$$

A sequência de passos do Método- 2 é análoga a do Método-1 diferenciando apenas no fato de que a matriz plataforma (B.17) é de dimensão 5 e tem um grau de liberdade maior na escolha dos movimentos das matrizes clientes.

Método-2. Este método apresenta as matrizes circulantes peso 2 como matrizes clientes e suas concatenações. São superpostas nos elementos não nulos da matriz (B.17).

Algorítmo-2.

Este algoritmo constrói matriz de verificação de paridade de dimensão *5pv* e taxa $\approx (p-1)/p$ a partir da concatenação das *p* cópias da matriz binária B definida em (B.17). Dada uma sequencia $\{\mathbf{B}_1, \mathbf{B}_2, ..., \mathbf{B}_p\}$ de *p* cópias de B têm-se:

Passo 1: Tome $\{\mathbf{B}_1, \mathbf{B}_2, ..., \mathbf{B}_p\}$ *p* cópias de matrizes binárias de B.

Passo 2: Construa as matrizes plataformas $\mathbf{H}_1,\mathbf{H}_2,...,\mathbf{H}_p$ atribuindo por superposição em $\{\mathbf{B}_1,\mathbf{B}_2,...,\mathbf{B}_p\}$ matrizes $\mathbf{I}_v^{(m)}$ de modo que a soma dos movimentos aleatórios sejam mod5.

Passo 3: Sobreponha a matriz O_v nos elementos nulos de $\{\mathbf{B}_1,\mathbf{B}_2,...,\mathbf{B}_p\}$

Passo 4: Construa a matriz de verificação de paridade **H** de 4-ciclo livre, dimensão $5\times 5p$ e taxa $\approx (p-1)/p$ concatenando $\{\mathbf{H}_1,\mathbf{H}_2,...,\mathbf{H}_p\}$

Exemplos 3.3: *A seguir tem-se cinco matrizes plataforma G_i construídas a partir do método2, cujos elementos são denotados pelas letras A,B,C,D,E,F,J,L,M que também indicam as matrizes clientes superpostas, Tabela 3.5. Os elementos nulos foram desprezados.*

$$\mathbf{G}_i=\begin{bmatrix} A & C & & & & \\ & & E & J & & \\ B & & & & & L \\ & D & F & & & \\ & & & K & M & \end{bmatrix}$$

Tabela 3.5 – Matrizes Clientes e Movimentos Aleatórios de circulantes – algoritmo 2

Matriz plataforma	*A*	*B*	*C*	*D*	*E*	*F*	*J*	*K*	*L*	*M*
$\mathbf{G}_1$	$\mathbf{I}^{(3)}$	$\mathbf{I}^{(1)}$	$\mathbf{I}^{(2)}$	$\mathbf{I}^{(3)}$	$\mathbf{I}^{(4)}$	$\mathbf{I}^{(5)}$	$\mathbf{I}^{(5)}$	$\mathbf{I}^{(5)}$	$\mathbf{I}^{(6)}$	$\mathbf{I}^{(6)}$
$\mathbf{G}_2$	$\mathbf{I}^{(6)}$	$\mathbf{I}^{(6)}$	$\mathbf{I}^{(6)}$	$\mathbf{I}^{(6)}$	$\mathbf{I}^{(6)}$	$\mathbf{I}^{(6)}$	$\mathbf{I}^{(6)}$	$\mathbf{I}^{(6)}$	$\mathbf{I}^{(6)}$	$\mathbf{I}^{(6)}$
$\mathbf{G}_3$	$\mathbf{I}^{(4)}$	$\mathbf{I}^{(3)}$	$\mathbf{I}^{(2)}$	$\mathbf{I}^{(1)}$	$\mathbf{I}^{(5)}$	$\mathbf{I}^{(6)}$	$\mathbf{I}^{(7)}$	$\mathbf{I}^{(9)}$	$\mathbf{I}^{(2)}$	$\mathbf{I}^{(1)}$
$\mathbf{G}_4$	$\mathbf{I}^{(5)}$	$\mathbf{I}^{(5)}$	$\mathbf{I}^{(5)}$	$\mathbf{I}^{(5)}$	$\mathbf{I}^{(5)}$	$\mathbf{I}^{(5)}$	$\mathbf{I}^{(5)}$	$\mathbf{I}^{(5)}$	$\mathbf{I}^{(5)}$	$\mathbf{I}^{(5)}$
$\mathbf{G}_5$	$\mathbf{I}^{(6)}$	$\mathbf{I}^{(5)}$	$\mathbf{I}^{(4)}$	$\mathbf{I}^{(3)}$	$\mathbf{I}^{(2)}$	$\mathbf{I}$	$\mathbf{I}^{(3)}$	$\mathbf{I}^{(2)}$	$\mathbf{I}$	$\mathbf{I}$

A partir das matrizes propostas no Exemplo 3.3, Tabela 3.5, método-2, são gerados os códigos LDPC C6(1500,1200); C7(4200,3600) cujas propriedades são mostradas na Tabela 3.6.

Tabela 3.6: Códigos propostos pelo méto-2

Comprimento do código	**1500**	**4200**
Matriz de Verificação de Paridade	$[\mathbf{G_1G_1G_1G_1G_1}]$ $[\mathbf{G_1G_2G_2G_2G_2}]$ $[\mathbf{G_2G_2G_2G_2G_2}]$	$[\mathbf{G_1G_1G_2G_2G_3G_3G_3}]$ $\mathbf{[G_1G_2G_3G_4G_5G_5G_4}]$
Cópias (*p*)	5	7
Vetor das circulantes (*v*)	60	24
Taxa	0.8	0,857

Capítulo 4

Resultados obtidos

Neste capítulo são apresentados os resultados das simulações implementadas no software Matlab. São feitas as análises de desempenho dos códigos LDPC na correção de apagamentos em rajada gerados a partir das matrizes de verificação de paridade propostos pelos métodos 1 e 2.

4.1 Desempenho dos códigos propostos

Para verificar o desempenho dos códigos propostos foram produzidas três simulações de 100 iterações cada utilizando mensagem aleatória em canal com ruído branco (canal AWGN) no qual é utilizada uma modulação BPSK. O canal introduz apagamentos em pontos aleatório da palavra código e uma rajada de apagamento é introduzida na palavra código a partir do sorteio de um ponto (bit inicial) condicionado a taxa do código e ao tamanho da rajada. É feita uma análise da média dos bits recuperados nas simulações e a média da taxa de erro de bits. O modelo utilizado para simulação por ser entendido por maio da Figura 4.1, que mostra cada etapa do processo utilizando um diagrama de blocos.

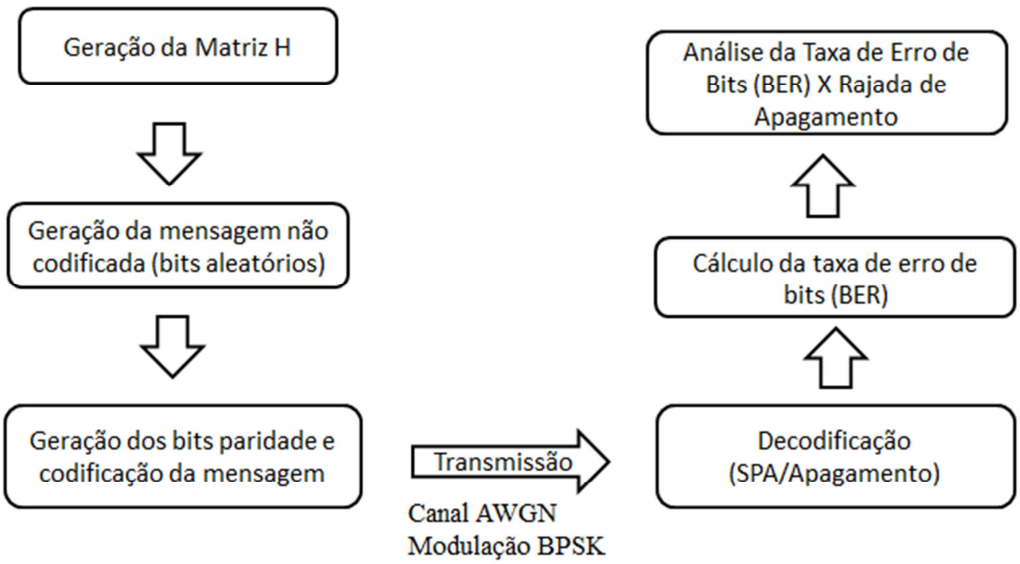


Figura 4.1 – Modelo utilizado para simulação dos códigos LDPC propostos

Para análise dos códigos LDPC pelo método-1 foram gerados os códigos C1(500, 250) e C2(1800, 1200) cujas matrizes de verificação de paridades estão descritas na Tabela 3.4. No receptor as rajadas de apagamento são recuperadas e o L_{max} é calculado com média das três simulações.

O código C1(500,250) de concatenação $\mathbf{H}_1\mathbf{H}_2$ recuperou 240 bits, tendo L_{max} =240, isto significa que dado uma rajada de 240 bits apagados todos foram recuperado. A Tabela 4.1 mostra a eficiência do código C1(500,250) a partir das concatenações propostas.

Tabela 4.1: Eficiência dos códigos propostos gerados pelo método 1

Comprimento do código	**Matriz de verificação de Paridade**	L_{max}	**Eficiência**
500	$[\mathbf{H}_1\mathbf{H}_2]$, $[\mathbf{H}_1\mathbf{H}_3]$, $[\mathbf{H}_1\mathbf{H}_4]$, $[\mathbf{H}_1\mathbf{H}_2\mathbf{H}_3\mathbf{H}_4]$	240; 241; 248 118	0,960; 0,964; 0,992 0,944
1800	$[\mathbf{H}_1\mathbf{H}_2\mathbf{H}_3]$, $[\mathbf{H}_1\mathbf{H}_3\mathbf{H}_4]$	572; 576	0,953; 0,960
3000	$[\mathbf{H}_1\mathbf{H}_4]$	1475	0,983
4158	$\mathbf{H}_{b2}$	682	0,927

O código C1(500,250) gerado pelas concatenações $\mathbf{H}_1\mathbf{H}_2$, $\mathbf{H}_1\mathbf{H}_3$, $\mathbf{H}_1\mathbf{H}_4$ obtém os melhores desempenhos. As matrizes $\mathbf{H}_1$, $\mathbf{H}_2$, $\mathbf{H}_3$ não possuem diferenças nos seus elementos livres, F=**I**; no entanto $\mathbf{H}_4$ possui elemento livre, F=$\mathbf{I}^{(5)}$, significativamente diferente de $\mathbf{H}_1$, $\mathbf{H}_2$, $\mathbf{H}_3$ e a concatenação, $\mathbf{H}_1\mathbf{H}_4$, obteve a melhor eficiência (99,2%). Após a recuperação dos bits apagados o decodificador calcula a taxa de erro de bits (BER). O gráfico da Figura 4.2 mostra o desempenho do código C1(500,250) pelo método-1.

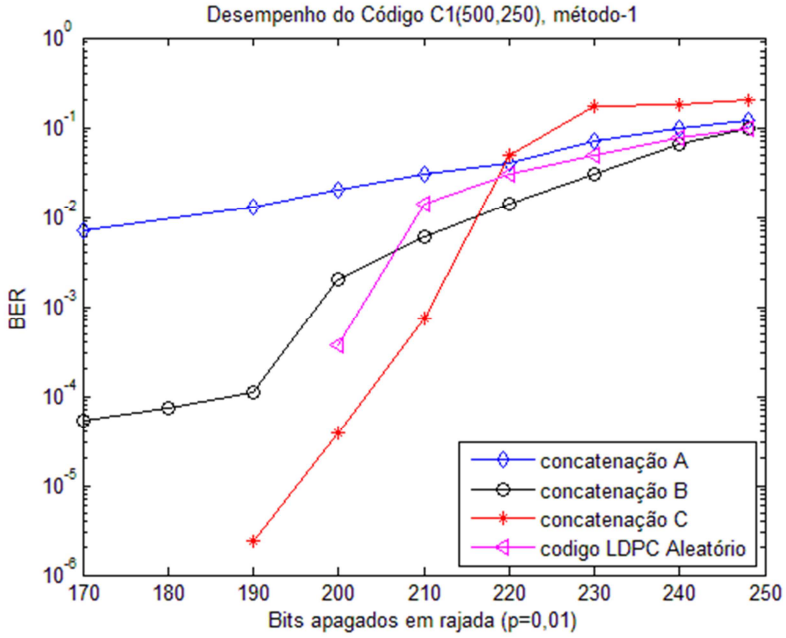


Figura 4.2. Desempenho do código C1(500,250) taxa 0,5. Concatenações: A=[$\mathbf{H}_1\mathbf{H}_2$], B=[$\mathbf{H}_1\mathbf{H}_3$], C=[$\mathbf{H}_1\mathbf{H}_4$]. Canal com apagamento em rajada aleatório, probabilidade *p=0,01*.

Para o código C2(1800, 1200), foram propostas as concatenações das matrizes $\mathbf{H}_1\mathbf{H}_2\mathbf{H}_3$, $\mathbf{H}_1\mathbf{H}_2\mathbf{H}_4$ obtendo L_{max} 572 e 576 bits respectivamente e observou-se que o melhor rendimento é a concatenação $\mathbf{H}_1\mathbf{H}_2\mathbf{H}_4$ que possui elemento livre $\mathbf{I}^{(5)}$ em $\mathbf{H}_4$. Em uma análise preliminar foi observado que o movimento "maior" do elemento livre é o responsável pelo melhor desempenho do código.

Para análise dos códigos C6(1500, 1200) e C7(4200, 3600), pelo método-2, foram propostas as concatenações descritas no Exemplo 3.3, Tabela 3.5, seção 3.1. A eficiência dos códigos é mostrada na Tabela 4.2. O gráfico da Figura 4.3 mostra o desempenho dos códigos propostos pelo método-2 para correção de apagamento em rajada.

Tabela 4.2: Eficiência dos códigos propostos gerados pelo método 2

Comprimento do código	Matriz de verificação de paridade	L_{max}	Eficiência
1500	$[\mathbf{G}_1\mathbf{G}_1\mathbf{G}_1\mathbf{G}_1\mathbf{G}_1]$	290;	0,966;
	$[\mathbf{G}_1\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2]$	292;	0,973;
	$[\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2]$	296	0,986
4200	$[\mathbf{G}_1\mathbf{G}_1\mathbf{G}_2\mathbf{G}_2\mathbf{G}_3\mathbf{G}_3\mathbf{G}_3]$	582	0,970
	$[\mathbf{G}_1\mathbf{G}_2\mathbf{G}_3\mathbf{G}_4\mathbf{G}_5\mathbf{G}_5\mathbf{G}_4]$	580	0,966

.

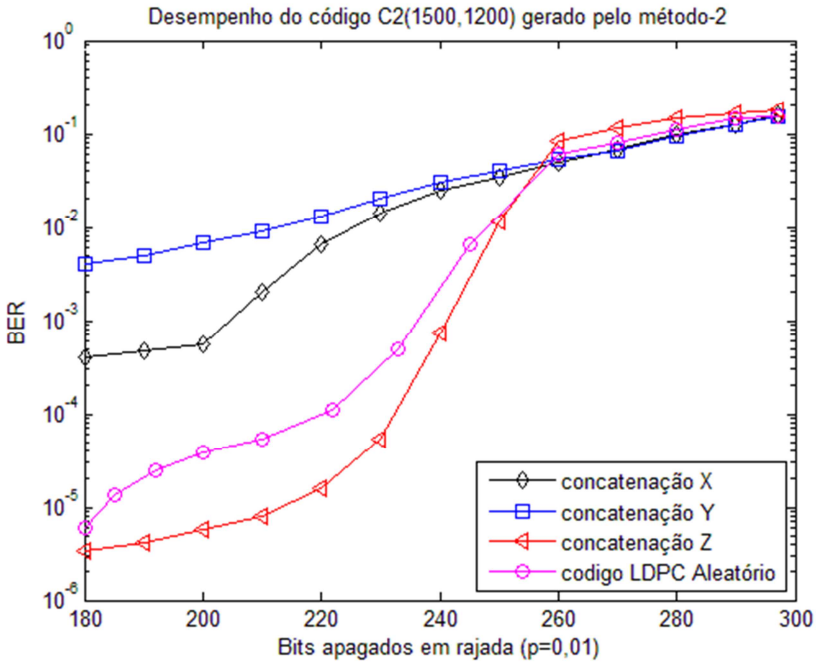


Figura 4.3. Desempenho do código LDPC de comprimento C6(1500, 1200) taxa 0,8. Concatenações: X=[$\mathbf{G}_1\mathbf{G}_1\mathbf{G}_1\mathbf{G}_1\mathbf{G}_1$], Y=[$\mathbf{G}_1\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2$] e Z=[$\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2$]. Canal com apagamento em rajada aleatório e probabilidade de apagamento *p=0,01*

A Tabela 4.3 apresenta um resumo dos códigos de comprimento 500, 1500 e 1800 gerados pelos métodos propostos.

Tabela 4.3 L_{max} dos códigos propostos em relação aos movimentos das matrizes clientes

Comprimento do código	**Matriz de verificação de paridade**	**Método**	**Posicionamento das matrizes clientes não nulas**	**L_{max}**
500	[$\mathbf{H}_1\mathbf{H}_2$]	Método-1	$\mathbf{H}_2$; F=**I** (ele. Livre)	240
	[$\mathbf{H}_1\mathbf{H}_3$]	Método-1	$\mathbf{H}_3$; F=**I**	241
	[$\mathbf{H}_1\mathbf{H}_4$]	Método-1	$\mathbf{H}_4$; F= $\mathbf{I}^{(5)}$	**248**
1800	[$\mathbf{H}_1\mathbf{H}_2\mathbf{H}_3$]	Método-1	$\mathbf{H}_2\mathbf{H}_3$; F=**I**	572
	[$\mathbf{H}_1\mathbf{H}_3\mathbf{H}_4$]	Método-1	$\mathbf{H}_3$; F=**I**; $\mathbf{H}_4$; F=$\mathbf{I}^{(5)}$	**576**
1500	[$\mathbf{G}_1\mathbf{G}_1\mathbf{G}_1\mathbf{G}_1\mathbf{G}_1$]	Método-2	$\mathbf{I}^{(m)}$, m= 1,2,3,4,5,6	290
	[$\mathbf{G}_1\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2$]	Método-2	$\mathbf{G}_2$; $\mathbf{I}^{(m)}$, m=6	292
	[$\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2$]	Método-2	$\mathbf{G}_2$; $\mathbf{I}^{(m)}$, m=6	**296**

A Figura 4.4 e a Figura 4.5 mostram o desempenho de um código LDPC proposto, de comprimento 4170, taxa 0.833, gerado com o Algoritmo 1 (N=5, v=139, p=6) e Algoritmo 2 em canal com apagamentos em rajada..

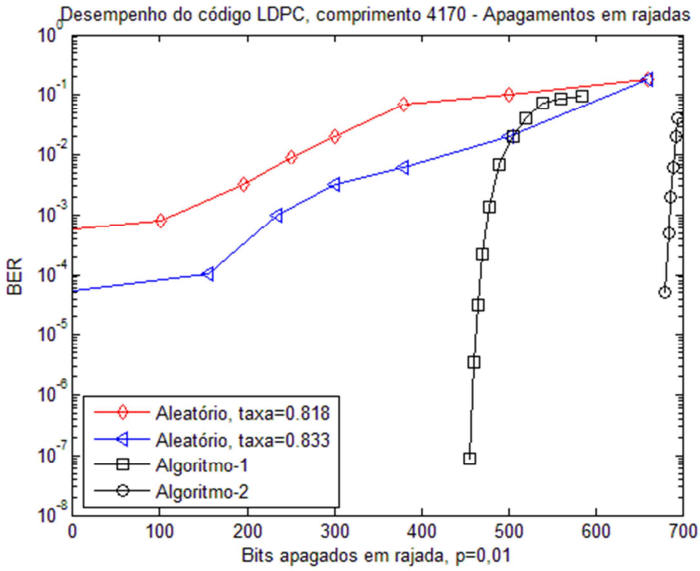


Figura 4.4: Desempenho do código LDPC de comprimento 4170, taxa 0,833, em canal com apagamento aleatório e probabilidade *p=0,01*

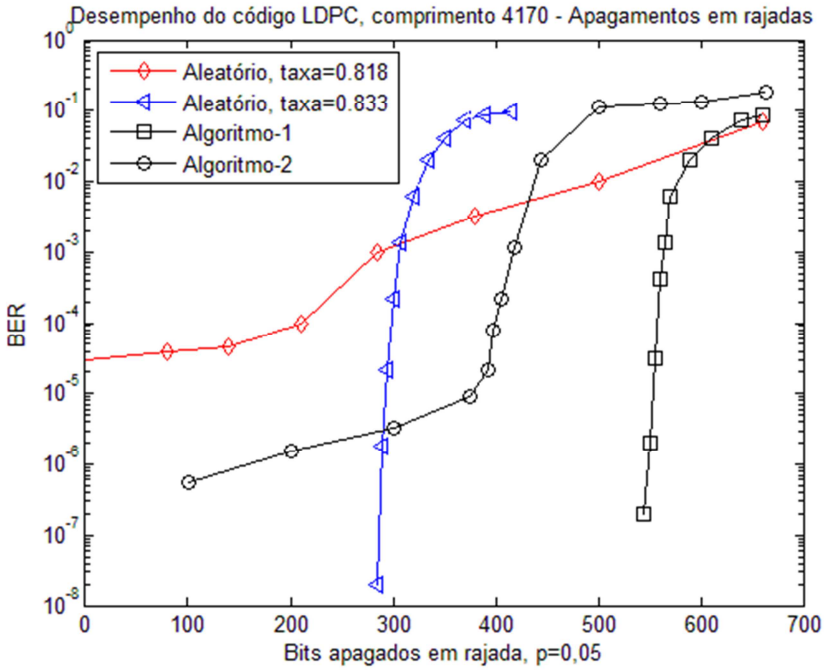


Figura 4.5: Desempenho do código LDPC de comprimento 4170, taxa 0,833, em canal com apagamento aleatório e probabilidade *p=0,05*.

Para grandes rajadas, os códigos LDPC tem se mostrado um eficiente corretor de erros de apagamentos [24-26]. A Tabela 4.4 descreve a comparação entre a eficiência dos códigos propostos com outros códigos obtidos na literatura. A primeira coluna da Tabela 4.4 indica as referências bibliográficas e as informações comparativas dos códigos; os códigos propostos estão indicados nesta coluna especificamente pelo uso da palavra “algorítmo”.

Tabela 4.4 – Comparação dos códigos LDPC propostos com códigos obtidos da literatura de apagamentos em rajadas com códigos selecionados. *Lmax* é o número de bits apagados recuperados

Tipo de código	Comprimento	Taxa	L_{max} (bits)	Eficiência
[26, N=5,p=2,v=50]	500	0,5	248	0,992
[23, Tabela I]	2040	0,515	509	0,51
[24, Tabela IV]	2000	0,5	786	0,786
[25, Margulis-Tabela I]	2640	0,5	1033	0,782
[25, PEG IRA-Tabela I]	2000	0,5	403	0,403
[26, Tabela I]	3000	0,5	1468	0,978
Algoritmo 1, N=5, p=2, v=50	500	0,5	248	0,992
Algoritmo 1, N=5, p=2, v=300	3000	0,5	1475	0,983
Algoritmo 2, p=2,v=300	3000	0,5	1498	0,999
[24, Tabela VI]	500	0,7	121	0,345
[23, Tabela I]	8176	0,753	1021	0,50
Algoritmo 1, N=5, p=4, v=25	500	0,75	180	0,994
Algoritmo 1, N=5, p=4, v=408	8180	0,75	2037	0,996
[26, N=5,v=300]	1500	0,8	291	0,97
[23, PEG regular-Tabela I]	4608	0,8752	287	0,499
[26, Tabela I]	4158	0,8333	682	0,927
[26, Tabela I]	16500	0,9	1648	0,999
Algoritmo 2, p=5,v=60	1500	0,8	296	0,986
Algoritmo 2, p=5,v=120	3000	0,8	592	0,986
Algoritmo 2, p=6,v=100	3000	0,833	490	0,980
Algoritmo 1, N=3, p=6,v=231	4158	0,8333	682	0,927
Algoritmo 2, p=6,v=550	16500	0,833	2748	0,999

4.2 Recuperação de apagamentos em canal ruidoso usando o código proposto – Simulação

Para comparar o desempenho dos códigos propostos, foram implementadas simulações da transmissão de uma imagem através de um canal ruidoso utilizando decodificador SPA e algoritmo corretor de apagamento. Foi utilizada nas simulações a imagem Lena de dimensão 512x512 (262.144 pixels) mostrada na Figura 4.7. A palavra código enviada através do canal tem comprimento de C(1512,1260), taxa 0,83, com matrizes de verificação de paridade desenvolvida pelos métodos 1 e 2 propostos, a qual foi adaptada à codificação do software. O modelo utilizado para simulação por ser entendido por meio da Figura 4.6, que mostra cada etapa do processo utilizando um diagrama de blocos. Nestas simulações foram utilizados computadores AMD A6-5400B de 3,6 GHz com 4GB de memória RAM.

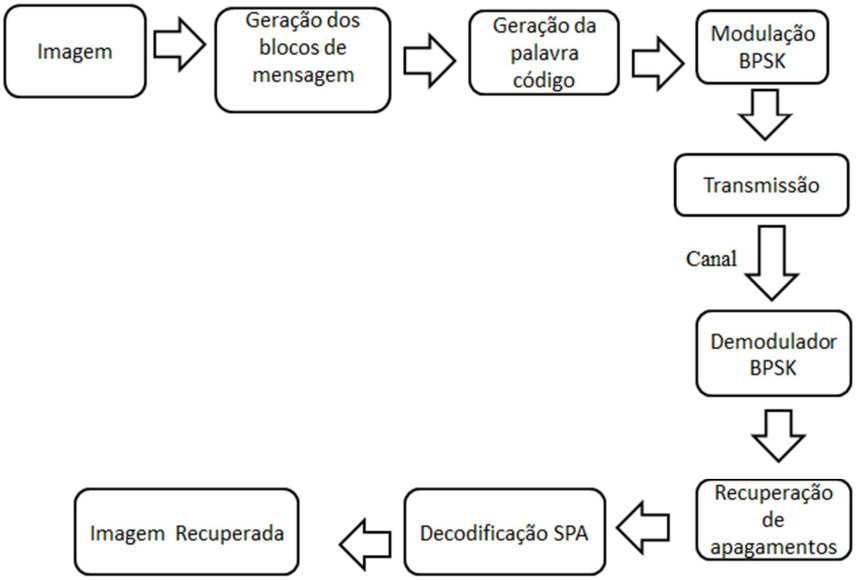


Figura 4.6: Modelo a ser usado Modelo utilizado para simulação da imagem

Figura 4.7: Imagem original usada nas simulações Lena(512x512)

Foram realizadas simulações em que a imagem ao passar pelo canal, apresenta uma faixa de apagamento. Em cada simulação o canal introduz perdas de bits (rajada de apagamentos) iniciados em um ponto aleatório da palavra código. O comprimento da rajada de apagamento varia de 2 a 246 bits; A probabilidade de apagamento do canal foi variada de 0,01 a 0,99 em incrementos de 0,05. A seguir são apresentados os resultados de algumas simulações.

Simulação 1

Nesta simulação, o código foi gerado pelo método-1 usando elemento livre F=**I** nas matrizes cliente. A Figura 4.8 mostra a imagem com bits apagados após sair do canal. Em seguida as Figuras 4.8(a) e Figuras 4.8(b) mostram as imagens após a Figura 4.11

passar por dois processos de recuperação utilizando o código proposto. A Tabela 4.5 mostra o tempo de resposta do programa e os números de pixels recuperados.

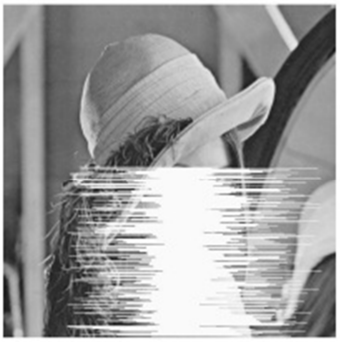


Figura 4.8: Imagem após sair do canal: 58.623 pixels apagados

Recuperação-1

Figura 4.8(a): Recuperação-1

Recuperação-2

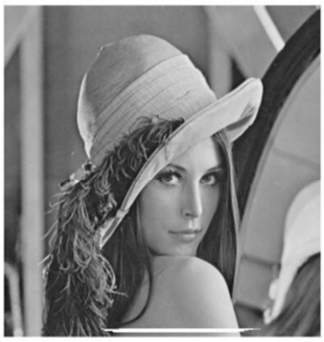


Figura 4.8(b): Recuperação-2

Tabela 4.5: Recuperação de imagem usando código método-1, elemento livre F=**I**

Imagem	Iterações	Tempo de Resposta da recuperação	N. de Pixels Recuperados
Figura 4.8(a)	80	10024.532s	56864
Figura 4.8(b)	100	12423.406s	57102

Simulação 2

Nesta simulação o código foi gerado pelo método-1 usando elemento livre F=$\mathbf{I}^{(5)}$ nas matrizes cliente. A Figura 4.9 mostra a imagem com bits apagados após sair do canal. Em seguida a Figuras 4.9(a) mostra a imagem após passar por processo de recuperação utilizando o código proposto. A Tabela 4.6 mostra a característica da recuperação

Figura 4.9 Imagem após sair do canal-71.034 pixels apagados

Figura 4.9 (a) Imagem recuperada

Tabela 4.6: Recuperação de imagem usando código método-1, elemento livre F= $\mathbf{I}^{(5)}$

Imagem	Iterações	Tempo de resposta da recuperação	N. de Pixels recuperados
Figura 4.9(a)	80	9507.402s	70.893

As Tabelas 4.5 e Tabelas 4.6 mostram que os códigos gerados pelo método-1 com elemento livre F= $\mathbf{I}^{(5)}$ tem o melhor desempenho na recuperação dos apagamentos

Simulação 3

Nesta simulação o código foi gerado pelo método-2 usando matrizes clientes alternadas $\mathbf{I}^{(5)}$ e $\mathbf{I}^{(6)}$. A Figura 4.10 mostra a imagem com bits apagados após sair do canal. Em seguida a Figuras 4.10(a) mostra a imagem após passar por processo de recuperação utilizando o código proposto. A Tabela 4.7 mostra a característica da recuperação

Figura 4.10 Imagem após sair do canal: 57834 pixels apagados

Figura 4.10(a) Imagem Recuperada

Tabela 4.7: Recuperação de imagem usando código método-2, matrizes alternadas $\mathbf{I}^{(5)}$ e $\mathbf{I}^{(6)}$

Imagem	Iterações	Tempo de resposta da recuperação	N. de Píxeis recuperados
Figura 4.10(a)	80	8527.281s	57813

A Tabela 4.7 mostra que o código gerado pelo método-2 alternando $\mathbf{I}^{(5)}$ e $\mathbf{I}^{(6)}$ com matrizes clientes reduz o tempo de resposta na execução do programa com boa eficiência na recuperação dos bits apagados.

Foram ainda gerados outros códigos LDPC, de comprimentos 3000, 4158 e 8180. Estes códigos foram aplicados para recuperar a imagem da Lena transmitida. A Tabela 4.8 apresenta os resultados das simulações.

Tabela 4.8 Desempenho de outros códigos proposto na recuperação dos pixels apagados da imagem Lena após passar pelo canal com apagamento em rajada.

Comprimento	Método	Matrizes Circulantes usadas	N. pixeis apagados	Tempo de resposta	N. pixeis recuperados
3000	2	$I^{(5)}$ e $I^{(6)}$- alternadas	82347	10507.206s	81868
4158	1	F= $\mathbf{I}^{(6)}$ (ele. Livre)	40138	17816.332s	39965
8180	1	F= $\mathbf{I}^{(6)}$ (ele. Livre)	55073	18360.328s	55010

Capítulo 5

Conclusão

Nesta tese foram apresentados dois métodos para gerar matrizes de verificação de paridade esparsas, 4 ciclo-livre, baseados em concatenações de matrizes bases e superposição de matrizes circulantes. Para o código C1(500,250) foram propostas, pelo método-1, as concatenações $\mathbf{H}_1\mathbf{H}_2$, $\mathbf{H}_1\mathbf{H}_3$, $\mathbf{H}_1\mathbf{H}_4$ com duas cópias da matriz base [B.13] e $\mathbf{H}_1\mathbf{H}_2\mathbf{H}_3\mathbf{H}_4$ com quatro cópias da mesma matriz base. A Tabela 4.1 mostrou que os códigos gerados com duas cópias, pelos métodos-1, têm eficiência melhor na correção de apagamentos em rajadas do que o código gerado por quatro cópias. O código de concatenação $\mathbf{H}_1\mathbf{H}_4$ tem desempenho satisfatório em comparação com os outros códigos mostrados na Figura 4.2; recuperando 190 bits apagados, este código possui a menor taxa de erro de bits em comparação com os demais códigos, após 220 bits recuperados os códigos possuem taxa de erros de bits com pouca diferença. A melhor eficiência do código C1(500,250) é descrita pela concatenação de duas cópia ($[\mathbf{H}_1\mathbf{H}_4]$) onde aparecem o elemento livre F=$\mathbf{I}^{(5)}$.

Para o código C6(1500,1200), foram propostas as concatenações $\mathbf{G}_1\mathbf{G}_1\mathbf{G}_1\mathbf{G}_1\mathbf{G}_1$, $\mathbf{G}_1\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2\ \mathbf{G}_2$, $\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2$ geradas pelo método-2, todas com cinco cópias da matriz base de [B.17]. A Tabela 4.2 mostrou que concatenação $\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2$ tem a melhor eficiência na correção de apagamentos em rajada podendo recuperar até 296 bits. O código de concatenação $\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2\mathbf{G}_2$ tem a menor taxa de erro de bits no intervalo de 180 a 250 rajadas de apagamento em comparação com os outros códigos mostrados na Figura 4.3, após 260 bits recuperados os códigos possuem taxa de erros de bits com pouca diferença. A melhor eficiência do código C6(1500,1200) é descrita pela concatenação de cinco cópias da matriz base $\mathbf{G}_2$ onde os elementos não nulos da matriz $\mathbf{G}_2$ são superpostos pela matriz $\mathbf{I}^{(5)}$.

As simulações 1,2 e 3 da seção 4.2 comparam o desempenho do código C(1512,1260) gerado pelos métodos-1 e 2 na recuperação dos pixels apagados da imagem Lena após passar por um canal que introduz apagamentos em rajada. Os resultados das Tabelas 4.5 e 4.6 mostram que o código gerado pelo método-1, em

relação ao tempo de resposta do programa e ao número de pixels recuperados, no tratamento de recuperação da imagem borrada, tem melhor desempenho quando este é gerado pela concatenação com elemento livre $F=\mathbf{I}^{(5)}$. A tabela 4.7 mostrou que o mesmo código gerado pelo método-2, com alternância das matrizes superpostas $I^{(5)}$ e $I^{(6)}$, tem rendimento melhor na recuperação, quando se considera o número de iteração e o tempo de resposta do programa, mesmo tendo recuperado um número um pouco menor de pixels apagado.

A Tabela 4.3 mostrou que os códigos comprimento variados gerados pelos métodos 1 e 2 tem a mesma, ou melhor, eficiência que os códigos propostos em outras literaturas. O código de comprimento 16500, pelo método-1, tem a mesma eficiência do código proposto por Sarah [26] de mesmo comprimento.

Apesar dos métodos 1 e 2 serem implementados com matrizes de dimensões mod5 foi possível avaliar simulações com diferentes dimensões, pois o vetor das matrizes circulantes possibilitou a combinação de números que exprimem dimensões não múltiplas de 5.

Trabalhos futuros

Em futuros trabalhos podem-se focar outras matrizes bases, outros movimentos de matrizes circulantes na correção não só de um apagamento em rajada em uma palavra código, mas em múltiplos apagamentos na palavra código ou em múltiplas palavras.

Publicações do Autor no Período

Artigo em Revista

Burst erasure correction using LDPC codes constructed on base matrices generated by matched groups, nested polygons and superposed circulant matrices. Cassio André Sousa da Silva e Evaldo Pelaes. **DYNA**, vol.83, Nº 197[Online]. 2016

Referências Bibliográficas

[1] RIASCOS ERAZO, Sandra Cristina**. A model for assessing information technology effectiveness in the business environment**. Ingeniería e Investigación [S.l.], v. 28, n. 2, pp 158-166, Apr. 2010.

[2] MARTÍNES ROSO, Ricardo. **El papel de la comunicación en los sistemas generales**. Ingeniería e Investigación**.** [S.l.], n. 22, p. 68-71, nov. 2012.

[3] CORREA ESPINAL, Alexander; GÓMES Montoya ; ANDRÉS, Rodrigo. **Tegnologías de la información en la cadena de suministro**. DYNA, [S.l.], v. 76, n. 157, p. 37-48, may. 2009.

[4] MANRIQUE-LOSADA, Bell; ECHEVERRI-ARIAS, Jaime Alberto; PELÁZ-RODRIGUES, Marlon. **Hydroinformatics' contribution to Amazonian water resources and ecosystem management**. Ingeniería e Investigación**,** [S.l.], v. 31, n. 1, p. 108-116, may. 2011.

[5] MORENO LÓPEZ, Gustavo Alberto; JIMÉNEZ BUILES; JOVANI Alberto. **Cycle of PDCA T-Learing Model and its Application on Interactive digital TV**. DYNA, [S.l.], v. 79, n. 173, p. 61-70, jan. 2012

[6] P. ELIAS. **Coding for two noisy channels**. Information Theory, Third London Symposium, p. 61-76, 1955.

[7] P. R. FREITAS, V. C. Da Rocha Jr.; J. S. Lemos-Neto. **On the iteractive decoding of binary product codes over the binary erasure channe**l. In: Proceedings of 8th International Symposium on Wireless Communication Systems, Aachen, Germany, November 2011, p. 126–130.

[8] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi; D. A. Spielman. **Improved low-density parity-check codes using irregular graphs**. IEEE Trans. Inform. Theory, vol. 47, no. 2, pp. 585–598, Feb. 2001.

[9] G. D. Forney, Jr. **Burst-correcting codes for the classic bursty channel.** Trans. Comms, vol. 19, no. 5, pp. 772–781, Oct. 1971.

[10] G.D. Forney, Jr. **Geometrically uniform codes.** IEEE Trans. Inform. Theory, vol.37, pp. 1241-1260,1991.

[11] A.S. Tanenbaum. **Computer Networks**. 4ª ed. Prentice Hall, Agosto 2002

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

[13] J. CAI, M. Tomlinson; C. Tjhai; M. Ambroze; M. Ahmed. **Comparison of concatenated Reed-Solomon coding with hard-decision to soft-decision LDPC coding for the Rayleigh fading channel**. In: Proceedings of IEEE Information Theory Workshop, Chengdu, China, 2006, p. 135–139.

[14] R. G. Gallager, "**Low-density parity-check codes**," IRE Trans. Inform. Theory, vol. IT-8, no. 1, pp. 21–28, Jan. 1962.

[15] C. Berrou, A. Glavieux; P. Thitimajshima. **Near Shannon limit error-correcting coding and decoding**. Proceedings of the IEEE International Conference on Communications (ICC '93), Maio de 1993, pp. 1064-1070.

[16] D. J. C. MacKay. **Good error-correcting codes based on very sparse matrices**. IEEE Trans. Inform. Theory, vol. 45, no. 2, pp. 399–431, Mar. 1999.

[17] D.J.C. Mackay; R.M. Neal. **Near Shannon Limit performance of low density parity check codes**. Eletronic Letters, v.32, p. 1645-1646, 1996.

[18] Z. Li, L. Chen; L. Zeng, S. Lin; W. Fong. **Efficient encoding of quasi-cyclic low-density parity-check codes**. IEEE Trans. Commun., vol. 53, no. 11, pp. 1973–1973, Nov. 2005.

[19] B. Liu; S. Mei; D. Bai. **Construction of LDPC Codes with Cycles Hold in Tanner Graph**. IEEE Trans. Inform

[20] R. Michael Tanner; D. Sridhara; A. Sridharan, T. E. Fuj; D. J. Costello. **LDPC Block and Convolutional Codes Based on Circulant Matrices.** IEEE Trans. Inf. Theory, vo1.50, no.12,pp. 2966-2984, Dec.2004.

[21] Y. Mal, M. Lee; Y. Xiao. **LDPC Codes Based on Circulant Permutation Matrices for Fast Encoding**. ICWMMN Proceedings 2006

[22] Z. Du; ZHEN Ji. **LDPC Codes by Circulant Decomposition Based on Difference Family**. Proceedings of the International MultiConference of Engineers and Computer Scientists 2009 Vol I IMECS 2009, March 18 - 20, Hong Kong, 2009

[23] Y. Y. Tai, L. Lan, L. Zeng, S. Lin; K. A. S. Abdel-Ghaffar. **Algebraic construction of quasi-cyclic LDPC codes for the AWGN and erasure channels.** IEEE Trans. Commun., vol. 54, no. 10, pp. 1765– 1774, 2006.

[24] G. Hosoya, H. Yagi, T. Matsushima, and S. Hirasawa, **A modification method for constructing low-density parity-check codes for burst erasures.** *ICICE Trans. Fundamentals*, vol. E89-A, no. 10, pp. 2501–2509, Oct. 2006.

[25] E. Paolini, M. Chiani; G. P. Calzolari. **Construction of Near-Pptimum Burst Erasure Correcting Low-Density Parity-Check codes.** *IEEE Trans. Inform. Theory*, vol. 57, no. 5, pp. 1320–1328, May 2009.

[26] S. J. Johnson. **Burst Erasure Correcting LDPC Codes**. IEEE Trans. Commun., vol. 57, no. 3, pp. 641–652, March. 2009.

[27] A. Hefez; M. L Villela. **Códigos Corretores de Erros**. IMPA, Rio de Janeiro, 2002

[28] D. Slepain. **Grup Codes for the Gaussian Channel.** Bell Syst. Tech. J., vol 47, pp.575-602, 1968

[29] C. Jones, A. Matache, T. Tian, J. Villasenor; R. D. Wesel, **The universality of LDPC codes on wireless channels**. In Proc. MilCom, Boston, MA, Oct. 2003, pp. 440–445

[30] H. Domingues; G. Iezzi, **Álgebra moderna**, 2º edição Atual Editora, São Paulo 1982.

[31] J. C. Moreira; P. G Farrel, **Essentials of error-control coding**, John Wiley & Son, Ltd 2006

[32] W. W. Peterson; E. J. Weldon. **Error-Correcting Codes**. 2nd ed. Cambridge, MA: MIT Press, 1972.

[33] S. J. Johnson**. Iterative Error Correction - Turbo, Low-Density Parity-Check and Repeat–Accumulate Codes.** Cambridge University Press , New York USA, 2010

[34] J. H van Lint. **Introduction to Coding Theory**. 3º edição, Springer, Press 1998.

[35] LIN. S., COSTELLO Jr. D.J. **Error Control Coding.** 2º edição, Pearson, 2004.

[36] M. Yang; W. Ryan; Y. Li. **Design of efficiently encodable moderatelength high-rate irregular LDPC codes**. IEEE Trans. Communications, vol. 52, no. 4, pp. 564-571, Abr. 2004.

[37] ——. **Performance of efficiently encodable low-density parity-check codes in noise bursts on the EPR4 channel.** IEEE Trans. Magn., vol. 40, no. 2, pp. 507–512, Mar.

[38] R. M. Tanner; **A Recursive Approach to Low Complexity Codes.** IEEE Transactions on Information Theory, vol., IT-27, no. 25. 1981.

[39] J. Xu; L. Chen; L.-Q. Zeng, L. Lan; S. Lin. **Construction of lowdensity parity-check codes by superposition**. *IEEE Trans. Commun.*, vol. 53, no. 2, pp. 243–251, Feb. 2005.

[40] M. Fossorier. **Quasi-cyclic low density parity check codes from circulant permutation matrices**. IEEE Trans. Inf. Theory, no. 8, pp. 1788–1793, Aug. 2004.

[41] T. Richardson; R. Urbanke. **The capacity of low-density parity-check codes under message-passing decoding**. IEEE Trans. Inf. Theory, vol. 47, no. 2, pp. 599–618, Feb. 2001.

[42] D.R. Hankerso, D.G. Hoffman. **Coding Theory and Cryptography: The Essentials**. segunda edição, New York, Marcel Dekker, 1991.

[43] P. J Davis. **Circulant Matrices**. Chelsea Publishing, New York, 1994

[44] J. Xu, L. Chen; L.-Q. Zeng; L. Lan; S. Lin. **Construction of lowdensity parity-check codes by superposition.** IEEE Trans. Commun., vol. 53, no. 2, pp. 243–251, Feb. 2005.

[45] H. Song; J. R. Cruz. **Reduced-complexity decoding of ary LDPC codes for magnetic recording**. IEEE Trans. Magn., vol. 39, no. 2, pp. 1081–1087, Mar. 2003

[46] F. Peng; M. Yang; W. E. Ryan. **Simplified eIRA code design and performance analysis for correlated Rayleigh fading channels**. IEEE Trans. Wireless Commun., vol. 5, no. 4, pp. 720–725, Apr. 2006.

[47] H.A. Loeliger. **Signal sets matched to grups**. IEEE Trans. Inform. Theory, vol IT-37, pp. 1675-1682, Nov. 1991

[48] C. Di; D. Proietti; I. E. Telatar. T. J. Richardson; R. L. Urbanke. **Finite-length analysis of low-density parity-check codes on the binary erasure channel**. IEEE Trans. Inform. Theory, vol. 48, no. 6, pp. 1570–1579, June 2002.

Apêndice A

Conceitos de álgebra

Neste apêndice são mostrados os conceitos matemáticos utilizados no capítulo 2.

A.1 Grupos

Um conjunto não vazio G munido com uma operação binária $(a,b) \to a * b$ é um grupo se as seguintes condições são satisfeitas:

1. A operação é associativa: $a * (b * c) = (a * b) * c$, $\forall a,b,c \in G$.

2. Existe um elemento neutro, isto é, $\exists e \in G$ tal que $e * a = a * e = a$, $\forall a \in G$.

3. Todo elemento possui um elemento inverso, isto é, $\forall a \in G$, $\exists b \in G$ tal que $a * b = b * a = e$

O grupo é abeliano ou comutativo se também vale:

4. A operação é comutativa, isto é, $a * b = b * a$, $\forall a,b \in G$.

Para simplificar a notação usaremos ab em vez de $a * b$.

Seja G um grupo. Diz-se que G é *finito* se ele contém um número finito de elementos. Caso contrário, ele é *infinito*. A *ordem* ou *cardinalidade* de G, denotada por $|G|$, é o número de elementos de G.

Se G e H são dois grupos, então o *produto direto* (*externo*) de G com H, denotado por $G \times H$, é o conjunto de todos os pares ordenados (g , h), onde $g \in G$ e $h \in H$, com a operação binária $(g , h) * (g', h') = (gg', hh')$.

Tem-se que $G \times H$ é um grupo com elemento identidade (e , e) e o elemento inverso de (g , h) é (g^{-1} , h^{-1}). Assim, $G^2 = G \times G$. Generalizando $G^n = G \times G \times \cdots \times G$.

Definição A.1 *Um subconjunto não vazio H de um grupo G é um subgrupo de G, denotado por* $H \leq G$ *,quando com a operação herdada de G, H é um grupo.*

Proposição A.1 *Seja G um grupo e H um subconjunto não vazio de G. Então H é um subgrupo de G se, e somente se, as seguintes condições são satisfeitas:*

$$\begin{aligned} &1.\ \textit{Para todos } h_1, h_2 \in H, \text{ tem-se } h_1 h_2 \in H. \\ &2.\ \textit{Para todo } h \in H, \text{ tem-se } h^{-1} \in H. \end{aligned} \tag{A.1}$$

Seja X um subconjunto não vazio de G e $\mathcal{F} = \{H : H \leq G \text{ e } X \leq H\}$. Então $\langle X \rangle = \bigcap_{H \in \mathcal{F}} H$ é o menor subgrupo de G contendo X e chamado o *subgrupo gerado* por X. Se X é um conjunto finito, denota-se $\langle X \rangle$ o conjunto gerado por X.

Uma *partição* de um conjunto não vazio Ω é um conjunto $P(\Omega) = \{\Gamma : \Gamma \subset \Omega, \Gamma \neq \varnothing\}$ tal que as seguintes condições são satisfeitas:

$$\begin{aligned} &1.\ \Gamma_1 \cap \Gamma_2 = \varnothing, \forall \Gamma_1, \Gamma_2 \in P(\Omega), \Gamma_1 \neq \Gamma_2 \\ &2.\ \Omega = \bigcup_{\Gamma \in P(\Omega)} \Gamma \end{aligned} \tag{A.2}$$

Sejam G um grupo e H um subgrupo de G. Dado $a \in G$, o conjunto $aH = \{ah: \forall h \in H\}$ é chamado a *classe lateral à esquerda* de H em G determinada por a. De modo semelhante, define-se a classe lateral à direita Ha de H em G. O conjunto de todas as classes laterais à esquerda de H em G formam uma partição de G, que denotamos por G/H .

Definição A.2 *Dados $a,b \in G$, dize-se que "a" é congruente a "b" módulo H se $a^{-1}b \in H$, que denota-se por $a \equiv b \pmod{H}$.*

Em [30] verifica-se que a relação de congruência "≡" é uma relação de equivalência em G e que a classe de equivalência determinada por a é igual a classe lateral à esquerda aH. O elemento a é chamado um *representante* da classe de equivalência. Nota-se também que existe uma correspondência biunívoca entre o conjunto das classes laterais à esquerda de H em G e o conjunto das classes laterais à direita de H em G. A cardinalidade do conjunto das classes laterais à esquerda (ou à direita) de H em G é chamado o índice de H em G, que denota-se por $[G : H]$.

Teorema A.1 (Lagrange) Seja *G* um grupo e *H* um subgrupo de *G*. Então A ordem de H divide a ordem de G. Em particular, se *G* é finito, tem-se que a ordem de *G* é o produto da ordem de *H* pelo índice de H em G.

A.1.1 Grupo Linear $GL(n,\mathbb{R})$ das Matrizes n x n

Seja $M_n(\mathbb{R})$ o conjunto de todas as matrizes $n \times n$ sobre $\mathbb{R}$. Então $GL(n,\mathbb{R}) = \{A \in M_n(\mathbb{R}): \det(A) \neq 0\}$ com a operação usual de multiplicação de matrizes é um grupo não abeliano, chamado grupo linear geral. De fato, sejam A**,** B $\in GL(n, \mathbb{R})$. Então, pelo Teorema de Binet, $\det(AB) = \det(A)\det(B) \neq 0$. Logo, AB $\in$ GL(n, $\mathbb{R}$). Assim, o produto usual de matrizes é uma operação binária em $GL(n,R)$. Tem-se que esta operação binária é associativa e $I \in M_n(\mathbb{R})$ é o elemento identidade de $GL(n, \mathbb{R})$. Finalmente, se $A \in M_n(\mathbb{R})$ é tal que $D = \det(A) \neq 0$, então $A^{-1} = \frac{1}{D}\text{adj}(A)$ é a inversa de A e $\det(A^{-1}) = \frac{1}{\det(A)} \neq 0$. Assim, $A^{-1} \in GL(n,\mathbb{R})$ e $A^{-1}A = AA^{-1} = I$.

A.1.2 Grupo das classes restos

Seja $m > 1$ um número inteiro. De acordo com a Definição 2.2 dados $a, b \in \mathbb{Z}$, *a é congruente* a *b* módulo *m* se, e somente se, $m \mid (a-b)$, leia-se "*m* divide $(a-b)$", ou seja, existe $q \in \mathbb{Z}$ tal que $a-bmq$, ou ainda $a = mq + b$. Em [30] mostra-se que toda relação de equivalência esta associada a uma partição e um conjunto quociente. O conjunto $\mathbb{Z}_m$ denota o conjunto de todas essas classes de equivalência gerada por um elemento *a* côngruo a *m*. Com a notação $\bar{a}$ descreve-se todos os elementos inteiros que são côngruos a *a* modulo *m* ou que deixa resta *m* na divisão por *a*. Por este motivo diz-se que $\mathbb{Z}_m$ é o conjunto das *classes resto ou residuais módulo m* ou *segundo m*. Assim, por exemplo, $\bar{1},\bar{2},\bar{3},\bar{4}$, denota todas as divisões que deixam resto 0, 1, 2, 3, 4 por um inteiro qualquer *a*.

Seja $m > 1$ um número inteiro. O conjunto quociente de *a* por *m* ou $\mathbb{Z}_m$ será denotado por $\mathbb{Z}_m = \{\bar{0}, \bar{1}, \bar{2}, \ldots, \overline{m-1}\}$.

Sejam $\bar{a}$ e $\bar{b}$ elementos em $\mathbb{Z}_m$, defini-se $\bar{a} \oplus \bar{b} = \overline{a \oplus b}$ e $\bar{a} \bullet \bar{b} = \overline{a \bullet b}$. Em [30] mostra-se que $(\mathbb{Z}_m, \oplus)$ e $(\mathbb{Z}_m^*, \bullet)$ munido dessas operações forma um Grupo comutativo. Para $m = 2$, o conjunto $\mathbb{Z}_2 = \{\bar{0}, \bar{1}\}$, cujas operações $\oplus$ e $\bullet$ são definidas pelas tábuas a seguir, é usualmente chamado de *Corpo binário* e é denotado por *GF*(2). Este conjunto tem um importante papel na teoria da codificação de sinais e é bastante usado em sistemas de transmissão digital.

$\oplus$	$\bar{0}$	$\bar{1}$
$\bar{0}$	$\bar{0}$	$\bar{1}$
$\bar{1}$	$\bar{1}$	$\bar{0}$

$\bullet$	$\bar{0}$	$\bar{1}$
$\bar{0}$	$\bar{0}$	$\bar{0}$
$\bar{1}$	$\bar{0}$	$\bar{1}$

Seja *p* um número primo. Quando $m = p$ o conjunto $\mathbb{Z}_p = \{\bar{0}, \bar{1}, \bar{2}, \ldots, \overline{p-1}\}$ é também um grupo sob as operações $\oplus$ e $\bullet$ anteriormente definidas. Este grupo é conhecido como um *corpo primo* e denotado por $GF(p)$. Para qualquer inteiro *m*, é possível estender o corpo primo $GF(p)$ para o corpo de p^m elementos, o qual é chamado uma *extensão* de $GF(p)$ e denotado por $GF(p^m)$. Além disso, é possível provar que a ordem de qualquer corpo finito é uma potência de um primo. Corpos finitos são chamados corpos de *Galois* em homenagem ao seu descobridor *Evarist Galois*. As tábuas seguintes mostram as operações anteriormente definidas com o conjunto $\mathbb{Z}_7$ onde, por simplificação, utilizou-se o símbolo + e foi extraída a barra sob os números para notação deste conjunto.

+	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	0
2	2	3	4	5	6	0	1
3	3	4	5	6	0	1	2
4	4	5	6	0	1	2	3
5	5	6	0	1	2	3	4
6	6	0	1	2	3	4	5

.	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

Definição A.3 *Um subgrupo H de um grupo G é chamado normal de G, em símbolos* $H \triangleright G$, *se* $aha^{-1} \in H, \forall a \in G, h \in H$.

Neste caso, *G/H* com a operação binária $(aH)(bH) = abH$ é um grupo.

Sejam G e H dois grupos. Uma função ϕ de *G* em *H* é um *homomorfismo de grupos se* ϕ(*ab*) = ϕ(*a*)ϕ(*b*), para todos *a,b* ∈ *G*. Neste caso, a *imagem* de ϕ é o conjunto $I_m\phi$={*h*:*h*= *ϕ(g)* para algum $g \in$ G} = {ϕ(*g*):*g* ∈ G}. O *núcleo* de ϕ é o conjunto $\ker\phi = \{g \in G : \phi(g) = e\}$. Em [30] verifica-se que Imϕ é um subgrupo de *H* e kerϕ é um subgrupo normal de G.

Um homomorfismo de grupos ϕ:$G \to H$ é um isomorfismo se ϕ é bijetora. Quando existir um isomorfismo entre *G* e *H* diz-se que *G* e *H* são isomorfos e denota-se por $G \simeq H$. Um endomorfismo de um grupo *G* é um homomorfismo ϕ :G → G e denota-se por $\mathrm{End}(G) = \{\phi : G \to G : \phi \text{ é um homomorfismo}\}$.Um *automorfismo* de um grupo G é um isomorfismo ϕ:$G \to H$, denota-se por $\mathrm{Aut}(G) = \{\phi : G \to G : \phi \text{ é um isomomorfismo}\}$.

Teorema A.2 Sejam *G, H dois grupos e ϕ:G → H um homomorfismo de grupos. Então*

$$\frac{G}{\ker\phi} \simeq \mathrm{Im}\,\phi \tag{A.3}$$

Definição A.4. Seja *G* um grupo e *X* um conjunto. Então *G age em X* se existe uma aplicação $\sigma : G \times X \to X$, denotado por $\sigma((g,x)) = gx$, tal que:

(1) $g(hx) = (gh)x, \forall g, h \in G, x \in X$; (A.4)

(2) $ex = x, \forall x \in X$.

A aplicação *σ* é chamada de *ação* de *G* em *X*.

Para cada $g \in G$ a aplicação $\sigma_g : X \to X$ definida por $\sigma_g(x) = gx$ é uma permutação de *X*, isto é, σ_g é um elemento do grupo simétrico S_X. A aplicação $\varphi : G \to S_X$ definida $\varphi(g) = \sigma_g$ é um homomorfismo de grupo chamado uma *representação* de G em S_X. Reciprocamente, qualquer homomorfismo $\phi : G \to S_X$ define uma ação, $gx = \phi(g)(x)$.

Definição A.5: Seja G um grupo agindo em um conjunto X e $x \in X$. Então o conjunto

$$O(x) \triangleq \{gx : g \in G\} \tag{A.5}$$

é chamado a orbita de x em G.

Definição A.6: Seja G um grupo agindo em um conjunto X. Então o conjunto dos elementos de G que deixam x fixo, isto é,

$$E(x) \triangleq \{g \in G : gx = x\} \tag{A.6}$$

é chamado o *estabilizador* de $x \in$ X.

A.2 Espaços Vetoriais

Definição A.7: Seja F um corpo qualquer. Um *espaço vetorial sobre F* é um conjunto V, não vazio, com duas operações: soma, $+ : V \times V \to V$, e multiplicação por escalar, tais que, para quaisquer **u**, **v**, **w**$\in$ V e a ,$b \in \mathbb{F}$, as propriedades seguintes sejam satisfeita:

i) (**u** + **v**) + **w** = **u** + (**v** + **w**)

ii) **u** + **v** = **v** + **u**

iii) Existe $\mathbf{0} \in V$ tal que **u** + **0** = **u** (**0** é chamado vetor nulo)

iv) Existe **–u**$\in V$ tal que **u** + (**-u**) = 0

v) a(**u**+**v**) = a**u** +a**v**

vi) (a + b)**v** = a**v** +b**v**

vii) (ab)**v** = a(b**v**)

viii) 1**u** = **u**

Utiliza-se a palavra vetor para designar os elementos de V. Assim, por exemplo, se for considerado o espaço vetorial $V = M_{2\times 2}$ os vetores serão matrizes.

Exemplo A.1: *O conjunto dos vetores do espaço* $V = \mathbb{R}^3 = \{(x_1, x_2, x_3);\ x_i \in \mathbb{R}\}$ *é evidentemente um espaço vetorial*

Exemplo A.2.: *Seja* $V = \mathbb{R}^n$. *Considere* $\mathbf{u} = (x_1, x_2, ..., x_n), \mathbf{v} = (y_1, y_2, ..., y_n)$ *vetores em V tais que* $\mathbf{u} + \mathbf{v} = (x_1 + y_1, x_2 + y_2, ..., x_n + y_n)$ *e* $a\mathbf{u} = (ax_1, ax_2, ..., ax_n)$, *então mostra-se que V com esta operação é um espaço vetorial*.

Exemplo A.3.: *Seja* $V = M_{2\times2}$ *o conjunto das matrizes constituídas por duas linhas e duas colunas com elementos reais. Então se* **u** *e* **v**∈ *V, tem-se:*

$$\mathbf{u} = \left\{ \begin{bmatrix} a_1 & b_1 \\ c_1 & d_1 \end{bmatrix} \; a_1, b_1, c_1, d_1 \in \mathbb{R} \right\} \tag{A.7}$$

$$\mathbf{v} = \left\{ \begin{bmatrix} a_2 & b_2 \\ c_2 & d_2 \end{bmatrix} \; a_2, b_2, c_2, d_2 \in \mathbb{R} \right\}$$

e portanto o conjunto V com a operação usual de adição de matrizes e produto escalar usual é um espaço vetorial sobre $\mathbb{R}$

Por simplificação de notação, daqui por diante será escrito apenas o espaço vetorial *V* para designar o espaço vetorial sobre o corpo F, real quando $F = \mathbb{R}$ ou complexo quando $F = \mathbb{C}$.

A.2.1 Subespaços Vetoriais

Definição A.8: Dado um espaço vetorial *V*, um subconjunto *W*, não vazio, será subespaço vetorial de V se:

i) Para quaisquer **u**, **v**∈ *W* tem-se $\mathbf{u} + \mathbf{v} \in W$.

ii) Para quaisquer $a \in \mathbb{R}$ e **u**∈ *W* tem-se $a\mathbf{u} \in W$.

Observações:

a) Denota-se W subespaço de V pela notação $W \subseteq V$
b) As condições da definição acima garantem que ao operar-se em *W*, não se obtem um vetor fora de *W*. Isto é, o fechamento de *W* o torna também em um espaço vetorial.
c) O vetor nulo está contido em *W*
d) Todo espaço vetorial admite pelo menos dois subespaços vetoriais: o conjunto formado pelo vetor nulo e o próprio espaço vetorial. Estes subespaços são chamados subespaços triviais

Exemplo A.4.: *Considere o sistema linear homogêneo abaixo:*

$$\begin{cases} 2x + 4y + z = 0 \\ x + y + 2z = 0 \\ x + 3y - z = 0 \end{cases} \quad \text{(A.8)}$$

e na forma matricial tem-se:

$$\begin{bmatrix} 2 & 4 & 1 \\ 1 & 1 & 2 \\ 1 & 3 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad \text{(A.9)}$$

e sejam $\mathbf{u} = (x_1, x_2, x_3), \mathbf{v} = (y_1, y_2, y_3)$ *vetores solução do sistema. Se* **A** *representa a matriz* (3x3) e **0** *o vetor nulo coluna do sistema* (A.9) *tem-se* $\mathbf{Au} = \mathbf{0}$ e $\mathbf{Av} = \mathbf{0}$ *portanto*:

$$\mathbf{A(u + v) = Au + Av = 0 + 0 = 0} \quad \text{(A.10)}$$
$$a\mathbf{Au} = a\mathbf{0} = \mathbf{0}$$

Isto nos diz que o conjunto formado pelas soluções de um sistema homogêneo é um subespaço vetorial do $\mathbb{R}^3$.

Exemplo A.5.: Seja $V = \mathbb{R}^2$ e *W* uma reta deste plano que não passa pela origem, por exemplo, $y = x+1$. Então, *W* não é subespaço de *V*, pois o vetor nulo (0,0) não pertence a *W*.

Teorema A.3: *Sejam* $W_1, W_2 \subseteq V$ *. Então* $W_1 \cap W_2 \subseteq V$.

Teorema A.4: *Sejam* $W_1, W_2 \subseteq V$ *. Então* $W_1 + W_2 \subseteq V$ *, onde*

$$W_1 + W_2 = \{\mathbf{v} \in V; \mathbf{v} = \mathbf{w_1} + \mathbf{w_2},\ \mathbf{w_1} \in W_1 \text{ e } \mathbf{w_1} \in W_1\}$$

A.2.2 Dependência e Independência Linear

Seja V um espaço vetorial. Considere $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_n \in V$ e $a_1, a_2, ..., a_n \in F$. Então o veto $\mathbf{v} = a_1\mathbf{v}_1 + a_2\mathbf{v}_2, ..., a_n\mathbf{v}_n = \sum_{i=1}^{n} a_i\mathbf{v}_i \in V$ é chamado uma *combinação linear* dos vetores $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n$

Uma vez fixados vetores $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n$ em *V*, o conjunto *W* de todos os vetores de *V* que são combinação linear destes, é um subespaço vetorial de *V*. O conjunto *W* é chamado subespaço gerado por $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n$ e usa-se a notação $W = \langle \mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n \rangle$ ou $W = \langle V \rangle$. Desta forma, pode-se escrever:

$$W = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n] = \left\{ \mathbf{v} \in V; \mathbf{v} = \sum_{i=1}^{n} a_i \mathbf{v}_i, \ a_i \in F \right\} \tag{A.11}$$

Exemplo A.6 $V = \mathbb{R}^3, \mathbf{v} \in V, \mathbf{v} \neq 0$. *Então,* $[\mathbf{v}] = \{a\mathbf{v} : a \in \mathbb{R}\}$, isto é, $[\mathbf{v}]$ *é a reta que contém o vetor* **v**.

Definição A.9: Seja V um espaço vetorial $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n \in V$. Diz-se que o conjunto $\{\mathbf{v}_1, ..., \mathbf{v}_n\}$ é lineamente independente (**LI**) ou que os vetores $\mathbf{v}_1, ..., \mathbf{v}_n$ são LI, se a equação

$$a_1\mathbf{v}_1 + a_2\mathbf{v}_2, ..., a_n\mathbf{v}_n = 0 \tag{A.12}$$

implica que $a_1 = a_2 = ... = a_n = 0$. No caso em que exista algum $a_i \neq 0$ dizemos que $\{\mathbf{v}_1, ..., \mathbf{v}_n\}$ é linearmente dependente (LD), ou que os vetores $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n$ são LD.

Teorema A.5: $\{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n\}$ *é LD se, e somente se, um destes vetores for uma combinação linear dos outros.*

Exemplo A.7 Para $V = \mathbb{R}^3$ *os vetores* $\mathbf{e}_1 = (1,0,0), \mathbf{e}_2 = (0,1,0), \text{e}\ \mathbf{e}_3 = (0,0,1)$ *são LI*

Exemplo A.8 $V = \mathbb{R}^2$ *os vetores* $\{(1,-1),(1,0),(1,1)\}$ *são LD, pois*

$$(0,0) = \frac{1}{2}(1,-1) - (1,0) + \frac{1}{2}(1,1) \tag{A.13}$$

A.2.3 Base de um Espaço Vetorial

Definição A.10: Um conjunto $\{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n\}$ de vetores de V será uma base de V se:

i) $\{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n\}$ é LI.

ii) $[\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n] = V$

Exemplo A.9 $V = \mathbb{R}^2, \mathbf{e}_1 = (1,0)$ e $\mathbf{e}_2 = (0,1)$ *o conjunto* $\{\mathbf{e}_1, \mathbf{e}_2\}$ *é base de V, conhecida como base canônica de* $\mathbb{R}^2$.

Exemplo A.10 *O conjunto* $\{(1,1),(0,1)\}$ *é uma base de.* $V = \mathbb{R}^2$. *De fato:*

i) *V é L.I, pois se* $(0,0) = a(1,0) + b(0,1) = (a, a+b)$, *então* $a = b = 0$.

ii) *Se S* = {(1,1),(0,1)} *então V* =[*S*], *pois dado* **v** = (*x*,*y*)∈ V, *temos*

$$(x, y) = x(1,1) + (y - x)(0,1)$$

ou seja, todo vetor de $\mathbb{R}^2$ *é uma combinação linear dos vetores* $(1,1)$ e $(0,1)$.

Exemplo A.11.: $V = \mathrm{M}_{2\times 2}$. *O conjunto* $\left\{\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}, \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}, \begin{bmatrix}0 & 0\\ 1 & 0\end{bmatrix}, \begin{bmatrix}0 & 0\\ 0 & 1\end{bmatrix}\right\}$ é *uma base de V*

Existem espaços que não tem base finita. Isto acontece principalmente quando se trabalha com espaços de funções. Isto não quer dizer que se trabalha com combinações lineares infinitas, mais sim, que cada vetor do espaço é uma combinação linear finita daquela "base infinita". Ou seja, para cada vetor, pode-se escolher uma quantidade finita de vetores da "base" para, com eles, escrever o vetor dado.

Para obter propriedades acerca das bases de um espaço vetorial, considere as proposições seguintes:

Teorema A.6 *Sejam* $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n$ *vetores não nulos, tais que,* $[\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n] = V$. *Então, dentre estes vetores pode-se extrair uma base de V.*

Teorema A.7 *Seja* $V = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n]$. *Então, qualquer conjunto com mais de n vetores é necessariamente LD (e portanto, qualquer conjunto LI tem no máximo n vetores).*

A.2.4 Operadores Lineares

Daqui por diante um espaço vetorial *V*, salvo menção explícita em contrário, significa um espaço vetorial sobre $\mathbb{R}$ com dimensão dim(*V*) = *n*.

Definição A.11: Um operador linear de V é uma aplicação $T:V \rightarrow V$ tal que

$$T(\alpha\mathbf{u}+\mathbf{v}) = \alpha T(\mathbf{u}) + T(\mathbf{v}), \forall \mathbf{u},\mathbf{v} \in V \text{ e } \alpha \in \mathbb{R} \quad \text{(A.14)}$$

Denota-se por $\mathcal{L}(V,V)$ o conjunto de todos os operadores lineares de V. Neste caso, $\mathcal{L}(V,V)$ é um espaço vetorial com $\dim(\mathcal{L}(V,V)) = n^2$. Denota-se por GL($V$) o conjunto de todos elementos de invertíveis de $\mathcal{L}(V,V)$.

Afirmação: GL(V) é um grupo isomorfo ao grupo GL(n,$\mathbb{R}$).

De fato, sejam $T \in$ GL(V) e $\{\mathbf{u}_1,...,\mathbf{u}_n\}$ uma base fixada de V. Como $T(\mathbf{u}_j) \in V$, $j=1,\ldots,n$, tem-se que existem únicos $a_{i,j} \in \mathbb{R}$, tais que $T(\mathrm{u}_j) = \sum_{i=1}^{n} a_{ij}\mathbf{u}_j$, $j = 1,...,n$. Fazendo $\mathrm{A} = (a_{ij})$, obtém-se para cada $T \in$ GL(V) uma única matriz A. Assim, a aplicação ϕ:GL(V)$\rightarrow$GL(n, $\mathbb{R}$) definida por ϕ(T) = A é um isomorfismo.

Definição A.12: Um escalar $\lambda \in \mathbb{R}$ é um autovalor de $T \in \mathcal{L}(V, V)$, se existir u $\in V$, u $\neq 0$ tal que T(u) = λu. O vetor u é chamado o *autovetor* de T associado ao autovalor λ.

Lema A.1 *Autovetores associados a autovalores distintos são sempre linearmente independentes*.

Seja W um subespaço de V e T $\in \mathcal{L}(V, V)$. Diz-se que W é invariante em relação a T se T(W) $\subseteq W$, isto é, T(w) $\in W$, $\forall$w $\in W$.

Definição A.13: Seja V um espaço vetorial de dimensão n sobre $\mathbb{R}$ equipado com o produto interno usual. *A norma quadrática* ou *peso Euclidiano* $N(\mathbf{x}) = \|\mathbf{x}\|^2$ de um vetor x $\in$ V é a soma dos quadrados de suas componentes, isto é,

$$\mathrm{N}(\mathbf{x}) = \langle \mathbf{x}, \mathbf{x} \rangle = \mathbf{x} \cdot \mathbf{x}^{\mathrm{t}}. \quad \text{(A.15)}$$

A *distância Euclidiana quadrática* entre dois vetores x, y $\in V$ é a norma quadrática de sua diferença, isto é,

$$\mathrm{d}^2(\mathbf{x}, \mathbf{y}) = N(\mathbf{x} - \mathbf{y}). \quad \text{(A.16)}$$

Seja $\mathcal{B} = \{\mathbf{x}_1,\ldots,\mathbf{x}_n\}$ uma base para $\mathbb{R}^n$. Diz-se que a base $\{\mathbf{y}_1,\ldots,\mathbf{y}_n\}$ para $\mathbb{R}^n$ é uma base dual de B se $\langle \mathbf{x}_i, \mathbf{y}_i \rangle = \delta_{ij}$, pois existe um único funcional $f_x(\mathbf{y}) = \langle \mathbf{x},\mathbf{y} \rangle$ associado com $x \in \mathbb{R}^n$ e, assim, pode-se identificar o espaço dual $\left(\mathbb{R}^n\right)^*$ com $\mathbb{R}^n$.

Definição A.14: Uma *isometria* ou um *movimento rígido* em $\mathbb{R}^n$ é uma aplicação T: $\mathbb{R}^n \rightarrow \mathbb{R}^n$ que preserva distância, isto é,

$$\|T(\mathbf{x}) - T(\mathbf{y})\| = \|\mathbf{x} - \mathbf{y}\|, \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n \tag{A.17}$$

Denota-se por Isom($\mathbb{R}^n$) o conjunto de todas as isometrias de $\mathbb{R}^n$.

Afirmação: Isom($\mathbb{R}^n$) é um grupo.

De fato, sejam S, $T \in$ Isom($\mathbb{R}^n$). Se $S \circ T$ representa a composição usual de aplicações tem-se:

$$\begin{aligned} \|S \circ T(\mathbf{x}) - S \circ T(\mathbf{y})\| &= \|S(T(\mathbf{x})) - S(T(\mathbf{y}))\| \\ &= \|T(\mathbf{x}) - T(\mathbf{y})\| \\ &= \|\mathbf{x} - \mathbf{y}\|, \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n \end{aligned} \tag{A.18}$$

Logo, $S \circ T \in$ Isom($\mathbb{R}^n$). Assim, a composição usual de aplicações é uma operação binária em Isom($\mathbb{R}^n$). É claro que esta operação binária é associativa e $\mathbf{I} \in \mathcal{F}(\mathbb{R}^n, \mathbb{R}^n)$

é o elemento identidade de Isom($\mathbb{R}^n$). Sejam $\mathbf{x},\mathbf{y} \in \mathbb{R}^n$ e $T \in$ Isom($\mathbb{R}^n$). Se $T(\mathbf{x}) = T(\mathbf{y})$, então $0 = \|T(\mathbf{x}) - T(\mathbf{y})\| = \|\mathbf{x} - \mathbf{y}\| \Rightarrow \mathbf{x} = \mathbf{y}$. Logo, T é injetor. Para provar que todo elemento $T \in$ Isom($\mathbb{R}^n$) é sobrejetor, primeiro apresenta-se algumas definições:

Sejam x, y $\in \mathbb{R}^n$. O *segmento de reta* de extremos x e y é o conjunto $[\mathbf{x}, \mathbf{y}] = \{(1-t)\mathbf{x} + t\mathbf{y} : 0 \le t \le 1\}$. Um subconjunto não vazio S de $\mathbb{R}^n$ é chamado *convexo*

se $[\mathbf{x},\mathbf{y}] \subseteq S, \forall \mathbf{x},\mathbf{y} \in S$. Dados x,y,z $\in \mathbb{R}^n$ tais que $\|\mathbf{x}-\mathbf{z}\| = \|\mathbf{x}-\mathbf{y}\| + \|\mathbf{y}-\mathbf{z}\|$. Então $\|T(\mathbf{x})-T(\mathbf{z})\| = \|T(\mathbf{x})-T(\mathbf{y})\| + \|T(\mathbf{y})-T(\mathbf{z})\|$, isto é, *T*(*S*) é convexo, para todo subconjunto convexo *S* de $\mathbb{R}^n$. Seja $\{x_1,...,x_n\}$ uma base qualquer de $\mathbb{R}^n$. Então $S_0 \subset S_1 \subset ... \subset S_n$, onde $S_i = \langle \mathbf{x}_1,...,\mathbf{x}_i \rangle, i$ =1,..., *n-1* e $S_n=\mathbb{R}^n$. Logo, $T(S_0) \subset T(S_1) \subset ... T(S_n)$ e $\dim T(S_{i+1}) > \dim T(S_i)$, i= 0,...,*n*-1,, pois os S_i são convexos. Assim, $\dim T(S_n) \geq n$. Portanto, $T(S_n) = \mathbb{R}^n$, isto é, T é sobrejetora.

Exemplo A.12.: *Se t ∈ ℝⁿ, então a aplicação* $T_t : \mathbb{R}^n \to \mathbb{R}^n$, *definida por* $T_t(\mathbf{x}) = \mathbf{x} + \boldsymbol{t}, \forall \mathbf{x} \in \mathbb{R}^n$, *é um movimento rígido, chamado a translação à direita por t. É claro que* $T_t(0) = t$, *de modo que* T_t *não é um operador linear se t ≠ 0. Denota-se por T(ℝⁿ) o conjunto de todas as translações à direita, então T(ℝⁿ) é um subgrupo de Isom(ℝⁿ) isomorfo ao grupo aditivo* $(\mathbb{R}^n, +), T \leftrightarrow t$.

Definição A.15: Uma operador linear $S : \mathbb{R}^n \to \mathbb{R}^n$ é *ortogonal* se $\|S\mathbf{x}\| = \|\mathbf{x}\|, \forall \mathbf{x} \in \mathbb{R}^n$. Note que todo operador ortogonal é um movimento rígido. Denota-se por $O(\mathbb{R}^n)$ o conjunto de todos os operadores ortogonais, então $O(\mathbb{R}^n)$ é um subgrupo de Isom($\mathbb{R}^n$).

Lema A.2 Sejam $S : \mathbb{R}^n \to \mathbb{R}^n$ *um operador linea*r e $\{\mathbf{e}_1,...\mathbf{e}_n\}$a *base canônica* de $\mathbb{R}^n$. Então $S \in O(\mathbb{R}^n)$ se, e somente se, $\{S\mathbf{e}_1,...,S\mathbf{e}_n\}$ é uma base *ortonormal* de $\mathbb{R}^n$.

Lema A.3 *Todo movimento rígido que fixa a origem é um operador linear*.

Proposição A.2 Todo elemento $f \in \text{Isom}(\mathbb{R}^n)$ pode se escrito de modo único como $f = T \circ S$, *onde* $T \in T(\mathbb{R}^n)$ e $S \in O(\mathbb{R}^n)$.

Corolário A.1 *Seja* $T \in \text{Isom}(\mathbb{R}^n)$. *Então* $T(0) = 0$ *se, e somente se,* $\langle T(\mathbf{x}), T(\mathbf{y})\rangle = \langle \mathbf{x}, \mathbf{y}\rangle$, $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n$.

Definição A.16: Sejam V um espaço vetorial com produto interno e S um subconjunto não vazio de V. O conjunto $S^{\perp} = \{\mathbf{u} \in V : \langle \mathbf{u}, \mathbf{v}\rangle = 0, \forall \mathbf{v} \in S\}$, é um subespaço de V, chamado o complemento ortogonal de $\langle S\rangle$. Além disso, se W é um subespaço de V invariante com relação a T, então $W^{\perp}$ é invariante com relação a T^t.

Seja $T \in \text{Isom}(\mathbb{R}^n)$ tal que $T(\mathbf{0}) = \mathbf{0}$. Diz-se que T é uma rotação (ou preserva orientação) se $\det T = 1$ e inverte orientação se $\det T = -1$.

Sejam V um espaço vetorial e U, W subespaços de V tais que $V = U \oplus W$. Uma reflexão em U ao longo de W é um operador linear R: $V \to V$ definida por $R(\mathbf{u} + \mathbf{w}) = \mathbf{u} - \mathbf{w}$ para todo $\mathbf{u} \in U$ e $\mathbf{w} \in W$.

Seja T um operador linear. Considere $\beta = \{\mathbf{v}_1, ..., \mathbf{v}_n\}$ bases de V. Então $T(\mathbf{v}_1), ..., T(\mathbf{v}_n) \in V$ e portanto:

$$\begin{aligned} T(\mathbf{v}_1) &= a_{11}\mathbf{v}_1 + \cdots + a_{n1}\mathbf{v}_n \\ &\vdots \qquad\qquad \vdots \\ T(\mathbf{v}_n) &= a_{1n}\mathbf{v}_1 + \cdots + a_{nn}\mathbf{v}_n \end{aligned} \tag{A.19}$$

A transposta da matriz de coeficiente deste sistema, denotada por $[T]_{\beta'}^{\beta}$, é chamada matriz de T em relação às bases β.

$$[T]^{\beta} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} = \text{A} \tag{A.20}$$

Proposição A.3 *Seja V um espaço vetorial de dimensão finita sobre R e* $T \in \mathcal{L}(V,V)$. *Então as a seguintes condições são equivalentes*:

1.T é uma reflexão;

2. $V = \ker(T - I) \oplus \ker(T + I)$;

3. Existe uma base de V tal que T é representada pela matriz diagonal da forma

$$\begin{bmatrix} I & 0 \\ 0 & -I \end{bmatrix}$$

4. $T^2 = I$.

Apêndice B

Grupos casados e polígonos em ninho

Neste apêndice são apresentados os procedimentos matemáticos que geram as matrizes bases utilizadas no capítulo 3, o leitor interessado em mais detalhes pode consultar [10], [43].

B.1 Grupos casados

Uma *constelação de sinais S* é qualquer subconjunto discreto em $\mathbb{R}^n$. Os elementos de uma constelação de sinais *S* são chamados *pontos de sinais*. Um *código do espaço Euclidiano* é um subconjunto de s^I, onde $I \subseteq \mathbb{Z}$. Uma constelação de sinais S é *geometricamente uniforme*, [10], se dados $s_1, s_2 \in S$ existe $\varphi \in Isom\left(\mathbb{R}^N\right)$ tal que

$$\varphi(s_1) = s_2 \text{ e } \varphi(S) = S. \tag{B.1}$$

Se $\Gamma(S) = \left\{\varphi \in Isom\left(\mathbb{R}^N\right) : \varphi(s) = s\right\}$, então S é orbita (A.2.5) de qualquer ponto $s_0 \in S$ sob $\Gamma(S)$, isto é,

$$S = \left\{\varphi(s_0) : \varphi \in \Gamma(S)\right\} = \bigcup_{\varphi \in \Gamma(S)} \left\{\varphi(s)\right\}. \tag{B.2}$$

Uma constelação de sinais S é casada com um grupo G, se existe um mapeamento μ de G sobre S tal que [47]

$$d(\mu(g), \mu(h)) = d(\mu(e), \mu(g^{-1}h)), \quad \forall g, h \in G \tag{B.3}$$

onde *d*(.,.) é a distância Euclidiana quadrática. O mapeamento μ é chamado de *mapeamento casado*. Quando o mapeamento μ é um *rotulamento casado*, isto é, se G é isomorfo a G(S) então μ é um *rotulamento isométrico* . Seja C um código linear sobre $\mathbb{Z}_m$ de comprimento *n*. Define-se [47],

$$\phi : C \to \mathbb{R}^N, \phi\left((c_1, ..., c_n)\right) \triangleq \sum_{j=1}^{n} A_j B_j \tag{B.4}$$

onde

$$A_j = \left[r_j \cos\left(\frac{2\pi c_j}{m}\right), r_j \operatorname{sen}\left(\frac{2\pi c_j}{m}\right)\right], B_j = \begin{bmatrix} b_{2j-1} \\ b_{2j} \end{bmatrix} \quad \text{(B.5)}$$

e $\{b_1, b_2, ..., b_{2n}\}$ é uma base ortonormal em $\mathbb{R}^{2n}$. Note que $r_j = 1$, para todo $j = 1, ..., n$. O mapeamento ϕ é chamado de mapeamento canônico.

Teorema B.1 *Sejam* $H = \langle h \rangle \simeq \mathbb{Z}_m$ *e* $K = \langle k \rangle \simeq \mathbb{Z}_{2m}$. *Então, uma constelação de sinais S é casada com um grupo* $G = H \times_\theta K$ *se, e somente se,* $S = S_1 \times_\varphi S_2$ *para* $\varphi(\mu(k)) = \mu_1 \theta(k) \mu_1^{-1}$, *onde* $\mu_1 : H \to S_1$, $\mu_2 : K \to S_2$ *são rotulamentos casados,* $\theta(k) \in Aut(H), \forall k \in K$ *e* $\theta(k)(h) = h^{-1}, \forall h \in H$.]

Exemplo B.1*: Sejam* $H = \mathbb{Z}_3 = \{0,1,2\}, K = \mathbb{Z}_2 = \{0,1\}$ *e o homomorfismo* $\theta : K \to Aut(H)$ *definido por* $\theta(k) = \tau^k, k = 0$, *onde* $\tau(h) = -h, h = 0,1,2$. *Então G* $G = H \times_\theta K$ *é um grupo não comutativo de ordem 6 com a operação*

$$(h,k) +_\theta (h', k') = (h + \theta(k)(h'), k + k'), \forall h, h' \in H \text{ e } k, k' \in K.$$

Tabela B.1 Tabela Caylei do grupo $\mathbb{Z}_3 \times_\theta \mathbb{Z}_2 \cong D_3$

$+_\theta$	0	1	2	3	4	5	*G*
0	0	1	2	3	4	5	(0,0)
1	1	2	0	4	5	3	(1,0)
2	2	0	1	5	3	4	(2,0)
3	3	5	4	0	2	1	(0,1)
4	4	3	5	1	0	2	(1,1)
5	5	4	3	2	1	0	(2,1)

Assim, G é isomorfo ao grupo diedral D_3. *Seja A={0,1,2,3,4,5} um conjunto de rótulos para G. A tabela de Cayley de A é mostrada na Tabela B.1 [10]. Pelo Teorema B.1, a constelação de sinais*

$$S = \left\{ (1,0,1), \left(-\frac{1}{2}, \frac{\sqrt{3}}{2}, 1\right), \left(-\frac{1}{2}, -\frac{\sqrt{3}}{2}, 1\right), (1,0,-1), \left(-\frac{1}{2}, \frac{\sqrt{3}}{2}, -1\right), \left(-\frac{1}{2}, -\frac{\sqrt{3}}{2}, -1\right) \right\}$$

ou a constelação de sinais normalizada $\overline{S} = \frac{1}{\sqrt{2}} S$ é casada com o grupo D3[28].

Exemplo B.2: *Sejam* $H=\mathbb{Z}_4=\{0,1,2,3\}, K=\mathbb{Z}_2=\{0,1\}$ *e o homomorfismo* $\theta: K \to Aut(H)$ *definido por* $\theta(k)=\tau^k, k=0,1$, *onde* $\tau(h)=-h, h=0,1,2,3$. *Então G* $G=H\times_\theta K$ *é um grupo não comutativo de ordem 8 com a operação*

$$(h,k)+_\theta (h',k')=(h+\theta(k)(h'),k+k'), \forall h,h' \in H \text{ e } k,k' \in K.$$

Seja A um conjunto de rótulos para G. a tabela de Cayley de A é mostrada na Tabela x,y, da qual conclui-se que $G=\langle 1,4\rangle$, *é isomorfo ao grupo diedral* D_4. *Pelo Teorema B.1 a constelação de sinais*

$$S=\{(1,0,1),(0,1,1),(-1,0,1),(0,-1,1),(1,0,-1),(0,1,-1),(-1,0,-1),(0,-1,-1)\}$$

ou a constelação de sinais normalizada $\overline{S}=\frac{1}{\sqrt{2}}S$ é casada com o grupo D_4.

Tabela B.2 Tabela Cayley do grupo $\mathbb{Z}_4\times_\theta \mathbb{Z}_2 \cong D_4$

$+_\theta$	0	1	2	3	4	5	6	7	G
0	0	1	2	3	4	5	6	7	(0,0)
1	1	2	3	0	5	6	7	4	(1,0)
2	2	3	0	1	6	7	4	5	(2,0)
3	3	0	1	2	7	4	5	6	(3,0)
4	4	7	6	5	0	3	2	1	(0,1)
5	5	4	7	6	1	0	3	2	(1,1)
6	6	5	4	7	2	1	0	3	(2,1)
7	7	6	5	4	3	2	1	0	(3,1)

Para construção de constelação de sinais casados de um grupo qualquer é necessário o conhecimento da ideia de d-Cadeia desenvolvida na proposta de [10]. Sejam $S=\{\mathbf{x}_{gi}: 0\le i\le N-1\}\subset \mathbb{R}^N$, onde $\mathbf{x}=(x_0,...,x_{N-1})\in \mathbb{R}^N$,

$$\mathbf{x}_{gi}=(x_{\sigma_{gi}(0)},...,x_{\sigma_{gi}(N-1)}) \tag{B.6}$$

com $g_i\in G,\ g_0=e \text{ e } \sigma_{gi}(0)\in S_G, 0\le i\le N-1.$

Supondo que todos os vetores distantes d de $\mathbf{x}=\mathbf{x}_{g_0}$ são $\mathbf{x}_{gi},...,\mathbf{x}_{gr}$, pode-se construir uma tabela dos elementos do grupo associados aos vetores $\mathbf{x}_{gi}, 0\le i\le r$, da seguinte forma: a primeira linha da tabela será $g_0,...,g_r$. O primeiro elemento a ser localizado na primeira coluna da tabela na (k+1)-ésima linha será o primeiro elemento na k-ésima linha que ainda não apareceu na primeira coluna da tabela. Seja g o primeiro

elemento na k-ésima linha que ainda não apareceu na primeira coluna da tabela. Então a (k+1)-ésima linha será $g, gg_1, ..., gg_r$. A Tabela B.3 ilustra o procedimento.

Tabela B.3 – Construção das d-cadeias

	g_0	g_1	...	g_r
g_0	g_0	g_1	...	g_r
g_1	g_1	g_1^2	...	$g_1 g_r$
⋮	⋮	⋮	⋱	⋮
g	g	gg_1	...	gg_r
⋮	⋮	⋮	⋱	⋮

Quando a j-ésima linha estiver sido escrita e todo elemento nessa j-ésima linha tenha aparecido uma vez na primeira coluna da tabela, o processo é interrompido e a tabela é considerada completa. Note que a tabela tem no máximo $|G|$ linhas. Se

$$d(\mathbf{x}_{g_i}, \mathbf{x}) = d(\mathbf{x}_{g_j g_i}, \mathbf{x}_{g_j}), \; 0 \le i, j \le N-1 \tag{B.7}$$

os vetores representados pelos elementos do grupo na coluna na 2ª,...,*r*-ésima coluna na k-ésima linha estão distante *d* do vetor representado pelo elemento do grupo na primeira coluna dessa linha. Assim, os elementos do grupo na primeira coluna da tabela representam vetores com a propriedade de que todo vetor subsequente está à distância *d* do vetor anterior e também do próximo. O conjunto de vetores que podem ser alcançados de x dessa maneira é chamado de *d-cadeia* iniciando de x. Nota-se que o vetor **x** está incluído nessa d-cadeia e todas as d-cadeias iniciando de x são obtidas da Tabela B.1.

Algoritmo 4.1 - Construção geométrica de constelação de sinais casados com grupos

Passo 1. Associe a G um conjunto qualquer de rótulo *A*.

Passo 2. Construa a tabela de Cayley para *A*.

Passo 3. Para cada linha da tabela de Cayley do Passo 2 associe uma permutação $\sigma_i, \; i \in A$.

Passo 4. Dado $\mathbf{x} = \mathbf{x}_0 = (\mathbf{x}_0, ..., \mathbf{x}_{|A|-1}) \in \mathbb{R}^{|A|}$ encontre

$$\mathbf{x}_i = (\mathbf{x}_{\sigma_i(0)}, ..., \mathbf{x}_{\sigma(|A|-1)}), \; 1 \le i \le |A| - 1. \tag{B.8}$$

Passo 5. Selecione em cada conjunto A_j os elementos do grupo associado aos vetores $\mathbf{x}_j$ que estão a mesma distância Euclidiana quadrática d_j de $\mathbf{x}_0$.

Passo 6. Para cada conjunto A_j do Passo 5 associe uma d_j-cadeia.

Passo 7. Associe a cada do Passo 6 um subgrupo H_j de G.

Passo 8. Faça

$$S_j = \bigcup_{i \in I} S_{ji}, \tag{B.9}$$

onde S_{ji}, são as constelações de sinais casadas com as classes laterais de H_j em G.

Exemplo 3.1 *Encontrar as constelações de sinais com o grupo* $\mathbb{Z}_6$.

Passo 1. Seja A = {0,1,2,3,4,5} um conjunto de rótulo para $\mathbb{Z}_6$.

Passo 2. A Tabela 4.2 é uma tabela de Caylei para o grupo A.

Passo 3. *As permutações associadas ao* ***Passo*** 2 *são*:

$$\sigma_0 = (0),\ \sigma_1 = (012345),\ \sigma_2 = (024)(135),$$
$$\sigma_3 = (03)(14)(25),\ \sigma_4 = (042)(153),$$
$$\sigma_5 = (054321)$$

Passo 4. Dado $\mathbf{x_0} = (x_0, x_1, x_2, x_3, x_4, x_5) \in \mathbb{R}^6$. Então

$$\mathbf{x_1} = (x_1, x_2, x_3, x_4, x_5, x_0),$$
$$\mathbf{x_2} = (x_2, x_3, x_4, x_6, x_0, x_1),$$
$$\mathbf{x_3} = (x_3, x_4, x_5, x_0, x_1, x_2),$$
$$\mathbf{x_4} = (x_4, x_5, x_0, x_1, x_2, x_3),$$
$$\mathbf{x_5} = (x_5, x_0, x_1, x_2, x_3, x_4)$$

Passo 5. *Sejam* $A_1 = \{1,5\}, A_2 = \{2,4\},$ e $A_3 = \{3\}$ *os conjuntos das distâncias Euclidianas quadráticas* d_1, d_2, d_3 *respectivamente.*

Passo 6.

	0	1	5
0	0	1	5
1	1	2	0
2	2	3	1
3	3	4	2
4	4	5	3
5	5	0	4

	0	2	4
0	0	2	4
2	2	4	0
4	4	0	2

	0	3
0	0	3
3	3	0

Passo 7. *Sejam*

$$H_1 = \mathbb{Z}_6,$$
$$H_2 = \{0,2,4\},$$
$$H_3 = \{0,3\}$$

os subgrupos de $\mathbb{Z}_6$ *associados as* d_j*-cadeias* $j = 0,1,2,3$ *do passo* 6.

Passo 8. Tomando

$$S_1 = S_{11} = \{0,1,2,3,4,5\},$$

$$S_2 = \{0,2,4\} \cup \{1,3,5\}, \text{ e}$$

$$S_3 = \{0,3\} \cup \{1,4\} \cup \{2,5\}.$$

As outras constelações de sinais são, a menos de permutações de rótulos, similares a estas.

Os grafos associados a S_1, S_2 e S_3 são o hexágono regular, o antiprisma e o prisma. A Figura B.1 mostra o antiprisma e o prisma casados com $\mathbb{Z}_6$

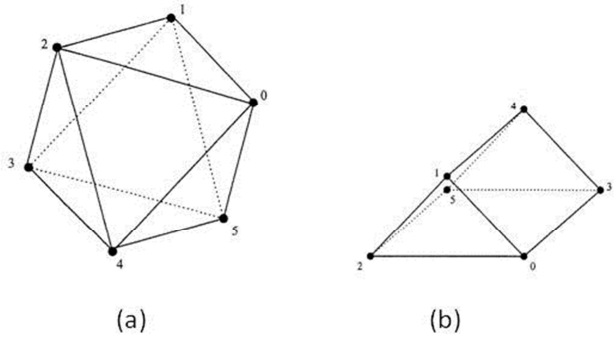


Figura B .1. a) Antiprisma casado com $\mathbb{Z}_6$ e b) prisma casado com $\mathbb{Z}_6$

Seguindo os passos do algoritmo descrito, como no Exemplo B.1, constrói-se o antiprisma e o prisma casado com o grupo diedral D_4. A Figura B.2 mostra o antiprisma e o prisma casados com o grupo D_4.

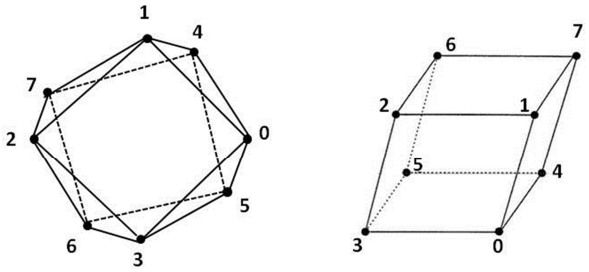


Figura B.2: Antiprisma e prisma casado com D_4

Considere o antiprisma casado com $\mathbb{Z}_6$, Figura B.3, onde as aresta são representadas pelos vetores $\mathbf{w}_i$ (segmento tracejado) e $\mathbf{u}_j$ (segmento contínuo) com i,j = 1,2,3,4,5,6;

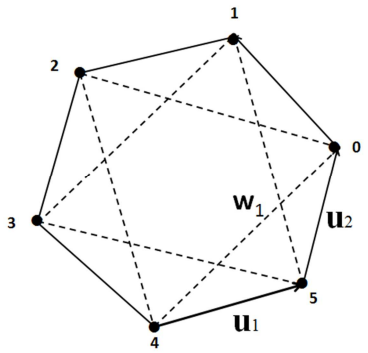


Figura B.3 Soma de vetores no antiprisma

tem-se que o vetor $\mathbf{w}_1 = \mathbf{u}_1 + \mathbf{u}_2$ e que $\mathbf{w}_2,...,\mathbf{w}_6$ são somas dos vetores $\mathbf{u}_i$ descritos nas equações de (B.10), isto é:

$$
\begin{aligned}
\mathbf{w}_1 &= 1\mathbf{u}_1 + 1\mathbf{u}_2 + 0\mathbf{u}_3 + 0\mathbf{u}_4 + 0\mathbf{u}_5 + 0\mathbf{u}_6 \\
\mathbf{w}_2 &= 0\mathbf{u}_1 + 1\mathbf{u}_2 + 1\mathbf{u}_3 + 0\mathbf{u}_4 + 0\mathbf{u}_5 + 0\mathbf{u}_6 \\
\mathbf{w}_3 &= 0\mathbf{u}_1 + 0\mathbf{u}_2 + 1\mathbf{u}_3 + 1\mathbf{u}_4 + 0\mathbf{u}_5 + 0\mathbf{u}_6 \\
\mathbf{w}_4 &= 0\mathbf{u}_1 + 0\mathbf{u}_2 + 0\mathbf{u}_3 + 1\mathbf{u}_4 + 1\mathbf{u}_5 + 0\mathbf{u}_6 \\
\mathbf{w}_5 &= 0\mathbf{u}_1 + 0\mathbf{u}_2 + 0\mathbf{u}_3 + 0\mathbf{u}_4 + 1\mathbf{u}_5 + 1\mathbf{u}_6 \\
\mathbf{w}_6 &= 1\mathbf{u}_1 + 0\mathbf{u}_2 + 0\mathbf{u}_3 + 0\mathbf{u}_4 + 0\mathbf{u}_5 + 1\mathbf{u}_6
\end{aligned}
\qquad \text{(B.10)}
$$

Tomando-se a matriz **P** formada pelos coeficientes de cada uma das equações em (B.10), tem-se que a matriz A (B.11) é dada por $\mathbf{A}=\mathbf{P}^T$.

$$[\mathrm{A}] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \tag{B.11}$$

Considerando-se o antiprisma casado com D_4, analogamente aos passos descritos no antiprisma casado com $\mathbb{Z}_6$, chega-se a matriz A (B.12):

$$[\mathrm{A}] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \tag{B.12}$$

A matriz B (B.13) é a generalização dos resultados descritos anteriormente de um antiprisma casado com um grupo cíclico contendo um grande número de elementos.

$$\mathrm{B} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & \vdots & \vdots & \vdots \\ 0 & 0 & 1 & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \tag{B.13}$$

B.2 Matriz associada a polígonos em ninho

Uma matriz de ordem n $\mathrm{A} = [a_{ij}]$ é uma g-circulante, denotada por $\mathrm{A} = g - \mathrm{circ}(a_1, a_2, ..., a_n)$ se A é escrita da forma:

$$\mathrm{A} = \begin{bmatrix} a_1 & a_2 & ... & a_n \\ a_{n-g+1} & a_{n-g+2} & ... & a_{n-g} \\ a_{n-2g+1} & a_{n-2g+2} & ... & a_{n-2g} \\ \vdots & \vdots & & \vdots \\ a_{g+1} & a_{g+2} & ... & a_g \end{bmatrix} \tag{B.14}$$

Se 0< g <n, cada linha de A é uma linha anterior movida g posições à direita, ou movidas à esquerda n-g posições cíclicas. Se $g > n$, então um movimento de g posições é o mesmo que o movimento de g mod$n$ posições. Por convenção, se g é negativo, o movimento a direita de g posições será equivalente ao movimento a esquerda (-g) posições. Assim, para quaisquer inteiros g e g' com g'=g(mod$n$) uma g'-circulante e uma g-circulante são sinônimas.

Exemplo B.2 : Polígonos em Ninho de ordem 8. Uma 3-circulante de ordem 8 é dada por

$$\mathrm{A} = 3 - \mathrm{circ}(a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8) = \begin{bmatrix} a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 \\ a_6 & a_7 & a_8 & a_1 & a_2 & a_3 & a_4 & a_5 \\ a_3 & a_4 & a_5 & a_6 & a_7 & a_8 & a_1 & a_2 \\ a_8 & a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 \\ a_5 & a_6 & a_7 & a_8 & a_1 & a_2 & a_3 & a_4 \\ a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 & a_1 \\ a_7 & a_8 & a_1 & a_2 & a_3 & a_4 & a_5 & a_6 \\ a_4 & a_5 & a_6 & a_7 & a_8 & a_1 & a_2 & a_3 \end{bmatrix} \qquad \text{(B.15)}$$

Tomando $a_1 = a_3 = 1/2$ e os demais elementos iguais a zero determina-se uma matriz cujo resultado esta associado à construção de polígonos em ninho, Davis [43] faz uma construção desses polígonos começando pelo triângulo. A ideia de "ninho" parte da divisão dos segmentos desta figura em segmentos que somados resultam na unidade. Constroem-se os "ninhos" ligando os pontos desta divisão; o segundo polígono é proporcional ao primeiro, o terceiro é proporcional ao segundo e assim por diante. A Figura B.4 mostra um polígono P_1 proporcional ao um polígono P_2 conforme uma proporção qualquer cuja soma é unitária.

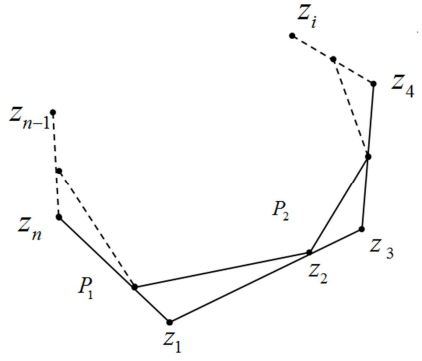


Figura B.4: Polígonos em ninho

Substituindo os valores de a_1 e a_3 em (B.19) chega-se na matriz:

$$A = 3 - \mathrm{circ}(\tfrac{1}{2}, 0, \tfrac{1}{2}, 0, 0, 0, 0, 0) = \begin{bmatrix} \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \end{bmatrix} \tag{B.16}$$

que está associada ao ninho formado por dois quadrados e um octógono, mostrado na Figura B.5.

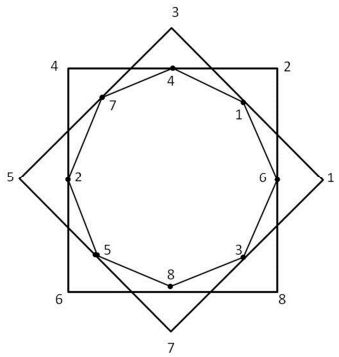


Figura B.5: Ninho formado por dois quadrados e um hexágono

Tomando-se a matriz B=2A e com alguns ajustes algébricos, obtém-se a seguinte matriz:

$$[B]_2 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \tag{B.17}$$