



Rafael Gregorio Lucas D'Oliveira

## **Raio de Empacotamento de Códigos Poset**



**UNIVERSIDADE ESTADUAL DE CAMPINAS  
INSTITUTO DE MATEMÁTICA, ESTATÍSTICA  
E COMPUTAÇÃO CIENTÍFICA**

**Rafael Gregorio Lucas D'Oliveira**

**Raio de Empacotamento de Códigos Poset**

**ORIENTADOR: Prof. Dr. Marcelo Firer**

DISSERTAÇÃO de MESTRADO  
APRESENTADA AO INSTITUTO DE MATEMÁTICA, ESTATÍSTICA E  
COMPUTAÇÃO CIENTÍFICA DA UNICAMP PARA OBTENÇÃO DO  
TÍTULO DE MESTRE EM MATEMÁTICA

**ESTE EXEMPLAR CORRESPONDE À VERSÃO FINAL DA TESE/DISSERTAÇÃO  
DEFENDIDA PELO ALUNO Rafael Gregorio Lucas D'Oliveira,  
E ORIENTADA PELO PROF.DR Marcelo Firer**

A handwritten signature in blue ink that reads "Marcelo Firer". The signature is written in a cursive style and is positioned above a horizontal line.

**Assinatura do Orientador**

**CAMPINAS  
2012  
iii**

FICHA CATALOGRÁFICA ELABORADA POR  
MARIA FABIANA BEZERRA MULLER - CRB8/6162  
BIBLIOTECA DO INSTITUTO DE MATEMÁTICA, ESTATÍSTICA E  
COMPUTAÇÃO CIENTÍFICA - UNICAMP

L962r Lucas D'Oliveira, Rafael Gregorio, 1988-  
Raio de empacotamento de códigos poset / Rafael Gregorio  
Lucas D'Oliveira. – Campinas, SP : [s.n.], 2012.

Orientador: Marcelo Firer.  
Dissertação (mestrado) – Universidade Estadual de Campinas,  
Instituto de Matemática, Estatística e Computação Científica.

1. Conjuntos ordenados. 2. Códigos corretores de erros (Teoria da informação). 3. Métricas sobre ordens parciais. I. Firer, Marcelo, 1961-. II. Universidade Estadual de Campinas. Instituto de Matemática, Estatística e Computação Científica. III. Título.

Informações para Biblioteca Digital

**Título em inglês:** The packing radius of poset codes

**Palavras-chave em inglês:**

Ordered sets

Error-correcting codes (Information theory)

Poset metric

**Área de concentração:** Matemática

**Titulação:** Mestre em Matemática

**Banca examinadora:**

Marcelo Firer [Orientador]

Yoshiharu Kohayakawa

Luciano Panek

**Data de defesa:** 08-08-2012

**Programa de Pós-Graduação:** Matemática

**Dissertação de Mestrado defendida em 08 de agosto de 2012 e aprovada**

**Pela Banca Examinadora composta pelos Profs. Drs.**

*Marcelo Firer*

---

**Prof.(a). Dr(a). MARCELO FIRER**

*Yoshiharu Kohayakawa*

---

**Prof. (a). Dr (a). YOSHIHARU KOHAYAKAWA**

*Luciano Panek*

---

**Prof. (a). Dr (a). LUCIANO PANEK**

# Agradecimentos

Ao CNPq, pelo apoio financeiro.

Ao meu orientador Marcelo Firer, cujas contribuições não podem ser resumidas a uma frase.

Aos examinadores da banca, por ajudarem a melhorar a qualidade da dissertação.

Ao Luciano Panek, cujo trabalho inspirou esta dissertação.

A minha namorada Mariana, por corrigir os meus erros de gramática.

Aos meus pais, por sempre me apoiarem.

A minha família e amigos.

# Sumário

<b>1</b>	<b>Códigos Corretores de Erros</b>	<b>3</b>
1.1	O modelo de Shannon-Weaver . . . . .	3
1.2	Sistemas Discretos de Comunicação . . . . .	4
1.3	Métrica de Hamming . . . . .	11
1.4	Raio de Empacotamento . . . . .	14
<b>2</b>	<b>Códigos Poset</b>	<b>17</b>
2.1	Conjuntos Parcialmente Ordenados . . . . .	17
2.2	Códigos Poset . . . . .	20
<b>3</b>	<b>Raio de Empacotamento de Códigos Poset I</b>	<b>24</b>
3.1	O Raio de Empacotamento de um Vetor . . . . .	24
<b>4</b>	<b>O Problema da Partição</b>	<b>33</b>
4.1	Introdução . . . . .	33
4.2	A Heurística KK . . . . .	35
4.3	CKK . . . . .	39
<b>5</b>	<b>Raio de Empacotamento de Códigos Poset II</b>	<b>42</b>
5.1	O caso dos ideais disjuntos . . . . .	42
5.2	O caso geral . . . . .	45
5.3	O método da diferenciação para posets . . . . .	48
5.4	O raio de empacotamento de códigos poset . . . . .	61

# Abstract

Until the present work, the packing radius of a poset code was only known in the cases where the poset was a chain, hierarchy, a union of disjoint chains of the same size, and for some families of codes. Our objective is to approach the general case of any poset. To do this, we will divide the problem into two parts.

The first part consists in finding the packing radius of a single vector. We will show that this is equivalent to a generalization of a famous NP-hard problem known as “the partition problem”. Then, we will review the main results known about this problem giving special attention to the algorithms to solve it. The main ingredient to these algorithms is what is known as the differentiating method, and therefore, we will extend it to the general case.

The second part consists in finding the vector that determines the packing radius of the code. For this, we will show how it is sometimes possible to compare the packing radius of two vectors without calculating them explicitly.

# Resumo

Até o trabalho presente, só era conhecido o raio de empacotamento de um código poset nos casos do poset ser uma cadeia, hierárquico, a união disjunta de cadeias do mesmo tamanho, e para algumas famílias de códigos. Nosso objetivo é abordar o caso geral de um poset qualquer. Para isso, iremos dividir o problema em dois.

A primeira parte consiste em encontrar o raio de empacotamento de um único vetor. Veremos que este problema é equivalente à uma generalização de um problema NP-difícil famoso conhecido como “o problema da partição”. Veremos então os principais resultados conhecidos sobre este problema dando atenção especial aos algoritmos para resolvê-lo. A receita principal destes algoritmos é o método da diferenciação, e sendo assim, iremos estendê-la para o caso geral.

A segunda parte consiste em encontrar o vetor que determina o raio de empacotamento do código. Para isso, mostraremos como é as vezes possível comparar o raio de empacotamento de dois vetores sem calculá-los explicitamente.

# Introdução

Tradicionalmente a teoria de códigos é feita num espaço vetorial  $\mathbb{F}_q^n$  sobre um corpo finito  $\mathbb{F}_q$  usando-se a métrica de Hamming. Neste âmbito um problema clássico é, fixado  $n$  e  $k$ , encontrar a maior distância mínima que um código linear de comprimento  $n$  e dimensão  $k$  pode atingir. Este problema foi generalizado por Niederreiter em 1991 ([16]) e mais tarde, em 1995 ([2]), Brualdi, Graves e Lawrence, expandindo sobre esse trabalho, introduziram uma nova família de métricas, chamadas de métricas poset, que generalizam a métrica de Hamming.

Um fato surpreendente sobre o raio de empacotamento de um código linear ao considerar a métrica poset é que, diferentemente do que ocorre no caso da métrica de Hamming, este não é necessariamente determinado pelo vetor de peso mínimo, conforme foi observado em [17]. A importância disto se reflete no fato de que a distância mínima costuma ser um dos parâmetros principais de um código, sendo muitas vezes utilizado na hora da decodificação. É útil, então, conseguir determinar o raio de empacotamento de um código poset.

Até o trabalho presente, só era conhecido o raio de empacotamento de um código poset nos casos do poset ser uma cadeia, hierárquico, a união disjunta de cadeias do mesmo tamanho, e para algumas famílias de códigos. Nosso objetivo é abordar o caso geral de um poset qualquer. Para isso, iremos dividir o problema em dois.

A primeira parte consiste em encontrar o raio de empacotamento de um único vetor. Veremos que este problema é equivalente à uma generalização de um problema NP-difícil famoso conhecido como “o problema da partição”. Veremos então os principais resultados conhecidos sobre este problema dando atenção especial aos algoritmos para resolvê-lo. A receita principal destes algoritmos é o método da diferenciação, e sendo assim, iremos estendê-la para o caso geral.

A segunda parte consiste em encontrar o vetor que determina o raio de empacotamento do código. Para isso, mostraremos como é as vezes possível comparar o raio de empacotamento de dois vetores sem calculá-los explicitamente.

## Organização

Salvo menção em contrário, os resultados originais deste trabalho se encontram nos capítulos 3 e 5.

No primeiro capítulo damos uma introdução breve ao assunto dos códigos corretores de erros. Tudo o que é discutido pode ser encontrado em livros textos.

No segundo capítulo introduzimos as definições e conceitos básicos dos códigos posets que serão necessários mais adiante.

No terceiro capítulo começam nossas contribuições. Nele, abordamos o problema de encontrar o raio de empacotamento de um vetor, e mostramos que este problema pode ser visto como um problema de partição. A primeira vez que o problema de encontrar o raio de empacotamento de um código poset é identificado como um problema de partição é em [24].

No quarto capítulo interrompemos a discussão sobre os códigos posets para falar sobre um problema famoso chamado de “problema da partição”. Discutimos os resultados principais já conhecidos sobre o assunto, dando ênfase ao método da diferenciação, receita principal dos melhores algoritmos conhecidos para resolver o problema.

No quinto capítulo, que pode ser visto como uma continuação do terceiro, mostramos que o problema de encontrar o raio de empacotamento de um vetor é uma generalização do problema da partição, justificando a existência do quarto capítulo. Daí, estendemos o método da diferenciação para poder utilizá-lo no caso geral. Após fazer isso, desenvolvemos formas de poder comparar o raio de empacotamento de dois vetores sem calculá-los explicitamente de modo a facilitar o problema de encontrar o raio de empacotamento de um código.

# Capítulo 1

## Códigos Corretores de Erros

Neste capítulo damos uma introdução breve ao assunto dos códigos corretores de erros. Todos os tópicos discutidos podem ser encontrados em livros textos.

Nas primeiras duas seções discutimos os fundamentos da teoria da informação. A referência principal é o artigo [21] e como livro texto recomendamos [3].

Na terceira seção introduzimos a métrica de Hamming nomeada assim em homenagem a Richard Hamming. O livro texto que recomendamos é [6] escrito pelo proprio Hamming.

Na quarta seção introduzimos o raio de empacotamento, cuja determinação no caso das métricas posets (a serem introduzidas no próximo capítulo) é o foco principal deste trabalho. Para o leitor interessado no assunto geral do raio de empacotamento recomendamos o livro [13].

### 1.1 O modelo de Shannon-Weaver

A teoria da informação, como teoria matemática, começou em 1948 com a publicação do artigo “A Mathematical Theory of Communication” [21], escrito por Claude E. Shannon. Neste artigo ele constrói um modelo matemático para um sistema de comunicação genérico. O sistema, esquematizado pela figura abaixo, consiste em cinco partes:

1. Uma fonte de informação (information source), que produz uma ou várias mensagens a serem enviadas pelo canal ao destinatário (destination). A mensagem pode ser de vários tipos: uma sequência de letras, uma imagem, uma música, etc. A natureza da informação é irrelevante.
2. Um transmissor (transmitter), que transforma a mensagem num sinal (signal) adequado para ser transmitido pelo canal. Na telefonia, por exemplo,

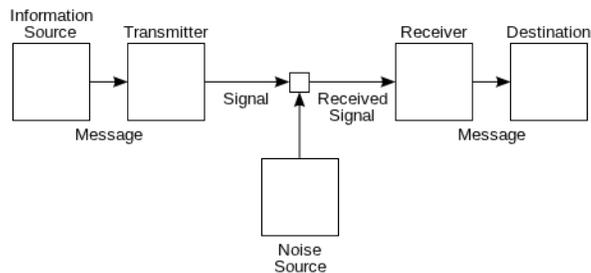


Figura 1.1: Modelo Original de Shannon [21]

isto consiste em transformar os sons emitidos pela fonte em sinais elétricos para poder enviá-los por meio de cabos. É comum chamar o transmissor de codificador e o sinal de código.

3. Um canal, o meio pelo qual o sinal é transmitido para o receptor (receiver). Na telefonia o canal seria o cabo telefônico. Dizemos que o canal tem ruído quando o sinal enviado pelo transmissor não é necessariamente o recebido pelo receptor (receiver). Um exemplo clássico de ruído é o “chiado” às vezes presente numa conversa telefônica. O que produz o ruído é chamado de fonte de ruído (noise source).
4. Um receptor, cuja função é transformar o sinal recebido na mensagem original produzida pela fonte de informação. Na telefonia, o receptor transformaria o sinal elétrico em som, de preferência, no som original emitido pela fonte. É comum chamar o receptor de decodificador.
5. Um destinatário, a pessoa ou “coisa” a qual se destina a mensagem.

Este modelo de comunicação é chamado de modelo de Shannon-Weaver, em homenagem ao Shannon, já mencionado, e a Warren Weaver. Ambos são co-autores do livro “The Mathematical Theory of Communication” [22] que contém o artigo original do Shannon e uma popularização deste escrito por Weaver. Para poder trabalhar de forma mais concreta com o modelo teremos que dar representações matemáticas para as partes descritas acima. De modo geral, os sistemas de comunicação podem ser classificados de três formas: discretos, contínuos e mistos. Neste trabalho estamos apenas interessados nos sistemas discretos.

## 1.2 Sistemas Discretos de Comunicação

Chamamos de **alfabeto** a um conjunto finito  $A$ , e de **letra** a cada elemento deste conjunto. Uma sequência finita de  $n$  elementos de  $A$  é chamado de **palavra de comprimento  $n$** . Sempre consideramos palavras de comprimento finito. No sistema de comunicação, as mensagens produzidas pela fonte são as letras de um alfabeto

chamado de **alfabeto da fonte**. Estas letras estão associadas a uma distribuição de probabilidade que dita suas probabilidades de ocorrência.

**Exemplo 1.** *Nossa fonte de informação pode ser uma moeda considerada "justa". Com isto queremos dizer que o alfabeto da fonte é*

$$S = \{cara, coroa\}$$

com a seguinte distribuição de probabilidade

$$p(cara) = p(coroa) = \frac{1}{2}.$$

O codificador tem o trabalho de converter as letras do alfabeto da fonte em palavras de um outro alfabeto chamado de **alfabeto do canal**, mais adequando para a transmissão pelo canal. Ele fará isso por meio de uma **função de codificação**.

**Definição 1.** *Seja  $S$  o alfabeto da fonte e  $P$  o conjunto de todas as palavras do alfabeto do canal. Uma função de codificação é uma função injetora  $\phi : S \mapsto P$ . A imagem da função é chamada de **código**, e cada um de seus elementos de **palavra-código**.*

Como o alfabeto da fonte é finito, temos que o código também é finito.

Neste trabalho estamos apenas interessados em códigos em que todas as palavras-código tem um mesmo comprimento.

**Definição 2.** *Chamamos de **código de bloco** a um código  $C$  em que todas as palavras-código têm um mesmo comprimento fixo  $n$ . Se, além disso, o código tem  $M$  palavras-código, dizemos que  $C$  é um  $(n, M)$  código de bloco.*

**Exemplo 2.** *Continuando com o exemplo da moeda. Suponha que gostaríamos de comunicar os resultados dos lançamentos da moeda a alguém que se encontra distante e que, para isto, temos duas bandeirinhas, uma vermelha e uma azul. Podemos combinar com a pessoa que toda vez que der cara vamos levantar a bandeira vermelha três vezes e toda vez que der coroa vamos levantar a bandeira azul três vezes. O alfabeto do canal é então*

$$A = \{\text{vermelho}, \text{azul}\},$$

e a função de codificação é

$$\phi(\text{cara}) = (\text{vermelho}, \text{vermelho}, \text{vermelho})$$

e

$$\phi(\text{coroa}) = (\text{azul}, \text{azul}, \text{azul}).$$

Daí o código é

$$C = \{(\text{vermelho}, \text{vermelho}, \text{vermelho}), (\text{azul}, \text{azul}, \text{azul})\},$$

um  $(3, 2)$  código de bloco.

O canal é o meio pelo qual o codificador se comunica com o decodificador. A comunicação é feita enviando-se uma letra do alfabeto do canal de cada vez. Quando há ruído, este é caracterizado por uma distribuição de probabilidade condicionada que dita as probabilidades de que letra será recebida dado o que realmente foi enviado. A letra enviada pelo codificador num determinado momento é chamada de **entrada** e a recebida pelo decodificador de **saída**.

Duas classes importantes de canal são os simétricos e os sem memória.

**Definição 3.** *Um canal é dito **simétrico** quando a probabilidade do sinal original chegar é uma constante igual para todos os sinais e a distribuição de probabilidade da saída, condicionada ao fato do sinal ter sido corrompido, é uniforme. Ou seja, se  $p$  é a probabilidade de chegar a mensagem correta, e  $A$  é o alfabeto do canal, então para todo  $x, y \in A$*

$$P(\text{receber } x | \text{enviado } x) = p$$

e

$$P(\text{receber } x | \text{enviado } y) = \frac{1 - p}{|A| - 1},$$

onde  $|A|$  denota a cardinalidade do conjunto  $A$ .

**Definição 4.** *Um canal é dito **sem memória** quando a distribuição de probabilidade da saída depende apenas da entrada atual do canal. Ou seja, se  $A$  é o alfabeto do canal e  $x_i, y_i \in A$ , então*

$$P(\text{receber } x_1 x_2 \dots x_n | \text{enviado } y_1 y_2 \dots y_n) = \prod_i P(\text{receber } x_i | \text{enviado } y_i).$$

**Exemplo 3.** *Continuando com o exemplo da moeda. Suponha que a pessoa para quem estamos sinalizando com as bandeirinhas tem uma chance de 10% de confundir a cor das bandeirinhas, e que o desempenho dela é sempre o mesmo, ou seja, o passado não afeta o presente no que diz respeito a ela identificar corretamente a cor da bandeirinha. Em termos técnicos estamos dizendo que o canal sendo utilizado é simétrico e sem memória determinado pelas seguintes probabilidades condicionais:*

$$p(\text{ver bandeira azul} | \text{mostramos bandeira azul}) = 90\%$$

$$p(\text{ver bandeira vermelha} | \text{mostramos bandeira vermelha}) = 90\%$$

$$p(\text{ver bandeira azul} | \text{mostramos bandeira vermelha}) = 10\%$$

$$p(\text{ver bandeira vermelha} | \text{mostramos bandeira azul}) = 10\%.$$

O trabalho do decodificador consiste em duas etapas. Na Primeira, ele deve inferir qual foi a palavra-código original enviada pelo codificador. Utilizar códigos de bloco facilita esta etapa pois daí o decodificador sabe quantas letras deve receber para

para inferir qual palavra foi enviada. A segunda etapa consiste em inverter a função de codificação de modo a que o destinatário receba a mensagem, de preferência a original. Muitas vezes inverter a função de codificação pode ser muito difícil computacionalmente e é preferível aproximar esta inversão.

Consideramos também a possibilidade do decodificador não decidir qual palavra-código foi enviada e simplesmente declarar que houve um erro.

O processo pelo qual o decodificador decide qual palavra-código foi enviada ou se declara erro é chamado de **regra de decisão**.

**Definição 5.** *Seja  $C$  um código sobre um alfabeto  $A$ ,  $P$  o conjunto de todas as palavras que têm possibilidade de chegar ao decodificador quando uma palavra-código é enviada e  $?$  um símbolo distinto de qualquer palavra-código de  $C$ . Uma **regra de decisão** é uma função  $f : P \mapsto C \cup \{?\}$  tal que se  $x \in P$  é uma palavra recebida pelo decodificador, ele decodifica  $x$  como  $f(x)$  se  $f(x) \neq ?$  ou declara um erro de decodificação se  $f(x) = ?$ .*

Note que, se o código  $C$  for um  $(n, M)$  código de bloco então  $P = A^n$ .

**Exemplo 4.** *Continuando com o exemplo da moeda. Como o código combinado é de bloco a pessoa sabe que cada vez que três bandeirinhas são levantadas, uma palavra foi enviada. Uma possível regra de decisão é ele interpretar a palavra enviada da seguinte forma: se as três vezes que a bandeirinha foi levantada ele viu a mesma cor ele assume que não houve erro na transmissão, mas se ele observar bandeirinhas de cores distintas ele assume que fez um único erro e considera que a bandeirinha de cor diferente das outras duas está errada. Em termos técnicos a sua regra de decisão é (para não ocupar muito espaço representamos vermelho pela letra  $v$  e azul pela letra  $a$ ):*

$$f((v, v, v)) = f((v, v, a)) = f((v, a, v)) = f((a, v, v)) = (v, v, v)$$

$$f((a, a, a)) = f((a, a, v)) = f((a, v, a)) = f((v, a, a)) = (a, a, a).$$

Após fazer isso a pessoa inverte a função de decodificação e interpreta  $(v, v, v)$  como cara e  $(a, a, a)$  como coroa.

Existem vários possíveis critérios para escolher uma regra de decisão. A princípio queremos aumentar a probabilidade de que o decodificador acerte qual palavra foi enviada, no entanto, muitas vezes esse critério não é viável do ponto de vista computacional e optamos por uma aproximação.

Uma das possíveis expressões para a probabilidade de acerto é a seguinte.

**Proposição 1.** *Seja  $A$  o alfabeto do canal,  $C$  um código sobre  $A$ ,  $P$  o conjunto de todas as palavras que têm possibilidade de chegar ao decodificador quando uma palavra-código é enviada, e  $f$  uma regra de decisão. Então, a probabilidade do decodificador acertar a palavra enviada é:*

$$p(\text{acerto}) = \sum_{d \in P} p(\text{enviado } f(d) | \text{recebido } d) p(\text{recebido } d).$$

*Demonstração.* Basta condicionar sobre as palavras recebidas. □

Logo, a probabilidade de acerto pode ser maximizada escolhendo-se um critério de decisão  $f$  que maximize cada uma das probabilidades

$$p(\text{enviado } f(d) | \text{recebido } d).$$

Temos então a seguinte definição e resultado.

**Definição 6.** *Seja  $C$  um código,  $P$  o conjunto de todas as palavras que têm possibilidade de chegar ao decodificador quando uma palavra-código é enviada, e  $f$  uma regra de decisão. Então,  $f$  é dita um **observador ideal** se para todo  $d \in P$*

$$p(\text{enviado } f(d) | \text{recebido } d) = \max_{c \in C} p(\text{enviado } c | \text{recebido } d).$$

**Proposição 2.** *Uma regra de decisão que é um observador ideal, maximiza a probabilidade de que o decodificador acerte na decodificação.*

*Demonstração.* Resultado direto da definição 6. □

Note que, a propriedade de ser um observador ideal depende da distribuição de probabilidade da fonte, pois por Bayes

$$p(\text{enviado } c | \text{recebido } d) = \frac{p(\text{recebido } d | \text{enviado } c)p(\text{enviado } c)}{\sum_{c' \in C} p(\text{recebido } d | \text{enviado } c')p(\text{enviado } c')}.$$

Se a distribuição da fonte for uniforme teremos

$$p(\text{enviado } c') = \frac{1}{|C|},$$

e portanto,

$$p(\text{enviado } c | \text{recebido } d) = \frac{p(\text{recebido } d | \text{enviado } c)}{\sum_{c' \in C} p(\text{recebido } d | \text{enviado } c')}.$$

Como o denominador não depende de  $c$ , maximizar  $p(\text{enviado } c | \text{recebido } d)$  é equivalente a maximizar  $p(\text{recebido } d | \text{enviado } c)$ . Temos então a seguinte definição e resultado.

**Definição 7.** *Seja  $C$  um código,  $P$  o conjunto de todas as palavras que têm possibilidade de chegar ao decodificador quando uma palavra-código é enviada, e  $f$  uma regra de decisão. Então,  $f$  é dita um **decodificador por máxima verossimilhança** se para todo  $d \in P$*

$$p(\text{recebido } d | \text{enviado } f(d)) = \max_{c \in C} p(\text{recebido } d | \text{enviado } c).$$

**Proposição 3.** *Se a distribuição de probabilidade da fonte é a uniforme, então uma regra de decisão é um observador ideal se e somente se é um decodificador por máxima verossimilhança.*

*Demonstração.* Resultado direto da definição 7. □

**Exemplo 5.** *Continuando com o exemplo da moeda. É fácil ver que a regra de decisão utilizada é um decodificador por máxima verossimilhança, pois é mais provável que não haja erros ou que haja apenas um do que dois ou três. Como a distribuição de probabilidade da fonte (a moeda) é uniforme, temos que o decodificador é um observador ideal e, portanto esta regra de decisão é a que maximiza a chance do decodificador - a pessoa - decodificar a mensagem corretamente. Fazendo as contas, temos que a chance da pessoa receber a mensagem correta é de 97,2%.*

No modelo de comunicação é comum subdividir o codificador em mais duas partes: o codificador de fonte, que busca retirar redundância da mensagem recebida da fonte, e o codificador de canal, que busca adicionar redundância à mensagem enviada pelo codificador de fonte, de modo a proteger o sinal contra erros introduzidos pelo canal. Nesse caso devemos também subdividir o decodificador em duas partes, cada uma para desfazer o que foi feito por um dos codificadores. Representamos isto na **Figura 1.2**.

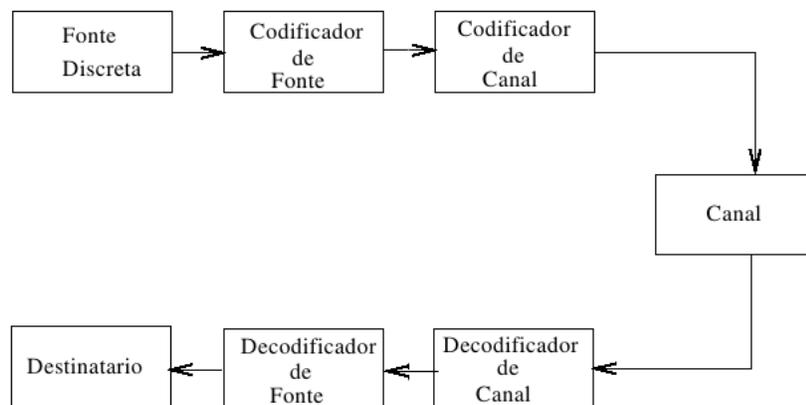


Figura 1.2: Modelo Estendido de Comunicação

Quando a distribuição de probabilidade da fonte não é uniforme, há redundância nas mensagens emitidas por ela. Uma boa codificação de fonte retira esta redundância e além disso, o alfabeto do codificador de fonte é o mesmo do codificador de canal. Como estamos apenas interessados na codificação de canal, iremos assumir que foi feita uma boa codificação de fonte. Isto equivale a considerar o modelo original da **Figura 1.1** com as hipóteses adicionais de que a distribuição de probabilidade da fonte é uniforme e o alfabeto dela é o mesmo que o do canal.

**Exemplo 6.** Continuando com o exemplo da moeda. Neste caso, a fonte já tem distribuição de probabilidade uniforme. Basta que o codificador de fonte transforme

$\text{cara} \mapsto v$

$\text{coroa} \mapsto a.$

Isto é equivalente a trocar a nossa fonte por outra que emite  $v$  e  $a$ , ambas com probabilidade de 50%.

Temos então uma nova variável a considerar, o comprimento das palavras da fonte. Como as mensagens são enviadas pelo canal letra por letra, o ideal é que a codificação não aumente muito o tamanho da mensagem. Teremos então uma forma de quantificar a redundância em uma mensagem.

**Definição 8.** A taxa de informação de um código é a proporção de letras não redundantes, ou seja, se a taxa de informação é  $\frac{k}{n}$  então para cada  $k$  letras emitidas pela fonte o codificador emite em média  $n$  letras para transmitir pelo canal.

Obviamente existe uma correlação entre a taxa de informação e a capacidade do código de corrigir erros.

**Exemplo 7.** Continuando com o exemplo da moeda. Ambas as palavras da fonte tem comprimento um, e ambas são codificadas em palavras de comprimento três. Logo, a taxa de informação do código é  $\frac{1}{3}$ . Isto quer dizer que para que a pessoa interprete uma mensagem, devemos transmitir o sinal três vezes. Enviando o sinal sem codificá-lo, a pessoa saberia o resultado de cada lançamento instantaneamente com 10% de probabilidade de erro. Usando o código, a pessoa deve esperar que se levante a bandeirinha 3 vezes para saber o resultado de cada lançamento, mas a probabilidade de erro é de 2,8%.

Como apareceu uma nova variável, o comprimento da palavra da fonte, teremos que estender a definição de código de bloco.

**Definição 9.** Um código  $C$  é um  $(n, |C|)$  código de bloco quando todas as palavras do domínio de sua função de codificação têm um tamanho fixo  $k$  e todas as palavras da imagem desta função têm um tamanho fixo  $n$ .

Existe uma outra forma de proteger mensagens contra erros sem prejudicar tanto a taxa de informação. A ideia é fazer o que se chama de extensão de fonte.

**Definição 10.** Seja  $F$  o conjunto das mensagens de uma fonte. A  $n$ -ésima extensão da fonte é a fonte cujo conjunto de mensagens é  $F^n$ .

O custo de se estender uma fonte é que para iniciar o processo de codificação temos que esperar o tempo que demora para a fonte gerar as  $n$  mensagens.

Um dos teoremas mais importantes de Shannon, publicado em [21], é o teorema de codificação para canais ruidosos (Noisy-channel coding theorem). O teorema diz que dado um canal discreto sem memória, existe um número  $C$ , chamado de **capacidade do canal**, que depende apenas das características do canal, que satisfaz a seguinte propriedade: Para todo  $\epsilon > 0$  e  $R < C$ , existe um  $n$  e um código sobre a  $n$ -ésima extensão da fonte tal que sua taxa de informação é maior do que  $R$  e a probabilidade do decodificador errar é menor do que  $\epsilon$ . A recíproca também vale, ou seja, se a taxa de informação é maior do que a capacidade do canal, não há decodificação com probabilidade arbitrariamente pequena. A demonstração do teorema não é construtiva no entanto, e encontrar códigos cuja taxa atinge a capacidade do canal continua um problema em aberto.

**Exemplo 8.** *Continuando com o exemplo da moeda. A capacidade do canal é aproximadamente 0,53. Isso quer dizer que não importa o quão pequeno queremos a probabilidade de erro, existe uma extensão de fonte e um código que satisfazem nossas condições e que transmite a uma taxa de informação de aproximadamente  $\frac{1}{2}$ .*

### 1.3 Métrica de Hamming

Nesta seção trataremos de códigos com mais estrutura. Os alfabetos que utilizamos são os corpos finitos. Denotamos estes por  $\mathbb{F}_q$ , onde  $q$ , uma potência de primo, é a cardinalidade do corpo. Além disso, estamos interessados apenas em códigos de bloco. Isso quer dizer que a função de codificação é sempre da forma

$$f : \mathbb{F}_q^k \mapsto \mathbb{F}_q^n,$$

e portanto, o código é sempre um subconjunto de  $\mathbb{F}_q^n$ . Definimos a seguir o que é um código linear.

**Definição 11.** *Seja  $C \subseteq \mathbb{F}_q^n$  um código. Então,  $C$  é dito um **código linear** se é um espaço vetorial.*

**Exemplo 9.** *Tomemos  $\mathbb{F}_2 = \{0, 1\}$  como sendo o alfabeto e  $C = \{000, 111\}$  como o código. Interpretando 1 como sendo vermelho e 0 como sendo azul, este código é essencialmente o mesmo utilizado no exemplo da moeda da seção anterior.*

Na seção anterior vimos dois tipos de decodificadores. Nesta, iremos estudar um chamado de decodificador por máxima proximidade. Para usá-lo teremos que definir uma métrica no nosso espaço.

**Definição 12.** *Seja  $C \subseteq \mathbb{F}_q^n$  um código e  $d$  uma métrica sobre  $\mathbb{F}_q^n$ . Um **decodificador por máxima proximidade** interpreta uma palavra  $x \in \mathbb{F}_q^n$  como sendo a palavra-código mais perto de  $x$  usando a métrica  $d$ . No caso em que a palavra-código mais perto de  $x$  não seja única, escolhe-se arbitrariamente uma destas palavras.*

Uma das vantagens do decodificador por proximidade quando comparado aos apresentados no capítulo anterior é que este não depende das probabilidades envolvidas no canal. É o que chamamos de um decodificador universal.

Existem várias possíveis métricas para se definir no espaço. Nesta seção estudamos a métrica de Hamming. No próximo capítulo estudamos uma classe de métricas, chamadas de métricas posets que generalizam a métrica de Hamming. Definimos primeiro o peso de Hamming.

**Definição 13.** *Seja  $x \in \mathbb{F}_q^n$ . O peso de Hamming de  $x$  é*

$$\omega_H(x) = |\text{supp}(x)|,$$

onde  $\text{supp}(x) = \{i : x_i \neq 0\}$ .

O peso de Hamming de uma palavra é então a quantidade de coordenadas não nulas da palavra. Este peso, como provamos a seguir, induz uma métrica em  $\mathbb{F}_q^n$  chamada de métrica de Hamming.

**Proposição 4.** *A função*

$$d_H : \mathbb{F}_q^n \times \mathbb{F}_q^n \mapsto \mathbb{N},$$

tal que

$$d_H(x, y) = \omega_H(x - y),$$

define uma métrica, chamada de **métrica de Hamming**.

Provamos apenas a desigualdade triangular, pois as outras propriedades saem direto da definição.

*Demonstração.* Queremos provar que dados  $x, y, z \in \mathbb{F}_q^n$  temos

$$d_H(x, z) \leq d_H(x, y) + d_H(y, z).$$

Seja  $i \in \text{supp}(x - z)$ , logo  $x_i \neq z_i$ . Se  $x_i = y_i$  daí  $y_i \neq z_i$  e, portanto  $i \in \text{supp}(y - z)$ . No entanto se  $x_i \neq y_i$  então  $i \in \text{supp}(x - y)$ . Portanto,

$$\text{supp}(x - z) \subseteq \text{supp}(x - y) \cup \text{supp}(y - z).$$

Mas daí,

$$\begin{aligned} d_H(x, z) &= |\text{supp}(x - z)| \\ &\leq |\text{supp}(x - y) \cup \text{supp}(y - z)| \\ &\leq |\text{supp}(x - y)| + |\text{supp}(y - z)| \\ &= d_H(x, y) + d_H(y, z). \end{aligned}$$

□

Note que, a distância de Hamming entre a palavra enviada pelo codificador e a recebida pelo decodificador é a quantidade de erros que ocorreram na transmissão.

**Exemplo 10.** Usando o código do exemplo anterior,

$$\omega_H(111) = 3,$$

e portanto,

$$d_H(111, 000) = 3.$$

Como é usual em qualquer espaço métrico, dados  $x \in \mathbb{F}_q^n$  e  $r \in \mathbb{R}$ , definimos a bola de centro  $x$  e raio  $r$  como

$$B(x, r) = \{y \in \mathbb{F}_q^n : d(x, y) \leq r\}.$$

Esta abordagem torna o problema de decodificação num problema geométrico. Uma pergunta natural é: quão bom é um decodificador por máxima proximidade, usando a métrica de Hamming, se queremos minimizar a probabilidade de erro na decodificação? Provaremos a seguir que num canal sem memória em que a probabilidade de erro é menor do que meio, o decodificador por máxima proximidade usando a métrica de Hamming é equivalente a um decodificador por máxima verossimilhança.

**Proposição 5.** Se num canal discreto sem memória a probabilidade de uma letra enviada ser corrompida é  $p < \frac{1}{2}$ , então um decodificador por máxima proximidade é equivalente a um decodificador por máxima verossimilhança.

*Demonstração.* Seja  $x \in \mathbb{F}_q^n$  a palavra enviada,  $y \in \mathbb{F}_q^n$  a palavra recebida e  $d = d_H(x, y)$ . Como o canal é sem memória e sabemos que houveram  $d$  erros na transmissão temos que:

$$\begin{aligned} P(y \text{ recebido} | x \text{ enviado}) &= (1-p)^{n-d} p^d \\ &= (1-p)^n p^d \left(\frac{1}{1-p}\right)^d \\ &= (1-p)^n \left(\frac{p}{1-p}\right)^d. \end{aligned}$$

Mas como  $p < \frac{1}{2}$ , maximizar a probabilidade é equivalente a minimizar  $d$ .  $\square$

Como vimos na seção anterior, se a distribuição de probabilidade da fonte é uniforme, então o decodificador por máxima verossimilhança é equivalente ao observador ideal. Logo,

**Proposição 6.** *Se a distribuição de probabilidade da fonte é uniforme e o canal é discreto sem memória com probabilidade de uma letra enviada ser corrompida menor do que meio, então o decodificador por máxima proximidade maximiza a chance de o decodificador decodificar corretamente.*

**Exemplo 11.** *Continuando o exemplo das moedas, o último resultado confirma o que vimos na seção anterior de que a nossa regra de decisão é ótima para minimizar a chance de erro do decodificador.*

Na próxima seção abordaremos o problema de achar qual é a quantidade mínima de erros que um código corrige. Isto certamente depende da distância entre as palavras do código. São naturais então as seguintes definições:

**Definição 14.** *Seja  $C \subseteq \mathbb{F}_q^n$  um código e  $d$  uma métrica sobre  $\mathbb{F}_q^n$ . O peso mínimo de  $C$  é definido como*

$$\omega_H(C) = \min \{ \omega_H(x) : x \in C - \{0\} \}.$$

A distância mínima de  $C$  é definida como

$$d_H(C) = \min \{ d_H(x, y) : x, y \in C, x \neq y \}.$$

Note que, se um código é linear a diferença de duas palavras-código também é uma palavra-código e, portanto a distância mínima e o peso mínimo do código coincidem.

## 1.4 Raio de Empacotamento

Nesta seção abordamos a seguinte questão: Qual é a quantidade máxima de erros que um decodificador por máxima proximidade pode corrigir, independentemente da palavra-código, ao utilizar-se a métrica de Hamming?

Como vimos na seção anterior, a distância de Hamming entre a palavra enviada e a recebida é a quantidade de erros que ocorreram na transmissão. Sendo assim, se todas as bolas de raio  $r$  centradas em palavras-código não se interceptam, o codificador irá interpretar todas as palavras de uma bola como sendo a palavra-código do centro, corrigindo no mínimo  $r$  erros. O que queremos então é saber qual é o maior valor possível para  $r$ . A este valor damos o nome de raio de empacotamento.

**Definição 15.** *Seja  $C \subseteq \mathbb{F}_q^n$  um código e  $d$  uma métrica sobre  $\mathbb{F}_q^n$ . O raio de empacotamento de  $C$  é o maior inteiro, denotado por  $R_d(C)$ , tal que*

$$B_H(x, R_d(C)) \cap B_H(y, R_d(C)) = \emptyset$$

para todo  $x, y \in C$  com  $x \neq y$ .

No caso da métrica de Hamming, o raio de empacotamento é determinado pela distância mínima do código.

**Proposição 7.** Seja  $C \subseteq \mathbb{F}_q^n$  um código. Então,

$$R_{d_H}(C) = \left\lfloor \frac{d_H(C) - 1}{2} \right\rfloor,$$

onde  $\lfloor x \rfloor$  é a parte inteira de  $x \in \mathbb{R}$ .

*Demonstração.* Sejam  $x, y \in C$ ,  $d = d_H(C)$  e

$$R = \left\lfloor \frac{d - 1}{2} \right\rfloor.$$

Suponha que existe

$$z \in B_H(x, R) \cap B_H(y, R).$$

Mas daí,

$$\begin{aligned} d &\leq d_H(x, y) \\ &\leq d_H(x, z) + d_H(z, y) \\ &\leq 2R \\ &\leq d - 1. \end{aligned}$$

Uma contradição. Logo,

$$B_H(x, R) \cap B_H(y, R) = \emptyset.$$

Resta mostrar que  $R$  é o maior inteiro satisfazendo isto. Primeiro note que, como  $d$  é inteiro,

$$\left\lfloor \frac{d - 1}{2} \right\rfloor \geq \frac{d - 1}{2} - \frac{1}{2},$$

e portanto,

$$\begin{aligned} d &\leq 2 \left\lfloor \frac{d - 1}{2} \right\rfloor + 2 \\ &\leq 2(R + 1). \end{aligned}$$

Sejam então  $v, w \in C$  tais que  $d_H(v, w) = d$ , e  $k_1, k_2, \dots, k_d$  as coordenadas em que  $v$  e  $w$  diferem. Defina  $z \in C$  como sendo a palavra cujas coordenadas  $k_1, k_2, \dots, k_{R+1}$  coincidem com  $v$  e o resto delas coincide com  $w$ . Por definição,

$$d_H(z, w) = R + 1$$

e

$$d_H(z, v) = d - (R + 1).$$

Aplicando a desigualdade provada acima temos

$$\begin{aligned} d_H(z, v) &= d - (R + 1) \\ &\leq R + 1, \end{aligned}$$

e portanto,

$$z \in B_H(v, R + 1) \cap B_H(w, R + 1).$$

□

Em suma, um código  $C$  corrige ao menos até  $\left\lfloor \frac{d_H(C) - 1}{2} \right\rfloor$  erros.

Na primeira parte da demonstração, usamos apenas as propriedades de espaço métrico e nenhuma especial da métrica de Hamming. Logo, num espaço métrico qualquer, temos um limitante inferior para o raio de empacotamento.

**Proposição 8.** *Seja  $C \subseteq \mathbb{F}_q^n$  um código e  $d$  uma métrica sobre  $\mathbb{F}_q^n$ . Então,*

$$\left\lfloor \frac{d(C) - 1}{2} \right\rfloor \leq R_d(C) \leq d(C) - 1$$

*Demonstração.* A desigualdade do lado esquerdo foi provada na **Proposição 7**. A do lado direito é óbvia, pois se  $x, y \in C$  são tais que  $d(C) = d(x, y)$  então

$$x \in B(x, d(C)) \cap B(y, d(C)).$$

□

O raio de empacotamento das métricas estudadas no próximo capítulo atinge todos os valores da desigualdade acima.

Uma classe importante de códigos são os chamados códigos perfeitos.

**Definição 16.** *Um código  $C \subseteq \mathbb{F}_q^n$  é chamado de código perfeito se*

$$\bigcup_{x \in C} B(x, R_d(C)) = \mathbb{F}_q^n,$$

*i.e. se toda palavra dista no máximo  $R_d(C)$  de uma palavra código.*

Para decodificar uma palavra  $x \in \mathbb{F}_q^n$  de um código perfeito  $C \subseteq \mathbb{F}_q^n$ , basta interpretá-la como a única palavra-código  $c \in C$  que pertence a  $B(x, R_d(C))$ .

## Capítulo 2

# Códigos Poset

O foco deste capítulo é introduzir as definições e conceitos básicos relacionados a códigos poset.

Na primeira seção, introduziremos os conceitos básicos de conjuntos parcialmente ordenados (posets) que serão utilizados no restante do trabalho. Estes conceitos podem ser encontrados em livros textos que abordem o assunto de posets como, por exemplo, [23].

Na segunda seção definiremos o que são os códigos poset. Na literatura eles apareceram inicialmente de forma generalizada em [2]. Para o leitor interessado nas possíveis aplicações para estes códigos recomendamos [19] e [18].

### 2.1 Conjuntos Parcialmente Ordenados

Uma ordem parcial é a generalização da ideia de ordenamento.

**Definição 17.** *Um conjunto parcialmente ordenado, também chamado de poset (partially ordered set), é uma dupla  $P = (X, \preceq)$  onde  $X$  é um conjunto qualquer, e  $\preceq$  é uma operação binária em  $X$ , chamada de **ordem parcial**, que satisfaz as seguintes propriedades. Para todo  $x, y, z \in X$ :*

- $x \preceq x$ . (reflexividade)
- Se  $x \preceq y$  e  $y \preceq x$ , então  $x = y$ . (anti simetria)
- Se  $x \preceq y$  e  $y \preceq z$ , então  $x \preceq z$ . (transitividade)

*Se considerarmos mais de uma ordem em um conjunto, para evitar ambiguidade, denotamos a ordem parcial por  $\preceq_P$ .*

É comum abusar da notação e identificar o poset com o conjunto adjacente. Naturalmente diremos que  $x$  é menor ou igual do que  $y$  quando  $x \preceq y$ . Do mesmo modo que se faz com o símbolo de menor ou igual, interpretamos  $x \succeq y$  como  $y \preceq x$ ,  $x \prec y$  como  $x \preceq y$  e  $x \neq y$ , e  $x \succ y$  como  $y \prec x$ .

**Exemplo 12.** *O conjunto dos números naturais com a ordem natural é um poset.*

Note que, não é necessário que para dois elementos  $x, y$  de um poset  $P$  tenhamos que  $x \preceq y$  ou  $y \preceq x$  (como acontece com os naturais). Quando isto ocorre, dizemos que os elementos são comparáveis, e caso contrário, dizemos que são incomparáveis. Sendo assim, não é necessário que exista um elemento máximo, mesmo num conjunto finito. Um elemento de um poset é **maximal** quando não existe nenhum outro elemento maior do que ele. Analogamente, um elemento é **mínimal** quando não existe nenhum outro elemento menor do que ele.

**Exemplo 13.** *Dado um conjunto  $\mathbb{A}$ , o conjunto das partes de  $A$ , denotado por  $2^{\mathbb{A}}$ , com a relação binária de continência como ordem é um poset. Ou seja, se  $A \subseteq B \subseteq \mathbb{A}$  dizemos que  $A \preceq B$ . Se o conjunto tem pelo menos dois elementos  $a, b \in \mathbb{A}$ , então  $\{a\}$  e  $\{b\}$  não são comparáveis.*

Um poset em que todos os elementos são comparáveis é chamado de **cadeia**, e se nenhum elemento é comparável a outro diferente, chamamos o poset de **anti-cadeia**. Os números naturais, por exemplo, com a ordem natural é uma cadeia.

Dizemos que um elemento  $y$  de um poset cobre outro elemento  $x$  se  $x \preceq y$  e não existe nenhum elemento  $z$  distinto de  $x, y$  tal que  $x \preceq z \preceq y$ .

**Exemplo 14.** *No conjunto dos naturais, o sucessor  $n + 1$  de um número  $n$  é o único que o cobre.*

Dois posets são ditos isomorfos se a estrutura destes é essencialmente a mesma. Mais precisamente,

**Definição 18.** *Sejam  $P = (X, \preceq)$  e  $Q = (X', \preceq')$  dois posets. Eles são ditos **isomorfos** se existir uma bijeção entre eles,*

$$f : X \mapsto X'$$

*que preserve a ordem, i.e.  $x \preceq_P y$  se e somente se  $f(x) \preceq_Q f(y)$ .*

**Exemplo 15.** *O conjunto dos números naturais com a ordem natural é isomorfa ao dos números pares positivos com a ordem natural. Basta usar a bijeção que leva um número no seu dobro.*

Daqui por diante consideraremos apenas posets finitos. Sendo assim, dado um poset  $P$  com  $n$  elementos podemos definir uma ordem parcial no conjunto  $[n] = \{1, 2, \dots, n\}$  que o torne isomorfo a  $P$ . Logo, se estamos apenas interessados nas propriedades satisfeitas pela estrutura do poset podemos assumir, sem perda de generalidade, que o poset é sempre sobre o conjunto  $[n] = \{1, 2, \dots, n\}$ .

**Exemplo 16.** Seja  $P = \{a, b, c\}$  com a ordem  $a \preceq c$ . Basta substituir  $a$  por 1,  $b$  por 2 e  $c$  por 3 que temos um poset isomorfo a  $P$  sobre  $[3]$ .

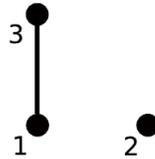


Figura 2.1: Diagrama de Hasse do **Exemplo 16**.

Os posets finitos podem ser representados por **diagramas de Hasse**. Dado um poset finito  $P$ , representamos seus elementos como vértices num plano e ligamos dois vértices  $x$  e  $y$  por uma curva (geralmente um segmento de reta) se um cobre o outro. Fazemos isto de modo que o elemento maior fique acima do menor no plano.

**Exemplo 17.** Um Poset de Niederreiter-Rosenbloom-Tsfasman (NRS) é a união disjunta de cadeias de mesmo comprimento. A **Figura 2.2** é um poset NRS de 4 cadeias de comprimento 3.

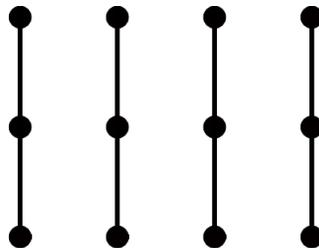


Figura 2.2: Poset de Niederreiter-Rosenbloom-Tsfasman.

Outra forma de representar um poset é por uma matriz binária onde as entradas da matriz indicam as relações entre os elementos do poset.

**Definição 19.** Seja  $P$  um poset de cardinalidade  $n$ . A **matriz de adjacência** de  $P$  é a matriz quadrada  $A_{n \times n}$  cujas coordenadas são definidas da seguinte forma:

$$A_{ij} = \begin{cases} 1 & \text{se } i \leq j \\ 0 & \text{caso contrário} \end{cases}.$$

**Exemplo 18.** A matriz de adjacência do poset do **Exemplo 16** é

$$\begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Dado a representação de Hasse de um poset, se enumerarmos os elementos da esquerda para a direita e de baixo para cima a matriz de adjacência do poset é triangular superior.

Para definir a métrica poset precisamos definir o que é um ideal.

**Definição 20.** *Seja  $P$  um poset. Um ideal em  $P$  é um subconjunto  $J \subseteq P$  tal que se  $y \in J$  e  $x \preceq y$ , então  $x \in J$ .*

Ou seja, um subconjunto de um poset é um ideal se todo elemento menor do que algum elemento do subconjunto pertence ao subconjunto.

Dado um subconjunto qualquer  $A$  de um poset  $P$ , definimos o **ideal gerado pelo conjunto**  $A$  como sendo o menor ideal  $J \subseteq P$  que o contém e o denotamos por  $\langle A \rangle$ .

Se  $x$  é um elemento de um poset, em vez de escrever  $\langle \{x\} \rangle$  escrevemos  $\langle x \rangle$ .

A **altura** de um elemento  $x$  de um poset  $P$ , denotado por  $l(x)$ , é a cardinalidade da maior cadeia contida no ideal gerado por  $\{x\}$ .

**Exemplo 19.** *Um poset  $P$  é hierárquico quando todo elemento  $x \in P$  é coberto por todos os elementos de altura  $l(x) + 1$  e apenas por estes. Podemos representar um poset hierárquico cujos elementos maximais têm altura  $l$  por uma sequência finita  $(n_1, n_2, \dots, n_l)$  tal que  $n_i$  é a quantidade de elementos com altura  $i$ . Deste modo, existe uma bijeção entre a quantidade de posets hierárquicos de cardinalidade  $n$  e as partições de  $n$ .*

## 2.2 Códigos Poset

Nesta seção definiremos uma classe de métricas para  $\mathbb{F}_q^n$  que vão generalizar a métrica de Hamming. Assim como fizemos para a métrica de Hamming, primeiro definiremos uma função peso.

**Definição 21.** *Seja  $P$  um poset sobre  $[n]$ . Definimos o  $P$ -peso de  $x \in \mathbb{F}_q^n$  como sendo*

$$\omega_P(x) = |\langle \text{supp}(x) \rangle|.$$

O  $P$ -peso de um elemento  $x$  é então a quantidade de elementos no ideal gerado pelo suporte de  $x$ . Se  $P$  é uma anti-cadeia, o suporte de um elemento e o ideal gerado por este coincidem e assim o  $P$ -peso é o peso de Hamming. Por isso é natural que denotemos o poset anti-cadeia por  $H$  e o chamemos de **poset de Hamming**. Analogamente ao peso de Hamming, o peso define uma métrica.

**Proposição 9.** *A função*

$$d_P : \mathbb{F}_q^n \times \mathbb{F}_q^n \mapsto \mathbb{R},$$

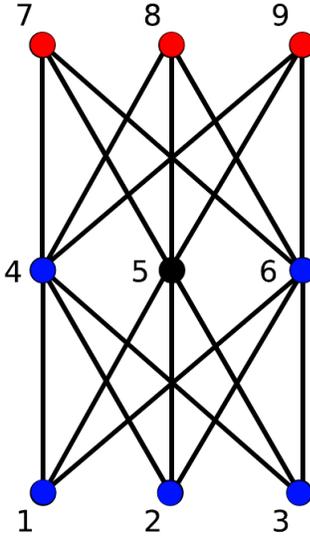


Figura 2.3: Poset Hierárquico  $(3, 3, 3)$ . Os elementos pintados em vermelho são os maximais. Os elementos pintados em azul determinam o ideal gerado pelo conjunto  $\{1, 4, 6\}$ . A altura de 5 é  $l(5) = 2$ .

tal que

$$d_P(x, y) = \omega_P(x - y),$$

defina uma métrica em  $\mathbb{F}_q^n$ , chamada de *P-métrica*.

Novamente demonstraremos apenas a desigualdade triangular, pois as outras propriedades são óbvias.

*Demonstração.* Queremos provar que

$$d_P(x, z) \leq d_P(x, y) + d_P(y, z).$$

Na demonstração da **Proposição 4** mostramos que

$$\text{supp}(x - z) \subseteq \text{supp}(x - y) \cup \text{supp}(y - z).$$

Mas daí,

$$\langle \text{supp}(x - z) \rangle \subseteq \langle \text{supp}(x - y) \cup \text{supp}(y - z) \rangle.$$

No entanto, o ideal da união é a união dos ideais e logo,

$$\langle \text{supp}(x - z) \rangle \subseteq \langle \text{supp}(x - y) \rangle \cup \langle \text{supp}(y - z) \rangle.$$

Daí segue o resultado. □

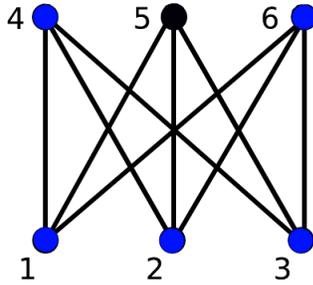


Figura 2.4: Os elementos pintados em azul são o ideal gerado pelo conjunto  $\{1, 4, 6\}$

**Exemplo 20.** Seja  $P$  o poset da Figura 2.4, e  $d_P$  a métrica induzida por esse poset em  $\mathbb{F}_2^6$ . Daí a distância entre 010010 e 110111 é

$$d_P(010010, 110111) = \omega_P(100101) = 5,$$

a cardinalidade do ideal gerado por  $\{1, 4, 6\}$ .

Quando utilizamos uma métrica poset  $d_P$  como critério de decodificação de um código  $C \subseteq \mathbb{F}_q^n$ , chamamos  $C$  de um  **$P$ -código**. No caso em que  $C$  é um código linear, o chamamos de  **$P$ -código linear**. Como vimos no capítulo anterior, no poset de Hamming o raio de empacotamento de um código  $C \subseteq \mathbb{F}_q^n$  é dado por

$$R_{d_H}(C) = \left\lfloor \frac{d_H(C) - 1}{2} \right\rfloor.$$

Um fato surpreendente é que em outros posets o raio de empacotamento não é necessariamente determinado pela distância mínima, como veremos no exemplo a seguir.

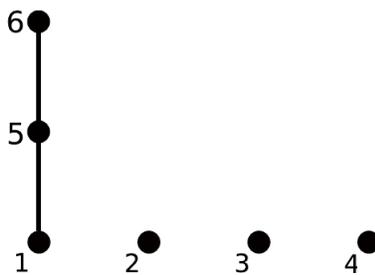


Figura 2.5: Poset do Exemplo 21

**Exemplo 21.** Seja  $P$  o poset da Figura 2.5,  $d_P$  a métrica induzida por esse poset em  $\mathbb{F}_2^6$ , e

$$C_1 = \{000000, 000001\}$$

$$C_2 = \{000000, 111100, 000001, 111101\}$$

*dois códigos lineares. As distâncias mínimas dos dois códigos são a mesma,*

$$d_P(C_1) = d_P(C_2) = 3,$$

*no entanto os raios de empacotamento são diferentes*

$$R_{d_P}(C_1) = 2$$

$$R_{d_P}(C_2) = 1.$$

## Capítulo 3

# Raio de Empacotamento de Códigos Poset I

Neste capítulo iniciamos as nossas contribuições na determinação do raio de empacotamento de um código poset. Previamente a este trabalho os únicos posets para os quais a determinação do raio de empacotamento foi estudado foram os casos de uma cadeia ([5]), hierárquicos ([4]) e para a união disjunta de várias cadeias de mesma altura ([17]). Além destes casos, o raio de empacotamento já é conhecido para uma certa família de códigos sobre um poset genérico ([18]).

Na primeira e única seção deste capítulo trataremos do problema de encontrar o raio de empacotamento de um vetor. Veremos que isto é equivalente a resolver um problema que chamaremos de “problema da partição para posets” que é uma generalização do “problema da partição”, tópico do próximo capítulo.

### 3.1 O Raio de Empacotamento de um Vetor

Como vimos, no poset de Hamming o raio de empacotamento de um código é determinado pela sua distância mínima. Este nem sempre é o caso para um poset qualquer.

Estamos apenas interessados em trabalhar com códigos lineares.

Note que, para um código linear  $C$ , como a diferença de duas palavras-código é também uma palavra código, o raio de empacotamento é o maior inteiro,  $R$ , tal que

$$B(0, R) \cap B(x, R) = \emptyset$$

para todo  $x \in C - \{0\}$ .

**Definição 22.** Seja  $d$  uma métrica em  $\mathbb{F}_q^n$  e  $x \in \mathbb{F}_q^n$ . O **raio de empacotamento do vetor**  $x$  e o raio de empacotamento do código  $C = \{0, x\}$  e é denotado por  $R_d(x)$ .

O próximo resultado mostra que podemos encontrar o raio de empacotamento de um código encontrando o raio de empacotamento de cada palavra-código e tomando o menor valor.

**Proposição 10.** Seja  $C \subseteq \mathbb{F}_q^n$  um código linear e  $d$  uma métrica. Então,

$$R_d(C) = \min_{x \in C - \{0\}} \{R_d(x)\}.$$

*Demonstração.* Seja  $y \in C$  tal que  $R_d(y) = \min_{x \in C - \{0\}} \{R_d(x)\}$ . Por definição, para todo  $x \in C$

$$B(0, R_d(x)) \cap B(x, R_d(x)) = \emptyset.$$

Mas  $R_d(y) \leq R_d(x)$ , e portanto,

$$B(0, R_d(y)) \cap B(x, R_d(y)) = \emptyset.$$

Como

$$B(0, R_d(y) + 1) \cap B(y, R_d(y) + 1) \neq \emptyset,$$

$$R_d(C) = R_d(y). \quad \square$$

Este resultado motiva a seguinte definição:

**Definição 23.** Seja  $C \subseteq \mathbb{F}_q^n$  um código linear e  $d$  uma métrica. Chamamos uma **palavra-código**  $x \in C - \{0\}$  tal que  $R_d(C) = R_d(x)$  de **vetor de empacotamento** de  $C$

Temos então que o vetor de empacotamento determina o raio de empacotamento do código. Deixaremos o problema de encontrar o vetor de empacotamento para mais tarde, observando que este não será necessariamente um vetor de peso mínimo. Nosso foco agora é como encontrar o raio de empacotamento de um vetor.

Um possível algoritmo por força bruta para encontrar o raio de empacotamento de um vetor  $v$  é: começando de  $r = 1$  comparamos todos os vetores na bola centrada em zero e de raio  $r$  com os vetores na bola centrada em  $v$  e de raio  $r$ . Se não encontramos nenhum elemento em comum, aumentamos o valor de  $r$  em um e repetimos o procedimento. Quando encontrarmos um elemento em comum, teremos determinado o raio de empacotamento que será  $r - 1$ .

**Exemplo 22.** Seja  $P$  o poset cadeia de altura três tal que  $1 \preceq 2 \preceq 3$  e  $d_P$  a métrica induzida por este poset em  $\mathbb{F}_2^3$ . Vamos encontrar o raio de empacotamento do vetor

$v = 001$ . A seguinte tabela lista os vetores encontrados nas esferas para cada raio  $r$  e centro  $c$ .

	$c=000$	$c=001$
$r=1$	100	101
$r=2$	110 010	111 011
$r=3$	111 101 001 011	110 100 000 010

Vemos que em  $r = 3$  encontramos elementos na bola centrada em 001 que já listamos nas bolas centradas em 000, e portanto, o raio de empacotamento de  $v$  é  $R_{d_P}(v) = 2$ .

Note que, no exemplo anterior, os vetores da segunda coluna e da terceira estão relacionados por uma translação  $T$  da forma  $T(x) = v - x$  com a propriedade de que  $x \in S(0, r)$  se e somente se  $T(x) \in S(v, r)$ . Isto ocorre pois a métrica poset é **invariante por translação**; i.e. para todo  $x, y, z \in \mathbb{F}_q^n$ ,  $d_P(x, y) = d_P(x+z, y+z)$ . Temos então a seguinte definição.

**Definição 24.** Sejam  $x, v \in \mathbb{F}_q^n$ . Definimos o **complementar de  $x$  com respeito à  $v$**  como  $x^v = v - x$ .

Como foi observado acima vale o seguinte resultado.

**Lema 1.** Seja  $d$  uma métrica invariante por translação sobre  $\mathbb{F}_q^n$  com  $x, v \in \mathbb{F}_q^n$  e  $r \in \mathbb{R}$ , com  $r \geq 0$ . Então  $x \in S(0, r)$  se e somente se  $x^v \in S(v, r)$ .

*Demonstração.* Como  $d$  é invariante por translação temos que

$$d(v, x^v) = d(v - x^v, 0) = d(x, 0)$$

donde segue o resultado. □

Logo, quando a métrica sendo usada é invariante por translação, como é no caso da métrica poset, podemos encontrar o raio de empacotamento de um vetor  $v \in \mathbb{F}_q^n$  da seguinte forma: fazemos duas listas; na primeira, vamos colocando os vetores de  $\mathbb{F}_q^n$  em ordem compatível com o peso; para cada vetor da primeira lista, colocamos o seu complementar com respeito à  $v$  na segunda; fazemos isto até que apareça um vetor  $x$  na primeira lista que já apareceu na segunda; daí o raio de empacotamento será  $\omega(x) - 1$ .

**Exemplo 23.** Usando o processo do parágrafo anterior no último exemplo chegaremos na seguinte tabela:

Lista 1	Lista 2
100	101
110	111
010	011
111	110

Daí, notamos que 111 já apareceu na segunda linha, e portanto, o raio de empacotamento será  $\omega(111) - 1 = 2$ .

Temos então a seguinte expressão para o raio de empacotamento de um vetor:

**Teorema 1.** *Seja  $d$  uma métrica invariante por translação sobre  $\mathbb{F}_q^n$  e  $v \in \mathbb{F}_q^n$ . Então,*

$$R_d(v) = \min_{x \in \mathbb{F}_q^n} \{\max\{\omega(x), \omega(x^v)\}\} - 1.$$

*Demonstração.* Seja  $y \in \mathbb{F}_q^n$  tal que,

$$\max\{\omega(y), \omega(y^v)\} = \min_{x \in \mathbb{F}_q^n} \{\max\{\omega(x), \omega(x^v)\}\}$$

e  $R = \max\{\omega(y), \omega(y^v)\}$ . Suponha que existe  $z \in B(0, R-1) \cap B(v, R-1)$ . Como  $z \in B(0, R-1)$  temos que  $\omega(z) \leq R-1$ , e portanto,

$$\omega(z) < \max\{\omega(y), \omega(y^v)\}.$$

Como  $z \in B(v, R-1)$ , usando o **Lema 1**, temos que  $z^v \in B(0, R-1)$ . Mas então,  $\omega(z^v) \leq R-1$ , e portanto,

$$\omega(z^v) < \max\{\omega(y), \omega(y^v)\}.$$

Mas isso quer dizer que

$$\max\{\omega(z), \omega(z^v)\} < \max\{\omega(y), \omega(y^v)\},$$

uma contradição. Logo,  $B(0, R-1) \cap B(v, R-1) = \emptyset$ . Como  $y \in B(0, R) \cap B(v, R)$ , nosso teorema está provado.  $\square$

O resultado motiva a seguinte definição.

**Definição 25.** *Seja  $d$  uma métrica invariante por translação sobre  $\mathbb{F}_q^n$  e  $v \in \mathbb{F}_q^n$ . Chamamos  $y \in \mathbb{F}_q^n$  de **vetor-raio de  $v$**  se*

$$\max\{\omega(y), \omega(y^v)\} = \min_{x \in \mathbb{F}_q^n} \{\max\{\omega(x), \omega(x^v)\}\}.$$

**Exemplo 24.** *No Exemplo 23 anterior todos os vetores são um vetor-raio de  $v$ .*

Para encontrar o raio de empacotamento de um vetor  $v \in \mathbb{F}_q^n$ , então, devemos encontrar um vetor-raio deste. Se procurarmos no espaço todo, no pior dos casos teríamos que avaliar o peso de  $q^n$  vetores. No próximo resultado iremos reduzir este espaço de busca para o caso de códigos posets. Mostraremos duas coisas. Uma delas é que sempre haverá um vetor-raio cujo suporte está contido no suporte de  $v$ , e a outra, que essencialmente não há nenhuma diferença entre o caso  $q = 2$  e o caso geral.

Antes de provar o próximo resultado precisamos da seguinte notação:

**Definição 26.** A notação de Iverson introduzida em [7] é definida da seguinte forma:

$$[P] = \begin{cases} 1 & \text{se } P \text{ é verdadeiro} \\ 0 & \text{se } P \text{ é falso} \end{cases}$$

onde  $P$  é uma sentença que pode ser verdadeira ou falsa.

Para uma discussão sobre as vantagens desta notação indicamos [10].

**Lema 2.** Seja  $P$  um poset e  $v \in \mathbb{F}_q^n$ . Então existe  $x \in \mathbb{F}_q^n$  tal que:

1.  $x$  é um vetor-raio de  $v$  ;
2.  $\text{supp}(x) \subseteq \text{supp}(v)$  ;
3. Dado  $i \in P$  temos que  $x_i = v_i$  ou  $x_i = 0$ .

*Demonstração.* Seja  $z \in \mathbb{F}_q^n$  um vetor-raio de  $v$ . Defina  $x \in \mathbb{F}_q^n$  tal que

$$x_i = v_i[z_i \neq 0].$$

Por definição,  $x$  satisfaz a terceira condição. Para ver que satisfaz a segunda, tome  $i \in \text{supp}(x)$ . Então,  $x_i \neq 0$ , mas isto só pode ocorrer se  $v_i \neq 0$  e  $z_i \neq 0$ . Logo,

$$i \in \text{supp}(v) \cap \text{supp}(z).$$

Para provar a primeira condição, note que

$$\begin{aligned} x_i^v &= v_i - x_i \\ &= v_i - v_i[z_i \neq 0] \\ &= v_i(1 - [z_i \neq 0]) \\ &= v_i[z_i = 0]. \end{aligned}$$

Mas como  $z_i^v = v_i - z_i$ , então

$$[z_i = 0] = [z_i^v = v_i]$$

e daí,

$$x_i^v = v_i[z_i^v = v_i].$$

Logo, se  $i \in \text{supp}(x^v)$ , então  $x_i^v \neq 0$ , e portanto,  $v_i \neq 0$  e  $z_i^v = v_i$ , ou seja,  $z_i^v \neq 0$ . Em outros termos, temos

$$\text{supp}(x^v) \subseteq \text{supp}(z^v).$$

Mas já vimos que

$$\text{supp}(x) \subseteq \text{supp}(z),$$

e portanto,

$$\omega_P(x) \leq \omega_P(z)$$

e

$$\omega_P(x^v) \leq \omega_P(z^v).$$

Ou seja,

$$\max\{\omega_P(x), \omega_P(x^v)\} \leq \max\{\omega_P(z), \omega_P(z^v)\},$$

e a primeira condição é satisfeita.  $\square$

Com este resultado, o espaço de busca para um vetor-raio de  $v \in \mathbb{F}_q^n$  é reduzido ao espaço  $2^{\text{supp}(v)}$ , o conjunto das partes do  $\text{supp}(v)$ . A cardinalidade do corpo não importa para determinar o vetor-raio, o que era de se esperar já que na métrica poset, ao tomarmos o suporte do vetor, estamos apenas distinguindo entre coordenadas com valor zero e diferente de zero. Também não importa a estrutura do poset fora do suporte de  $v$ . Sendo assim, o raio de empacotamento de um vetor é uma propriedade do poset gerado pelo seu suporte. Faz sentido então identificar um vetor com o seu suporte. Tendo isto em mente, as seguintes definições são naturais.

**Definição 27.** *Seja  $P$  um poset e  $A \subseteq P$ . Definimos o  **$P$ -peso de  $A$**  como sendo*

$$\omega_P(A) = |\langle A \rangle|,$$

*a cardinalidade do ideal gerado por  $A$ .*

**Definição 28.** *Seja  $P$  um poset,  $v \in \mathbb{F}_q^n$ , e  $A \subseteq \text{supp}(v)$ . Definimos o **complementar de  $A$  com relação a  $v$**  como sendo*

$$A^v = \text{supp}(v) - A.$$

Note que, por definição,  $(A, A^v)$  é uma partição de  $\text{supp}(v)$ . Estamos prontos para mostrar que o problema da determinação do vetor-raio de um vetor pode ser visto como um problema de partição.

**Lema 3.** *Seja  $P$  um poset e  $v \in \mathbb{F}_q^n$ . Então,*

$$\min_{x \in \mathbb{F}_q^n} \{\max\{\omega_P(x), \omega_P(x^v)\}\} = \min_{A \subseteq \text{supp}(v)} \{\max\{\omega_P(A), \omega_P(A^v)\}\}$$

*Demonstração.* Defina  $f_v : 2^{\text{supp}(v)} \mapsto \mathbb{F}_q^n$  como a função que leva  $A \subseteq \text{supp}(v)$  em  $f_v(A) = x$  onde  $x_i = v_i[i \in A]$ . Esta função é injetiva e preserva o peso, i.e.  $\omega_P(A) = \omega_P(f_v(A))$ . Além disso, ela respeita a complementação, pois, como  $(A, A^v)$  é uma partição de  $\text{supp}(v)$ ,

$$f_v(A) + f_v(A^v) = v_i[i \in A] + v_i[i \in A^v] = v_i,$$

e portanto,

$$f_v(A^v) = f_v(A)^v.$$

Mas daí,

$$\min_{A \subseteq \text{supp}(v)} \{\max\{\omega_P(A), \omega_P(A^v)\}\} = \min_{x \in f_v(2^{\text{supp}(v)})} \{\max\{\omega_P(x), \omega_P(x^v)\}\}.$$

Pelo **Lema 2**, existe um vetor-raio de  $v$  em  $f_v(2^{\text{supp}(v)})$ , e portanto, o resultado esta provado.  $\square$

Transformamos o nosso problema em um problema de partição. Definimos um conjunto-raio analogamente ao que foi feito com vetores.

**Definição 29.** *Seja  $P$  um poset e  $x \in \mathbb{F}_q^n$ . Chamamos um conjunto  $X \subseteq \text{supp}(v)$  de conjunto-raio de  $v$  se*

$$\max\{\omega_P(X), \omega_P(X^v)\} = \min_{A \subseteq \text{supp}(v)} \{\max\{\omega_P(A), \omega_P(A^v)\}\}$$

O espaço de soluções do nosso problema pode ser reduzido ainda mais. Para isso, precisaremos da seguinte definição:

**Definição 30.** *Seja  $P$  um poset e  $A \subseteq P$ . Denotamos o conjunto dos elementos maximais de  $A$  por  $M_A$ .*

Note que, o peso de um conjunto é determinado pelos seus elementos maximais, i.e.  $\omega_P(A) = \omega_P(M_A)$  pois o ideal gerado pelos dois é igual. É de se esperar, então, que os elementos maximais tenham um papel importante na determinação do raio de empacotamento.

**Lema 4.** *Seja  $P$  um poset e  $v \in \mathbb{F}_q^n$ . Então, existe um conjunto-raio  $X$  de  $v$  tal que  $(M_X, M_{X^v})$  é uma partição de  $M_{\text{supp}(v)}$ .*

*Demonstração.* Seja  $Z$  um conjunto-raio de  $v$ . Defina

$$X = \langle M_Z \cap M_{\text{supp}(v)} \rangle \cap \text{supp}(v).$$

Primeiro, mostraremos que  $X$  é um conjunto-raio de  $v$ . Da definição,

$$M_X = M_Z \cap M_{\text{supp}(v)},$$

e portanto,

$$\langle X \rangle = \langle M_Z \cap M_{\text{supp}(v)} \rangle \subseteq \langle M_Z \rangle$$

donde

$$\omega_P(X) \leq \omega_P(Z).$$

Seja  $i \in X^v$ . Por definição,  $i \in \text{supp}(v)$  e  $i \notin X$ , e portanto,

$$i \notin \langle M_Z \cap M_{\text{supp}(v)} \rangle.$$

Como  $i \in \text{supp}(v)$ , existe  $j \in M_{\text{supp}(v)}$  tal que  $i \leq j$ . Se tivéssemos  $j \in M_Z$  então

$$i \in \langle M_Z \cap M_{\text{supp}(v)} \rangle,$$

uma contradição. Logo,  $j \in M_{Z^v}$ , e portanto,  $i \in \langle M_{Z^v} \rangle$ . Então,  $X^v \subseteq \langle M_{Z^v} \rangle$  e consequentemente

$$\omega_P(X^v) \leq \omega_P(Z^v).$$

Usando as duas desigualdades esta provado que  $X$  é um conjunto-raio de  $v$ . Falta, então, mostrar que  $(M_X, M_{X^v})$  é uma partição de  $M_{\text{supp}(v)}$ . Para isso, basta mostrar que

$$M_{X^v} = M_{\text{supp}(v)} - M_X.$$

Seja  $i \in M_{X^v}$ , então certamente  $i \notin M_X$ . Suponha que  $i \notin M_{\text{supp}(v)}$ , então existe  $j \in M_{\text{supp}(v)}$  tal que  $i < j$ . Como  $j$  não pode estar em  $X^v$  devemos ter  $j \in X$ , mas isso implicaria em  $i \in M_X$ , uma contradição. Logo,  $i \in M_{\text{supp}(v)}$ , e portanto,

$$M_{X^v} \subseteq M_{\text{supp}(v)} - M_X.$$

Seja  $i \in M_{\text{supp}(v)} - M_X$ . Então  $i \in M_{\text{supp}(v)}$  e  $i \notin M_X$ . Como  $i$  não está em  $X$ , certamente está em  $X^v$ , mas além disso, está em  $M_{\text{supp}(v)}$ , e portanto, deve estar em  $M_{X^v}$ . Logo,

$$M_{X^v} \supseteq M_{\text{supp}(v)} - M_X.$$

O Lema esta provado. □

Logo, o problema de encontrar o raio de empacotamento de um vetor pode ser transformado num problema de partição sobre o conjunto dos elementos maximais do seu suporte.

**Teorema 2.** *Seja  $P$  um poset e  $v \in \mathbb{F}_q^n$ . Então,*

$$R_{d_P}(v) = \min_{A, B \subseteq M_{\text{supp}(v)}} \{\max\{\omega_P(A), \omega_P(B)\}\} - 1$$

onde  $(A, B)$  é uma partição de  $M_{\text{supp}(v)}$ .

*Demonstração.* Basta utilizar o **Lema 3** no **Teorema 1**, e daí usando a definição de conjunto-raio aplicar o **Lema 4**. □

Assim, reduzimos o espaço de busca para  $2^{|M_{\text{supp}(v)}|}$  elementos.

Podemos modificar a notação de modo a focar no problema de partição.

**Definição 31.** *Seja  $P$  um poset e  $M_P$  o conjunto dos elementos maximais de  $P$ . Definimos o **raio de empacotamento de  $P$**  como sendo*

$$R(P) = \min_{A, B \subseteq M_P} \{\max\{|\langle A \rangle|, |\langle B \rangle|\}\} - 1$$

onde  $(A, B)$  é uma partição de  $M_P$ .

Uma partição que minimiza a expressão é chamada de **partição ótima**.

Como corolário direto do **Teorema 2** temos:

**Corolário 1.** *Seja  $P$  um poset e  $v \in \mathbb{F}_q^n$ . Então,*

$$R_{d_P}(v) = R(\text{supp}(v)).$$

O problema de encontrar o raio de empacotamento de um vetor foi transformado então no problema de encontrar o raio de empacotamento de um poset, que chamaremos de **problema da partição para posets**. Este problema é uma generalização de um problema famoso chamado de “problema da partição” que será discutido no próximo capítulo.

## Capítulo 4

# O Problema da Partição

Este capítulo interrompe a linha de pensamento do capítulo anterior para discutir o problema da partição. No próximo capítulo veremos que este problema é um caso particular do “problema da partição para posets”.

Na primeira seção damos uma introdução ao problema. As referências principais são os artigos [9] em que é provado que o problema é NP-difícil e [1] em que se estuda o comportamento de transição de fase do problema.

Na segunda seção discutimos o método da diferenciação introduzida em [8] e demonstramos alguns resultados já conhecidos referentes a esta heurística.

Na terceira seção discutimos a extenuação do método da diferenciação para um algoritmo completo feito em [12].

### 4.1 Introdução

O **problema da partição**, que também chamaremos de **problema da partição clássica**, é definido da seguinte forma: dado uma lista finita  $S$  de inteiros positivos, encontrar uma partição  $(S_1, S_2)$  de  $S$  que minimize

$$\max \left\{ \sum_{x \in S_1} x, \sum_{y \in S_2} y \right\}.$$

Isso é equivalente a minimizar a **discrepância**:

$$\Delta(S_1, S_2) = \left| \sum_{x \in S_1} x - \sum_{y \in S_2} y \right|.$$

Este problema tem importância tanto teórica quanto prática. Em [9], Karp prova que este problema é NP-difícil. Sendo assim, a menos que  $P=NP$ , não há possibilidade de existir um algoritmo em tempo polinomial para o problema. Além disso, qualquer problema em NP pode ser reduzido ao problema da partição em tempo polinomial, i.e., se houver um algoritmo em tempo polinomial que resolve o problema da partição, então  $P=NP$ .

Uma aplicação simples deste problema é a seguinte: Dados duas máquinas idênticas, uma lista de tarefas, e o tempo que demora para a máquina executar cada uma das tarefas, distribuir as tarefas entre as duas máquinas de modo a que todas sejam executadas no menor tempo possível. Neste caso, a lista a ser particionada é o tempo de demora de cada uma das tarefas. A **discrepância** é o quanto uma máquina trabalhou mais do que a outra.

A priori, a menor discrepância possível é 0 quando a soma dos elementos da lista é par e 1 quando a soma é ímpar. Uma partição que atinge a menor discrepância possível é chamada de **partição perfeita**. A existência destas partições têm um papel fundamental nos algoritmos utilizados para resolver o problema, pois ao encontrar uma partição perfeita, o algoritmo pode terminar a busca. Veremos adiante que isto determina a dificuldade de resolver o problema. Sendo assim, distinguimos entre os casos fáceis, em que a probabilidade de haver uma partição perfeita é alta, e os casos difíceis, nos quais esta probabilidade é baixa. O espaço de probabilidade que estamos considerando é o definido no enunciado do **Teorema 3**.

Por simplicidade sempre assumiremos que a lista é ordenada começando pelos maiores elementos. Ou seja, se escrevermos  $S = (x_1, x_2, x_3)$  assumiremos que  $x_1 \geq x_2 \geq x_3$ . É um fato conhecido que uma lista qualquer pode ser ordenada em tempo polinomial, fazendo com que isto não afete a complexidade do problema.

**Exemplo 25.** *Seja  $S = (8, 7, 6, 5, 4)$ . Então a partição  $(8, 7) (6, 5, 4)$  tem discrepância 0, e portanto, é perfeita e ótima.*

Em [14], Mertens argumenta, utilizando métodos da mecânica estatística, que existe uma transição de fase que separa os casos fáceis e difíceis do problema da partição. Uma transição de fase num problema combinatorial é uma mudança brusca no comportamento qualitativo ao variar-se algum parâmetro. Neste caso, o parâmetro é  $\kappa = m/n$  onde  $n$  é a quantidade de números a serem particionados e  $m$  é a quantidade de bits necessários para expressar esses números. Quando  $m$  e  $n$  vão para infinito, a probabilidade de existir uma partição ideal vai para 1, quando  $\kappa < 1$  e vai para 0 quando  $\kappa > 1$ . Este resultado foi demonstrado de forma rigorosa por Borgs, Chayes e Pittel em [1]. Segue o enunciado do teorema.

**Teorema 3.** [1] *Suponha que escolhemos  $n$  números independentemente e uniformemente do conjunto  $[2^m] = \{1, 2, \dots, 2^m\}$  onde  $m$  é um inteiro maior do que 1 e que depende de  $n$ . Seja  $m = \kappa_n n$  e suponha que  $\lim_{n \rightarrow \infty} \kappa_n = \kappa \in [0, \infty]$ . Então*

$$\lim_{n \rightarrow \infty} P_n(\text{existe partição perfeita}) = \begin{cases} 1 & \text{se } \kappa < 1 \\ 0 & \text{se } \kappa > 1 \end{cases}$$

Os problemas difíceis então são os de tamanho intermediário. Uma explicação intuitiva para isto é a seguinte: Dado os  $n$  inteiros a serem particionados, a quantidade de partições possíveis é  $2^n$ . Se os inteiros são escolhidos no intervalo de 0 a  $M$ , então a discrepância pode atingir no máximo  $Mn$  valores, pois  $Mn$  é o maior valor possível a ser atingido. Se fixarmos  $M$  como uma constante e formos aumentando o valor de  $n$ , o número de partições possíveis cresce exponencialmente. No entanto, os valores possíveis de discrepância para estas partições cresce linearmente. Sendo assim, é de se esperar que a quantidade de partições com mesma discrepância vá aumentando, em particular as partições perfeitas. Como dissemos anteriormente, quanto mais partições perfeitas existem, maiores são as chances de um algoritmo poder encontrá-las rapidamente.

Discutiremos agora os algoritmos utilizados para resolver o problema da partição. O algoritmo mais óbvio é a busca por força bruta. O espaço de busca no entanto é muito grande,  $2^n$  para  $n$  números. Um outro algoritmo chamado de algoritmo de Schroepel e Shamir (SS) é apresentado em [20]. O espaço de busca é diminuído para  $2^{n/2}$ , no entanto isto tem um custo. O espaço de memória utilizado é exponencial, diferente dos outros algoritmos que foram ou serão discutidos. Isto faz com que o algoritmo SS não seja útil para casos em que se trabalhe com mais do que 100 números. Nos casos difíceis com menos de 100 números, SS é o melhor algoritmo conhecido no momento. Para resolver casos grandes, primeiro discutiremos heurísticas que, em tempo polinomial, nos dão soluções aproximadas para o problema, e depois mostraremos como transformá-los em algoritmos completos (que devolvem as soluções).

A heurística mais óbvia é a gananciosa. Dado os números a serem particionados, ordenamos os e colocamos o maior num subconjunto arbitrário. Daí vamos colocando o restante dos números de maior a menor no subconjunto com a menor soma (se ambos os subconjuntos estão com a mesma soma, escolha um arbitrariamente).

**Exemplo 26.** *Seja  $(8, 7, 6, 5, 4)$  a lista a ser particionada. Daí o algoritmo ganancioso passaria pelos seguintes estados (os dois números fora do parêntesis representam a soma em cada subconjunto):  $8, 0(7, 6, 5, 4)$ ;  $8, 7(6, 5, 4)$ ;  $8, 13(5, 4)$ ;  $13, 13(4)$ ;  $17, 13()$ , dando uma discrepância de 4. Neste caso, o algoritmo não encontrou a discrepância mínima. Note que, se estamos apenas interessados na discrepância, podemos apenas indicar a diferença entre os subconjuntos fora dos parêntesis. Daí a notação fica:  $8(7, 6, 5, 4)$ ;  $1(6, 5, 4)$ ;  $5(5, 4)$ ;  $0(4)$ ;  $4()$ .*

## 4.2 A Heurística KK

A melhor heurística conhecida atualmente é devido à Karmarkar e Karp ([8]) e é conhecida como **método da diferenciação** ou heurística *KK*. Novamente, o primeiro passo é ordenar os  $n$  números. Daí, substitui-se os dois maiores números pela diferença deles. Isso equivale a decidir que eles vão para subconjuntos diferentes sem especificar qual. Faz-se isso  $n - 1$  vezes e o número que sobra é a discrepância.

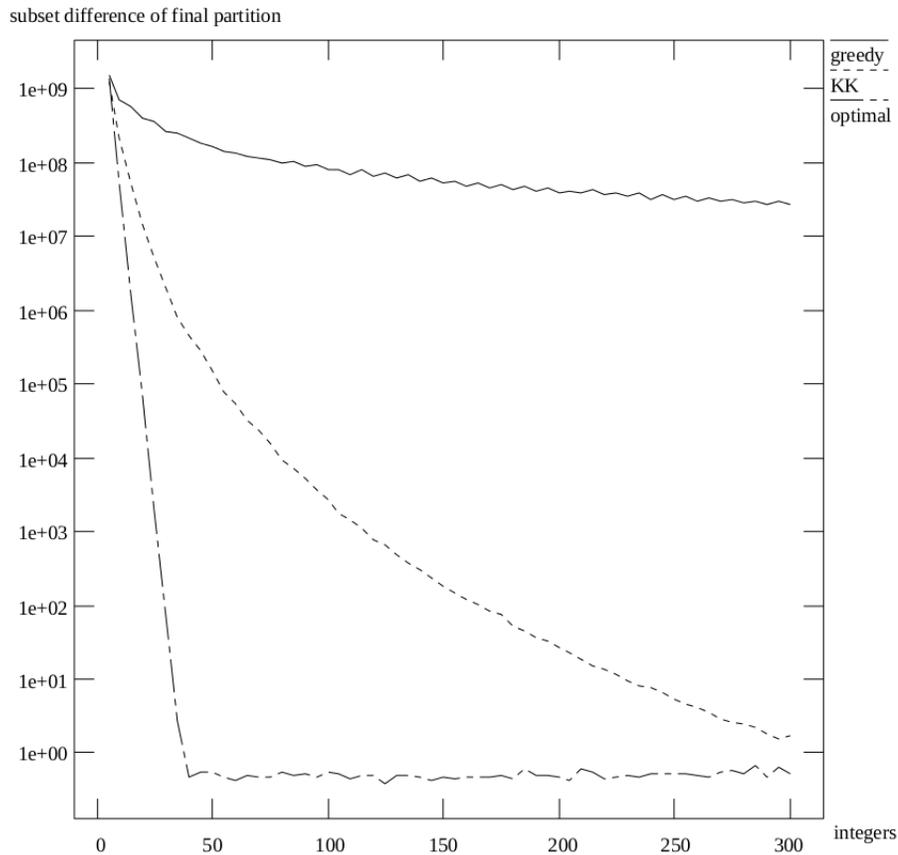


Figura 4.1: Fonte: [12]

**Exemplo 27.** Seja  $(8, 7, 6, 5, 4)$  a lista a ser particionada. O algoritmo KK passa pelos seguintes estados:  $(8, 7, 6, 5, 4)$ ,  $(6, 5, 4, 1)$ ,  $(4, 1, 1)$ ,  $(3, 1)$ ,  $(2)$ , dando uma discrepância de 2. Neste caso, a heurística KK também não encontrou a solução ótima, no entanto encontrou uma solução melhor do que a heurística gananciosa.

Em média a heurística KK encontra soluções bem melhores do que a gananciosa. A **Figura 4.1** retirado de [12] compara as duas heurísticas. Os números a serem particionados são escolhidos uniformemente distribuídos entre 0 e 10 bilhões. O eixo horizontal indica a quantidade de números particionados, e o vertical, as discrepâncias encontradas por cada heurística (escala logarítmica). Cada ponto nas duas linhas superiores é a média de 1000 instâncias aleatórias do problema, enquanto que os da linha de baixo é a média de 100 problemas. Ao aumentar-se a quantidade de números a serem particionados, o desempenho da heurística KK é várias ordens de magnitude melhor do que a gananciosa. A figura também mostra o fato de que mantendo-se o tamanho dos números fixos, e aumentando-se a

quantidade de números a serem particionados a existência de partições perfeitas vai crescendo.

Uma explicação para a diferença de desempenho entre a heurística KK e a gananciosa é a seguinte: A discrepância tem a ordem de tamanho do último número a ser designado um subconjunto. Na heurística gananciosa este é o tamanho do menor número da lista a ser particionada. Já no caso da heurística KK, a operação de diferenciar diminui dramaticamente o tamanho dos números que vão restando. Daí, com mais números na lista, o tamanho do último número tende a ser bem menor do que o menor número da lista inicial.

Como já vimos, a heurística KK nem sempre encontra o resultado certo. Para que isto ocorresse sempre, haveria uma partição ótima em que os dois maiores elementos não pertencessem ao mesmo subconjunto. O próximo resultado mostra, que para uma lista com menos de 4 números, isto sempre ocorre. O exemplo que demos no início da seção mostra que para 5 números este resultado não vale.

**Proposição 11.** *Seja  $S$  uma lista ordenada com 4 elementos ou menos. Então existe uma partição de  $S$  com discrepância mínima tal que os dois maiores elementos de  $S$  não pertencem ao mesmo subconjunto.*

*Demonstração.* Para uma lista com um elemento não há o que ser discutido. Com dois elementos a afirmação é óbvia, já que subtrair um do outro é melhor do que somá-los. Para três elementos fazemos da seguinte forma: Seja  $S = (x_1, x_2, x_3)$ . Se colocarmos  $x_1$  e  $x_2$  num mesmo subconjunto, nos resta apenas colocar  $x_3$  no outro. Daí ficamos com a partição  $(x_1, x_2) (x_3)$ . Mas a partição  $(x_1)(x_2, x_3)$  tem discrepância menor ou igual pois

$$x_1 \leq x_1 + x_2$$

$$x_2 + x_3 \leq x_1 + x_2.$$

Neste caso, essa partição é ótima, pois ao colocarmos  $x_1$  e  $x_2$  em subconjuntos diferentes é equivalente ao problema de particionar  $(x_1 - x_2, x_3)$  e quando temos dois elementos, o ideal é subtraí-los (colocá-los em subconjuntos diferentes), mas daí ficamos com a partição  $(x_1)(x_2, x_3)$ . Para quatro elementos: Seja  $S = (x_1, x_2, x_3, x_4)$ . Colocar  $x_1$  e  $x_2$  no mesmo subconjunto faz com que o problema seja equivalente a particionar  $(x_1 + x_2, x_3, x_4)$ . Mas já vimos que a solução para isso é a partição  $(x_1 + x_2) (x_3, x_4)$ , e portanto, a melhor partição mandando-se os maiores dois elementos para o mesmo subconjunto é  $(x_1, x_2) (x_3, x_4)$ . Mas a partição  $(x_1, x_3) (x_2, x_4)$  tem discrepância menor ou igual pois

$$x_1 + x_3 \leq x_1 + x_2$$

$$x_2 + x_4 \leq x_1 + x_2.$$

Neste caso, essa partição não é necessariamente ótima. Mas é garantido que existe solução ótima em que os dois maiores números não estão no mesmo subconjunto.  $\square$

Na demonstração vimos que no caso de três elementos  $(x_1, x_2, x_3)$  a partição  $(x_1)$   $(x_2, x_3)$  é sempre ótima. Podemos estender isso para mais elementos. Para isso é conveniente usar a seguinte notação: Dado uma partição, escolhemos o subconjunto que contém o maior elemento para ser o subconjunto dos positivos, e o outro para ser o dos negativos. Daí representamos a partição como uma sequência dos símbolos  $+$  e  $-$  onde um  $+$  ( $-$ ) na  $i$ -ésima posição representa o fato do  $i$ -ésimo elemento da lista pertencer ao subconjunto dos positivos (negativos). Em particular, o primeiro elemento da sequência é sempre  $+$ .

**Exemplo 28.** Seja  $(8, 7, 6, 5, 4)$  a lista a ser particionada. A partição  $(8, 7)$   $(6, 5, 4)$  fica representada na forma  $++----$ . Essa representação pode ser vista como as instruções para encontrar a discrepância:

$$\Delta(++----) = | +8 + 7 - 6 - 5 - 4 | = 0.$$

Na demonstração vimos que para 3 números  $+-$  é uma solução. Vimos também que para 4 números  $+-$  é dominado por  $++-$ . No entanto, não é garantido que  $+-$  seja uma solução. O próximo resultado mostra quais são as soluções garantidas.

**Proposição 12.** Seja  $S$  uma lista ordenada com 4 elementos. Então  $+-$  ou  $+-+$  é sempre ótima.

*Demonstração.* Seja  $S = (x_1, x_2, x_3, x_4)$ . No resultado anterior já vimos que neste caso sempre existe partição ótima, em que os dois maiores elementos não pertencem ao mesmo subconjunto. Logo, o problema é equivalente a particionar  $(x_1 - x_2, x_3, x_4)$  (lista não ordenada). Separamos o problema então em 3 casos: Caso 1:  $x_1 - x_2 \geq x_3$ . Neste caso, a lista ordenada é  $(x_1 - x_2, x_3, x_4)$ . Já vimos que, neste caso,  $+-$  é ótimo. Logo,

$$+(x_1 - x_2) - x_3 - x_4 = +x_1 - x_2 - x_3 - x_4$$

é ótimo. Logo, neste caso,  $+-$  é ótimo.

Caso 2:  $x_3 \geq x_1 - x_2 \geq x_4$ . Neste caso, a lista ordenada é  $(x_3, x_1 - x_2, x_4)$ . Análogo ao caso anterior temos que

$$x_3 - (x_1 - x_2) - x_4 = -x_1 + x_2 + x_3 - x_4$$

é ótimo. Logo, neste caso,  $-+$  é ótimo. Usando a convenção de que sempre mandamos o maior elemento para o subconjunto dos positivos temos que renomear os subconjuntos. Daí,  $+-$  é ótimo.

Caso 3:  $x_4 \geq x_1 - x_2$ . Neste caso, a lista ordenada é  $(x_3, x_4, x_1 - x_2)$ . Análogo aos casos anteriores temos que

$$x_3 - x_4 - (x_1 - x_2) = -x_1 + x_2 + x_3 - x_4$$

é ótimo. Este caso é idêntico ao anterior. □

Poderíamos continuar fazendo isto para listas com números maiores, mas como é de se esperar, os casos ótimos vão crescendo muito.

Vamos analisar agora o primeiro caso em que a heurística KK não nos dá necessariamente uma partição ótima. Este resultado será útil quando estendermos a heurística para um algoritmo completo.

**Teorema 4.** *Seja  $S$  uma lista ordenada com 5 elementos. Então, a heurística KK nos dá uma partição ótima ou  $++--$  é uma partição ótima.*

*Demonstração.* Seja  $S = (x_1, x_2, x_3, x_4, x_5)$ . Suponha que existe uma partição ótima em que os dois maiores elementos aparecem no mesmo subconjunto (se isto não ocorre a heurística KK nos dá uma partição ótima). Então o problema é equivalente a particionar  $(x_1 + x_2, x_3, x_4, x_5)$ . Mas neste problema a heurística KK funciona e, mandando os dois maiores elementos para subconjuntos diferentes, o problema fica equivalente a particionar  $(x_1 + x_2 - x_3, x_4, x_5)$ . Como sabemos que neste caso  $+-$  é ótimo, então

$$x_1 + x_2 - x_3 - x_4 - x_5$$

é ótimo. Logo,  $++--$  é ótimo. □

### 4.3 CKK

Estas heurísticas, a KK e a gananciosa, nos dão apenas soluções aproximadas. Em [12], Korf mostra como estas heurísticas podem ser estendidas para um “complete anytime algorithm”, i.e. um algoritmo que vai encontrando soluções melhores e melhores quanto mais tempo as deixamos rodando, até que em algum momento ela garante que a solução encontrada é ótima. Neste trabalho, discutiremos apenas a extensão da heurística KK, chamada de Complete Karmarkar Karp (CKK), pois ela é a que obtêm os melhores resultados em média.

Quando estamos utilizando a heurística KK, sempre mandamos os maiores elementos para subconjuntos diferentes. A outra opção seria mandá-los para o mesmo subconjunto. Podemos então construir uma árvore binária da seguinte forma: O primeiro nodo é a lista de  $n$  números a ser particionada. Deste, saem dois ramos, um para esquerda na qual colocamos a lista em que se substituiu os dois maiores elementos pela diferença deles, e um para a direita na qual colocamos uma lista em que se substituiu os dois maiores pela soma deles. Repetimos o processo para cada nodo. No final do processo teremos  $2^{n-1}$  folhas com os possíveis valores para a discrepância. O problema da partição vira, então, um problema de busca em árvore binária.

A princípio, se não existe solução perfeita, temos que olhar todas as folhas. Nesse caso, não faz muita diferença a ordem em que buscamos as folhas pois precisamos do valor de todas elas. No entanto, se existe alguma solução perfeita, faz toda a diferença a ordem da busca pois, ao encontrar uma folha com discrepância mínima,

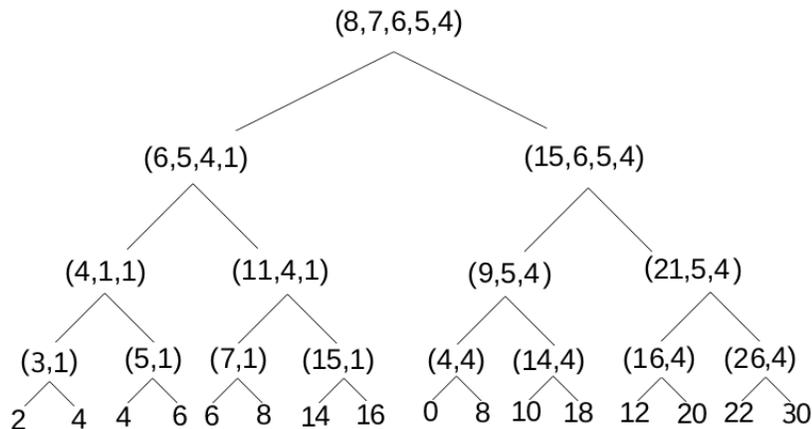


Figura 4.2: Árvore para particionar (8, 7, 6, 5, 4).

podemos parar o processo todo. Em [11], Korf compara os dois melhores algoritmos de busca para a árvore gerada no algoritmo CKK.

Um destes, chamado de “depth-first search” (DFS), consiste em buscar as folhas da esquerda para direita. Este algoritmo é bem útil se precisarmos olhar todas as folhas. No problema da partição esta situação ocorre nos problemas difíceis em que a chance de encontrar uma partição perfeita é baixa.

O outro, chamado de “Improved Limited Discrepancy Search” (ILDS), consiste em dar uma preferência por ramos que vão a esquerda. Primeiro buscamos as folhas na qual não precisamos tomar nenhum ramo que vai para a direita, em seguida buscamos as folhas nas quais temos que tomar um ramo para a direita, e assim sucessivamente. Esta busca usa fortemente a heurística KK e se mostra eficiente quando a chance de existir uma partição perfeita é alta, ou seja nos casos fáceis. Nestes casos, esta encontra as partições perfeitas mais rapidamente, em média, do que o DFS, e portanto, pode parar a busca antes.

Um outro aspecto importante da busca em uma árvore é conseguir formas de podar ramos. Por exemplo, já vimos que, ao particionar 4 números, podemos encontrar a solução ótima diferenciando. Sendo assim, ao encontrarmos uma lista com 4 números não precisamos olhar mais para ramos que vão para a direita. Nesse sentido, dizemos que estes ramos foram podados. Vimos também que no caso de 3 números a partição da forma  $+ - -$  é ótima. Logo, ao chegarmos numa lista dessas, já sabemos qual valor a folha ótima vai ter. No caso de 5 números se tomarmos o ramo que vai para a direita, já vimos que a solução ótima é do tipo  $+ + - - -$ . Um outro critério útil é ver se o primeiro número da lista é maior do que a soma do resto pois, daí, a solução ótima é da forma  $+ - - \dots -$ .

Em suma, para decidir qual algoritmo usar, podemos separar o problema da partição em três casos. Nos casos fáceis, em que as chances de existirem uma partição perfeita é alta, o melhor algoritmo é o CKK usando a busca ILDS. Nos casos

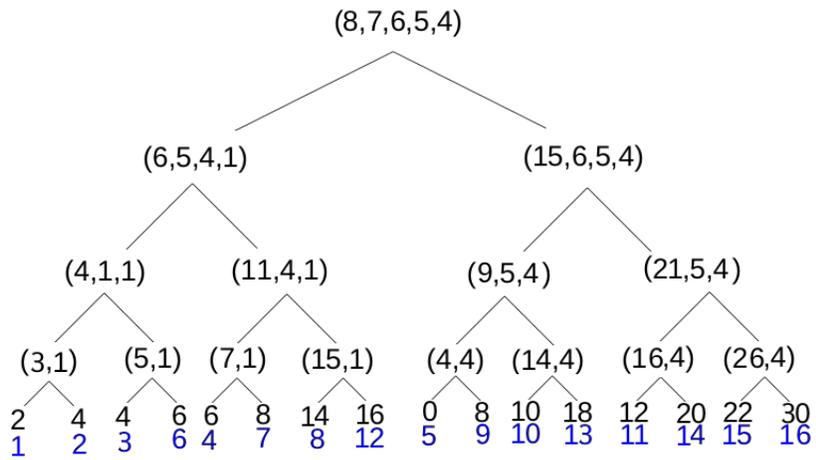


Figura 4.3: Os números azuis indicam a ordem em que o ILDS busca as folhas.

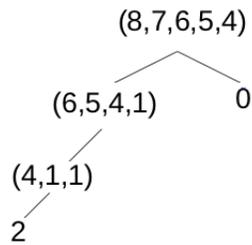


Figura 4.4: A árvore com os ramos podados.

difíceis com poucos números, digamos menos de cem por exemplo, o melhor é o SS. E nos casos difíceis com bastantes números, como não podemos usar o SS, o melhor algoritmo é o CKK usando a busca DFS.

## Capítulo 5

# Raio de Empacotamento de Códigos Poset II

Neste capítulo voltaremos ao assunto de determinar o raio de empacotamento de um código poset interrompido pelo capítulo anterior.

Na primeira seção mostraremos que, no caso de posets cujos elementos maximais têm ideais disjuntos, o problema de determinar o raio de empacotamento do poset é equivalente ao problema da partição, justificando o motivo pelo qual escrevemos o capítulo anterior.

Na segunda seção, abordaremos o problema no caso de um poset qualquer introduzindo o conceito da discordância que faz o papel que a discrepância fazia no problema da partição.

Na terceira seção, iremos generalizar o método da diferenciação e como completá-lo para o problema da partição de posets.

Na quarta seção voltaremos ao problema de encontrar o raio de empacotamento de um código. Mostraremos técnicas para encontrar desigualdades entre os raios de empacotamento de posets sem calculá-los explicitamente.

### 5.1 O caso dos ideais disjuntos

No final do capítulo 3 foi visto que, o problema da partição para posets pode ser visto como uma generalização do problema da partição. O próximo resultado mostra que, no caso em que os ideais gerados pelos elementos maximais do poset são disjuntos, o problema da partição para posets é equivalente ao problema da partição clássica.

**Teorema 5.** *Seja  $P$  um poset com elementos maximais  $M_P = \{x_1, x_2, \dots, x_m\}$*

tal que

$$\langle x_i \rangle \cap \langle x_j \rangle = \emptyset, \forall i \neq j$$

e denotemos  $\omega_P(x_i) = l_i$ . Então, encontrar o raio de empacotamento de  $P$  é equivalente a resolver o problema da partição para  $S = (l_1, l_2, \dots, l_m)$ .

*Demonstração.* Seja  $(A, B)$  uma partição de  $M_P$ . Defina

$$S_1 = (l_i : x_i \in A)$$

$$S_2 = (l_i : x_i \in B).$$

É claro que  $(S_1, S_2)$  é uma partição de  $S$ . Além disso, como

$$\langle x_i \rangle \cap \langle x_j \rangle = \emptyset, \forall i \neq j$$

temos que

$$\begin{aligned} \omega_P(A) &= \sum_{x \in A} \omega_P(x) \\ &= \sum_x \omega_P(x)[x \in A] \\ &= \sum_i \omega_P(x_i)[x_i \in A] \\ &= \sum_i l_i[l_i \in S_1] \\ &= \sum_{l \in S_1} l. \end{aligned}$$

Analogamente,

$$\omega_P(B) = \sum_{l \in S_2} l,$$

e portanto, encontrar uma partição  $(A, B)$  de  $M_P$  que minimize o máximo dentre  $\omega_P(A)$  e  $\omega_P(B)$  é equivalente a encontrar uma partição  $(S_1, S_2)$  de  $S$  que minimize o máximo dentre  $\sum_{l \in S_1} l$  e  $\sum_{l \in S_2} l$ .  $\square$

Observe que a condição do teorema equivale a exigir que cada componente conexa do diagrama de Hasse de  $P$  possui um único elemento maximal.

Logo, para encontrar o raio de empacotamento de um poset, podemos utilizar os métodos discutidos no capítulo anterior. Um dos conceitos importantes que foi utilizado é o da discrepância. Definimos a seguir a discrepância para o caso de posets.

**Definição 32.** Seja  $P$  um poset e  $(A, B)$  uma partição do conjunto  $M_P$  de elementos maximais de  $P$ . Definimos a **discrepância** entre  $A$  e  $B$  como sendo

$$\Delta(A, B) = |\omega_P(A) - \omega_P(B)|$$

e a **discrepância mínima** de  $P$  como sendo

$$\Delta^*(P) = \min_{X \sqcup Y = M_P} \Delta(X, Y),$$

onde  $X \sqcup Y$  indica uma união disjunta.

No capítulo anterior vimos que, no problema da partição, queremos minimizar a discrepância. Consequentemente, isso será verdade também quando quisermos encontrar o raio de empacotamento de um poset no caso em que os ideais gerados pelos elementos maximais não se interceptam. Neste caso, podemos escrever o raio de empacotamento do poset em função de sua discrepância mínima.

**Teorema 6.** Seja  $P$  um poset de cardinalidade  $n$  com elementos maximais  $M_P = \{x_1, x_2, \dots, x_m\}$  tal que

$$\langle x_i \rangle \cap \langle x_j \rangle = \emptyset \quad \forall i \neq j.$$

Então, o raio de empacotamento de  $P$  é

$$R(P) = \frac{n}{2} + \frac{\Delta^*(P)}{2} - 1.$$

*Demonstração.* Seja  $(A, B)$  uma partição de  $M_P$ . Então, valem as seguintes equações:

$$\Delta(A, B) = \max\{\omega_P(A), \omega_P(B)\} - \min\{\omega_P(A), \omega_P(B)\}$$

$$n = \max\{\omega_P(A), \omega_P(B)\} + \min\{\omega_P(A), \omega_P(B)\}.$$

A primeira equação sai direto da definição. A segunda sai do fato de não haver interseção dos ideais gerados por  $A$  e por  $B$ , e portanto,

$$n = \omega_P(A) + \omega_P(B).$$

Somando ambas e dividindo por dois ficamos com

$$\max\{\omega_P(A), \omega_P(B)\} = \frac{n}{2} + \frac{\Delta(A, B)}{2}.$$

Mas daí, pelo **Teorema 2**,

$$\begin{aligned} R(P) &= \min_{A \sqcup B = M_P} \max\{\omega_P(A), \omega_P(B)\} - 1 \\ &= \min_{A \sqcup B = M_P} \frac{n}{2} + \frac{\Delta(A, B)}{2} - 1 \\ &= \frac{n}{2} + \frac{\Delta^*(P)}{2} - 1 \end{aligned}$$

□

Na próxima seção veremos que a condição expressa no **Teorema 6** é de fato relevante, pois no caso geral de um poset qualquer, encontrar o raio de empacotamento não é equivalente a minimizar a discrepância.

**Exemplo 29.** *Seja  $H$  o poset de Hamming com  $n$  elementos. O problema de partição associado a este é o de particionar uma lista com  $n$  uns. Neste caso, a menor discrepância ocorre quando se divide os uns o mais igual possível. Se  $n$  é par, dá para dividir em duas listas iguais obtendo uma discrepância mínima de  $\Delta^*(H) = 0$ . No entanto, se  $n$  é ímpar, o melhor que se pode fazer é dividir em duas listas onde, uma tem um elemento a mais do que a outra, obtendo uma discrepância mínima de  $\Delta^*(H) = 1$ . Logo,*

$$\Delta^*(H) = [n \text{ é ímpar}].$$

*Sendo assim, o raio de empacotamento de  $H$  é*

$$R(H) = \frac{n}{2} + \frac{[n \text{ é ímpar}]}{2} - 1.$$

**Exemplo 30.** *Seja  $P$  um poset cadeia com  $n$  elementos. O problema de partição associado a este é o de particionar a lista  $(n)$ . A menor discrepância, é então,  $\Delta^*(P) = n$ . Logo,*

$$\begin{aligned} R(P) &= \frac{n}{2} + \frac{n}{2} - 1 \\ &= n - 1. \end{aligned}$$

No capítulo anterior vimos o papel importante da discrepância no problema da partição. Quando a discrepância é mínima chamamos a partição de perfeita e a existência destas é o que controla a dificuldade de se resolver o problema da partição. Nos posets estudados nesta seção quando a discrepância é mínima o raio de empacotamento assume o menor valor permitido pela desigualdade da **Proposição 8**, ou seja, o raio de empacotamento do poset de Hamming. Com isso em mente, podemos interpretar o que os resultados mencionados no capítulo anterior podem nos dizer sobre o raio de empacotamento dos posets desta seção.

Por exemplo, o **Teorema 3** nos diz que, se pegarmos aleatoriamente um poset que é uma união de cadeias em que a quantidade de elementos maximais é maior do que a quantidade de bits necessária para expressar os comprimentos das cadeias, então, com probabilidade alta, o raio de empacotamento do poset é o mesmo que o do poset de Hamming.

## 5.2 O caso geral

O caso geral do problema da partição para posets também tem uma interpretação como um problema de distribuição de tarefas. Dados duas máquinas idênticas e

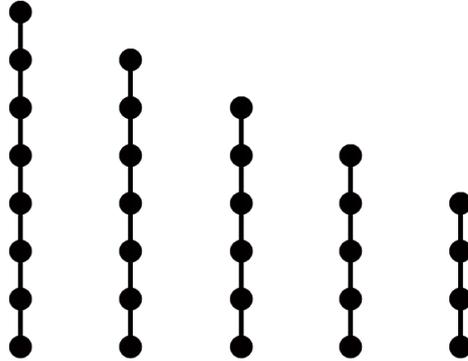


Figura 5.1: Encontrar o raio de empacotamento deste poset, que denotaremos por  $P$ , é equivalente a resolver o problema da partição para  $(8, 7, 6, 5, 4)$ . No capítulo anterior vimos que  $\Delta^*(8, 7, 6, 5, 4) = 0$ . Logo, como  $n = 30$  e  $\Delta^*(P) = 0$ , o raio de empacotamento do poset é  $R(P) = 14$ .

uma lista de tarefas em que certas tarefas dependem de outras, qual é o menor tempo para que ambas as máquinas executem todas as tarefas. A dependência entre as tarefas geram um poset no qual uma tarefa  $T$  é maior do que uma tarefa  $S$ , se  $T$  depende de  $S$ .

A principal diferença entre o caso geral e o abordado na seção anterior é o fato de que o problema não é mais equivalente a minimizar a discrepância, como vemos na **Figura 5.2**.

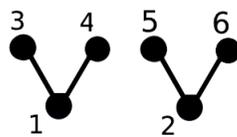


Figura 5.2: Neste poset, a partição  $\{3, 5\}\{4, 6\}$  tem discrepância  $\Delta(\{3, 5\}, \{4, 6\}) = 0$  no entanto ela não é ótima, pois o maior de ambos os pesos é 4. A partição ótima, neste caso, é  $\{3, 4\}\{5, 6\}$  com o maior dos pesos igual a 3.

O exemplo da **Figura 5.2** mostra que o fato de encontrarmos uma partição com discrepância mínima não garante que a partição é ótima. Neste caso, ambas as partições têm discrepância mínima, o que nos poderia levar a imaginar que a discrepância da partição ótima é menor ou igual do que a discrepância de qualquer outra partição. Isto também não é verdade. No exemplo da **Figura 5.3** mostramos um contra-exemplo.

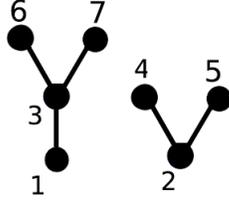


Figura 5.3: Neste poset, a partição  $\{6, 4\}\{7, 5\}$  tem discrepância  $\Delta(\{6, 4\}, \{7, 5\}) = 0$  e peso máximo igual a 5. A partição ótima é  $\{6, 7\}\{4, 5\}$  com peso máximo 4, no entanto esta partição tem discrepância 1.

O que devemos levar em conta no caso geral, é a possibilidade de haver interseção dos ideais gerados pelas partições. Definimos então a discordância de uma partição.

**Definição 33.** Seja  $P$  um poset e  $(A, B)$  uma partição do conjunto  $M_P$  de elementos maximais de  $P$ . Definimos a **discordância** entre  $A$  e  $B$  como sendo

$$\Lambda(A, B) = \Delta(A, B) + |\langle A \rangle \cap \langle B \rangle|$$

e a **discordância mínima** de  $P$  como sendo

$$\Lambda^*(P) = \min_{X \sqcup Y = M_P} \Lambda(X, Y).$$

No caso geral, a discordância faz o papel que a discrepância faz no caso analisado na seção anterior, observando que no caso tratado na seção anterior a discordância coincide com a discrepância.

**Teorema 7.** Seja  $P$  um poset de cardinalidade  $n$ . Então, o raio de empacotamento de  $P$  é

$$R(P) = \frac{n}{2} + \frac{\Lambda^*(P)}{2} - 1.$$

*Demonstração.* A demonstração é análoga ao do **Teorema 6**. A única diferença é que no caso geral, vale

$$n + |\langle A \rangle \cap \langle B \rangle| = \max\{\omega_P(A), \omega_P(B)\} + \min\{\omega_P(A), \omega_P(B)\}.$$

□

No caso de um poset hierárquico conseguiremos encontrar uma fórmula fechada para o seu raio de empacotamento de modo similar ao caso de Hamming.

**Proposição 13.** Seja  $P$  um poset hierárquico com  $n$  elementos. Seja  $M_P = \{x_1, x_2, \dots, x_m\}$  o conjunto dos elementos maximais de  $P$ . Então,

$$R(P) = n + \frac{[m \text{ é ímpar}]}{2} - \frac{m}{2} - 1.$$

*Demonstração.* Separamos em dois casos.

Se  $m = 1$ :

Então existe apenas uma partição para  $M_P$ , no caso  $(\{x_1\}, \emptyset)$ . Mas

$$\Lambda(\{x_1\}, \emptyset) = n,$$

e portanto,

$$\Lambda^*(P) = n.$$

Caso  $m > 1$ :

Neste caso, a partição trivial  $(M_P, \emptyset)$  certamente não é ótima. Sendo assim, seja  $(A, B)$  uma partição não trivial. Daí,

$$\langle A \rangle \cap \langle B \rangle = P - M_P$$

pois os elementos maximais não se interceptam e, por definição, os elementos de  $P - M_P$  são menores do que todos os elementos de  $M_P$ . Logo, a discordância de qualquer partição não trivial é

$$\Lambda(A, B) = \Delta(A, B) + |P - M_P|.$$

Neste caso então, minimizar a discordância é equivalente a minimizar a discrepância. Como vimos, se  $(A, B)$  é não trivial,  $P - M_P$  está contido no ideal de ambos os conjuntos, e portanto, o problema é equivalente a particionar o poset  $M_P$ , um poset de Hamming. Já vimos que a discrepância mínima no caso de Hamming é

$$\Delta^*(M_P) = [m \text{ é ímpar}].$$

Logo,

$$\Lambda^*(P) = [m \text{ é ímpar}] + |P - M_P|.$$

□

### 5.3 O método da diferenciação para posets

As heurísticas mais poderosas para o problema da partição clássica se utilizam do operador de diferenciação apresentado em [8] e já discutido no capítulo anterior. Esta operação consiste em tomar dois elementos  $x_i$  e  $x_j$  da lista a ser particionada e substituí-los por  $|x_i - x_j|$ . Fazer isto é equivalente a decidir que os elementos serão colocados em listas diferentes. A outra possibilidade, é decidir que os elementos irão para a mesma lista. Chamaremos o operador que faz isto de operador de associação. Neste caso, substituíam-se os elementos por  $x_i + x_j$ , e com estas duas operações

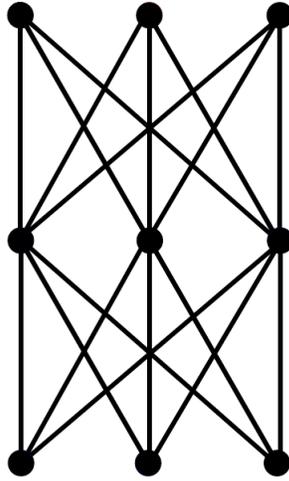


Figura 5.4: Usando a notação da **Proposição 13** o poset  $P$  acima tem parâmetros  $n = 9$  e  $m = 3$ . Logo,  $\Lambda^*(P) = 7$ , e portanto,  $R(P) = \frac{9+7}{2} - 1 = 7$ .

podemos fazer uma árvore em que as folhas representam todas as partições possíveis e têm o valor da discrepância. Nesta seção, iremos generalizar este método para poder aplicá-lo ao problema mais geral de particionar um poset.

Já vimos na seção anterior que o problema da partição clássica pode ser vista como um problema da partição de posets que são a união disjunta de cadeias. Neste caso, como não há interseção dos ideais gerados pelos elementos maximais, apenas os pesos dos elementos maximais são necessários para determinar a discordância. No caso geral, temos que levar em conta a relação dos elementos maximais com o resto do poset. Para poder generalizar o método da diferenciação iremos representar cada elemento maximal pelo seu vetor coluna da matriz de adjacência do poset, i.e. o vetor binário que indica quais elementos do poset são menores do que este.

**Definição 34.** *Seja  $P = ([n], \preceq)$  um poset. Dado  $x \in P$ , denotamos o seu **vetor de adjacência** por  $\hat{x}$  onde as coordenadas deste são definidas por*

$$\hat{x}_i = [i \preceq x].$$

**Exemplo 31.** *Seja  $P = ([n], \preceq)$  um poset. Então, a matriz de adjacência de  $P$  é*

$$\begin{pmatrix} \vdots & \vdots & \vdots & \vdots \\ \hat{1} & \hat{2} & \cdots & \hat{n} \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

Dado então, o conjunto dos elementos maximais  $M_P = \{x_1, x_2, \dots, x_m\}$  a serem particionados, teremos a este associado uma lista de vetores de adjacência

$(\hat{x}_1, \hat{x}_2, \dots, \hat{x}_m)$ . Os operadores de diferenciação ( $\ominus$ ) e associação ( $\oplus$ ) serão definidos de modo que possamos tratar o problema de particionar a lista  $(\hat{x}_1, \hat{x}_2, \dots, \hat{x}_m)$  analogamente ao que é feito no caso do problema da partição, i.e. substituir dois elementos  $\hat{x}_i$  e  $\hat{x}_j$  da lista por  $\hat{x}_i \ominus \hat{x}_j$ , deverá ser equivalente a decidir que  $x_i$  e  $x_j$  vão para conjuntos diferentes e substituí-los por  $\hat{x}_i \oplus \hat{x}_j$ , deverá ser equivalente a decidir que vão para o mesmo conjunto.

Os vetores  $\hat{x}_1, \hat{x}_2, \dots, \hat{x}_m$  advindos do conjunto de elementos maximais  $M_P = \{x_1, x_2, \dots, x_m\}$  que queremos particionar são binários. Os vetores que aparecerão ao diferenciarmos e associarmos, no entanto, não serão necessariamente binários. As suas coordenadas poderão assumir os seguintes valores:  $0, 1, -1$  e  $i$ . O  $i$  pode ser interpretado formalmente como a unidade imaginária, no sentido de que podemos somar os valores, e teremos uma parte real e outra imaginária.

Além disso, faremos um abuso de notação. Os operadores vão agir coordenada a coordenada, e usaremos o mesmo símbolo e nome para o operador que age na coordenada e o que age no vetor (analogamente ao que é feito na soma e diferença de vetores).

Seguem as definições dos operadores.

**Definição 35.** *Seja  $X = \{0, 1, -1, i\}$ . Os operadores de diferenciação e associação são definidos respectivamente pelas tabelas abaixo:*

$\ominus$	0	1	-1	i
0	0	-1	1	i
1	1	i	1	i
-1	-1	-1	i	i
i	i	i	i	i

$\oplus$	0	1	-1	i
0	0	1	-1	i
1	1	1	i	i
-1	-1	-1	-1	i
i	i	i	i	i

O valor de  $x \ominus y$  se encontra na linha do  $x$  e coluna do  $y$ , por exemplo,

$$1 \ominus -1 = 1.$$

Análogo para o caso da associação.

No caso de dois vetores  $\hat{x}, \hat{y} \in X^n$ , para algum  $n$ , os operadores são definidos coordenada por coordenada da seguinte forma:

$$(\hat{x} \oplus \hat{y})_i = \hat{x}_i \oplus \hat{y}_i$$

$$(\hat{x} \ominus \hat{y})_i = \hat{x}_i \ominus \hat{y}_i.$$

As operações  $\oplus$  e  $\ominus$  se comportam de forma semelhante a soma e a diferença. Podemos também definir  $\oplus x$  como sendo  $0 \oplus x$  e  $\ominus x$  como sendo  $0 \ominus x$ . Listamos algumas propriedades:

**Proposição 14.** *Sejam  $x, y, z \in \{0, 1, -1, i\}$ . Então valem as seguintes propriedades:*

- $x \oplus y = y \oplus x$

$$2. (x \oplus y) \oplus z = x \oplus (y \oplus z)$$

$$3. 0 \oplus x = x \oplus 0 = x$$

$$4. \oplus(x \oplus y) = \oplus x \oplus y$$

$$5. \oplus(x \ominus y) = \oplus x \ominus y$$

$$6. \ominus(x \oplus y) = \ominus x \oplus y$$

$$7. \ominus(x \ominus y) = \ominus x \oplus y$$

Como as operações agem coordenada a coordenada nos vetores  $\hat{x}, \hat{y}, \hat{z} \in \{0, 1, -1, i\}^n$ , as propriedades acima também valem no caso vetorial.

*Demonstração.* Direto da definição. □

Analogamente ao caso do problema da partição clássica, cada expressão envolvendo os vetores a serem particionados e as operações  $\oplus$  e  $\ominus$  estão associados a uma partição.

Para fazer esta associação e utiliza-la de modo a obter resultados precisamos de algumas definições.

**Definição 36.** *Seja  $P$  um poset e  $M_P = \{x_1, x_2, \dots, x_m\}$  o conjunto dos elementos maximais de  $P$ . A lista de vetores associada a  $M_P$  é  $(\hat{x}_1, \hat{x}_2, \dots, \hat{x}_m)$ .*

Primeiro iremos associar uma partição a um tipo especial de expressão que chamaremos de simples.

**Definição 37.** *Seja  $(\hat{x}_1, \hat{x}_2, \dots, \hat{x}_m)$  a lista de vetores associada aos elementos maximais de um poset. Uma expressão simples envolvendo os elementos desta lista e as operações  $\oplus$  e  $\ominus$  é uma sequência da forma*

$$*_1 \hat{x}_{k_1} *_2 \hat{x}_{k_2} \dots *_n \hat{x}_{k_n},$$

onde  $k_i \in [m]$  e  $*_i = \oplus$  ou  $*_i = \ominus$ .

A partição associada a uma expressão simples é a partição  $(A, B)$  definido da seguinte forma:

$$A = \{x_{k_i} : *_i = \oplus\}$$

$$B = \{x_{k_i} : *_i = \ominus\}.$$

O conjunto  $A$  é chamado de **conjunto primário**, e o  $B$  de **conjunto secundário**.

As definições acima são estendidas para expressões quaisquer, notando que, utilizando as propriedades da **Proposição 14**, podemos transformar qualquer expressão utilizando os elementos da lista de vetores associada e as operações  $\oplus$  e  $\ominus$  em uma expressão simples.

**Definição 38.** A *partição associada a uma expressão* qualquer é a partição associada a expressão simples, obtida transformando a expressão original, utilizando as propriedades da **Proposição 14**.

Por abuso de notação denotamos uma expressão como se fosse o vetor resultante de se calculá-la. Sendo assim, quando nos referirmos a  $k$ -ésima coordenada de uma expressão, estamos nos referindo a  $k$ -ésima coordenada do vetor resultante do cálculo da expressão. Denotamos o conjunto primário de uma expressão  $\hat{v}$  por  $\text{Primario}(\hat{v})$  e o conjunto secundário por  $\text{Secundario}(\hat{v})$ .

O próximo exemplo ilustra as definições acima.

**Exemplo 32.** Seja  $P$  um poset e  $M_P = \{x_1, x_2, x_3, x_4\}$  o conjunto dos elementos maximais de  $P$ . A lista de vetores associada a  $M_P$  é  $(\hat{x}_1, \hat{x}_2, \hat{x}_3, \hat{x}_4)$ . A forma simples da expressão

$$(\hat{x}_1 \ominus \hat{x}_2) \ominus (\hat{x}_3 \ominus \hat{x}_4)$$

é

$$\oplus \hat{x}_1 \ominus \hat{x}_2 \ominus \hat{x}_3 \oplus \hat{x}_4$$

cuja partição associada é  $(\{x_1, x_4\}, \{x_2, x_3\})$  onde  $\{x_1, x_4\}$  é o conjunto primário e  $\{x_2, x_3\}$  é o conjunto secundário.

Valem as seguintes propriedades:

**Proposição 15.** Seja  $P$  um poset,  $M_P = \{x_1, x_2, \dots, x_m\}$  o conjunto dos elementos maximais de  $P$  e  $(\hat{x}_1, \hat{x}_2, \dots, \hat{x}_m)$  a lista de vetores associada a  $M_P$ . Se  $\hat{v}$  e  $\hat{w}$  são duas expressões utilizando elementos da lista então valem as seguintes propriedades:

1.  $\text{Primario}(\hat{v} \oplus \hat{w}) = \text{Primario}(\hat{v}) \cup \text{Primario}(\hat{w})$
2.  $\text{Secundario}(\hat{v} \oplus \hat{w}) = \text{Secundario}(\hat{v}) \cup \text{Secundario}(\hat{w})$
3.  $\text{Primario}(\hat{v} \ominus \hat{w}) = \text{Primario}(\hat{v}) \cup \text{Secundario}(\hat{w})$
4.  $\text{Secundario}(\hat{v} \ominus \hat{w}) = \text{Secundario}(\hat{v}) \cup \text{Primario}(\hat{w})$

*Demonstração.* Basta escrever  $\hat{v}$  e  $\hat{w}$  na forma simples e notar que  $\oplus$  não altera em nada os sinais na frente dos vetores e  $\ominus$  troca todos os sinais de  $\hat{w}$ .  $\square$

Com estas propriedades estamos prontos para mostrar o seguinte resultado:

**Lema 5.** Seja  $P$  um poset de cardinalidade  $n$ ,  $M_P = \{x_1, x_2, \dots, x_m\}$  o conjunto dos elementos maximais de  $P$ ,  $(\hat{x}_1, \hat{x}_2, \dots, \hat{x}_m)$  a lista de vetores associada a  $M_P$ , e  $\hat{v}$  e  $\hat{w}$  duas expressões usando os vetores da lista associada. Suponha que para todo  $k \in [n]$ :

1.  $\hat{v}_k = 0 \Leftrightarrow k \notin \langle \text{Primario}(\hat{v}) \rangle \cup \langle \text{Secundario}(\hat{v}) \rangle$
2.  $\hat{v}_k = 1 \Leftrightarrow k \in \langle \text{Primario}(\hat{v}) \rangle - \langle \text{Secundario}(\hat{v}) \rangle$
3.  $\hat{v}_k = -1 \Leftrightarrow k \in \langle \text{Secundario}(\hat{v}) \rangle - \langle \text{Primario}(\hat{v}) \rangle$
4.  $\hat{v}_k = i \Leftrightarrow k \in \langle \text{Primario}(\hat{v}) \rangle \cup \langle \text{Secundario}(\hat{v}) \rangle$

e que as propriedades listadas continuam valendo quando substituimos  $\hat{v}$  por  $\hat{w}$ . Então, as propriedades listadas continuam valendo quando substituimos  $\hat{v}$  por  $\hat{v} \oplus \hat{w}$  ou  $\hat{v} \ominus \hat{w}$ .

*Demonstração.* A demonstração consiste em separar em todos os casos possíveis e utilizar as propriedades da **Proposição 15**. Como temos 4 valores possíveis para  $\hat{v}_k$  e para  $\hat{w}_k$  e temos duas operações, a quantidade total de casos é 32. Vamos demonstrar apenas para um caso, como exemplo. A demonstração dos outros casos é totalmente análoga.

**Caso da associação com  $\hat{v}_k = 1$  e  $\hat{w}_k = -1$ :**

Neste caso,

$$\hat{v}_k \oplus \hat{w}_k = i.$$

As duas hipóteses nos dizem que

$$k \in \langle \text{Primario}(\hat{v}) \rangle - \langle \text{Secundario}(\hat{v}) \rangle$$

e

$$k \in \langle \text{Secundario}(\hat{w}) \rangle - \langle \text{Primario}(\hat{w}) \rangle.$$

Mas então,

$$k \in \langle \text{Primario}(\hat{v}) \rangle \subseteq \langle \text{Primario}(\hat{v}_k \oplus \hat{w}_k) \rangle$$

e

$$k \in \langle \text{Secundario}(\hat{w}) \rangle \subseteq \langle \text{Secundario}(\hat{v}_k \oplus \hat{w}_k) \rangle,$$

e portanto,

$$k \in \langle \text{Primario}(\hat{v}_k \oplus \hat{w}_k) \rangle \cup \langle \text{Secundario}(\hat{v}_k \oplus \hat{w}_k) \rangle.$$

□

Note que, os vetores  $(\hat{x}_1, \hat{x}_2, \dots, \hat{x}_m)$  da lista de vetores associada aos elementos maximais de um poset satisfazem as hipóteses do Lema. Sendo assim, ao utilizarmos o método da diferenciação, os vetores que aparecem nas folhas terão as informações necessárias para calcular a discordância da partição associada. Para isto precisaremos definir a seguinte função.

**Definição 39.** Seja  $\hat{v} \in \{0, 1, -1, i\}^n$ . Definimos a função soma das entradas do vetor como

$$S(\hat{v}) = \sum_{k=1}^n \hat{v}_k$$

onde  $i$  é tratado formalmente como se fosse a unidade imaginária.

**Exemplo 33.** Se  $\hat{v} = (1, 1, -1, i, 1, i)$  então

$$S(\hat{v}) = 2 + 2i.$$

Faz sentido então falar sobre a parte real de  $S(\hat{v})$ , denotado por  $\Re(S(\hat{v}))$ , e da parte imaginária, denotada por  $\Im(S(\hat{v}))$ . Estamos prontos para encontrar explicitamente a discordância das folhas da árvore do método da diferenciação.

**Teorema 8.** Seja  $P$  um poset,  $M_P = \{x_1, x_2, \dots, x_m\}$  o conjunto dos elementos maximais de  $P$ ,  $(\hat{x}_1, \hat{x}_2, \dots, \hat{x}_m)$  a lista de vetores associada a  $M_P$  e  $\hat{v}$  uma expressão utilizando todos os vetores da lista associada uma única vez. Denotemos por  $(A, B)$  a partição de  $M_P$  associada a  $\hat{v}$ . Então

$$\Delta(A, B) = |\Re(S(\hat{v}))|$$

e

$$|\langle A \rangle \cap \langle B \rangle| = \Im(S(\hat{v}))$$

de modo que

$$\Lambda(A, B) = |\Re(S(\hat{v}))| + \Im(S(\hat{v})).$$

*Demonstração.* Sem perda de generalidade suponha que  $A = \text{Primario}(\hat{v})$  e  $B = \text{Secundario}(\hat{v})$ . Então, como a lista de vetores associada a  $M_P$  satisfaz as hipóteses do **Lema 5**,

$$|\langle A \rangle| = \sum_{k=1}^n [v_k = 1] + \sum_{k=1}^n [v_k = i]$$

e

$$|\langle B \rangle| = \sum_{k=1}^n [v_k = -1] + \sum_{k=1}^n [v_k = i].$$

Logo,

$$\begin{aligned} \Delta(A, B) &= \left| \sum_{k=1}^n [v_k = 1] - \sum_{k=1}^n [v_k = -1] \right| \\ &= \left| \sum_{k=1}^n v_k [v_k \neq i] \right| \\ &= |\Re(S(\hat{v}))|. \end{aligned}$$

Pelo **Lema 5** também temos que

$$\begin{aligned} |\langle A \rangle \cap \langle B \rangle| &= \sum_{k=1}^n [v_k = i] \\ &= \mathfrak{S}(S(\hat{v})). \end{aligned}$$

□

As operações que definimos, então, servem para generalizar o método da diferenciação para o caso de posets. Iremos no entanto fazer algumas modificações de notação no método. Em vez de considerar a lista de vetores  $(\hat{x}_1, \hat{x}_2, \dots, \hat{x}_m)$  associada aos elementos maximais, vamos considerar a matriz cujas colunas são os vetores da lista.

**Definição 40.** *Seja  $P$  um poset de cardinalidade  $n$ ,  $M_P = \{x_1, x_2, \dots, x_m\}$  o conjunto dos elementos maximais de  $P$  e  $(\hat{x}_1, \hat{x}_2, \dots, \hat{x}_m)$  a lista de vetores associada a  $M_P$ . Definimos a **matriz-raio** de  $P$  como sendo a matriz*

$$\begin{pmatrix} \vdots & \vdots & \vdots & \vdots \\ \hat{x}_1 & \hat{x}_2 & \cdots & \hat{x}_m \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}.$$

Note que, esta matriz pode ser obtida a partir da matriz de adjacência de  $P$  eliminando-se as colunas que não são referentes a elementos maximais.

**Teorema 9.** *O raio de empacotamento de um poset é determinado pela sua matriz-raio, i.e. precisamos saber apenas a matriz-raio de um poset para determinar o seu raio de empacotamento.*

*Demonstração.* Direto do **Teorema 8** e da definição de matriz-raio. □

Deste modo, podemos enxergar o raio de empacotamento como a propriedade de uma matriz.

**Definição 41.** *Chamamos de **raio de empacotamento de uma matriz  $M$**  o raio de empacotamento de um poset  $P$  com matriz-raio  $M$ . Estendemos a definição também para o caso das matrizes que aparecem nos nodos da árvore ao aplicarmos o método da diferenciação.*

Um fato óbvio sobre a matriz-raio, devido a como ela foi construída, é que se trocarmos duas colunas ou duas linhas o raio de empacotamento não muda. Podemos trocar duas colunas pelo fato de poder trocar dois vetores da lista associada aos elementos maximais. As linhas podem ser trocadas, pois as operações de associação e diferenciação agem coordenada por coordenada.

Estamos prontos para construir um algoritmo análogo ao CKK ([12]) para o problema da partição de posets. Construimos uma árvore começando pela raiz que será a matriz-raio. Depois disso, escolhemos duas colunas da matriz. Da raiz vão sair dois ramos, um para a esquerda, cujo nodo vai ser a matriz em que se substitui as colunas escolhidas pela sua diferenciação, e um para a direita, cujo nodo vai ser a matriz em que se substitui as colunas escolhidas pela sua associação. Fazemos isso para cada nodo até chegarmos a uma folha, um nodo em que a matriz tem uma única coluna. Daí, para calcular a discordância associada a uma folha basta usar o **Teorema 8**.

No caso do problema da partição os melhores resultados ocorrem quando se escolhe os dois maiores valores para diferenciar ([12]). No caso de posets não temos ainda resultados com relação a que critérios usar para escolher os valores a diferenciar. O critério que iremos utilizar nos exemplos, no entanto, generaliza a ideia de escolher os dois maiores valores e consiste em escolher o vetor  $\hat{v}$  tal que  $|\Re(S(\hat{v}))| + \Im(S(\hat{v}))$  seja máximo e o vetor  $\hat{w}$  que minimize  $|\Re(S(\hat{v} \ominus \hat{w}))| + \Im(S(\hat{v} \ominus \hat{w}))$ . Quando utilizarmos este critério nos exemplos os vetores  $\hat{v}$  e  $\hat{w}$  sempre serão as primeiras duas colunas da matriz.

**Exemplo 34.** *Seja  $P$  o poset com matriz de adjacência*

$$\begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

*O conjunto de elementos maximais de  $P$  é  $M_P = \{4, 5, 6, 7\}$ . Logo, a matriz-raio de  $P$  é*

$$\begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

**A Figura 5.5** mostra a árvore decorrente do método da diferenciação aplicada na matriz-raio de  $P$ . Dela concluímos que  $\Lambda^*(P) = 3$ , e portanto, o raio de empacotamento do poset é  $R(P) = 4$ .

Podemos deixar a notação um pouco mais sucinta ainda. Note que, se na matriz de um nodo da árvore uma das entradas assume o valor  $i$ , então todos os vetores folhas decorrentes desse nodo vão ter a coordenada correspondente a linha em que

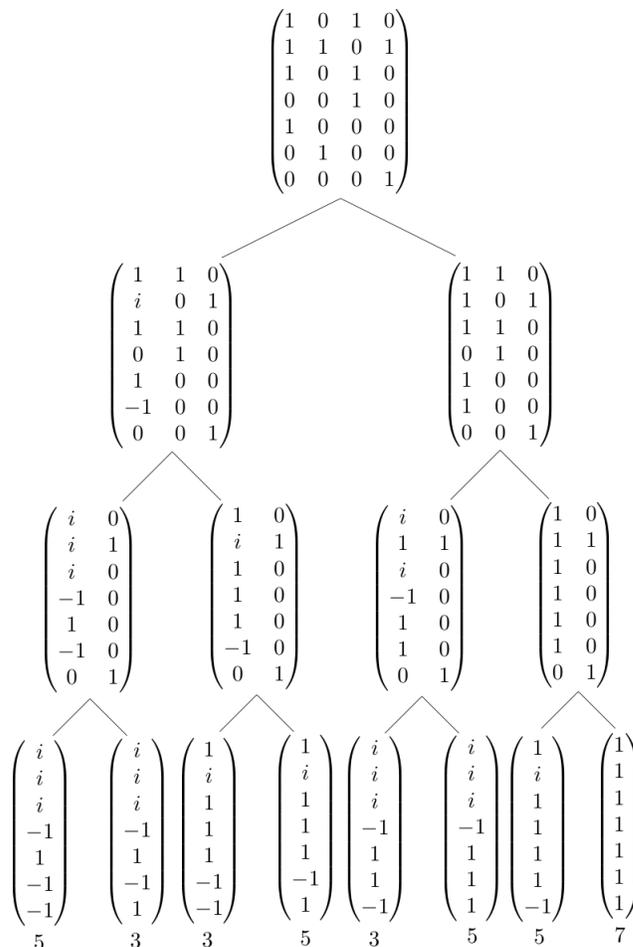


Figura 5.5: Árvore referente ao **Exemplo 34**.

aparece o  $i$  igual a  $i$ . Isto ocorre pois as operações  $\oplus$  e  $\ominus$  sempre preservam o  $i$ . Sendo assim, podemos apagar as linhas em que aparecem  $i$ 's e colocar um número que aparece na frente da matriz que conta quantas linhas foram apagadas. Fazendo isto, quando chegamos na folha a discordância é calculada somando-se o número fora do vetor com a soma das entradas do vetor.

Mais precisamente, substituímos uma matriz  $L$  por uma dupla  $(\alpha, M)$  chamada de **número-matriz**, onde  $\alpha$  é um inteiro e  $M$  é uma matriz, da seguinte forma:  $\alpha$  é a quantidade de linhas da matriz  $L$  em que aparecem ao menos um  $i$  e  $M$  é obtida a partir de  $L$  retirando-se as linhas em que aparecem ao menos um  $i$ . Daí, ao invés de aplicar o método da diferenciação em  $L$ , o aplicamos em  $M$  mas retirando as linhas em que aparece ao menos um  $i$ . Chamemos de  $N$  a matriz obtida por este processo, e suponha que tenhamos retirado  $\beta$  linhas da matriz, então, neste nodo

colocaremos o número-matriz  $(\alpha + \beta, N)$ . Fazemos isto até chegar a uma das folhas que será uma dupla  $(\gamma, \hat{v})$ , e como  $\gamma$  é a quantidade de  $i$ 's que apareceriam no final, pelo **Teorema 8**, a discordância da folha será  $\gamma + S(\hat{v})$ .

Denotaremos o número-matriz  $(\alpha, M)$  por  $\alpha M$  omitindo o  $\alpha$  se ele for 0. Esta dupla não deve ser confundida com um número multiplicando uma matriz.

**Exemplo 35.** Na árvore da **Figura 5.5** a primeira matriz do lado esquerdo é

$$\begin{pmatrix} 1 & 1 & 0 \\ i & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Usando a notação de número-matriz, substituimos a matriz por

$$1 \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

No caso do problema da partição vimos várias formas de podar os ramos da árvore. Mostraremos a seguir formas de podar ramos no caso de posets.

**Teorema 10.** *Seja*

$$\alpha M = \alpha \begin{pmatrix} \vdots & \vdots & \vdots & \vdots \\ \hat{w}_1 & \hat{w}_2 & \cdots & \hat{w}_k \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

um número-matriz que aparece na árvore do método da diferenciação aplicada a um poset e seja  $\beta \hat{v}$  uma folha decorrente do nodo em que aparece o número-matriz tal que  $\Lambda(\beta \hat{v})$  seja mínimo. Então,

$$\Lambda(\beta \hat{v}) \geq \alpha + \Delta^*(S(\hat{w}_1), S(\hat{w}_2), \dots, S(\hat{w}_k))$$

*Demonstração.* Note que,

$$\Lambda^*(\alpha M) = \alpha + \Lambda^*(M).$$

Mas então,  $\Lambda^*(M) \geq \Delta^*(S(\hat{w}_1), S(\hat{w}_2), \dots, S(\hat{w}_k))$  pois a aparição de  $i$ s apenas acrescenta o valor de  $\Lambda^*(M)$ .  $\square$

Com este resultado, podemos podar a árvore da seguinte forma. Após encontrar uma folha  $\beta\hat{v}$  podemos podar os ramos de todos os número-matrizes

$$\alpha M = \alpha \begin{pmatrix} \vdots & \vdots & \vdots & \vdots \\ \hat{w}_1 & \hat{w}_2 & \cdots & \hat{w}_k \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

tais que

$$\alpha + \Delta^*(S(\hat{w}_1), S(\hat{w}_2), \dots, S(\hat{w}_k)) - 1 \geq \beta + S(\hat{v}).$$

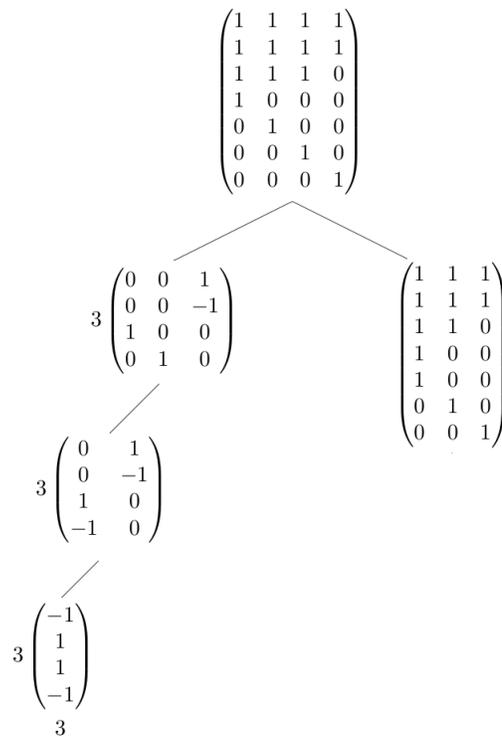


Figura 5.6: Árvore referente ao **Exemplo 36**

**Exemplo 36.** Seja  $P$  o poset com matriz de adjacência

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

O conjunto de elementos maximais de  $P$  é  $M_P = \{4, 5, 6, 7\}$ . Logo, a matriz-raio de  $P$  é

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

A **Figura 5.6** mostra a árvore decorrente do método da diferenciação aplicada na matriz-raio de  $P$ . Assim como no caso do problema da partição de mos preferênciã para operação de diferenciação. A primeira folha que achamos nos deu um valor de  $\Lambda = 3$ . Isso já podou todos os ramos que sobraram do lado esquerdo da árvore pois o número associado a matriz é 3. Indo para a direita nos deparamos com a matriz

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Mas a soma das entradas de cada vetor nos dá a lista  $(5, 4, 3)$ . Como  $\Delta^*(5, 4, 3) = 2$  e  $\Lambda^*(P)$  é ímpar, o menor valor possível para a discordância de uma folha decorrente dessa matriz é 3. Logo,  $\Lambda^*(P) = 3$  e o raio de empacotamento do poset é

$$R(P) = 4.$$

Com isto, concluímos o nosso estudo sobre a determinação do raio de empacotamento de um poset. Na próxima seção voltaremos ao problema de encontrar o raio de empacotamento de um  $P$ -Código Linear.

Deixamos para futuras investigações os seguintes problemas:

- Fazer um estudo sobre as propriedades estatísticas da diferenciação, por exemplo, provar resultados do tipo do **Teorema 3** para o caso de posets.
- Fazer experimentos computacionais análogos aos encontrados em [12], testando critérios diferentes para a seleção dos vetores a serem diferenciados. É a opinião do autor que o critério proposto neste trabalho é uma generalização boa para o critério de escolher os dois maiores números no problema da partição.
- Encontrar outras aplicações para o método da diferenciação.

## 5.4 O raio de empacotamento de códigos poset

No começo do capítulo 3 mostramos que o raio de empacotamento de um código linear é o mínimo dos raios de empacotamento de suas palavras-código. Após isto, nos concentramos puramente no problema de encontrar o raio de empacotamento de um vetor. Nesta seção, voltamos a abordar o problema original e vamos buscar formas de determinar o vetor de empacotamento de um código.

Primeiro vamos re-escrever a **Proposição 10**, no contexto de códigos poset.

**Teorema 11.** *Seja  $P$  um poset e  $C \subseteq \mathbb{F}_q^n$ . Denotemos o ideal gerado pelo suporte de uma palavra-código  $x \in C$  por  $I_x$ . Então, o raio de empacotamento de  $C$  é*

$$R_P(C) = \min_{x \in C} R(I_x).$$

*Demonstração.* Basta re-escrever a **Proposição 10** utilizando a notação apresentada no contexto de códigos poset.  $\square$

Já vimos que encontrar do raio de empacotamento é NP-difícil. Seria interessante então não precisar calcular o raio de empacotamento de todas as palavras-código. O que queremos é um método tal que dado dois posets  $P$  e  $Q$  possamos saber qual tem o menor raio de empacotamento, sem ter que calculá-los explicitamente.

O critério mais simples é usar a desigualdade da **Proposição 8**.

**Proposição 16.** *Sejam  $P$  e  $Q$  dois posets com cardinalidades  $n$  e  $m$  respectivamente. Então,*

$$n \leq \frac{m}{2} + \frac{[m \text{ é ímpar}]}{2} \Rightarrow R(P) \leq R(Q).$$

*Demonstração.* Suponha que

$$n \leq \frac{m}{2} + \frac{[m \text{ é ímpar}]}{2}.$$

Então, pela desigualdade da **Proposição 8**

$$R(P) + 1 \leq n,$$

e portanto,

$$R(P) \leq \frac{m}{2} + \frac{[m \text{ é ímpar}]}{2} - 1.$$

Mas, pela desigualdade da **Proposição 8**, temos também que

$$\frac{m}{2} + \frac{[m \text{ é ímpar}]}{2} - 1 \leq R(Q)$$

donde segue o resultado.  $\square$

Um outro critério é o seguinte.

**Teorema 12.** *Sejam  $P$  e  $Q$  dois posets. Suponha que existe um poset  $P_2$  isomorfo a  $P$  tal que  $P_2$  é um subposet de  $Q$ . Então,*

$$R(P) \leq R(Q).$$

*Demonstração.* Seja  $(A, B)$  uma partição de  $Q$ . Mas daí,  $(A \cap P_2, B \cap P_2)$  é uma partição em  $P_2$  com a propriedade de que

$$\omega_{P_2}(A) \leq \omega_Q(A)$$

e

$$\omega_{P_2}(B) \leq \omega_Q(B),$$

e portanto,  $R(P_2) \leq R(Q)$ . Mas o raio de empacotamento é uma propriedade do poset, e portanto, é invariante por isomorfismo fazendo com que  $R(P_2) = R(P)$ .  $\square$

Com isto, podemos calcular explicitamente o raio de empacotamento de qualquer código quando o poset é hierárquico. Como foi dito no início da seção este resultado já é conhecido.

**Proposição 17.** *Seja  $P$  um poset hierárquico,  $C \subseteq \mathbb{F}_q^n$  um  $P$ -código linear e  $c \in C$  uma palavra-código de  $P$ -peso mínimo. Se  $M_{I_c} = \{x_1, x_2, \dots, x_m\}$  é conjunto dos elementos maximais do ideal gerado pelo suporte de  $c$ , então,*

$$R_P(C) = \omega_P(C) + \frac{[m \text{ é ímpar}]}{2} - \frac{m}{2} - 1.$$

*Demonstração.* Sejam  $I$  e  $J$  dois ideais contidos em  $P$ . Como  $P$  é hierárquico,  $I$  e  $J$  também serão. Suponha que  $|I| \leq |J|$ . Então existem duas possibilidades:  $I \subseteq J$  ou os elementos maximais de  $I$  estão no mesmo nível dos de  $J$ . No primeiro caso, já temos que  $R(I) \leq R(J)$ . No segundo caso,  $I$  certamente tem menos elementos maximais que  $J$ , então basta considerar um isomorfismo que leve os elementos maximais de  $I$  num subconjunto dos elementos maximais de  $J$  e que não altera os outros elementos de  $I$ , e daí usando o **Teorema 12**,  $R(I) \leq R(J)$ .

O vetor-raio então, é tal que o ideal gerado pelo seu suporte seja mínimo, ou seja, a palavra-código de  $P$ -peso mínimo. Usando a **Proposição 13** chegamos na expressão acima para o raio de empacotamento de  $C$ .  $\square$

**Exemplo 37.** *Como o caso do poset  $P$  ser uma anti-cadeia ou uma cadeia é um caso particular dele ser hierárquico, a **Proposição 17** nos possibilita calcular o raio de empacotamento de um código explicitamente nestes casos também.*

Na seção anterior vimos que a matriz-raio do poset define seu raio de empacotamento. É de se esperar, então, que haja alguma forma de comparar duas matrizes raio para determinar qual é maior. O próximo resultado mostra que todo poset vai poder ser transformado num poset mais simples preservando, no entanto, o seu raio de empacotamento.

**Teorema 13.** *Seja  $P$  um poset. Então existe um poset  $Q$  que chamaremos de forma padrão de  $P$  tal que:*

- $R(Q) = R(P)$ .
- *Todo elemento de  $Q$  é maximal ou minimal.*

*Demonstração.* A matriz-raio  $M$  de  $P$  é uma submatriz da matriz  $A$  de adjacência de  $P$ , que determina unicamente o seu raio de empacotamento. Sendo assim, qualquer modificação nas entradas de  $A$  que não altere  $M$  e tal que  $A$  continue sendo uma matriz de adjacência de um poset irá preservar o raio de empacotamento. Definimos, então, a matriz de adjacência de  $Q$  como sendo uma matriz cujas entradas coincidem com as de  $A$  na diagonal e na submatriz  $M$ , e são zero no resto da matriz. Como  $Q$  tem a mesma matriz-raio de  $P$ ,  $R(Q) = R(P)$ . Além disso, como as únicas relações que existem em  $Q$  envolvendo elementos distintos se encontra na submatriz  $M$ , a matriz das colunas maximais de  $P$ , todo elemento que não for maximal, será minimal.  $\square$

**Exemplo 38.** *Seja  $P$  o poset com matriz de adjacência*

$$A_P = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

*Para encontrar os elementos maximais basta buscar as linhas em que aparece um único 1. Isto ocorre nas linhas 7, 8, 9 e 10. Logo, a matriz raio de  $P$  vai ser a matriz cujas colunas são as últimas 4 colunas de  $A_P$ . Zerando todos os elementos*

que não pertencem a estas colunas ou a diagonal obtemos a matriz

$$A_Q = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix},$$

que será a matriz de adjacência da forma padrão de  $P$ .

Juntando isto com o **Teorema 12** aumentam-se as possibilidades de comparações entre posets para determinar qual tem raio de empacotamento menor. O nosso proximo objetivo é mostrar como comparar posets cujas matrizes de adjacência e matrizes-raio tem tamanhos diferentes. Para isso iremos generalizar a ideia de matriz-raio.

**Definição 42.** *Seja  $P$  um poset. Chamamos uma matriz  $M$  uma matriz-raio-estendida ou matriz-RE de  $P$  se for possível encontrar o raio de empacotamento de  $P$  aplicando-se o método da diferenciação na matriz  $M$ .*

A matriz-raio de  $P$ , por exemplo, é uma matriz-RE de  $P$ .

A seguir listamos modificações que podem ser aplicadas a uma matriz-RE de um poset para construir outras matrizes-RE do mesmo poset.

**Proposição 18.** *Seja  $P$  um poset e  $M$  uma matriz-RE de  $P$ . Então as seguintes modificações em  $M$  não alteram o fato desta ser uma matriz-RE.*

1. Trocar duas linhas.
2. Trocar duas colunas.
3. Adicionar ou apagar uma linha nula.
4. Adicionar ou apagar uma coluna cujo suporte esteja contido no suporte de uma coluna de  $M$ .

*Demonstração.* As primeiras duas propriedades valem pelo mesmo motivo que valem no caso da matriz-raio de  $P$ . A terceira propriedade vale pois associar ou diferenciar zeros continua dando zero, e no final do método da diferenciação não altera em nada o valor das discordâncias das folhas. A quarta propriedade vale pelo seguinte motivo: Seja  $\hat{v}$  um vetor coluna de  $M$ . Se adicionarmos a  $M$  uma coluna  $\hat{w}$  tal que  $\text{supp}(\hat{w}) \subseteq \text{supp}(\hat{v})$  então  $\hat{v} \oplus \hat{w} = \hat{v}$  e além disso,  $\hat{v} \ominus \hat{w}$  é  $\hat{v}$  mas com entradas  $i$  nos lugares em que  $\hat{w} = 1$  o que faz com que certamente associar  $\hat{v}$  com  $\hat{w}$  seja melhor do que diferenciá-los.  $\square$

Note que, a matriz de adjacência de um poset é uma matriz-RE dele, pois todas as colunas correspondentes a elementos não maximais têm suporte contido em alguma coluna correspondente a um elemento maximal, e portanto, utilizando a propriedade 4 da **Proposição 18** podemos modificar a matriz de adjacência de modo a chegar na matriz-raio de  $P$ .

**Exemplo 39.** *Sejam  $P$  e  $Q$  dois posets com matrizes de adjacência*

$$A_P = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

e

$$A_Q = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

respectivamente.

Vamos mostrar que  $A_P$  é uma matriz raio equivalente de  $Q$ .

Começando pela matriz  $A_Q$ , utilizando a propriedade 4 da **Proposição 18** podemos eliminar as colunas 1, 2 e 3 pois o suporte delas está contido na coluna 5. Com isto, ficamos com a matriz-RE de  $Q$

$$\begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

Utilizando as propriedades 1 e 2 trocamos a ordem das duas colunas e trocamos as linhas 2 com a 3 e a 4 com a 5 ficando com a matriz-RE

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \\ 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$$

de  $Q$ . Mas daí, utilizando a propriedade 4 podemos chegar na matriz  $A_P$  pois o suporte das colunas 1, 2 e 3 está contido na primeira coluna da matriz acima.

Logo,  $R(P) = R(Q)$ .

No próximo resultado vemos como comparar o raio de empacotamento de dois posets utilizando as matrizes-RE destes.

**Teorema 14.** *Sejam  $P$  e  $Q$  dois posets e  $A_P$  e  $A_Q$  duas de suas matrizes-RE respectivamente. Então,*

$$\text{supp}(A_P) \subseteq \text{supp}(A_Q) \Rightarrow R(P) \leq R(Q).$$

*Demonstração.* Mostraremos que zerar uma entrada de uma matriz-RE sempre diminui seu raio de empacotamento. Seja  $M$  uma matriz-RE de algum poset. Definimos  $N$  como a matriz cujas entradas coincidem com as de  $M$  a não ser em uma, que denotaremos de  $m_{ij}$ , onde assume o valor zero, i.e.  $n_{ij} = 0$ .

Se a  $i$ -ésima linha de  $N$  contem apenas zeros, certamente  $R(N) \leq R(M)$ .

Se a  $i$ -ésima linha de  $N$  contem algum 1, a coluna  $k$  por exemplo, então associar ou diferenciar as colunas  $i$  e  $k$  de  $N$  nos dará discordâncias menores do que associar ou diferenciar as mesmas colunas da matriz  $M$ , e daí  $R(N) \leq R(M)$ .

Análogo para os outros casos. □

Logo, podemos tentar comparar o raio de empacotamento de dois posets da seguinte forma: modificar a matriz-RE de um dos posets ou de ambos de forma descrita na **Proposição 18** de modo a que o suporte de uma matriz esteja contida na outra.

**Exemplo 40.** *Sejam  $P$  e  $Q$  posets com matriz de adjacência respectivamente*

$$A_P = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

e

$$A_Q = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

*Os elementos maximais de  $Q$  podem ser encontrados buscando as linhas com apenas um 1, ou seja, as linhas 5, 6, 7 e 8. Logo, a matriz-raio de  $Q$ , que é também uma*

matriz-RE é

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

De modo análogo, a matriz-raio de  $P$  é

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Adicionando uma linha de zeros entre a quinta e sexta linha, e daí adicionando uma coluna de zeros entre a primeira e segunda linha, obtemos a seguinte matriz-RE de  $P$ :

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Mas o ideal desta matriz está contido no ideal da matriz-raio de  $Q$ . Logo,  $R(P) \leq R(Q)$ .

O **Teorema 14** nos dá, então, uma forma de reduzir bastante a quantidade de palavras-código para quais precisaríamos calcular o raio de empacotamento.

No nosso último resultado damos um limitante superior para o raio de empacotamento de um poset e conseqüentemente para um  $P$ -código linear.

**Proposição 19.** *Seja  $Q$  um poset com  $n$  elementos contendo  $m$  elementos máximos. Então,*

$$R(P) \leq n + \frac{[m \text{ é ímpar}]}{2} - \frac{m}{2} - 1.$$

Em particular, se  $P$  é um poset e  $C$  é um  $P$ -código linear e  $x \in C$  então

$$R_Q(C) \leq \omega_Q(x) + \frac{[|M_{\text{supp}(x)}| \text{ é ímpar}]}{2} - \frac{|M_{\text{supp}(x)}|}{2} - 1$$

onde  $M_{\text{supp}(x)}$  é conjunto dos elementos maximais de  $\text{supp}(x)$ .

*Demonstração.* Basta notar que a o suporte da matriz-raio de  $P$  esta contida na matriz-raio de um poset hierárquico com mesma quantidade de elementos maximais que  $P$ . Utilizando a **Proposição 13** o resultado esta demonstrado.  $\square$

Com isto, temos o seguinte método para encontrar o raio de empacotamento de um  $P$ -código linear  $C$ : Primeiro, usamos os métodos apresentados nesta seção para diminuir a quantidade de palavras-código candidatas a serem um vetor de empacotamento. Das que sobrarem, teremos que calcular o raio de empacotamento, utilizando as técnicas da seção anterior, no entanto a **Proposição 19** nos sugere começar pelas palavras com menor peso e maior quantidade de elementos maximais.

# Referências Bibliográficas

- [1] Christian Borgs, Jennifer Chayes and Boris Pittel, *Phase transition and finite-size scaling for the integer partitioning problem*, Random Structures and Algorithms 19, 247-288 (2001).
- [2] Richard A. Brualdi, Janine S. Graves and K. Mark Lawrence, *Codes with a poset metric*, Discrete Mathematics 147, Issues 1–3, 57–72 (1995).
- [3] Thomas M. Cover, Joy A. Thomas, *Elements of Information Theory*, New York: Wiley, 1991.
- [4] Luciano V. Felix and Marcelo Firer, *Canonical-Systematic Form of Hierarchical Codes*, to appear in Advances in Mathematics of Communication , 2011.
- [5] Marcelo Firer, Luciano Panek, Marcelo Muniz Silva Alves, *Classification of Niederreiter-Rosenbloom-Tsfasman block codes*, IEEE Transactions on Information Theory 56, Issue: 10 , 5207 - 5216 (2010).
- [6] Richard Hamming, *Coding and Information Theory*, Prentice Hall 1980, second edition, 1986.
- [7] Kenneth E. Iverson, *A Programming Language*, New York: Wiley, 1962.
- [8] Narendra Karmarkar and Richard M. Karp, *The differencing method of set partitioning*, Technical Report UCB/CSD 82/113, Computer Science Division (EECS), University of California, Berkley, 1982.
- [9] Richard M. Karp, *Reducibility Among Combinatorial Problems*, em Complexity of Computer Computations, 85–103 (1972).
- [10] Donald Knuth, *Two Notes on Notation*, American Mathematical Monthly 99, 403 - 422 (1992).
- [11] Richard E. Korf, *Improved Limited Discrepancy Search*, Proceedings of AAAI-96, 286-291 (1996).
- [12] Richard E. Korf, *A Complete anytime algorithm for number partitioning*, Artificial Intelligence 106, Issue 2, 181 - 203 (1998).

- [13] Florence J. MacWilliams and Neil J. A. Sloane, *The Theory of Error-Correcting Codes*, Elsevier/North-Holland, 1977.
- [14] Stephan Mertens, *Phase transition in the number partitioning problem*. Phys. Rev. Lett. 81, 4281–4284 (1998).
- [15] Stephan Mertens, *The Easiest Hard Problem: Number Partitioning*, Computational complexity and statistical physics, Oxford University Press US, 125-140 (2006).
- [16] Harald Neiderreiter, *A combinatorial problem for vector spaces over finite fields*, Discrete Mathematics 96, Issue 3, 221 - 228 (1991).
- [17] Luciano Panek, Marcelo Muniz e Marcelo Firer, *Raio de Empacotamento e o Limitante de Hamming sobre os Espaços de Rosenbloom-Tsfasman*, XXX CNMAC, Florianópolis (2007).
- [18] Luciano Panek, *Codificação na Presença do Valor Semântico da Informação*, Tese de Doutorado, UEM (2012).
- [19] M. Yu. Rosenbloom and Michael A. Tsfasman, *Codes for  $m$ -metric*, Problems of Information Transmission, 45-52 (1997).
- [20] Schroepfel, R., and A. Shamir, *A  $T = O(2^{n/2})$ ,  $S = O(2^{n/4})$  algorithm for certain NP-Complete Problems*, SIAM Journal of Computing, Vol. 10, No. 3, 456-464 (1981).
- [21] Claude E. Shannon, *A Mathematical Theory of Communication*, Bell System Technical Journal 27, 379–423 (1948).
- [22] Claude E. Shannon and Warren Weaver, *The Mathematical Theory of Communication*, Univ of Illinois Press, 1949.
- [23] Richard P. Stanley, *Enumerative Combinatorics, Volume 1*, Second Edition, Cambridge University Press, 2000.
- [24] Jong Yoon Hyun and Hyun Kwang Kim, *The poset structures admitting the extended binary Hamming code to be a perfect code*, Discrete Mathematics 288, 37-47 (2004).

# Índice Remissivo

- P*-código, 22
- P*-código linear, 22
- P*-métrica, 21
- P*-peso, 20
- P*-peso de um poset, 29
  
- Alfabeto, 5
- Alfabeto da fonte, 5
- Alfabeto do canal, 5
- Altura, 20
- anti-cadeia, 18
  
- Código, 5
- Código de bloco, 5, 10
- Código linear, 11
- Código perfeito, 16
- Cadeia, 18
- Canal sem memória, 6
- Canal simétrico, 6
- Capacidade do canal, 11
- Complementar de um poset com relação a um vetor, 29
- Complementar de um vetor com respeito a outro, 26
- Conjunto parcialmente ordenado, 18
- Conjunto Primário, 51
- Conjunto Secundário, 51
- Conjunto-raio, 30
  
- Decodificador por máxima proximidade, 12
- Decodificador por máxima verossimilhança, 9
- Diagrama de Hasse, 19
- Discordância, 47
- Discrepância, 33, 44
  
- Entrada do canal, 6
- Expressão simples, 51
- Extensão da fonte, 10
  
- Forma padrão de um poset, 63
- Função de codificação, 5
  
- Ideal, 20
- Ideal gerado, 20
- Invariante por translação, 26
- Isomorfismo de posets, 18
  
- Letra, 5
- Lista de vetores associada, 51
  
- Método da diferenciação, 35
- Métrica de Hamming, 12
- Matriz de adjacência, 19
- Matriz-raio, 55
- Maximal, 18
- Minimal, 18
  
- Número-matriz, 58
- Notação de Iverson, 28
  
- Observador ideal, 8
- Operador de associação, 50
- Operador de diferenciação, 50
- Ordem parcial, 18
  
- Palavra de comprimento  $n$ , 5
- Palavra-código, 5
- Partição ótima, 32
- Partição associada a uma expressão, 52
- Partição associada a uma expressão simples, 51
- Partição perfeita, 34
- Peso de Hamming, 12

Peso mínimo, 14  
Poset, 18  
Poset de Hamming, 20  
Poset Hierárquico, 20  
Problema da partição, 33  
Problema da partição clássica, 33  
Problema da partição para posets, 32

Raio de empacotamento de um código,  
14  
Raio de empacotamento de um poset, 32  
Raio de empacotamento de um vetor, 25  
Raio de empacotamento de uma matriz,  
55  
Regra de decisão, 7

Saída do canal, 6

Taxa de informação, 10

Vetor de adjacência, 49  
Vetor de empacotamento, 25  
Vetor-raio, 27