**Operações sobre Coleções
de Intervalos e suas aplicações
à Programação Automática de
Máquinas Morfológicas**

Guillermo Pablo Salas

DISSERTAÇÃO APRESENTADA
AO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA
DA
UNIVERSIDADE DE SÃO PAULO

PARA OBTENÇÃO DO GRAU
DE
MESTRE EM MATEMÁTICA APLICADA

Área de Concentração: **COMPUTAÇÃO**
Orientador: **Prof. Dr. Junior Barrera**

-São Paulo, Março de 1996-

*aos meus Pais,
irmã e Adriana
pelo estímulo e apoio*

Resumo

A Morfologia Matemática sobre conjuntos pode ser entendida como uma linguagem formal cujo vocabulário são erosões, dilatações, complementação, interseção e união. Esta linguagem é completa, isto é, ela é suficiente para realizar qualquer operador. Desde a década de sessenta tem sido construídas máquinas especiais para implementar esta linguagem, chamadas aqui *Máquinas Morfológicas* (MMach's, do inglês Morphological Machines). Na literatura, encontramos centenas de programas para MMach's que são usados para resolver problemas de Análise de Imagens. Apesar disso, o projeto destes programas não é uma tarefa elementar. Recentemente, muito esforço tem sido direcionado para a automação da programação de MMach's. Uma abordagem muito promissora deste problema é a descrição do operador desejado por pares de imagens (entrada-saída) e a tradução destes dados em programas eficientes de MMach's. Esta abordagem pode ser decomposta em dois passos igualmente importantes: i-) aprendizado do operador desejado a partir de pares de imagens; ii-) busca de representações económicas para o operador aprendido. A teoria apresentada nesta tese é útil em ambos os passos deste procedimento. Apresentamos algumas operações de conjunto sobre coleções de intervalos fechados e damos algoritmos eficientes para computá-los. Estas operações serão usadas para a paralelização de programas de MMach's, para provar a equivalência de programas distintos de MMach's e para definir um novo algoritmo eficiente de aprendizado.

Abstract

Mathematical Morphology on sets can be understood as a formal language, whose vocabulary are erosions, dilations, complementation, intersection and union. This language is complete, that is, it is enough to perform any set operator. Since the sixties special Machines, called Morphological Machines (MMach's), have been built to implement this language. In the literature, we find hundreds of MMach programs that are used to solve image analysis problems. However, the design of these programs is not an elementary task. Thus, recently much research effort has been addressed to automating the programming of MMach's. A very promising approach to this problem is the description of the target operator by input-output pairs of images and the translation of these data into efficient MMach programs. This approach can be decomposed in two equally important steps: i-learning of the target operator from the pairs of images; ii-search of economical representations for the operators learned. The theory presented in this paper is useful in both steps of this procedure. We will present some set operations on collections of closed intervals and give efficient algorithms to perform them. These operations will be used to parallelize MMach programs, to prove the equivalence between distinct MMach programs and to define a new efficient learning algorithm.

Agradecimentos

Gostaria de expressar o meu agradecimento a todos que contribuíram para que este trabalho fosse possível.

Em primeiro lugar, gostaria de agradecer ao Brasil pela oportunidade que me foi oferecida de fazer o mestrado neste país. Em particular, ao Instituto de Matemática e Estatística da Universidade de São Paulo e a todos os professores deste instituto que contribuíram na minha formação.

Ao meu orientador, o professor Junior Barrera, pelo seu apoio na realização deste trabalho, por discussões úteis, comentários interessantes, que muitas vezes me levaram a evitar erros, e pela cuidadosa revisão do trabalho.

À minha família pelo constante estímulo e apoio.

À Adriana pelo seu carinho e compreensão.

A meus professores da Universidad de Mar del Plata, Manuel Gonzalez, Raul Rivera e Ricardo Marcangeli, pelo apoio e incentivo.

À Sonia Trepode, Junior Barrera, Yoshiko Wakabayashi e Anil D'Souza, pelo apoio em um momento difícil.

Ao Ronaldo Fumio Hashimoto e Francisco de Assis Zampirolli pelo companheirismo e amizade.

Ao CNPq, UNMDP e Olivetti do Brasil pelo seu apoio financeiro.

Conteúdo

1	Introdução	6
2	Fundamentos Matemáticos	9
2.1	Teoria de Conjuntos e Teoria de Reticulados	9
2.2	Alguns reticulados Booleanos particulares	14
2.3	Teoria de aprendizado PAC	21
2.4	Descrição do algoritmo Quine-McCluskey	25
3	Operações sobre coleções de intervalos	31
4	Decomposição canônica de operadores de conjuntos	49
5	Composição de operadores na forma canônica	61
6	Computação Incremental da base de um operador	78
7	Algoritmos para a Computação da Base	93
7.1	Estrutura de dados	93
7.2	Rotinas Auxiliares	94
7.3	Rotina do Ínfimo	96
7.4	Rotina de Complementação	99
7.5	Avaliação do Desempenho	106
8	Algoritmos para o Aprendizado PAC	108
8.1	Divisão Incremental de Intervalos	108

8.2	Divisão Incremental de Intervalos Generalizada	111
8.3	Avaliação de Desempenho	111
9	Descrição da Toolbox Implementada	115
10	Conclusão	120

Lista de Figuras

2.1	Diagramas representando o Exemplo 2.1 . a) $\mathcal{X}$ e o intervalo $[000, 110]$ contido em $\mathcal{X}$. b) $\mathcal{X}$ e o intervalo $[000, 011]$ contido in $\mathcal{X}$. c) $\cup \mathcal{Y}$ e os intervalos $[000, 001]$ e $[010, 011]$ em $\mathcal{Y}$	17
2.2	Diagramas representando o Exemplo 2.2 a) $\mathcal{X}$ e o intervalo $[000, 110]$ contido em $\mathcal{X}$. b) $\mathcal{X}$ e o intervalo $[100, 111]$ contido em $\mathcal{X}$. c) $\cup \mathcal{Y}$ e os intervalos $[000, 100]$, $[000, 010]$ e $[100, 111]$ em $\mathcal{Y}$	19
2.3	Tabela verdade.	26
3.1	Interseção de duas subcoleções. a) $\mathcal{X}$. b) $\mathcal{Y}$. c) $\mathcal{X} \cap \mathcal{Y}$ e $Max(\mathcal{X} \cap \mathcal{Y})$. . .	35
3.2	Complemento de uma subcoleção a partir de um conjunto maximal de intervalos. $\cup \mathcal{X}$ e $\overline{\mathcal{X}} = \{[000, 010], [101, 111]\}$	40
3.3	Complemento de uma subcoleção a partir de um conjunto não maximal de intervalos. a) $\cup \mathcal{X}$ e $\{[1010, 1111], [0101, 1101]\} \subseteq \overline{Max(\cup \mathcal{X})}$. b) $\cup \mathcal{X}$ e $\{[1001, 1111]\} \subseteq \overline{Max(\cup \mathcal{X})}$	42
3.4	O intervalo $[1000, 1111] = \cup \mathcal{X}$ e a coleção unitária $\{[0000, 0111]\}$	44
3.5	Computação de $\mathcal{X} \sqcup \mathcal{Y}$ usando a lei de Morgan. a) $\mathcal{X}$ e $\cup \mathcal{X}$. b) $\mathcal{Y}$ e $\cup \mathcal{Y}$. c) $\cup(\mathcal{X} \sqcup \mathcal{Y})$ e $[100.111] \in \mathcal{X} \sqcup \mathcal{Y}$. d) $\cup(\mathcal{X} \sqcup \mathcal{Y})$ e $[010.111] \in \mathcal{X} \sqcup \mathcal{Y}$	47
3.6	Duas maneiras alternativas para computar a operação $\mathcal{X} \sqcup \mathcal{Y}$. a) Corolário 3.1. b) Corolário 3.2.	48
4.1	Os quatro isomorfismos de reticulados básicos.	59
4.2	Três representações equivalentes de um operador de conjuntos ψ i.t. e localmente definido, que realiza a detecção de bordas verticais. O subconjunto W é a janela horizontal de três pontos, centrada no origem. a) X e $\psi(X)$. b) f_ψ . c) $\mathcal{K}_W(\psi)$. d) $B_W(\psi)$	60
5.1	Ínfimo de dois operadores.	68

5.2	Composição do operador ψ com o operador de negação.	69
5.3	Supremo de dois operadores.	71
5.4	Composição do operador ψ com a dilatação δ_B	75
5.5	Composição do operador ψ com a erosão ε_B	77
6.1	a) Ícone da base do operador de identidade. b) Ícones do ínfimo, supremo, negação, erosão e dilatação, respectivamente, de cima para baixo.	79
6.2	Computação incremental da base de $\psi = \iota \wedge \nu \varepsilon_B \iota$	81
6.3	Computação incremental da base de $\psi = \iota \wedge \nu(\varepsilon_A \iota \wedge \nu \delta_B \iota)$	82
6.4	Comparação de igualdade entre dois operadores.	85
6.5	Verificação de equivalência de ψ_1 e ψ_2	86
6.6	Estruturas de decomposição altamente paralelas. a) Sup-decomposição canônica. b) Inf-decomposição canônica.	87
6.7	Duas representações do operador ψ . a) Definição de ψ . b) Inf-decomposição de ψ	89
6.8	Duas representações do operador. a) Definição de ψ . b) Inf-decomposição de ψ	90
6.9	Duas representações do operador. a) Definição de ψ . b) Inf-decomposição de ψ	92
8.1	a) Estágio inicial. b) Primeiro estágio. c) Segundo estágio. d) Terceiro estágio.	110
8.2	Árvore de intervalos do algoritmo ISI.	110
8.3	a) Estágio inicial. b) Primeiro estágio. c) Segundo estágio.	113
8.4	Árvore de intervalos do algoritmos G-ISI.	113
9.1	Menu de programas implementados.	117
9.2	Exemplo do uso da toolbox para computar a base de $\psi = \iota \wedge (\nu \varepsilon_A \iota \wedge \nu \delta_B \iota)$	118
9.3	Uso da toolbox que mostra duas alternativas para computar o ínfimo de dois operadores.	119

Lista de Tabelas

2.1	Tabela de definições.	25
7.1	Lista de instruções de controle.	93
7.2	Avaliação de desempenho dos algoritmos que computam o ínfimo. $B_{W=W_1 \cup W_2}(\psi = \psi_1 \wedge \psi_2) = B_{W_1}(\psi_1) \sqcap B_{W_2}(\psi_2)$	107
7.3	Avaliação de desempenho dos algoritmos que computam o complemento.	107
8.1	Avaliação de desempenho dos algoritmos ISI e G-ISI.	114

Capítulo 1

Introdução

A *Análise de Imagens Binárias* é uma ferramenta importante para várias áreas tais como controle de processos industriais, automação de escritórios, microscopia quantitativa, etc.

Um modelo natural para um procedimento usado em Análise de Imagens Binárias é um mapeamento definido entre subconjuntos aplicado sobre um Conjunto Aleatório Discreto [9]. Os mapeamentos definidos entre subconjuntos serão chamados aqui de *operadores de conjuntos*. A *Morfologia Matemática* (MM) é uma teoria geral para estudar os operadores de conjuntos [19, 22, 23].

Duas famílias simples de operadores de conjuntos são as chamadas de *erosões* e *dilatações*. O paradigma central da MM é a representação de operadores de conjuntos por uma concatenação de erosões e dilatações através de operações de *composição*, *interseção*, *união* e *complementação*.

Este paradigma pode ser formalizado pelo uso de uma linguagem formal, chamada aqui *Linguagem Morfológica*, cujo vocabulário são erosões, dilatações, interseção, união e complementação [5]. Esta linguagem é completa (i.e., qualquer operador de conjunto pode ser representado como uma frase desta linguagem) e expressiva (i.e., muitos operadores úteis podem ser construídos usando relativamente poucos operadores elementares). Uma frase da Linguagem Morfológica será chamada aqui *Operador Morfológico*.

Desde a década de sessenta têm sido construídas máquinas especiais para implementar esta linguagem, chamadas aqui *Máquinas Morfológicas* (MMach, do inglês Morphological Machines) [5, 15, 10, 14, 6]. Um *programa* para uma MMach é uma implementação de um Operador Morfológico.

Na literatura, encontramos centenas de aplicações de Operadores Morfológicos para

a solução de Problemas de Análise de Imagens [22, 2]. Apesar disso, o projeto de Operadores Morfológicos não é uma tarefa elementar. Classicamente esta tarefa se realiza empiricamente, baseada apenas na experiência do usuário com o uso de Operadores Morfológicos.

A abordagem clássica para o projeto de operadores morfológicos limita o universo de programadores de MMach's aos especialistas em MM. Para superar esta restrição, recentemente foram desenvolvidos muitos esforços de pesquisa em modelos para a automação da programação de MMach's [18, 17, 6, 7]. A ideia é encontrar formalismos adequados de representação de conhecimento e desenvolver ferramentas que traduzam estas representações em programas para MMach's. Algumas destas ferramentas usam coleções de pares de imagens entrada-saída como formalismo de representação de conhecimento.

O problema de projeto de um Operador Morfológico a partir de pares de imagens pode ser decomposto em dois passos igualmente importantes: i-) estimação ou aprendizado do operador desejado a partir de pares de imagens; ii-) busca de representações econômicas (i.e., que usam um número mínimo de palavras) para o operador estimado. A teoria apresentada neste trabalho é útil para efetuar ambos os passos deste procedimento.

Uma classe particular de Operadores Morfológicos usados na programação automática de MMach's é a *decomposição canônica*: união de interseções de erosões e dilatações complementadas. Esta estrutura de decomposição é paralela e suficiente para realizar qualquer operador de conjuntos. Os pares de erosões e dilatações usadas na decomposição canônica de um dado operador de conjuntos se caracterizam por uma coleção de intervalos fechados, chamado a *base do operador*. Uma propriedade importante da base de um operador é que ela caracteriza este operador de forma única, i.e., cada operador de conjuntos tem uma base que é distinta da base de qualquer outro operador de conjunto [3].

Apresentamos aqui algumas operações sobre coleções de intervalos e fornecemos algoritmos eficientes para realizá-las. Este estudo será o fundamento para a definição de um conjunto de regras que permitem a geração de Operadores Morfológicos que são sinônimos de um dado Operador Morfológico. Particularmente, estas regras permitirão computar a base de qualquer operador de conjuntos a partir de qualquer Operador Morfológico que representa este operador. Assim, serão úteis para paralelizar programas de MMach's, uma vez que a base caracteriza uma representação paralela, e para provar a equivalência entre Operadores Morfológicos distintos, uma vez que qualquer representação pode ser reduzida à base que determina unicamente o operador de conjunto.

Recentemente, Barrera et al. [7] propuseram um novo algoritmo para a estimação ou aprendizado de operadores de conjuntos a partir de pares de imagens [7]. Este algoritmo é

mais eficiente do que o algoritmo clássico de Quine-McCluskey [21, 20] em muitas situações práticas e é baseado em uma idéia simples: a *Divisão Incremental de Intervalos*, i.e., a representação, em termos de uma coleção de intervalos maximais, do conjunto formado pela eliminação de um ponto em um intervalo. O conceito de divisão de um intervalo pode ser generalizado pela representação da coleção formada pela eliminação de um intervalo de pontos de outro intervalo. A teoria apresentada aqui pode ser usada para desenvolver este novo conceito de divisão e, assim, aumentar a eficiência do algoritmo de aprendizado.

No capítulo dois relembramos alguns aspectos da teoria de conjuntos e da teoria de reticulados. Apresentamos algumas propriedades do conjunto de coleções de intervalos maximais e uma revisão da teoria de aprendizado PAC, que é usada para o projeto de operadores de conjuntos. Finalmente, descrevemos o algoritmo de Quine-McCluskey para a minimização de funções Booleanas. No capítulo três apresentamos alguns resultados que permitem a computação de operações sobre coleções de intervalos. No capítulo quatro revemos alguns conceitos de MM e apresentamos a generalização da decomposição canônica para operadores localmente definidos ou de janela. No capítulo cinco usamos os resultados do capítulo três para computar a base da interseção, da união e da composição com uma erosão (com uma dilatação ou com a complementação) de dados operadores. No capítulo seis mostramos como as regras apresentadas no capítulo cinco podem ser usadas para computar a base de qualquer operador de conjuntos. No capítulo sete apresentamos alguns algoritmos eficientes para implementar a teoria apresentada nos capítulos anteriores. No capítulo oito apresentamos uma generalização do Algoritmo de Divisão Incremental de Intervalos e estudamos a sua dinâmica. No capítulo nove descrevemos a toolbox implementada. Finalmente, apresentamos algumas conclusões e sugerimos alguns passos futuros para este trabalho de pesquisa.

Capítulo 2

Fundamentos Matemáticos

O objetivo deste capítulo é apresentar as definições e resultados teóricos necessários para a compressão dos resultados apresentados nesta dissertação.

Começamos discutindo na seção 2.1 conceitos básicos da teoria de conjuntos e teoria de reticulados [8, 22]. Na seção 2.2 estudamos alguns reticulados Booleanos que serão utilizados nos próximos capítulos. Na seção 2.3 apresentamos conceitos elementares da teoria de aprendizado PAC. Finalmente, discutimos o algoritmo de minimização de funções Booleanas de Quine-Mccluskey.

2.1 Teoria de Conjuntos e Teoria de Reticulados

Definição 2.1 *A igual a B:*

$$A = B \iff (x \in A \iff x \in B).$$

□

Definição 2.2 *A contido (subconjunto de) em B:*

$$A \subseteq B \iff (x \in A \longrightarrow x \in B).$$

□

Definição 2.3 *A união B:*

$$A \cup B = \{x : x \in A \text{ ou } x \in B\}.$$

□

Definição 2.4 A interseção B :

$$A \cap B = \{x : x \in A \text{ e } x \in B\}.$$

□

Definição 2.5 O conjunto vazio, denotado por $\emptyset$, é o conjunto que não contém elementos.

□

Definição 2.6 O produto cartesiano $A \times B$ do conjunto $A = \{a, \dots, a_n\}$ e $B = \{b, \dots, b_m\}$ é o conjunto que consiste dos pares ordenados (a_i, b_j) para todo $a_i \in A$ e $b_j \in B$

□

Definição 2.7 Uma relação binária α do conjunto A no conjunto B é um subconjunto do produto cartesiano $A \times B$ ($\alpha \subseteq A \times B$).

□

Definição 2.8 A inversa de uma relação binária α do conjunto A no conjunto B é a relação binária α^{-1} do conjunto B no conjunto A especificado por

$$\forall a \in A \text{ e } b \in B, b \alpha^{-1} a \iff a \alpha b.$$

□

Definição 2.9 Uma relação α sobre um conjunto A é dita reflexiva se e somente se, para todo $a \in A$, $a \alpha a$.

□

Definição 2.10 Uma relação α sobre um conjunto A é dita simétrica se e somente se, para todo $a, b \in A$, $a \alpha b \implies b \alpha a$.

□

Definição 2.11 Uma relação α sobre um conjunto A é dita anti-simétrica se e somente se, para todo $a, b \in A$, $a \alpha b \implies b = a$.

□

Definição 2.12 Uma relação α sobre um conjunto A é dita transitiva se e somente se, para todo $a, b, c \in A$, $a \alpha b$ e $b \alpha c \longrightarrow a \alpha c$.

□

Definição 2.13 Uma função $f : A \longrightarrow B$ (ou mapeamento ou transformação) de um conjunto A em um conjunto B é uma relação binária de A em B tal que para cada $a \in A$ existe um único $b \in B$ para o qual $(a, b) \in f$. Os conjuntos A e B são chamados, respectivamente, domínio e contradomínio de f . O conjunto $\{f(a) : a \in A\}$ é chamado imagem de A sob f , e denotado $f(A)$.

□

Definição 2.14 Uma função $f : A \longrightarrow B$ é dita sobrejetora se para todo $b \in B$ existe pelo menos um $a \in A$ tal que $b = f(a)$.

□

Definição 2.15 Uma função $f : A \longrightarrow B$ é dita injetora se $a_1 \neq a_2$ em A implica que $f(a_1) \neq f(a_2)$ em B .

□

Definição 2.16 Uma função $f : A \longrightarrow B$ é dita bijetora se e somente se, é injetora e sobrejetora.

□

Definição 2.17 Uma operação binária sobre o conjunto A é uma função: $A \times A \longrightarrow A$.

□

Definição 2.18 Uma relação de ordem parcial é uma relação reflexiva, anti-simétrica e transitiva sobre um conjunto.

□

Definição 2.19 Um conjunto parcialmente ordenado consiste de um conjunto $\mathcal{L}$ e uma relação de ordem parcial sobre $\mathcal{L}$, denotada por $(\mathcal{L}, \leq)$.

□

Definição 2.20 Seja $(\mathcal{L}, \leq)$ um conjunto parcialmente ordenado. Se $A \subseteq \mathcal{L}$, $u \in \mathcal{L}$, e $a \leq u$ para todo $a \in A$, então u é chamado um limitante superior de A . Similarmente, se

$l \in \mathcal{L}$ e $l \leq a$ para todo $a \in A$, então l é chamado um limitante inferior de A . Se A tem ambos um limitante superior e um limitante inferior, então A é chamado um subconjunto limitado de $\mathcal{L}$.

□

Definição 2.21 Seja A um subconjunto de um conjunto parcialmente ordenado $(\mathcal{L}, \leq)$. Então s é chamado mínimo (ou menor) elemento de A , denotado como $\min(A)$, se e somente se $s \in A$ e $s \leq a$ para todo $a \in A$. Similarmente, g é chamado o máximo (ou o maior) elemento de A , denotado como $\max(A)$, se e somente se $g \in A$ e $a \leq g$ para todo $a \in A$.

□

Definição 2.22 Seja $\mathcal{L}$ um conjunto parcialmente ordenado. Suponha que $A \subseteq \mathcal{L}$ e A tem um limitante superior em $\mathcal{L}$. O elemento mínimo, se existe, do conjunto de limitantes superiores de A é chamado de supremo de A e denotado por $\sup(A)$. Similarmente, suponha que A tem um limitante inferior em $\mathcal{L}$. O elemento máximo, se ele existe, do conjunto de limitantes inferiores de A , é chamado o ínfimo de A e denotado por $\inf(A)$. O $\sup(A)$ e o $\inf(A)$ podem ou não pertencer a A . Se $\sup(A) \in A$, então $\sup(A) = \max(A)$; do mesmo modo, se $\inf(A) \in A$, então $\inf(A) = \min(A)$.

□

Definição 2.23 Para a e b em um conjunto parcialmente ordenado $(\mathcal{L}, \leq)$ o supremo de a e b é um elemento $c \in \mathcal{L}$ que satisfaz a relação $a \leq c$ e $b \leq c$, e não há nenhum outro $x \in \mathcal{L}$ para o qual $a \leq x \leq c$ e $b \leq x \leq c$. Se dois elementos a e b em $\mathcal{L}$ tem um único supremo, então ele é denotado por $a \vee b$.

□

Definição 2.24 Para a e b em um conjunto parcialmente ordenado $(\mathcal{L}, \leq)$ o ínfimo de a e b é um elemento $d \in \mathcal{L}$ que satisfaz a relação $d \leq a$ e $d \leq b$, e não há nenhum outro $x \in \mathcal{L}$ para o qual $d \leq x \leq a$ e $d \leq x \leq b$. Se dois elementos a e b em $\mathcal{L}$ tem um único ínfimo, então ele é denotado por $a \wedge b$.

□

Definição 2.25 Seja $\mathcal{L}$ um conjunto parcialmente ordenado pela relação de ordem $\leq$.

$x \in \mathcal{L}$ é o menor elemento de $\mathcal{L}$ ($\min(\mathcal{L})$ ou O) se $x \leq y$ para todo $y \in \mathcal{L}$, isto é,

$$O = \min(\mathcal{L}) = x \iff (x \in \mathcal{L} \text{ e } (\forall y \in \mathcal{L} \implies x \leq y)),$$

$x \in A$ é o maior elemento de $\mathcal{L}$ ($\max(\mathcal{L})$ ou I) se $y \leq x$ para todo $y \in \mathcal{L}$, isto é,

$$I = \max(\mathcal{L}) = x \iff (x \in \mathcal{L} \text{ e } (\forall y \in \mathcal{L} \implies y \leq x)).$$

□

Proposição 2.1 *O menor e maior elemento de $\mathcal{L}$, se existirem, são únicos.*

□

Definição 2.26 *Um reticulado é um conjunto parcialmente ordenado $(\mathcal{L}, \leq)$, onde quaisquer dois elementos tem um único supremo e um único ínfimo.*

□

Definição 2.27 *Um reticulado $\mathcal{L}$ é :*

- *completo se e somente se qualquer subconjunto de $\mathcal{L}$ possui um ínfimo e um supremo em $\mathcal{L}$*

- *distributivo se e somente se $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$, $\forall x, y, z \in \mathcal{L}$*

- *complementado se e somente se todos os seus elementos possuem complemento. O complemento de um elemento x em um reticulado $\mathcal{L}$ com O e I é um elemento $y \in \mathcal{L}$ tal que $x \wedge y = O$ e $x \vee y = I$, e é denotado por x^c .*

- *Booleano se e somente se ele é distributivo e complementado.*

□

Teorema 2.1 *(Lei de Morgan) Em um reticulado Booleano vale $(a \wedge b)^c = a^c \vee b^c$ e $(a \vee b)^c = a^c \wedge b^c$*

□

Definição 2.28 *Sejam $\mathcal{L}_1$ e $\mathcal{L}_2$ dois reticulados ordenados pelas relações de ordem $\leq_{\mathcal{L}_1}$ e $\leq_{\mathcal{L}_2}$ respectivamente. Seja $f : \mathcal{L}_1 \longrightarrow \mathcal{L}_2$ uma função bijetora de $\mathcal{L}_1$ em $\mathcal{L}_2$. A função f é um isomorfismo de $\mathcal{L}_1$ em $\mathcal{L}_2$ se e somente se, quaisquer que sejam x e y em $\mathcal{L}_1$, tem-se:*

$$x \leq_{\mathcal{L}_1} y \iff f(x) \leq_{\mathcal{L}_2} f(y).$$

□

Definição 2.29 O conjunto potência do conjunto W , denotado por $\mathcal{P}(W)$, é a coleção de todos os subconjuntos de W .

□

Por causa da unicidade da união e da interseção em $\mathcal{P}(W)$, a conclusão imediata é que o conjunto parcialmente ordenado $(\mathcal{P}(W), \subseteq)$ é um reticulado.

Definição 2.30 Um conjunto G , munido com uma operação binária $+$, é um grupo se satisfaz os seguintes axiomas:

- $a + (b + c) = (a + b) + c, \quad \forall a, b, c \in G,$
- G contém um elemento e tal que $e + a = a + e = a \quad \forall a \in G,$
- $\forall a \in G$, existe um elemento em G , denotado por $-a$, tal que $A + (-a) = (-a) + a = e.$

□

Definição 2.31 Um grupo G é chamado de grupo abeliano se satisfaz

$$a + b = b + a \quad a, b \in G$$

□

2.2 Alguns reticulados Booleanos particulares

Seja E um conjunto não vazio e seja W um subconjunto finito de E . Seja $\mathcal{P}(W)$ a coleção de todos os subconjuntos de W . Seja $\subseteq$ a relação de inclusão usual sobre subconjuntos. O par $(\mathcal{P}(W), \subseteq)$ é um reticulado completo Booleano [8]. O menor e o maior elementos de $\mathcal{P}(W)$ são, respectivamente, $\emptyset$ e W . O ínfimo e o supremo de X_1 e X_2 em $\mathcal{P}(W)$ são, respectivamente, a interseção e a união de X_1 e X_2 , i.e., $X_1 \cap X_2$ e $X_1 \cup X_2$. O subconjunto complementar de um subconjunto X em $\mathcal{P}(W)$, com respeito a W , é denotado X^c , i.e., $X^c = \{x \in W : x \notin X\}$.

Uma propriedade particular de um Reticulado Booleano é a Lei de Morgan: $\forall X_1, X_2 \in \mathcal{P}(W)$,

$$(X_1^c \cap X_2^c)^c = X_1 \cup X_2.$$

Dados $A, B \in \mathcal{P}(W)$, a subcoleção $[A, B]$ de $\mathcal{P}(W)$ definida por

$$[A, B] = \{X \in \mathcal{P}(W) : A \subseteq X \subseteq B\}$$

é chamada um *intervalo fechado* ou, simplesmente, um *intervalo*. Se $A \subseteq B$, então os elementos A e B são chamados, respectivamente, as *extremidades esquerda* e *direita* de $[A, B]$. Para todos os pares (A, B) tais que $A \not\subseteq B$, $[A, B]$ representa a coleção vazia também denotada $\emptyset$.

Neste trabalho denotaremos as subcoleções de $\mathcal{P}(W)$ por letras caligráficas maiúsculas, i.e., $\mathcal{A}, \mathcal{B}, \dots, \mathcal{X}, \mathcal{Y}, \mathcal{Z}$. As coleções de intervalos fechados serão denotadas por letras caligráficas maiúsculas em negrito, isto é, $\mathbf{\mathcal{A}}, \mathbf{\mathcal{B}}, \dots, \mathbf{\mathcal{X}}, \mathbf{\mathcal{Y}}, \mathbf{\mathcal{Z}}$.

Um elemento de uma coleção de intervalos fechados $\mathbf{\mathcal{X}}$ é chamado *maximal em $\mathbf{\mathcal{X}}$* se nenhum outro elemento de $\mathbf{\mathcal{X}}$ o contém propriamente, i.e., $\forall [A, B] \in \mathbf{\mathcal{X}}$,

$$[A, B] \text{ é maximal em } \mathbf{\mathcal{X}} \iff \forall [A', B'] \in \mathbf{\mathcal{X}}, [A, B] \subseteq [A', B'] \implies [A, B] = [A', B'].$$

A coleção de todos intervalos fechados maximais em $\mathbf{\mathcal{X}}$ é denotada $Max(\mathbf{\mathcal{X}})$.

Seja $\mathcal{X}$ uma subcoleção de $\mathcal{P}(W)$. A coleção de todos intervalos fechados maximais contidos em $\mathcal{X}$ é denotada por $Max(\mathcal{X})$, i.e.,

$$Max(\mathcal{X}) = Max(\{[A, B] \subseteq \mathcal{P}(W) : [A, B] \subseteq \mathcal{X}\}).$$

Usualmente, denotaremos uma subcoleção de $\mathcal{P}(W)$ e o conjunto de intervalos maximais contidos nela pela mesma letra. Por exemplo, $\mathbf{\mathcal{X}} = Max(\mathcal{X})$.

Denotaremos por $\cup \mathbf{\mathcal{X}}$ a coleção de todos os elementos de $\mathcal{P}(W)$ que são elementos de intervalos fechados em $\mathbf{\mathcal{X}}$, i.e.,

$$\cup \mathbf{\mathcal{X}} = \{X \in \mathcal{P}(W) : \exists [A, B] \in \mathbf{\mathcal{X}} : X \in [A, B]\}.$$

Note que, para qualquer $\mathcal{X} \subseteq \mathcal{P}(W)$, $\cup \text{Max}(\mathcal{X}) = \mathcal{X}$. Assim, qualquer coleção $\mathcal{X}$ pode ser representada por seus intervalos fechados maximais. Note também que, para qualquer $\mathcal{X} \subseteq \mathcal{P}(W)$, $\text{Max}(\text{Max}(\mathcal{X})) = \text{Max}(\mathcal{X})$.

Uma vez que W é finito e $\mathcal{X} \subseteq \mathcal{P}(W)$, para qualquer $[A, B] \subseteq \mathcal{X}$, existe $[A', B'] \in \text{Max}(\mathcal{X})$ tal que $[A, B] \subseteq [A', B']$.

Por simplicidade de notação, representaremos os subconjuntos de W por sucessões de 0's e 1's, onde 0 significa que o ponto não pertence ao subconjunto e 1 significa que pertence. Por exemplo, se W é o subconjunto $\{(-1, 0), (0, 0), (1, 0)\}$, o subconjunto $\{(0, 0), (1, 0)\}$ será representado por 011.

As subcoleções de um Reticulado Booleano finito podem ser representadas por diagramas como os apresentados nas Figuras 2.1 e 2.2. As caixinhas representam os elementos do reticulado e os segmentos de reta (i.e., as arestas do cubo) representam a ordem parcial. As caixinhas em negrito representam os elementos da subcoleção. Os intervalos maximais dentro de uma subcoleção são representados por arestas em negrito. Se dois intervalos maximais de uma subcoleção tem um aresta comum, apenas por clareza, os representaremos em dois diagramas distintos.

O Exemplo 2.1 ilustra alguns destes conceitos.

Exemplo 2.1 a) Seja $W = \{(-1, 0), (0, 0), (1, 0)\}$ e sejam

$$\mathcal{X} = \{000, 100, 010, 001, 110, 011\}.$$

A subcoleção de intervalos maximais contidos em $\mathcal{X}$ é:

$$\text{Max}(\mathcal{X}) = \{[000, 110], [000, 011]\}.$$

As Figuras 2.1a e 2.1b mostram $\mathcal{X}$ e $\text{Max}(\mathcal{X})$.

b) Seja

$$\mathcal{Y} = \{[000, 000], [000, 001], [010, 011]\}.$$

O conjunto de intervalos maximais em $\mathcal{Y}$ é:

$$\text{Max}(\mathcal{Y}) = \{[000, 001], [010, 011]\}.$$

Figura 2.1: Diagramas representando o Exemplo 2.1 . a) $\mathcal{X}$ e o intervalo $[000, 110]$ contido em $\mathcal{X}$. b) $\mathcal{X}$ e o intervalo $[000, 011]$ contido in $\mathcal{X}$. c) $\cup \mathcal{Y}$ e os intervalos $[000, 001]$ e $[010, 011]$ em $\mathcal{Y}$.

A Figura 2.1c mostra $\cup \mathcal{Y}$ e $Max(\mathcal{Y})$.

□

Note que $Max(\cup \mathcal{X})$ não é necessariamente igual a $Max(\mathcal{X})$. O Exemplo 2.2 ilustra este fato.

Exemplo 2.2 a) Seja $W = \{(-1, 0), (0, 0), (1, 0)\}$ e seja

$$\mathcal{X} = \{000, 100, 010, 110, 101, 111\}.$$

O conjunto de intervalos maximais contidos em $\mathcal{X}$ é:

$$Max(\mathcal{X}) = \{[000, 110], [100, 111]\}.$$

As Figuras 2.2a e 2.2b mostram $\mathcal{X}$ e $Max(\mathcal{X})$.

b) Seja

$$\mathcal{Y} = \{[000, 000], [000, 100], [000, 010], [100, 111], [110, 111]\}.$$

O conjunto de intervalos maximais em $\mathcal{Y}$ é:

$$Max(\mathcal{Y}) = \{[000, 100], [000, 010], [100, 111]\}.$$

A Figura 2.2c mostra $\cup \mathcal{Y}$ e $Max(\mathcal{Y})$.

□

Figura 2.2: Diagramas representando o Exemplo 2.2 a) $\mathcal{X}$ e o intervalo $[000, 110]$ contido em $\mathcal{X}$. b) $\mathcal{X}$ e o intervalo $[100, 111]$ contido em $\mathcal{X}$. c) $\cup \mathcal{Y}$ e os intervalos $[000, 100]$, $[000, 010]$ e $[100, 111]$ em $\mathcal{Y}$.

Seja $\mathcal{P}(\mathcal{P}(W))$ a coleção de todas as subcoleções de $\mathcal{P}(W)$. Seja $\subseteq$ a relação usual de inclusão entre subconjuntos. O par $(\mathcal{P}(\mathcal{P}(W)), \subseteq)$ é um reticulado completo Booleano. O menor e maior elementos de $\mathcal{P}(\mathcal{P}(W))$ são, respectivamente, $\emptyset$ e $\mathcal{P}(W)$. O ínfimo e o supremo de $\mathcal{X}_1$ e $\mathcal{X}_2$ são respectivamente a interseção e a união de $\mathcal{X}_1$ e $\mathcal{X}_2$ em $\mathcal{P}(\mathcal{P}(W))$, i.e., $\mathcal{X}_1 \cap \mathcal{X}_2$ e $\mathcal{X}_1 \cup \mathcal{X}_2$. A subcoleção complementar de uma subcoleção $\mathcal{X}$ em $\mathcal{P}(\mathcal{P}(W))$, com respeito a $\mathcal{P}(W)$, é denotada $\mathcal{X}^c$, i.e., $\mathcal{X}^c = \{X \in \mathcal{P}(W) : X \notin \mathcal{X}\}$.

Denotamos por Π_W o conjunto de todas as subcoleções de intervalos fechados maximais contidos em $\mathcal{P}(W)$, i.e.,

$$\Pi_W = \{Max(\mathcal{X}) : \mathcal{X} \subseteq \mathcal{P}(W)\}.$$

Definimos a relação de ordem parcial $\leq$ no conjunto de Π_W fazendo : $\forall \mathcal{X}, \mathcal{Y} \in \Pi_W$,

$$\mathcal{X} \leq \mathcal{Y} \iff \forall [A, B] \in \mathcal{X}, \exists [A', B'] \in \mathcal{Y} : [A, B] \subseteq [A', B'].$$

É imediato que a relação definida acima é reflexiva e transitiva. Esta relação é também anti-simétrica, já que as coleções de intervalos fechados consideradas são maximais. De fato, o poset $(\Pi_W, \leq)$ constitui um reticulado Booleano, onde as operações de ínfimo, supremo e negação são fornecidas, respectivamente, por: $\forall \mathcal{X}, \mathcal{Y} \in \Pi_W$,

$$\mathcal{X} \cap \mathcal{Y} = Max(\mathcal{X} \cap \mathcal{Y}),$$

$$\mathcal{X} \sqcup \mathcal{Y} = Max(\mathcal{X} \cup \mathcal{Y})$$

e

$$\overline{\mathcal{X}} = Max(\mathcal{X}^c).$$

Estas expressões surgem porque o mapeamento $Max(\cdot)$, definido de $\mathcal{P}(\mathcal{P}(W))$ em Π_W , é uma bijeção crescente. O inverso do mapeamento $Max(\cdot)$ é o mapeamento $\cup(\cdot)$.

Em particular, note que o menor e o maior elemento de $(\Pi_W, \leq)$ são, respectivamente, $Max(\emptyset) = \{\emptyset\}$ e $Max(\mathcal{P}(W)) = \{[\emptyset, W]\}$.

Seja $\{0, 1\}^{\mathcal{P}(W)}$ o conjunto de todas as funções Booleanas definidas sobre $\mathcal{P}(W)$. O par $(\{0, 1\}^{\mathcal{P}(W)}, \leq)$, onde $\leq$ é a relação de ordem parcial herdada da relação de ordem total na cadeia $\{0, 1\}$, constitui um reticulado Booleano. Este reticulado é isomorfo ao reticulado $(\Pi_W, \leq)$, uma vez que o mapeamento F , definido de Π_W em $\{0, 1\}^{\mathcal{P}(W)}$ por

$$F(\mathcal{X})(X) = \begin{cases} 1 & \text{se } X \in \cup \mathcal{X} \\ 0 & \text{caso contrario} \end{cases} \quad (X \in \mathcal{P}(W)),$$

é uma bijeção crescente de dois lados. O inverso do mapeamento F é o mapeamento F^{-1} , definido por

$$F^{-1}(f) = Max(\{X \in \mathcal{P}(W) : f(X) = 1\}) \quad (f \in \{0, 1\}^{\mathcal{P}(W)}).$$

A Figura 4.1 ilustra os isomorfismos de reticulado entre $(\mathcal{P}(\mathcal{P}(W)), \subseteq)$, $(\Pi_W, \leq)$, e $(\{0, 1\}^{\mathcal{P}(W)}, \leq)$.

2.3 Teoria de aprendizado PAC

A *teoria de aprendizado Computacional* [25],[1] é um das primeiras tentativas de construir um modelo matemático para os processos cognitivos de aprendizado de conceito. Ela constitui uma abordagem para estudar uma variedade de processos algorítmicos, tais como aqueles correntemente em uso para treinamento de redes neurais artificiais.

Entendemos *conceito* como um subconjunto de objetos em um domínio pré definido. Um *exemplo* de um conceito é um objeto deste o domínio junto com um rótulo indicando se o objeto pertence ao conceito. Se o objeto pertence ao conceito, é um *exemplo positivo* caso contrario, é um *exemplo negativo*. O *aprendizado de um Conceito* é o processo pelo qual um *aprendiz* constrói uma boa aproximação para um conceito desconhecido, dado um pequeno número de *exemplos* e alguma informação prévia a respeito do conceito a ser aprendido. Em seguida, formalizamos estas idéias.

Seja Σ um conjunto, chamado o *alfabeto* para descrever exemplos. Nesta trabalho, Σ será o *alfabeto Booleano* $\{0, 1\}$. Denotaremos o conjunto de n -tuplas de elementos de Σ por Σ^n . Seja X um subconjunto de Σ^n . Definimos um *conceito*, sobre o alfabeto Σ , como uma função $c : X \rightarrow \{0, 1\}$.

Vamos nos referir a X como o *espaço de exemplos*, e a seus membros como exemplos. Um exemplo $y \in X$ para o qual $c(y) = 1$ é conhecido como um *exemplo positivo* e um exemplo para o qual $c(y) = 0$ é conhecido como um *exemplo negativo*. Assim, se o domínio é conhecido, c determina e é determinado por seu conjunto de exemplos positivos. As vezes é útil pensar em um conceito como um subconjunto dessa forma.

O conjunto de todos os possíveis conceitos a serem aprendidos será referido como o *espaço de hipótese* e denotado por H . O conceito $t \in H$ a ser determinado é chamado o *conceito objetivo*. O problema é encontrar um conceito $h \in H$, chamado *hipótese*, que é uma boa aproximação para t .

Uma *amostra de comprimento* m é apenas uma sequência de m exemplos, isto é, uma m -tupla $\mathbf{x} = \{x_1, x_2, \dots, x_m\}$ em X^m . Uma sequência pode conter os mesmos valores mais de uma vez. Uma *amostra de treinamento* $\mathbf{s}$ é um elemento de $X^m \times \{0, 1\}$, isto é,

$$\mathbf{s} = ((x_1, b_1), (x_2, b_2), \dots, (x_m, b_m)),$$

onde os x_i são exemplos e os b_i são bits. O valor de b_i é fornecido por um “professor” e especifica se x_i é um exemplo positivo ou negativo. Não é permitida contradição, isto é, se $x_i = x_j$ então $b_i = b_j$.

Um *algoritmo de aprendizado* é simplesmente uma função L a qual atribui a qualquer exemplo de treinamento $\mathbf{s}$, para um conceito objetivo $t \in H$, uma hipótese $h \in H$. Escreveremos $h = L(\mathbf{s})$.

Seja μ uma *medida de probabilidade* sobre X . Dado um conceito objetivo $t \in H$, definimos o *erro de qualquer hipótese* $h \in H$, com respeito a t , como a probabilidade do evento $h(x) \neq t(x)$, isto é,

$$\text{er}_\mu(h, t) = \mu\{x \in X : h(x) \neq t(x)\}.$$

Quando um conjunto X é provido com a estrutura de uma espaço de probabilidade, o conjunto produto X^m herda esta estrutura de X . A correspondente distribuição sobre

X^m será denotada μ^m . As componentes da m -tupla $(x_1, x_2, \dots, x_m)$ são supostas variáveis "independentes", cada uma distribuída de acordo com a distribuição de probabilidade μ sobre X .

Denotaremos por $S(m, t)$ o conjunto de amostras de treinamento de comprimento m para um conceito objetivo t , onde os exemplos são extraídos de um espaço de exemplos X . Já que existe uma bijeção $\phi : X^m \rightarrow S(m, t)$ para a qual $\phi(\mathbf{x}) = \mathbf{s}$, a seguinte igualdade é válida

$$\mu^m\{\mathbf{s} \in S(m, t) : \mathbf{s} \text{ tem propriedade } P\} = \mu^m\{\mathbf{x} \in X^m : \phi(\mathbf{x}) \in S(m, t) \text{ tem propriedade } P\}.$$

Definição 2.32 Dizemos que um algoritmo L é um algoritmo de aprendizado provavelmente aproximadamente correto (PAC) para o espaço de hipótese H se, para dois números reais ε e δ ($0 < \varepsilon, \delta < 1$), existe um inteiro positivo $m_0 = m_0(\varepsilon, \delta)$ tal que, para qualquer conceito objetivo $t \in H$, e, para qualquer distribuição μ sobre X , sempre que $m \geq m_0$,

$$\mu^m\{\mathbf{s} \in S(m, t) : \mathbf{er}_\mu(L(\mathbf{s}), t) < \varepsilon\} > 1 - \delta.$$

A função $m_0 = m_0(\varepsilon, \delta)$ é chamada complexidade de exemplos do problema.

□

Um algoritmo de aprendizado L para H é consistente se e só se, dada qualquer amostra de treinamento $\mathbf{s}$, para um conceito objetivo $t \in H$, a hipótese de saída concorda com t para os exemplos em $\mathbf{s}$, isto é, $h(x_i) = t(x_i)$ ($1 \leq i \leq m$).

Para uma dada $\mathbf{s} \in S(m, t)$, denotaremos por $H[\mathbf{s}]$ o conjunto de todas as hipóteses consistentes com $\mathbf{s}$, isto é,

$$H[\mathbf{s}] = \{h \in H : h(x_i) = t(x_i) \text{ } (1 \leq i \leq m)\}.$$

Dado $\varepsilon \in (0, 1)$ o conjunto

$$\mathbf{B}_t = \{h \in H : \mathbf{er}_\mu(h, t) \geq \varepsilon\}$$

é chamada o conjunto de *hipóteses ruim* para t .

Dizemos que o espaço de hipótese H é potencialmente aprendível se, dados dois números reais ε e δ ($0 < \varepsilon, \delta < 1$), existe um inteiro positivo $m_0 = m_0(\varepsilon, \delta)$ tal que, sempre que $m \geq m_0$

$$\mu^m \{s \in S(m, t) : H[s] \cap B_\varepsilon = \emptyset\} > 1 - \delta,$$

Para qualquer distribuição de probabilidade μ sobre X e qualquer $t \in H$.

Prova-se que os seguintes teoremas valem [1].

Teorema 2.2 *Se H é potencialmente aprendível e L é um algoritmo de aprendizagem consistente para H , então L é PAC.*

□

Teorema 2.3 *Qualquer espaço de hipóteses finito é potencialmente aprendível.*

□

Um corolário imediato dos Teoremas 2.2 e 2.3 é que qualquer algoritmo de aprendizagem consistente aplicado a um espaço de hipóteses finito é PAC.

Denotaremos $|H|$ a cardinalidade do conjunto H . Segundo Anthony e Biggs, a complexidade de exemplos de um algoritmo de aprendizado PAC é

$$m_0(\varepsilon, \delta) = \left\lceil \frac{1}{\varepsilon} \ln \frac{|H|}{\delta} \right\rceil,$$

para qualquer distribuição μ e com a única restrição que o espaço de hipóteses H seja finito.

Aqui, usaremos a teoria de aprendizado PAC para aprender funções Booleanas que são equivalentes a operadores i.t. localmente definidos dentro de uma janela finita W . Uma vez que este espaço de hipóteses é finito, todos os algoritmos de aprendizado consistentes são PAC e, conseqüentemente, a última expressão pode ser usada para computar o número mínimo de exemplos requeridos (isto é, a complexidade de exemplos) para estes algoritmos.

2.4 Descrição do algoritmo Quine-McCluskey

Nesta seção, descreveremos o algoritmo de Quine-McCluskey (QM) para a minimização de funções Booleanas [11]. Antes de apresentar o algoritmo introduziremos algumas definições.

A Tabela 2.1 mostra as definições de *literal*, *produto* e *mintermo*.

Termo	Definição
Literal	Uma variável e o seu complemento (A , $\bar{A}$, B , $\bar{B}$, D , etc).
Produto	Uma série de literais relacionados por AND, p.e., $A \bar{B} C$, $A B C \bar{D}$.
Mintermo	Um produto no qual estão todas as variáveis da função.

Tabela 2.1: Tabela de definições.

Dizemos que uma função Booleana é expressa na *forma canônica*, se ela é expressa como a soma de todos os mintermos da função.

Por exemplo a função $f(A, B, C) = A + \bar{B} C$ expressa na forma canônica é $f(A, B, C) = A \bar{B} \bar{C} + A B \bar{C} + A \bar{B} C + A B C + \bar{A} \bar{B} C$.

A figura 2.3 mostra a tabela verdade de uma função de 3 variáveis. O número associado é simplesmente o equivalente decimal da combinação das variáveis de entrada interpretada como um número decimal. Agora, podemos especificar a função listando as linhas que tem valor 1 da seguinte maneira:

$$f(A, B, C) = \sum(0, 4, 5, 7).$$

O algoritmo de Quine-McCluskey requer que as funções Booleanas estejam expressas na forma canônica. A idéia básica deste algoritmo consiste em encarar os mintermos da função como pontos no n -espaço, ou seja como vértices de um n -cubo. A partir do conjunto destes vértices (ou 0-cubos) procura-se gerar todos os 1-cubos possíveis combinando-se dois deles (equivale a gerar as arestas do cubo que ligam dois 0-cubos considerados). A partir da combinação de dois 1-cubos procura-se gerar todos os possíveis 2-cubos e assim por diante, até que nenhum cubo de dimensão maior possa ser gerado a partir da combinação de dois cubos de dimensão menor.

O processo de geração de k -cubos ($1 \leq k \leq n$) a partir da combinação de dois

Número de fila	A B C	f
0	0 0 0	1
1	0 0 1	0
2	0 1 0	0
3	0 1 1	0
4	1 0 0	1
5	1 0 1	1
6	1 1 0	0
7	1 1 1	1

Figura 2.3: Tabela verdade.

$(k - 1)$ -cubos será denominado passo do algoritmo de minimização. O conjunto de cubos resultantes ao final de todo o processo são denominados **implicantes primos**.

A garantia de que o algoritmo QM realmente fornece uma expressão minimal, equivalente à expressão original, é uma decorrência do fato de que, em cada passo do algoritmo, são gerados todas as combinações possíveis entre os termos.

Descrição dos passos do algoritmo QM

A primeira parte do algoritmo QM [24] consiste de um processo para determinação de todos os implicantes primos. A seguir descrevemos os processos que constituem esta parte e, ao mesmo tempo, mostramos a sua aplicação sobre a função $f(x_1, x_2, x_3) = \sum m(0, 1, 4, 5, 6)$.

- Primeiro passo: converter os mintermos para a notação binária.

000, 001, 100, 101, 110

- Segundo passo: Separar os mintermos em grupos de acordo com o número de 1's em sua representação binária e ordená-los em ordem crescente, em sua coluna, separando os grupos com uma linha horizontal.

000
001
100
101
110

- Terceiro Passo: combinar todos os elementos de um grupo com todos os elementos do grupo adjacente inferior para a geração de cubos de dimensão maior. Para cada 2 grupos comparados entre si, gerar um novo grupo na próxima coluna e colocar os novos cubos. Marcar com $\checkmark$ os cubos que foram usados para gerar novos cubos.

$\checkmark$	000	$\Rightarrow$	00X
$\checkmark$	001		X00
$\checkmark$	100		X01
$\checkmark$	101		10X
$\checkmark$	110		1X0

Observação: o novo cubo gerado será inserido no novo conjunto se e somente se ele ainda não estiver contido nele.

Repetir o processo para cada nova coluna formada, até que nenhuma combinação mais seja possível.

$\checkmark$	000	$\Rightarrow$	$\checkmark$	00X	$\Rightarrow$	X0X
$\checkmark$	001		$\checkmark$	X00		
$\checkmark$	100		$\checkmark$	X01		
$\checkmark$	101		$\checkmark$	10X		
$\checkmark$	110			1X0		

- Quarto passo: Listar os implicantes primos. Os implicantes primos são aqueles que não foram combinados nenhum outro, ou seja, aqueles que estão com a marca $\checkmark$.

1X0 e X0X

Definição 2.33 Seja f uma função Booleana e seja $\mathcal{C}$ a coleção de implicantes primos correspondente a esta função. Seja $\mathcal{S} \subseteq \mathcal{C}$. Então $\mathcal{S}$ é uma subcoleção ótima que cobre todos os mintermos de f se:

não existe outra subcoleção $\mathcal{S}'$ que cobre todos os implicantes primos de f , $\mathcal{S}' \subseteq \mathcal{C}$, tal que $|\mathcal{S}'| < |\mathcal{S}|$;

□

Definição 2.34 Uma subcoleção ótima dos implicantes primos que cobre todos os min-terms da expressão Booleana original é denominada conjunto dos **implicantes primos essenciais**.

□

Os implicantes primos essenciais são calculados com a ajuda de uma Tabela de Implicantes Primos [11], no qual é selecionado um subconjunto ótimo de implicantes primos que cobre todos os mintermos da expressão Booleana inicial.

A seguir descrevemos os passos da segunda parte do algoritmo QM e mostramos sua aplicação sobre a função

$$f(x_1, x_2, x_3, x_4, x_5) = \sum(1, 2, 3, 5, 9, 10, 11, 18, 19, 20, 21, 23, 25, 26, 27)$$

1. Construir a Tabela de Implicantes Primos: no topo das colunas deve-se colocar os mintermos positivos e, à esquerda de cada linha, os implicantes primos. Os implicantes primos devem ser listados em ordem decrescente de acordo com sua dimensão, isto é, ordem crescente de acordo com o número de literais. Deve-se acrescentar uma coluna à esquerda e uma linha na parte inferior. Em cada linha, marcar as colunas com $\checkmark$ quando o implicante primo da linha cobre o mintermo da coluna.

		1	2	3	5	9	10	11	18	19	20	21	23	25	26	27
	XX01X		$\checkmark$	$\checkmark$			$\checkmark$	$\checkmark$	$\checkmark$	$\checkmark$					$\checkmark$	$\checkmark$
	X10X1					$\checkmark$		$\checkmark$						$\checkmark$		$\checkmark$
	0X0X1	$\checkmark$		$\checkmark$		$\checkmark$		$\checkmark$								
	00X01	$\checkmark$			$\checkmark$											
	X0101				$\checkmark$							$\checkmark$				
	1010X										$\checkmark$	$\checkmark$				
	10X11									$\checkmark$			$\checkmark$			
	10X11											$\checkmark$	$\checkmark$			

2. Selecionar os implicantes primos essenciais: deve-se procurar na tabela, as colunas que contém apenas uma marca $\checkmark$. A linha na qual estas colunas contém a marca $\checkmark$ corresponde a um implicante primo essencial. Em outras palavras, este implicante primo é o único que cobre o mintermo da coluna e, por tanto, não pode ser descartado. Então, deve-se marcar com um asterisco (*) esta linha na coluna mais à

esquerda, para indicar que este é um implicante primo essencial. A seguir, deve-se marcar, na última linha da tabela, todas as colunas cujo mintermo é coberto pelo implicante primo selecionado.

		1	2	3	5	9	10	11	18	19	20	21	23	25	26	27
*	XX01X		✓	✓			✓	✓	✓	✓					✓	✓
	X10X1					✓		✓						✓		✓
	0X0X1	✓		✓		✓		✓								
	00X01	✓			✓											
	X0101				✓							✓				
	1010X										✓	✓				
	10X11									✓			✓			
	10X11											✓	✓			
			✓	✓			✓	✓	✓	✓					✓	✓

Deve-se repetir este processo enquanto existir uma coluna cujo mintermo não está coberto pelo conjunto de implicantes primos selecionados até a momento e que satisfaz as condições descritas.

		1	2	3	5	9	10	11	18	19	20	21	23	25	26	27
*	XX01X		✓	✓			✓	✓	✓	✓					✓	✓
*	X10X1					✓		✓						✓		✓
	0X0X1	✓		✓		✓		✓								
	00X01	✓			✓											
	X0101				✓							✓				
*	1010X										✓	✓				
	10X11									✓			✓			
	10X11											✓	✓			
			✓	✓		✓	✓	✓	✓	✓	✓	✓		✓	✓	✓

3. Reduzir a tabela: manter apenas as colunas correspondentes aos mintermos não cobertos pelos implicantes primos essenciais, e as linhas correspondente aos implicantes primos que não foram selecionados e que cobrem pelo menos um mintermo ainda não coberto.

		1	5	23
	0X0X1	✓		
	00X01	✓	✓	
	X0101		✓	
	10X11			✓
	101X1			✓

4. Selecionar os implicantes primos secundariamente essenciais: eliminar as linhas dominadas e selecionar os essenciais, conforme definição a seguir.

Definição 2.35 Diz-se que uma linha A na tabela de implicantes Primos domina uma outra linha B se A cobre pelo menos todos os mintermos que estão nas colunas da tabela e que são cobertos por B .

□

Proposição 2.2 Sejam A e B duas linhas na tabela de implicantes Primos, tal que A domina B , existe uma forma minimal que não leva B .

□

A seguir, deve-se repetir o mesmo processo do passo 2, porém os implicantes primos essenciais serão chamados secundariamente essenciais e marcados com dois asteriscos (**), e em seguida fazer a redução da tabela (passo 3).

		1	5	23
	0X0X1	✓		
**	00X01	✓	✓	
**	10X11			✓
		✓	✓	✓

Deve-se repetir este processo até que isto não seja mais possível, ou seja, até atingir-se um estágio em que não mais existe, na tabela reduzida, nenhuma coluna com apenas um ✓ e não existe nenhuma linha que cobre outra.

5. Aplicar método de Petrick: este método (de busca exaustiva) fornece todas as possíveis combinações dos implicantes primos restantes que cobrem os mintermos restantes. Deve-se escolher dentre elas, aquela que envolve o menor número de termos caso existam mais de uma nestas condições, deve-se escolher a de custo mínimo (aquela que envolve menor número de literais).

Capítulo 3

Operações sobre coleções de intervalos

Em aplicações computacionais usualmente subcoleções de $\mathcal{P}(W)$ são representadas por seus intervalos maximais para minimizar o espaço armazenado. Assim, é útil expressar as operações de interseção, união e complementação de subcoleções em termos de seus intervalos maximais. Neste capítulo, apresentamos um procedimento que computa o conjunto de intervalos maximais contidos na união de duas subcoleções a partir dos conjuntos de intervalos maximais contidos em cada uma das subcoleções. Resultados similares serão dados para as operações de interseção e complementação.

Proposição 3.1 *Sejam $[A, B]$ e $[C, D]$ dois intervalos contidos em $\mathcal{P}(W)$, então*

$$[A, B] \cap [C, D] = [A \cup C, B \cap D].$$

Prova:

$$\begin{aligned} [A, B] \cap [C, D] &= \{X \in \mathcal{P}(W) : A \subseteq X \subseteq B\} \cap \{X \in \mathcal{P}(W) : C \subseteq X \subseteq D\} \\ &= \{X \in \mathcal{P}(W) : A \subseteq X \subseteq B \text{ e } C \subseteq X \subseteq D\} \\ &= \{X \in \mathcal{P}(W) : A \subseteq X \text{ e } C \subseteq X \text{ e } X \subseteq B \text{ e } X \subseteq D\} \\ &= \{X \in \mathcal{P}(W) : A \cup C \subseteq X \text{ e } X \subseteq B \cap D\} \end{aligned}$$

(como $\cap$ e $\cup$ são, respectivamente, o ínfimo e o supremo em $\mathcal{P}(W)$).

$$\begin{aligned} &= \{X \in \mathcal{P}(W) : A \cup C \subseteq X \subseteq B \cap D\} \\ &= [A \cup C, B \cap D] \end{aligned}$$

□

Teorema 3.1 *Seja $\mathcal{X}$ e $\mathcal{Y}$ dois elementos de Π_W , então*

$$\mathcal{X} \sqcap \mathcal{Y} = \text{Max}(\{[A \cup C, B \cap D] : [A, B] \in \mathcal{X}, [C, D] \in \mathcal{Y}\}).$$

Prova:

Denotemos $\mathcal{X} = \text{Max}(\mathcal{X})$ e $\mathcal{Y} = \text{Max}(\mathcal{Y})$.

$$\mathcal{X} \sqcap \mathcal{Y} = \text{Max}(\mathcal{X} \cap \mathcal{Y})$$

(pelo isomorfismo de reticulados entre $\mathcal{P}(\mathcal{P}(W))$ e Π_W)

$$= \text{Max}(\{[A, B] \subseteq \mathcal{P}(W) : [A, B] \subseteq \mathcal{X} \cap \mathcal{Y}\})$$

(pela definição de Max em uma subcoleção de $\mathcal{P}(W)$)

$$= \text{Max}(\{[A, B] \subseteq \mathcal{P}(W) : [A, B] \subseteq \mathcal{X} \text{ e } [A, B] \subseteq \mathcal{Y}\})$$

$$= \text{Max}(\{[A, B] \subseteq \mathcal{P}(W) : \exists [A', B'] \in \mathcal{X}, [A, B] \subseteq [A', B']$$

e

$$\exists [C', D'] \in \mathcal{Y}, [A, B] \subseteq [C', D']\})$$

(pela definição de Max com W finito)

$$= \text{Max}(\{[A, B] \subseteq \mathcal{P}(W) : \exists [A', B'] \in \mathcal{X} \text{ e } \exists [C', D'] \in \mathcal{Y},$$

$$[A, B] \subseteq [A', B'] \cap [C', D']\})$$

(como $\cap$ é o ínfimo em $\mathcal{P}(\mathcal{P}(W))$)

$$= \text{Max}(\{[A, B] \subseteq \mathcal{P}(W) : \exists [A', B'] \in \mathcal{X} \text{ e } \exists [C', D'] \in \mathcal{Y},$$

$$[A, B] \subseteq [A' \cup C', B' \cap D']\})$$

(pela Proposição 3.1)

$$= \text{Max}(\{[A, B] \subseteq \mathcal{P}(W) : \exists [A', B'] \in \mathcal{X} \text{ e } \exists [C', D'] \in \mathcal{Y},$$

$$[A, B] = [A' \cup C', B' \cap D']\})$$

(pela eliminação dos elementos não maximais em uma cadeia)

$$= \text{Max}(\{[A' \cup C', B' \cap D'] : [A', B'] \in \mathcal{X} \text{ e } [C', D'] \in \mathcal{Y}\})$$

□

Este resultado fornece um procedimento para computar a operação de interseção de duas subcoleções de $\mathcal{P}(W)$, representadas pelos seus intervalos maximais.

Exemplo 3.1 *Seja $E = \{(-1, 0), (0, 0), (1, 0)\}$. Sejam*

$$\mathcal{X} = \{000, 100, 010, 001, 110, 011\}$$

e

$$\mathcal{Y} = \{000, 100, 001, 110, 101, 111\}.$$

As Figuras 3.1a e 3.1b mostram $\mathcal{X}$ e $\mathcal{Y}$, respectivamente

Os intervalos maximais destas coleções são, respectivamente,

$$\mathcal{X} = \text{Max}(\mathcal{X}) = \{[000, 110], [000, 011]\}$$

e

$$\mathcal{Y} = \text{Max}(\mathcal{Y}) = \{ [000, 101], [100, 111] \}.$$

Aplicando o Teorema 3.1, temos:

$$\text{Max}(\mathcal{X} \cap \mathcal{Y}) = \mathcal{X} \cap \mathcal{Y}$$

$$\begin{aligned} & \text{Max}(\{ [000, 110] \cap [000, 101], [000, 110] \cap \\ & [100, 111], [000, 011] \cap [000, 101], \\ & [000, 011] \cap [100, 111] \}) \\ &= \text{Max}(\{ [000, 100], [100, 110], [000, 001], \emptyset \}) \\ &= \{ [000, 100], [000, 001], [100, 110] \} \end{aligned}$$

A Figura 3.1c mostra $\mathcal{X} \cap \mathcal{Y}$ e $\text{Max}(\mathcal{X} \cap \mathcal{Y})$.

□

Figura 3.1: Interseção de duas subcoleções. a) $\mathcal{X}$. b) $\mathcal{Y}$. c) $\mathcal{X} \cap \mathcal{Y}$ e $Max(\mathcal{X} \cap \mathcal{Y})$.

Seja $[A, B]$ um intervalo em $\mathcal{P}(W)$. Seja $[A, B]^c$ o complemento de $[A, B]$ em $\mathcal{P}(W)$, i.e.,

$$[A, B]^c = \{X \in \mathcal{P}(W) : X \notin [A, B]\}.$$

Agora estudaremos a representação da subcoleção $[A, W]^c$.

Proposição 3.2 *Seja A um subconjunto não vazio de W , então*

$$Max([A, W]^c) = \{[\emptyset, \{a\}^c] : a \in A\}.$$

Prova:

$$\begin{aligned} Max([A, W]^c) &= Max(\{X \in \mathcal{P}(W) : X \notin [A, W]\}) \\ &= Max(\{X \in \mathcal{P}(W) : A \not\subseteq X \text{ or } X \not\subseteq W\}) \\ &= Max(\{X \in \mathcal{P}(W) : A \not\subseteq X\}) \\ &= Max(\{X \in \mathcal{P}(W) : \exists a \in A : a \notin X\}) \\ &= Max(\cup\{[\emptyset, \{a\}^c] : a \in A\}) \\ &= \{[\emptyset, \{a\}^c] : a \in A\} \end{aligned}$$

A última passagem é válida, pois, para qualquer $a \in A$, $[\emptyset, \{a\}^c] \subseteq \cup\{[\emptyset, \{a\}^c] : a \in A\}$ e $[\emptyset, W]$, o único intervalo contido em $\mathcal{P}(W)$ que contém propriamente $[\emptyset, \{a\}^c]$, não está incluído em $[A, W]^c$.

□

Agora estudaremos a representação da subcoleção $[\emptyset, B]^c$.

Proposição 3.3 *Seja B um subconjunto de W , distinto de W , então*

$$Max([\emptyset, B]^c) = \{[\{b\}, W] : b \in B^c\}.$$

Prova:

$$\begin{aligned}
Max([\emptyset, B]^c) &= Max(\{X \in \mathcal{P}(W) : X \not\subseteq [\emptyset, B]\}) \\
&= Max(\{X \in \mathcal{P}(W) : \emptyset \not\subseteq X \text{ or } X \not\subseteq B\}) \\
&= Max(\{X \in \mathcal{P}(W) : X \not\subseteq B\}) \\
&= Max(\{X \in \mathcal{P}(W) : \exists b \in B^c : b \in X\}) \\
&= Max(\cup\{[\{b\}, W] : b \in B^c\}) \\
&= \{[\{b\}, W] : b \in B^c\}
\end{aligned}$$

A última passagem é válida, pois, para qualquer $b \in B^c$, $[\{b\}, W] \subseteq \cup\{[\{b\}, W] : b \in B^c\}$ e $[\emptyset, W]$, o único intervalo contido em $\mathcal{P}(W)$ que contém propriamente $[\{b\}, W]$, não está incluído em $[\emptyset, B]^c$.

□

Agora estudaremos a representação da subcoleção $[A, B]^c$.

Proposição 3.4 *Sejam A e B dois subconjuntos de W , tais que $A \subseteq B$, A distinto de vazio e B é distinto de W . O conjunto de intervalos maximais contidos em $[A, B]^c$ é fornecida pela seguinte expressão:*

$$Max([A, B]^c) = \{[\emptyset, \{a\}^c] : a \in A\} \cup \{[\{b\}, W] : b \in B^c\}.$$

Prova:

$$\begin{aligned}
Max([A, B]^c) &= Max((([A, W] \cap [\emptyset, B])^c) \\
&= Max([A, W]^c \cup [\emptyset, B]^c)
\end{aligned}$$

(pela lei de Morgan)

$$= Max(\cup\{[\emptyset, \{a\}^c] : a \in A\} \cup \cup\{[\{b\}, W] : b \in B^c\})$$

(pelas Proposições 3.2 e 3.3)

$$= \{[\emptyset, \{a\}^c] : a \in A\} \cup \{[\{b\}, W] : b \in B^c\}$$

A última passagem é válida, pois, para qualquer $a \in A$ e $b \in B^c$, $[\emptyset, \{a\}^c]$ e $[\{b\}, W]$ são elementos maximais contidos, respectivamente, em $[A, W]^c$ e $[\emptyset, B]^c$, e $[\emptyset, W]$, o único conjunto que é maior que $[\emptyset, \{a\}^c]$ ou $[\{b\}, W]$, não está incluído em $[A, B]^c$.

□

Obviamente, quando $[A, B] = \mathcal{P}(W)$ e $[A, B] = \emptyset$ temos, respectivamente, $\text{Max}([A, B]^c) = \{\emptyset\}$ e $\text{Max}([A, B]^c) = \{[\emptyset, W]\}$.

Aplicaremos o Teorema 3.1 e Proposição 3.4 para encontrar um procedimento para computar $\text{Max}(\mathcal{X}^c)$.

Teorema 3.2 *Sejam $\mathcal{X}$ uma coleção de intervalos contidos em $\mathcal{P}(W)$, então*

$$\overline{\text{Max}(\cup \mathcal{X})} = \cap \{ \text{Max}([A, B]^c) : [A, B] \in \mathcal{X} \}.$$

Prova:

$$\overline{\text{Max}(\cup \mathcal{X})} = \text{Max}((\cup \mathcal{X})^c)$$

(pelo isomorfismo de reticulados entre $(\mathcal{P}(\mathcal{P}(W)), \subseteq)$ e $(\Pi_W, \leq)$)

$$\begin{aligned} &= \text{Max}(\{X \in \mathcal{P}(W) : X \not\subseteq \cup \mathcal{X}\}) \\ &= \text{Max}(\{X \in \mathcal{P}(W) : X \not\subseteq [A, B], \forall [A, B] \in \mathcal{X}\}) \\ &= \text{Max}(\{X \in \mathcal{P}(W) : X \in [A, B]^c, \forall [A, B] \in \mathcal{X}\}) \\ &= \text{Max}(\cap \{ \{X \in \mathcal{P}(W) : X \in [A, B]^c\}, [A, B] \in \mathcal{X} \}) \\ &= \text{Max}(\cap \{ [A, B]^c : [A, B] \in \mathcal{X} \}) \\ &= \cap \{ \text{Max}([A, B]^c) : [A, B] \in \mathcal{X} \} \end{aligned}$$

(pelo isomorfismo de reticulados entre $(\mathcal{P}(\mathcal{P}(W)), \subseteq)$ e $(\Pi_W, \leq)$).

□

Note que no Teorema 3.2 não estamos assumindo que $\mathcal{X}$ seja maximal. Em seguida, fornecemos dois exemplos que ilustram a aplicação deste teorema.

Exemplo 3.2 Seja $W = \{(-1, 0), (0, 0), (1, 0)\}$ e seja

$$\mathcal{X} = \left\{ \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \right\}.$$

Neste caso, $\mathcal{X}$ é maximal, i.e., $\mathcal{X} = \text{Max}(\cup \mathcal{X})$.

Aplicando o Teorema 3.2, temos:

$$\begin{aligned} \overline{\mathcal{X}} &= \text{Max}(\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}^c) \cap \text{Max}(\begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}^c) \\ &= \left\{ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right\} \cap \left\{ \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \right\} \\ &= \text{Max}(\left\{ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \emptyset, \emptyset, \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right\}) \\ &= \left\{ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right\}. \end{aligned}$$

A Figura 3.2 mostra $\cup \mathcal{X}$ e $\overline{\mathcal{X}}$.

□

Figura 3.2: Complemento de uma subcoleção a partir de um conjunto maximal de intervalos. $\cup \mathcal{X}$ e $\overline{\mathcal{X}} = \{[000, 010], [101, 111]\}$.

No Exemplo 3.3, o Teorema 3.2 é aplicado a uma coleção não maximal de intervalos.

Exemplo 3.3 Seja $W = \{(-1, 0), (0, 0), (1, 0), (2, 0)\}$ e seja

$$\mathcal{X} = \{ [0000, 1100], [0000, 0011], [0010, 0111], [0100, 0110] \}.$$

Aplicando Teorema 3.2, temos

$$\begin{aligned} \overline{Max(\cup \mathcal{X})} &= Max([0000, 1100]^c) \sqcap Max([0000, 0011]^c) \sqcap \\ &\quad Max([0010, 0111]^c) \sqcap Max([0100, 0110]^c) \\ &= \{ [0010, 1111], [0001, 1111] \} \\ &\quad \sqcap \{ [1000, 1111], [0100, 1111] \} \\ &\quad \sqcap \{ [0000, 1101], [1000, 1111] \} \\ &\quad \sqcap \{ [0000, 1011], [1000, 1111], [0001, 1111] \} \\ &= Max(\{ [1010, 1111], [0110, 1111], [1001, 1111], \\ &\quad [0101, 1111] \}) \sqcap \{ [0000, 1101], [1000, 1111] \} \end{aligned}$$

$$\begin{aligned}
& \cap \{ [0000, 1011], [1000, 1111], [0001, 1111] \} \\
& = \{ [1010, 1111], [0110, 1111], [1001, 1111], \\
& \quad [0101, 1111] \} \cap \{ [0000, 1101], [1000, 1111] \} \\
& \cap \{ [0000, 1011], [1000, 1111], [0001, 1111] \} \\
& = \text{Max}(\{ \emptyset, [1010, 1111], \emptyset, [1110, 1111], [1001, 1101], \\
& \quad [1001, 1111], [0101, 1101], [1101, 1111] \}) \\
& \cap \{ [0000, 1011], [1000, 1111], [0001, 1111] \} \\
& = \{ [1010, 1111], [1001, 1111], [0101, 1101] \} \\
& \cap \{ [0000, 1011], [1000, 1111], [0001, 1111] \} \\
& = \text{Max}(\{ [1010, 1011] [1010, 1111], [1011, 1111], \\
& \quad [1001, 1011], [1001, 1111], [1001, 1111], \\
& \quad \emptyset, [1101, 1101], [0101, 1101] \}) \\
& = \{ [1010, 1111], [1001, 1111], [0101, 1101] \}.
\end{aligned}$$

As Figuras 3.3a e 3.3b mostram $\cup \mathcal{X}$ e $\overline{\text{Max}(\cup \mathcal{X})}$.

□

a)

b)

Figura 3.3: Complemento de uma subcoleção a partir de um conjunto não maximal de intervalos. a) $\cup \mathcal{X}$ e $\{[1010, 1111], [0101, 1101]\} \subseteq \overline{Max(\cup \mathcal{X})}$. b) $\cup \mathcal{X}$ e $\{[1001, 1111]\} \subseteq \overline{Max(\cup \mathcal{X})}$.

Agora estudaremos o problema de obter os intervalos maximais contidos em uma coleção de subconjuntos representados por uma coleção não maximal de intervalos.

Teorema 3.3 *Seja $\mathcal{X}$ uma coleção de intervalos, então*

$$Max(\cup \mathcal{X}) = \cap \{Max([A, B]^c) : [A, B] \in \cap \{Max([C, D]^c) : [C, D] \in \mathcal{X}\}\}.$$

Prova:

Uma vez que o operador $\bar{\cdot}$ é a operação de complemento em Π_W , temos que $Max(\cup \mathcal{X}) = \overline{Max(\cup \mathcal{X})}$. O resultado segue a partir do Teorema 3.2.

□

O Exemplo 3.4 ilustra a aplicação do Teorema 3.3.

Exemplo 3.4 *Seja $W = \{(-1, 0), (0, 0), (1, 0), (2, 0)\}$ e*

$$\mathcal{X} = \left\{ \left[\begin{array}{cccc} 1 & 1 & 0 & 0 \end{array}, \begin{array}{cccc} 1 & 1 & 1 & 1 \end{array} \right], \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \end{array}, \begin{array}{cccc} 1 & 1 & 0 & 1 \end{array} \right], \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \end{array}, \begin{array}{cccc} 1 & 0 & 1 & 1 \end{array} \right] \right\}.$$

Aplicamos o Teorema 3.3 para computar $Max(\cup \mathcal{X})$. Primeiro computamos $\overline{Max(\cup \mathcal{X})}$.

$$\begin{aligned} \overline{Max(\cup \mathcal{X})} &= Max\left(\left[\begin{array}{cccc} 1 & 1 & 0 & 0 \end{array}, \begin{array}{cccc} 1 & 1 & 1 & 1 \end{array}\right]^c\right) \cap Max\left(\left[\begin{array}{cccc} 1 & 0 & 0 & 0 \end{array}, \begin{array}{cccc} 1 & 1 & 0 & 1 \end{array}\right]^c\right) \\ &\quad \cap Max\left(\left[\begin{array}{cccc} 1 & 0 & 0 & 0 \end{array}, \begin{array}{cccc} 1 & 0 & 1 & 1 \end{array}\right]^c\right) \\ &= \left\{ \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \end{array}, \begin{array}{cccc} 0 & 1 & 1 & 1 \end{array}\right], \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \end{array}, \begin{array}{cccc} 1 & 0 & 1 & 1 \end{array}\right] \right\} \\ &\quad \cap \left\{ \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \end{array}, \begin{array}{cccc} 0 & 1 & 1 & 1 \end{array}\right], \left[\begin{array}{cccc} 0 & 0 & 1 & 0 \end{array}, \begin{array}{cccc} 1 & 1 & 1 & 1 \end{array}\right] \right\} \\ &\quad \cap \left\{ \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \end{array}, \begin{array}{cccc} 0 & 1 & 1 & 1 \end{array}\right], \left[\begin{array}{cccc} 0 & 1 & 0 & 0 \end{array}, \begin{array}{cccc} 1 & 1 & 1 & 1 \end{array}\right] \right\} \\ &= Max\left(\left\{ \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \end{array}, \begin{array}{cccc} 0 & 1 & 1 & 1 \end{array}\right], \left[\begin{array}{cccc} 0 & 0 & 1 & 0 \end{array}, \begin{array}{cccc} 0 & 1 & 1 & 1 \end{array}\right], \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \end{array}, \begin{array}{cccc} 0 & 0 & 1 & 1 \end{array}\right], \right. \right. \\ &\quad \left. \left[\begin{array}{cccc} 0 & 0 & 1 & 0 \end{array}, \begin{array}{cccc} 1 & 0 & 1 & 1 \end{array}\right] \right\}\right) \cap \left\{ \left[\begin{array}{cccc} 0 & 0 & 0 & 0 \end{array}, \begin{array}{cccc} 0 & 1 & 1 & 1 \end{array}\right], \left[\begin{array}{cccc} 0 & 1 & 0 & 0 \end{array}, \begin{array}{cccc} 1 & 1 & 1 & 1 \end{array}\right] \right\} \end{aligned}$$

Figura 3.4: O intervalo $[1000, 1111] = \cup \mathcal{X}$ e a coleção unitária $\{[0000, 0111]\}$.

$$\begin{aligned}
 &= \{ [0000, 0111], [0010, 1011] \} \\
 &\sqcap \{ [0000, 0111], [0100, 1111] \} \\
 &= \text{Max}(\{ [0000, 0111], [0100, 0111], [0010, 0011], \emptyset \}) \\
 &= \{ [0000, 0111] \}.
 \end{aligned}$$

Logo, computamos $\overline{\overline{\text{Max}(\cup \mathcal{X})}}$:

$$\overline{\overline{\text{Max}(\cup \mathcal{X})}} = \text{Max}([0000, 0111]^c) = \{ [1000, 1111] \}.$$

A Figura 3.4 mostra $\cup \mathcal{X}$ e $\text{Max}(\cup \mathcal{X})$.

□

Em seguida, mostramos como o Teorema 3.3 pode ser aplicado para computar o supremo de duas coleções de intervalos maximais.

Corolário 3.1 *Sejam $\mathcal{X}$ e $\mathcal{Y}$ elementos de Π_W , então*

$$\mathcal{X} \sqcup \mathcal{Y} = \sqcap \{ \text{Max}([A, B]^c) : [A, B] \in \sqcap \{ \text{Max}([C, D]^c) : [C, D] \in \mathcal{X} \cup \mathcal{Y} \} \}.$$

Prova:

Este corolário é uma consequência imediata do Teorema 3.3.

□

Corolário 3.2 *Sejam $\mathcal{X}$ e $\mathcal{Y}$ elementos de Π_W , então*

$$\mathcal{X} \sqcup \mathcal{Y} = \overline{\overline{\mathcal{X}} \sqcap \overline{\mathcal{Y}}}.$$

Prova:

Este corolário é a lei de Morgan para o reticulado completo Booleano $(\Pi_W, \leq)$.

□

O Corolário 3.2 constitui um caminho alternativo para computar a operação $\sqcup$ sobre $(\Pi_W, \leq)$. Exemplo 3.5 ilustra este caminho alternativo para computar $\sqcup$.

Exemplo 3.5 *Seja $E = \{(-1, 0), (0, 0), (1, 0)\}$ e sejam*

$$\mathcal{X} = \left\{ \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix} \right\}$$

e

$$\mathcal{Y} = \left\{ \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right\}.$$

A Figura 3.5a mostra $\cup \mathcal{X}$ a $\mathcal{X}$, enquanto a Figure 3.5b mostra $\cup \mathcal{Y}$ e $\mathcal{Y}$.

Aplicamos o Corolário 3.2 para computar $\mathcal{X} \sqcup \mathcal{Y}$. Primeiro computamos $\overline{\mathcal{X}}$.

$$\begin{aligned}
\overline{\mathcal{X}} &= \text{Max}([100, 110]^c) \cap \text{Max}([010, 110]^c) \\
&= \{ [000, 011], [001, 111] \} \cap \{ [000, 101], [001, 111] \} \\
&= \text{Max}(\{ [000, 001], [001, 011], [001, 101], [001, 111] \}) \\
&= \{ [000, 001], [001, 111] \}.
\end{aligned}$$

Agora computamos $\overline{\mathcal{Y}}$.

$$\begin{aligned}
\overline{\mathcal{Y}} &= \text{Max}([100, 101]^c) \cap \text{Max}([101, 111]^c) \cap \text{Max}([011, 111]^c) \\
&= \{ [000, 011], [010, 111] \} \cap \{ [000, 110], [000, 011] \} \\
&\quad \cap \{ [000, 101], [000, 110] \} \\
&= \text{Max}(\{ [000, 010], [000, 011], [010, 110], [010, 011] \}) \\
&\quad \cap \{ [000, 101], [000, 110] \} \\
&= \{ [000, 011], [010, 110] \} \cap \{ [000, 101], [000, 110] \} \\
&= \text{Max}(\{ [000, 001], [000, 010], \emptyset, [010, 110] \}) \\
&= \{ [000, 001], [000, 010], [010, 110] \}
\end{aligned}$$

Agora computamos $\overline{\mathcal{X}} \cap \overline{\mathcal{Y}}$.

$$\begin{aligned}
\overline{\mathcal{X}} \cap \overline{\mathcal{Y}} &= \{ [000, 001], [001, 111] \} \cap \{ [000, 001], [000, 010], \\
&\quad [010, 110] \} \\
&= \text{Max}(\{ [000, 001], [000, 000], \emptyset, [001, 001], \emptyset, \emptyset \}) \\
&= \{ [000, 001] \}.
\end{aligned}$$

Finalmente, computamos $\overline{\overline{\mathcal{X}} \cap \overline{\mathcal{Y}}}$.

Figura 3.5: Computação de $\mathcal{X} \sqcup \mathcal{Y}$ usando a lei de Morgan. a) $\mathcal{X}$ e $\cup \mathcal{X}$. b) $\mathcal{Y}$ e $\cup \mathcal{Y}$. c) $\cup(\mathcal{X} \sqcup \mathcal{Y})$ e $[100.111] \in \mathcal{X} \sqcup \mathcal{Y}$. d) $\cup(\mathcal{X} \sqcup \mathcal{Y})$ e $[010.111] \in \mathcal{X} \sqcup \mathcal{Y}$.

$$\overline{\mathcal{X} \sqcap \mathcal{Y}} = \cap \{ \text{Max}([000, 001]^c) \} = \{ [100, 111], [010, 111] \}.$$

As Figuras 3.5c e 3.5d mostram $\cup(\mathcal{X} \sqcup \mathcal{Y})$ e $\mathcal{X} \sqcup \mathcal{Y}$.

□

A figura 3.6 ilustra as duas possíveis maneiras de computar a operação $\mathcal{X} \sqcup \mathcal{Y}$.

Figura 3.6: Duas maneiras alternativas para computar a operação $\mathcal{X} \sqcup \mathcal{Y}$. a) Corolário 3.1. b) Corolário 3.2.

Capítulo 4

Decomposição canônica de operadores de conjuntos

Suponha que o conjunto E é um *grupo Abelian* com respeito a uma operação binária, denotada por $+$. O elemento zero de $(E, +)$ é denotado por o . Este elemento é também chamado origem de E e é representado por um caractere em negrito na descrição de um subconjunto de E por uma sequência de 0's e 1's. Por exemplo, seja E o conjunto $\{(-1, 0), (0, 0), (1, 0)\}$ e sejam X e Y , respectivamente, os subconjuntos $\{(-1, 0), (0, 0)\}$ e $\{(-1, 0), (1, 0)\}$. Os subconjuntos X e Y serão representados, respectivamente, pelas sequências 110 e 101.

Seja X^t o transposto de um subconjunto X , i.e., $X^t = \{y \in E : y = -x, x \in X\}$.

Para qualquer $h \in E$ e $X \subseteq E$, o conjunto $X + h = \{x \in E : x - h \in X\}$ chama-se o translado de X por h . Em particular, $X_o = X$.

Um *operador de conjunto* é qualquer mapeamento definido de $\mathcal{P}(E)$ em $\mathcal{P}(E)$. O conjunto Ψ de todos os operadores de $\mathcal{P}(E)$ em $\mathcal{P}(E)$ herda a estrutura de reticulado completo de $(\mathcal{P}(E), \subseteq)$: $\forall \psi_1, \psi_2 \in \Psi$,

$$\psi_1 \leq \psi_2 \iff \psi_1(X) \subseteq \psi_2(X) \quad (X \in \mathcal{P}(E)).$$

O supremo e ínfimo de um subconjunto Θ do reticulado completo $(\Psi, \leq)$ verificam

$$(\vee \Theta)(X) = \cup \{\theta(X) : \theta \in \Theta\} \quad (X \in \mathcal{P}(E))$$

e

$$(\wedge \Theta)(X) = \cap \{\theta(X) : \theta \in \Theta\} \quad (X \in \mathcal{P}(E)),$$

respectivamente.

Um operador de conjunto ψ é chamado *invariante por translação* (i.t.) se e só se, $\forall h \in E$,

$$\psi(X + h) = \psi(X) + h \quad (X \in \mathcal{P}(E)).$$

Seja W um subconjunto finito de E . Um operador de conjunto ψ chama-se *localmente definido dentro de uma janela* ou *operador de janela* W se e só se, $\forall h \in E$,

$$h \in \psi(X) \iff h \in \psi(X \cap (W + h)) \quad (X \in \mathcal{P}(E)).$$

Denotaremos Ψ_W a coleção de operadores i.t. localmente definidos dentro de uma janela W . O par $(\Psi_W, \leq)$ constitui um subreticulado do reticulado $(\Psi, \leq)$.

O *Núcleo* $K_W(\psi)$ de um operador de conjunto ψ i.t. e localmente definido dentro de uma janela W é a subcoleção de $\mathcal{P}(W)$ definida por

$$K_W(\psi) = \{X \in \mathcal{P}(W) : o \in \psi(X)\}.$$

Proposição 4.1 *O mapeamento K_W de Ψ_W a $\mathcal{P}(\mathcal{P}(W))$ definido por*

$$K_W(\psi) = \{X \in \mathcal{P}(W) : o \in \psi(X)\} \quad (\psi \in \Psi_W)$$

constitui um isomorfismo de reticulado entre os reticulados $(\Psi_W, \leq)$ e $(\mathcal{P}(\mathcal{P}(W)), \subseteq)$. O inverso do mapeamento K_W é o mapeamento K_W^{-1} , definido por

$$K_W^{-1}(\mathcal{X})(X) = \{x \in E : (X - x) \cap W \in \mathcal{X}\} \quad (X \in \mathcal{P}(E)).$$

Prova:

Vamos mostrar que $\mathcal{K}_W$ é uma bijeção entre Ψ_W e $\mathcal{P}(\mathcal{P}(W))$. Em primeiro lugar, provaremos que $\mathcal{K}_W$ é injetora. Para todo $\psi \in \Psi_W$ temos

$$\begin{aligned}
 \mathcal{K}_W^{-1}(\mathcal{K}_W(\psi))(X) &= \{x \in E : (X - x) \cap W \in \mathcal{K}_W(\psi)\} && \text{(pela definição de } \mathcal{K}_W^{-1}) \\
 &= \{x \in E : o \in \psi((X - x) \cap W)\} && \text{(pela definição de } \mathcal{K}_W) \\
 &= \{x \in E : o \in \psi((X - x) \cap (W + o))\} \\
 &= \{x \in E : o \in \psi(X - x)\} && \text{(porque } \psi \text{ é localmente definido)} \\
 &= \{x \in E : x \in \psi(X)\} && \text{(porque } \psi \text{ é i.t.)} \\
 &= \psi(X),
 \end{aligned}$$

i.e., $\psi = \mathcal{K}_W^{-1}(\mathcal{K}_W(\psi))$ e $\mathcal{K}_W$ é injetora.

Agora, provaremos que $\mathcal{K}_W$ é sobrejetora. Para todo $\mathcal{X} \in \mathcal{P}(\mathcal{P}(W))$ temos

$$\begin{aligned}
 \mathcal{K}_W(\mathcal{K}_W^{-1}(\mathcal{X})) &= \{X \in \mathcal{P}(W) : o \in \mathcal{K}_W^{-1}(\mathcal{X})(X)\} && \text{(pela definição de } \mathcal{K}_W) \\
 &= \{X \in \mathcal{P}(W) : o \in \{x \in E : (X - x) \cap W \in \mathcal{X}\}\} && \text{(pela definição de } \mathcal{K}_W^{-1}) \\
 &= \{X \in \mathcal{P}(W) : X \cap W \in \mathcal{X}\} \\
 &= \{X \in \mathcal{P}(W) : X \in \mathcal{X}\} \\
 &= \mathcal{X},
 \end{aligned}$$

i.e., $\mathcal{X} = \mathcal{K}_W(\mathcal{K}_W^{-1}(\mathcal{X}))$ e, então, $\mathcal{K}_W$ é sobrejetora. Assim, $\mathcal{K}_W$ é injetora e sobrejetora, conseqüentemente, é bijetora.

Agora, provaremos que $\mathcal{K}_W$ preserva a ordem parcial.

$$\begin{aligned}
 \psi_1 \leq \psi_2 &\iff \forall X \in \mathcal{P}(E), \psi_1(X) \subseteq \psi_2(X) \\
 &\iff \forall X \in \mathcal{P}(E), x \in \psi_1(X) \implies x \in \psi_2(X)
 \end{aligned}$$

$$\iff \forall X \in \mathcal{P}(E), o \in \psi_1(X - x) \implies o \in \psi_2(X - x)$$

(porque ψ_1 e ψ_2 são i.t.)

$$\iff \forall X \in \mathcal{P}(E), o \in \psi_1((X - x) \cap W) \implies o \in \psi_2((X - x) \cap W)$$

(porque ψ_1 e ψ_2 são localmente definidos)

$$\iff \forall X \in \mathcal{P}(E), (X - x) \cap W \in \mathcal{K}_W(\psi_1) \implies (X - x) \cap W \in \mathcal{K}_W(\psi_2)$$

(pela definição de $\mathcal{K}_W$)

$$\iff \forall Y \in \mathcal{P}(W), Y \in \mathcal{K}_W(\psi_1) \implies Y \in \mathcal{K}_W(\psi_2)$$

$$\iff \mathcal{K}_W(\psi_1) \subseteq \mathcal{K}_W(\psi_2).$$

Portanto,

$$\psi_1 \leq \psi_2 \iff \mathcal{K}_W(\psi_1) \subseteq \mathcal{K}_W(\psi_2), \quad \forall \psi_1, \psi_2 \in \Psi_W.$$

Então, $\mathcal{K}_W$ é uma bijeção que preserva a ordem parcial e, conseqüentemente, é um isomorfismo de reticulado.

□

Como uma consequência da Proposição 4.1, decorrem as seguintes igualdades: $\forall \psi_1, \psi_2 \in \Psi_W$,

$$\mathcal{K}_W(\psi_1 \wedge \psi_2) = \mathcal{K}(\psi_1) \cap \mathcal{K}(\psi_2)$$

e

$$\mathcal{K}_W(\psi_1 \vee \psi_2) = \mathcal{K}(\psi_1) \cup \mathcal{K}(\psi_2).$$

Os operadores de conjuntos ι e ν definidos por

$$\iota(X) = \{x \in E : x \in X\} \quad (X \in \mathcal{P}(E))$$

e

$$\nu(X) = \{x \in E : x \notin X\} \quad (X \in \mathcal{P}(E)).$$

são chamados, respectivamente, identidade e negação.

Seja $A, B \in \mathcal{P}(E)$. As operações

$$A \oplus B = \cup_{b \in B} A + b \text{ e } A \ominus B = \cap_{b \in B} A - b$$

são chamadas, respectivamente, *adição e subtração de Minkowski*.

Seja $B \in \mathcal{P}(W)$. Os operadores de conjuntos δ_B e ε_B definidos por

$$\delta_B(X) = X \oplus B \quad (X \in \mathcal{P}(E))$$

e

$$\varepsilon_B(X) = X \ominus B \quad (X \in \mathcal{P}(E))$$

são chamados, respectivamente, *dilatação e erosão* relativamente por B . O parâmetro B , que caracteriza uma dilatação ou uma erosão, chama-se elemento estruturante. A dilatação e a erosão por B são operadores i.t..

Seja $A, B \in \mathcal{P}(W)$, tal que $A \subseteq B$. Os operadores de conjuntos $\lambda_{[A,B]}^W$ e $\mu_{[A,B]}^W$ definidos por

$$\lambda_{[A,B]}^W(X) = \{x \in E : A \subseteq (X - x) \cap W \subseteq B\} \quad (X \in \mathcal{P}(E))$$

e

$$\mu_{[A,B]}^W(X) = \{x \in E : (X - x) \cap A^t \neq \emptyset \text{ ou } ((X - x) \cap W^t) \cup B^t \neq W^t\} \quad (X \in \mathcal{P}(E)),$$

são chamados, respectivamente, operador sup-gerador e inf-gerador.

Note que estes operadores são i.t. e localmente definidos, respectivamente, dentro de W e W^t . Os operadores sup-geradores e inf-geradores podem ser decompostos em termos de erosões e dilatações, respectivamente, por

$$\lambda_{[A,B]}^W(X) = \varepsilon_A(X) \cap \nu \delta_{B^{tc}}(X) \quad (X \in \mathcal{P}(E))$$

e

$$\mu_{[A,B]}^W(X) = \delta_A(X) \cup \nu \varepsilon_{B^{ct}}(X) \quad (X \in \mathcal{P}(E)),$$

onde o complemento de B é tomado relativamente a W .

Proposição 4.2 *O Núcleo de um operador sup-gerador localmente definido dentro de uma janela W é um intervalo, isto é,*

$$\mathcal{K}_W(\lambda_{[A,B]}^W) = [A, B].$$

prova:

$$\begin{aligned} \mathcal{K}(\lambda_{[A,B]}^W) &= \{X \in \mathcal{P}(W) : o \in \lambda_{[A,B]}^W(X)\} \\ &= \{X \in \mathcal{P}(W) : o \in \{x \in E : A + x \subseteq X \cap (W + x) \subseteq B + x\}\} \\ &= \{X \in \mathcal{P}(W) : A \subseteq X \cap W \subseteq B\} \\ &= [A, B]. \end{aligned}$$

□

Teorema 4.1 *Seja ψ um operador i.t. localmente definido dentro de uma janela W , então*

$$\psi(X) = \bigcup \{ \lambda_{[A,B]}^W(X) : [A, B] \subseteq \mathcal{K}_W(\psi) \} \quad (X \in \mathcal{P}(E)).$$

Prova:

$$\mathcal{K}_W(\psi) = \bigcup \{ [A, B] \subseteq \mathcal{P}(W) : [A, B] \subseteq \mathcal{K}_W(\psi) \}$$

(porque qualquer subconjunto de um reticulado completo pode ser construído pela união de seus intervalos)

$$\mathcal{K}_W(\psi) = \bigcup \{ \mathcal{K}_W(\lambda_{[A,B]}^W) : [A, B] \subseteq \mathcal{K}_W(\psi) \}$$

(pela Proposição 4.2)

$$\psi = \bigvee \{ \lambda_{[A,B]}^W : [A, B] \subseteq \mathcal{K}_W(\psi) \}$$

(já que $\mathcal{K}_W$ é um isomorfismo de reticulados entre $(\Psi_W, \leq)$ e $(\mathcal{P}(\mathcal{P}(W)), \subseteq)$).

□

O operador dual do operador ψ , denotado por ψ^* , é

$$\psi^* = \nu\psi\nu.$$

Um exemplo de operadores duais são $\lambda_{[A,B]}^W$ e $\mu_{[A^t,B^t]}^{W^t}$, isto é, $\mu_{[A^t,B^t]}^{W^t} = (\lambda_{[A,B]}^W)^*$.

Teorema 4.2 *Seja ψ um operador i.t. localmente definido dentro da janela W , então*

$$\psi(X) = \bigcap \{ \mu_{[A^t,B^t]}^{W^t}(X) : [A, B] \subseteq \mathcal{K}_W(\psi^*) \} \quad (X \in \mathcal{P}(E)).$$

Prova:

Pelo Teorema 4.1, temos:

$$\psi^*(X) = \bigcup \{ \lambda_{[A,B]}^W(X) : [A, B] \subseteq \mathcal{K}_W(\psi^*) \} \quad (X \in \mathcal{P}(E)).$$

Pela definição de ψ^* temos:

$$\psi = \nu \psi^* \nu.$$

Portanto, $\forall X \in \mathcal{P}(E)$,

$$\begin{aligned} \psi(X) &= \nu(\bigcup \{ \lambda_{[A,B]}^W \nu(X) : [A, B] \subseteq \mathcal{K}_W(\psi^*) \}) \\ &= \bigcap \{ \nu \lambda_{[A,B]}^W \nu(X) : [A, B] \subseteq \mathcal{K}_W(\psi^*) \} \\ &= \bigcap \{ \mu_{[A^t, B^t]}^{W^t}(X) : [A, B] \subseteq \mathcal{K}_W(\psi^*) \}. \end{aligned}$$

□

As decomposições dos Teoremas 4.1 e 4.2 são chamadas, respectivamente, sup-decomposição canônica e inf-decomposição canônica.

Contudo estas decomposições podem conduzir a representações computacionalmente ineficientes para a maioria dos operadores de conjuntos i.t., no sentido que uma família menor de operadores sup-geradores ou inf-geradores pode ser suficiente para representar os mesmos operadores.

O conjunto $\mathbf{B}_W(\psi)$ de todos os intervalos maximais contidos em $\mathcal{K}_W(\psi)$ é chamado a *base* de ψ , i.e., $\mathbf{B}_W(\psi) = \text{Max}(\mathcal{K}_W(\psi))$.

A base do operador sup-gerador $\lambda_{[A,B]}^W$ é a coleção unitária $\{[A, B]\}$. Particularmente, a base da erosão ε_A é $\{[A, W]\}$ e a base da identidade é $\{[\{o\}, W]\}$.

A base da dilatação δ_A é a coleção $\{[\{a\}, W] : a \in A^t\}$.

Teorema 4.3 *Seja ψ um operador i.t. localmente definido dentro de uma janela W , então*

$$\psi(X) = \bigcup \{ \lambda_{[A,B]}^W(X) : [A, B] \in \mathbf{B}_W(\psi) \} \quad (X \in \mathcal{P}(E)).$$

Prova:

Como W é finito, podemos construir $\mathcal{K}_W(\psi)$ em termos de seus intervalos maximais, i.e.,

$$\mathcal{K}_W(\psi) = \bigcup \{ [A, B] \subseteq \mathcal{P}(W) : [A, B] \in \mathbf{B}_W(\psi) \}.$$

O resultado segue pelos mesmos argumentos usados para provar o Teorema 4.1. □

Como uma consequência do Teorema 4.3, a decomposição do Teorema 4.2 pode também ser simplificada.

Teorema 4.4 *Seja ψ um operador i.t. localmente definido dentro de uma janela W , então*

$$\psi(X) = \bigcap \{ \mu_{[A^t, B^t]}^{W^t}(X) : [A, B] \in \mathbf{B}_W(\psi^*) \} \quad (X \in \mathcal{P}(E))$$

□

Prova:

Este resultado pode ser provado pelos mesmos argumentos usados na prova do Teorema 4.2, com o Teorema 4.3 no lugar do Teorema 4.1. □

Vimos que os operadores de conjuntos i.t. e localmente definidos podem ser representados por seus Núcleo ou por suas bases. Estudaremos, agora, uma terceira forma de representar estes operadores: a equivalência com funções Booleanas.

Seja T o mapeamento entre Ψ_W e $\{0, 1\}^{\mathcal{P}(W)}$ definido por

$$T(\psi)(X) = \begin{cases} 1 & \text{se } o \in \psi(X) \\ 0 & \text{c.c.} \end{cases} \quad (X \in \mathcal{P}(W)).$$

O mapeamento T constitui um isomorfismo de reticulado entre os reticulados completos $(\Psi_W, \leq)$ e $(\{0, 1\}^{\mathcal{P}(W)}, \leq)$ e sua inversa T^{-1} é definida por

$$T^{-1}(f)(X) = \{x \in E : f((X - x) \cap W) = 1\} \quad (X \in \mathcal{P}(E)).$$

Devemos notar que por trás de todos estes resultados de decomposição estão os isomorfismos entre os reticulados $(\Psi_W, \leq)$, $(\mathcal{P}(\mathcal{P}(W)), \subseteq)$, $(\Pi_W, \leq)$ e $(\{0, 1\}^{\mathcal{P}(W)}, \leq)$. A Figura 4.1 apresenta um diagrama que descreve graficamente estes quatro isomorfismos básicos.

A Figura 4.2 ilustra a teoria de decomposição apresentada neste capítulo, mostrando a representação do detector de bordas verticais por sua função Booleana, Núcleo e Base. É dado também uma aplicação deste operador a uma imagem particular.

Finalmente, devemos observar que se $W = E$, a classe dos operadores i.t. e localmente definidos coincide com a classe dos operadores i.t. e, sob estas condições, os resultados de decomposição apresentados neste capítulo reduzem-se aos apresentados por Banon e Barrera [3].

Figura 4.1: Os quatro isomorfismos de reticulados básicos.

Figura 4.2: Três representações equivalentes de um operador de conjuntos ψ i.t. e localmente definido, que realiza a detecção de bordas verticais. O subconjunto W é a janela horizontal de três pontos, centrada no origem. a) X e $\psi(X)$. b) f_ψ . c) $\mathcal{K}_W(\psi)$. d) $B_W(\psi)$.

Capítulo 5

Composição de operadores na forma canônica

Neste capítulo, usaremos os resultados dos capítulos anteriores para mostrar como achar a base da interseção, da união e da composição com uma erosão (com uma dilatação ou com a complementação) de operadores dados, a partir das bases destes operadores.

Os resultados apresentados nesta seção e na próxima estendem o trabalho de Jones [12, 13].

Seja $\mathcal{X}$ uma subcoleção de $\mathcal{P}(W)$. Denotaremos por $\mathcal{X} + h$ a coleção construída pela translação dos elementos de $\mathcal{X}$ pelo vetor h , i.e.,

$$\mathcal{X} + h = \{X \in \mathcal{P}(W + h) : X - h \in \mathcal{X}\}.$$

Particularmente, $[A, B] + h$ denota a translação de todos os elementos do intervalo $[A, B]$ pelo vetor h , i.e.,

$$[A, B] + h = \{X \in \mathcal{P}(W + h) : X - h \in [A, B]\}.$$

Seja $\mathcal{X}$ uma coleção de intervalos contidos em $\mathcal{P}(W)$. Denotamos por $\mathcal{X} + h$ a coleção de intervalos construídos pela translação dos elementos de $\mathcal{X}$ pelo vetor h , i.e.,

$$\mathcal{X} + h = \{[A, B] \subseteq \mathcal{P}(W + h) : [A, B] - h \in \mathcal{X}\}.$$

Seja $\mathcal{X}$ uma subcoleção de $\mathcal{P}(W)$, com $\mathcal{X} = \text{Max}(\mathcal{X})$. Para $h \in E$, temos:

$$\mathcal{X} + h = \cup(\mathcal{X} + h) = (\cup \mathcal{X}) + h.$$

Agora, estudaremos algumas propriedades de composições de operadores localmente definidos dentro de janelas distintas.

Proposição 5.1 *Se um operador de conjunto é localmente definido dentro de uma janela W , então ele é localmente definido dentro de qualquer janela $W' \supseteq W$.*

Prova:

Seja $W' \supseteq W$. Para qualquer $X \in \mathcal{P}(E)$ e $h \in E$,

$$h \in \psi(X \cap (W' + h)) \iff h \in \psi((X \cap (W' + h)) \cap (W + h))$$

(por que ψ é localmente definido dentro de W)

$$\iff h \in \psi(X \cap (W' + h) \cap (W + h))$$

$$\iff h \in \psi(X \cap (W + h))$$

(porque $W \subseteq W'$)

$$\iff h \in \psi(X)$$

(porque ψ é localmente definido dentro de W)

Assim, ψ é localmente definido dentro de W' .

□

Note que para mudar a representação da base de um operador ψ localmente definido dentro de uma janela W , desta janela para outra janela $W' \supseteq W$, é suficiente completar a extremidade direita dos intervalos com o complemento de W relativo a W' , i.e.,

$$B_{W'}(\psi) = \{[A', B'] \subseteq \mathcal{P}(W') : A' = A \text{ e } B' = B \cup W^c, [A, B] \in B_W(\psi)\}.$$

Observe que, $B_W(\psi)$ e $B_{W'}(\psi)$ são representações equivalentes de ψ . Note que para cada $[A', B'] \in B_{W'}(\psi)$, existe $[A, B] \in B_W(\psi)$ tal que $A = A'$ e $\{x \in W : x \notin B\} = \{x \in W' : x \notin B'\}$. O próximo exemplo ilustra esta mudança de representação para o operador apresentado na Figura 4.2.

Exemplo 5.1 *Seja $W = 111$ e $W' = \begin{smallmatrix} & 1 \\ 1 & 1 & 1 \\ & 1 \end{smallmatrix}$. Seja ψ o operador que realiza a detecção de bordas verticais. A base de ψ em W e W' são, respectivamente,*

$$B_W(\psi) = \left\{ \begin{bmatrix} 010 & , & 110 \end{bmatrix}, \begin{bmatrix} 010 & , & 011 \end{bmatrix} \right\}$$

e

$$B_{W'}(\psi) = \left\{ \begin{bmatrix} 0 & , & 1 \\ 010 & , & 110 \\ 0 & , & 1 \end{bmatrix}, \begin{bmatrix} 0 & , & 1 \\ 010 & , & 011 \\ 0 & , & 1 \end{bmatrix} \right\}.$$

□

Proposição 5.2 *Se ψ_1 e ψ_2 são dois operadores localmente definidos, respectivamente, dentro das janelas, W_1 e W_2 , então os operadores $\psi_1 \wedge \psi_2$ e $\psi_1 \vee \psi_2$ são localmente definidos dentro da janela $W_1 \cup W_2$.*

Prova:

$$\forall h \in E \text{ e } X \in \mathcal{P}(E),$$

$$h \in (\psi_1 \wedge \psi_2)(X) \iff h \in \psi_1(X) \cap \psi_2(X)$$

$$\iff h \in \psi_1(X) \text{ e } h \in \psi_2(X)$$

$$\iff h \in \psi_1(X \cap (W_1 + h)) \text{ e } h \in \psi_2(X \cap (W_2 + h))$$

(ψ_1 e ψ_2 são operadores localmente definidos, respectivamente, dentro de W_1 e W_2)

$$\iff h \in \psi_1(X \cap ((W_1 \cup W_2) + h)) \text{ e } h \in \psi_2(X \cap ((W_1 \cup W_2) + h))$$

(ψ_1 e ψ_2 são operadores localmente definidos dentro de qualquer $W \supseteq W_1 \cup W_2$)

$$\iff h \in (\psi_1 \wedge \psi_2)(X \cap ((W_1 \cup W_2) + h)).$$

Assim, $\psi_1 \wedge \psi_2$ é localmente definido dentro de $W_1 \cup W_2$.

Por argumentos similares, provaremos o mesmo resultado para $\psi_1 \vee \psi_2$.

□

Para qualquer $u \in E$, o operador de conjunto τ_u , definido por

$$\tau_u(X) = X + u \quad (X \in \mathcal{P}(E)),$$

é chamado *operador de translação por u* .

Proposição 5.3 *Se ψ é um operador localmente definido dentro de uma janela W e, para qualquer $u \in E$, τ_u é o operador de translação por u , $\tau_u\psi$ é um operador localmente definido dentro da janela $W - u$.*

Prova:

$\forall h \in E$ e $X \in \mathcal{P}(E)$,

$$h \in \tau_u\psi(X) \iff h \in \psi(X) + u$$

$$\iff h - u \in \psi(X)$$

$$\iff h - u \in \psi(X \cap (W + (h - u)))$$

(ψ é localmente definido dentro de W)

$$\iff h \in \psi(X \cap ((W - u) + h)) + u$$

$$\iff h \in \tau_u\psi(X \cap ((W - u) + h))$$

(pela definição de τ_u).

Assim, $\tau_u\psi$ é localmente definido dentro da janela $W - u$.

□

Proposição 5.4 *Se ψ é um operador localmente definido dentro de uma janela W , então os operadores $\delta_B\psi$ e $\varepsilon_B\psi$ são localmente definidos, respectivamente, dentro das janelas $W \oplus B^t$ e $W \oplus B$.*

Prova:

$\forall h \in E$ e $X \in \mathcal{P}(E)$,

$$\begin{aligned} h \in \varepsilon_B\psi(X) &\iff h \in \cap_{b \in B} \psi(X) - b \\ &\iff h \in (\wedge_{b \in B} \tau_{-b}\psi)(X) \\ &\iff h \in (\wedge_{b \in B} \tau_{-b}\psi)(X \cap ((\cup_{b \in B} W + b) + h)) \end{aligned}$$

(pelas Proposições 5.2 e 5.3)

$$\begin{aligned} &\iff h \in (\wedge_{b \in B} \tau_{-b}\psi)(X \cap ((W \oplus B) + h)) \\ &\iff h \in \cap_{b \in B} \psi(X \cap ((W \oplus B) + h)) - b \\ &\iff h \in \varepsilon_B\psi(X \cap ((W \oplus B) + h)) \end{aligned}$$

Assim, $\varepsilon_B\psi$ é localmente definido dentro da janela $W \oplus B$.

Uma prova similar pode ser feita para $\delta_B\psi$.

□

Note que os operadores ν e ι são operadores neutros em relação ao tamanho da janela, isto é, se ψ é um operador localmente definido dentro da janela W então $\nu\psi$, $\psi\nu$, $\iota\psi$, e $\psi\iota$ são também localmente definidos dentro de W .

Corolário 5.1 *Se ψ_1 e ψ_2 são dois operadores localmente definidos, respectivamente, dentro das janelas W_1 e W_2 , então o operador $\psi_2\psi_1$ é um operador localmente definido dentro da janela $W_1 \oplus W_2$.*

Prova:

Seja $[A, B] \in \mathbf{B}_{W_2}(\psi_2)$. O operador $\varepsilon_A \psi_1$ é localmente definido dentro da janela $W_1 \oplus A$ (pela Proposição 5.4), então ele é também localmente definido dentro da janela $W_1 \oplus W_2$, pois $A \subseteq W_2$. O operador $\nu \delta_{B^{ct}} \psi_1$ é localmente definido dentro de $W_1 \oplus (B^{ct})^t = W_1 \oplus B^c$ (pela Proposição 5.4 e porque ν é neutro em relação a janela). então ele é também localmente definido dentro da janela $W_1 \oplus W_2$, pois $B^c \subseteq W_2$. Consequentemente, o operador $\varepsilon_A \psi_1 \wedge \nu \delta_{B^{ct}} \psi_1$ também é localmente definido dentro da janela $W_1 \oplus W_2$. Assim, o operador $\psi_2 \psi_1 = \vee \{ \varepsilon_A \psi_1 \wedge \nu \delta_{B^{ct}} \psi_1 : [A, B] \in \mathbf{B}_{W_2}(\psi_2) \}$ é também localmente definido dentro da janela $W_1 \oplus W_2$.

□

Note que a Proposição 5.4 e o Corolário 5.1 fornecem condições de pior caso para o aumento da janela na composição de operadores. Pode acontecer que a composição de dois operadores específicos localmente definidos dentro da mesma janela produzem um novo operador morfológico que também depende desta janela. Um exemplo deste fato é a seguinte composição: $(\delta_A \varepsilon_A)(\delta_A \varepsilon_A)$.

Em seguida, mostraremos como computar a base do ínfimo de dois operadores de conjuntos i.t., a partir das bases destes operadores.

Proposição 5.5 *Se ψ_1 e ψ_2 são dois operadores de conjuntos i.t. e localmente definidos, respectivamente, dentro das janelas W_1 e W_2 , com bases, $\mathbf{B}_{W_1}(\psi_1)$ e $\mathbf{B}_{W_2}(\psi_2)$, então*

$$\mathbf{B}_{W_1 \cup W_2}(\psi_1 \wedge \psi_2) = \mathbf{B}_{W_1 \cup W_2}(\psi_1) \sqcap \mathbf{B}_{W_1 \cup W_2}(\psi_2).$$

Prova:

Este resultado é uma consequência imediata da Proposição 5.2 e do isomorfismo de reticulado entre $(\Psi_{W_1 \cup W_2}, \leq)$ e $(\Pi_{W_1 \cup W_2}, \leq)$.

□

O próximo exemplo ilustra a aplicação desta propriedade.

Exemplo 5.2 *Sejam ψ_1 e ψ_2 dois operadores de conjuntos i.t. e localmente definidos, com bases, respectivamente,*

$$\mathbf{B}_W(\psi_1) = \left\{ \begin{bmatrix} 010 & , & 111 \end{bmatrix} \right\}$$

e

$$\mathbf{B}_W(\psi_2) = \left\{ \begin{bmatrix} 000 & , & 011 \end{bmatrix}, \begin{bmatrix} 000 & , & 101 \end{bmatrix}, \begin{bmatrix} 000 & , & 110 \end{bmatrix} \right\},$$

onde $W = 111$

Aplicamos a Proposição 5.5 para computar $\mathbf{B}_W(\psi_1 \wedge \psi_2)$ a partir de $\mathbf{B}_W(\psi_1)$ e $\mathbf{B}_W(\psi_2)$.

$$\begin{aligned} \mathbf{B}_W(\psi_1 \wedge \psi_2) &= \mathbf{B}_W(\psi_1) \cap \mathbf{B}_W(\psi_2) \\ &= \left\{ \begin{bmatrix} 010 & , & 111 \end{bmatrix} \right\} \cap \left\{ \begin{bmatrix} 000 & , & 011 \end{bmatrix}, \begin{bmatrix} 000 & , & 101 \end{bmatrix}, \right. \\ &\quad \left. \begin{bmatrix} 000 & , & 110 \end{bmatrix} \right\} \\ &= \left\{ \begin{bmatrix} 010 & , & 011 \end{bmatrix}, \begin{bmatrix} 010 & , & 110 \end{bmatrix} \right\}. \end{aligned}$$

□

A Figura 5.1 mostra duas possíveis representações do operador apresentado no Exemplo 5.2. Neste capítulo, os *pixels* de valor um e zero da imagem são representados, por um quadrinho, respectivamente, preto e branco. Supoem-se que os pixels fora da janela da imagem são de valor zero. Na Figura 5.1 e em outras figuras, usamos a seguinte notação: $\Gamma_{\mathbf{B}(\psi)} = \vee \{ \varepsilon_A \wedge \delta_{Bct} : [A, B] \in \mathbf{B}(\psi) \}$.

Figura 5.1: Ínfimo de dois operadores.

Em seguida, mostramos como computar a base da composição de um operador i.t. e localmente definido com o operador de negação, a partir da base do operador de entrada.

Proposição 5.6 *Seja ψ um operador de conjunto i.t. e localmente definido dentro de uma janela W , com base $B_{W_1}(\psi_1)$, então*

$$B_W(\nu\psi) = \overline{B}_W(\psi).$$

Prova:

Este resultado é uma consequência imediata do isomorfismo de reticulados entre $(\Psi_W, \leq)$ e $(\Pi_W, \leq)$.

□

O próximo exemplo apresenta uma aplicação da Proposição 5.6.

Exemplo 5.3 *Seja ψ um operador de conjunto i.t. e localmente definido, com base*

Figura 5.2: Composição do operador ψ com o operador de negação.

$$B_W(\psi) = \left\{ \begin{bmatrix} 100 & 111 \\ 000 & 111 \\ 000 & 111 \end{bmatrix}, \begin{bmatrix} 000 & 111 \\ 010 & 111 \\ 000 & 111 \end{bmatrix}, \begin{bmatrix} 000 & 111 \\ 000 & 111 \\ 001 & 111 \end{bmatrix} \right\}.$$

A base do operador $\nu\psi$ será

$$B_W(\nu\psi) = \left\{ \begin{bmatrix} 000 & 011 \\ 000 & 101 \\ 000 & 110 \end{bmatrix} \right\}, \text{ com } W = \begin{matrix} 111 \\ 111 \\ 111 \end{matrix}.$$

□

Na Figura 5.2 mostramos duas possíveis representações do operador apresentado no Exemplo 5.3.

Em seguida, mostramos como computar a base do supremo de dois operadores i.t. e localmente definidos, a partir da base destes operadores.

Proposição 5.7 *Se ψ_1 e ψ_2 são dois operadores de conjuntos i.t. e localmente definidos, respectivamente, dentro de W_1 e W_2 , com bases $B_{W_1}(\psi_1)$ e $B_{W_2}(\psi_2)$, então*

$$B_{W_1 \cup W_2}(\psi_1 \vee \psi_2) = B_{W_1 \cup W_2}(\psi_1) \sqcup B_{W_1 \cup W_2}(\psi_2).$$

Prova:

Este resultado é uma consequência imediata da Proposição 5.2 e do isomorfismo de reticulado entre $(\Psi_{W_1 \cup W_2}, \leq)$ e $(\Pi_{W_1 \cup W_2}, \leq)$.

□

O próximo exemplo apresenta uma aplicação da Proposição 5.7.

Exemplo 5.4 *Sejam ψ_1 e ψ_2 dois operadores de conjuntos i.t. e localmente definidos, com bases, respectivamente,*

$$B_W(\psi_1) = \left\{ \begin{bmatrix} 1 & & 1 \\ 1 & 1 & 1 \\ & 1 & 1 \end{bmatrix} \right\}$$

e

$$B_W(\psi_2) = \left\{ \begin{bmatrix} 1 & & 1 \\ 1 & 0 & 1 \\ & 1 & 1 \end{bmatrix} \right\},$$

$$\text{onde } W = \begin{bmatrix} 1 \\ 1 & 1 & 1 \\ 1 \end{bmatrix}.$$

Aplicando a Proposição 5.7, computamos a base de $\psi_1 \vee \psi_2$:

Figura 5.3: Supremo de dois operadores.

$$\begin{aligned}
 B_W(\psi_1 \vee \psi_2) &= \left\{ \left[\begin{array}{cc} 1 & 1 \\ 1 \ 1 \ 1 & 1 \ 1 \ 1 \\ 1 & 1 \end{array} \right] \right\} \sqcup \left\{ \left[\begin{array}{cc} 1 & 1 \\ 1 \ 0 \ 1 & 1 \ 0 \ 1 \\ 1 & 1 \end{array} \right] \right\} \\
 &= \left\{ \left[\begin{array}{cc} 1 & 1 \\ 1 \ 0 \ 1 & 1 \ 1 \ 1 \\ 1 & 1 \end{array} \right] \right\}.
 \end{aligned}$$

□

A Figura 5.3 mostra duas formas alternativas para representar o operador apresentado no Exemplo 5.4.

Agora, mostraremos como computar a base da composição de um operador i.t. e localmente definido com uma dilação ou com uma erosão, a partir da base deste operador.

Proposição 5.8 *Se ψ é um operador de conjunto i.t e localmente definido dentro de uma janela W , com base $\mathbf{B}_W(\psi)$, e, $\forall b \in E$, τ_b é o operador translação pelo vetor b , então*

$$\mathbf{B}_{W-b}(\tau_b\psi) = \mathbf{B}_W(\psi) - b.$$

Prova:

$$\mathbf{B}_{W-b}(\tau_b\psi) = \text{Max}(\mathcal{K}_{W-b}(\tau_b\psi))$$

(pela definição de $\mathbf{B}_W$)

$$= \text{Max}(\{X \in \mathcal{P}(W-b) : o \in (\tau_b\psi)(X)\})$$

(pela definição de $\mathcal{K}_W$)

$$= \text{Max}(\{X \in \mathcal{P}(W-b) : o \in \psi(X) + b\})$$

(pela definição de τ_b)

$$= \text{Max}(\{X \in \mathcal{P}(W-b) : o \in \psi(X+b)\})$$

(ψ é i.t.)

$$= \text{Max}(\{X \in \mathcal{P}(W) : o \in \psi(X)\} - b)$$

(por uma propriedade da translação)

$$= \text{Max}(\mathcal{K}_W(\psi) - b)$$

(pela definição de $\mathcal{K}_W$)

$$= \mathbf{B}_W(\psi) - b$$

(pela definição de $\mathbf{B}_W$ e uma propriedade da translação).

□

Proposição 5.9 *Seja ψ um operador de conjunto i.t. e localmente definido dentro de uma janela W , com base $\mathbf{B}_W(\psi)$, e seja δ_B a dilação por B , então*

$$\mathbf{B}_{W \oplus B^t}(\delta_B \psi) = \sqcup_{b \in B} \mathbf{B}_{W \oplus B^t}(\tau_b \psi).$$

Prova:

$$\mathbf{B}_{W \oplus B^t}(\delta_B \psi) = \text{Max}(\mathcal{K}_{W \oplus B^t}(\delta_B \psi))$$

(pela definição de $\mathbf{B}_W$ e pela Proposição 5.4)

$$= \text{Max}(\{X \in \mathcal{P}(W \oplus B^t) : o \in (\delta_B \psi)(X)\})$$

(pela definição de $\mathcal{K}_W$)

$$= \text{Max}(\{X \in \mathcal{P}(W \oplus B^t) : o \in \cup_{b \in B} \psi(X) + b\})$$

(pela definição de δ_B)

$$= \text{Max}(\{X \in \mathcal{P}(W \oplus B^t) : \exists b \in B : o \in \psi(X) + b\})$$

(pela definição de $\cup$)

$$= \text{Max}(\{X \in \mathcal{P}(W \oplus B^t) : \exists b \in B : o \in \tau_b \psi(X)\})$$

(pela definição de τ_b)

$$= \text{Max}(\cup_{b \in B} \{X \in \mathcal{P}(W \oplus B^t) : o \in \tau_b \psi(X)\})$$

(pela definição de $\cup$)

$$= \text{Max}(\cup_{b \in B} \mathcal{K}_{W \oplus B^t}(\tau_b \psi))$$

(pela definição de $\mathcal{K}_W$ e porque $\tau_b \psi$ é localmente definido dentro de $W \oplus B^t$, já que $W - b \subseteq W \oplus B^t$)

$$= \sqcup_{b \in B} \mathbf{B}_{W \oplus B^t}(\tau_b \psi)$$

(pelo isomorfismo de reticulado entre $(\mathcal{P}(\mathcal{P}(W \oplus B^t)), \subseteq)$ e $(\Pi_{W \oplus B^t}, \leq)$).

□

O Exemplo 5.5 apresenta uma aplicação da Proposição 5.9

Exemplo 5.5 *Seja ψ um operador de conjunto i.t. e localmente definido, com base*

$$B_{W_1}(\psi) = \left\{ \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right\},$$

onde $W_1 = 111$, e seja $B = 011$. Aplicando a Proposição 5.9, computamos a base de $\delta_B \psi$:

$$\begin{aligned} B_W(\delta_B \psi) &= \left\{ \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \right\} \\ &\quad \sqcup \left\{ \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \right\} \\ &= \left\{ \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \right. \\ &\quad \left. \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \right\}, \end{aligned}$$

onde $W = W_1 \oplus B^t = 1111$.

□

A Figura 5.4 mostra duas formas alternativas para representar o operador apresentado no Exemplo 5.5.

Figura 5.4: Composição do operador ψ com a dilatação δ_B .

Proposição 5.10 *Seja ψ um operador de conjunto i.l. e localmente definido dentro de uma janela W , com base $\mathbf{B}_W(\psi)$, e seja ε_B uma erosão por B , então*

$$\mathbf{B}_{W \oplus B}(\varepsilon_B \psi) = \bigcap_{b \in B^t} \mathbf{B}_{W \oplus B}(\tau_b \psi).$$

Prova:

$$\mathbf{B}_{W \oplus B}(\varepsilon_B \psi) = \text{Max}(\mathcal{K}_{W \oplus B}(\varepsilon_B \psi))$$

(pela definição de $\mathbf{B}_W$ e pela Proposição 5.4)

$$= \text{Max}(\{X \in \mathcal{P}(W \oplus B) : o \in (\varepsilon_B \psi)(X)\})$$

(pela definição de $\mathcal{K}_W$)

$$= \text{Max}(\{X \in \mathcal{P}(W \oplus B) : o \in \bigcap_{b \in B} \psi(X) - b\})$$

(pela definição de ε_B)

$$= \text{Max}(\{X \in \mathcal{P}(W \oplus B) : o \in \psi(X) - b, \forall b \in B\})$$

(pela definição de $\bigcap$)

$$= \text{Max}(\bigcap_{b \in B} \{X \in \mathcal{P}(W \oplus B) : o \in \psi(X) - b\})$$

(pela definição de $\bigcap$)

$$= \text{Max}(\bigcap_{b \in B} \{X \in \mathcal{P}(W \oplus B) : o \in \tau_{-b} \psi(X)\})$$

(pela definição de τ_b)

$$= \text{Max}(\bigcap_{b \in B^t} \mathcal{K}_{W \oplus B}(\tau_b \psi))$$

(pela definição de $\mathcal{K}_W$ e porque $\tau_{-b} \psi$ é localmente definido dentro de $W \oplus B$, já que $W + b \subseteq W \oplus B$)

$$= \bigcap_{b \in B^t} \mathbf{B}_{W \oplus B}(\tau_b \psi)$$

(pelo isomorfismo de reticulado entre $(\mathcal{P}(\mathcal{P}(W \oplus B)), \subseteq)$ e $(\Pi_{W \oplus B}, \leq)$).

Figura 5.5: Composição do operador ψ com a erosão ε_B .

□

O Exemplo 5.6 apresenta uma aplicação da Proposição 5.10.

Exemplo 5.6 *Seja ψ um operador de conjunto i.t., com base*

$$B_{W_1}(\psi) = \left\{ \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \right\},$$

onde $W_1 = 11$, e seja $B = 110$. Aplicando a Proposição 5.10, computamos a base do operador $\varepsilon_B\psi$:

$$\begin{aligned} B_W(\varepsilon_B\psi) &= \left\{ \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right\} \cap \left\{ \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \right\} \\ &= \left\{ \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right\}, \end{aligned}$$

onde $W = W_1 \oplus B = 111$.

□

A Figura 5.5 mostra duas formas alternativas para representar o operador apresentado no Exemplo 5.6.

Capítulo 6

Computação Incremental da base de um operador

Os resultados apresentados no capítulo anterior são a chave para computar a base de qualquer operador de conjuntos i.t. para o qual é conhecida uma representação na Linguagem Morfológica. A idéia é usar recursivamente estes resultados, para computar incrementalmente a base do operador. Neste processo, a condição inicial é a base trivial do operador de identidade.

A Figura 6.1a mostra o ícone que representa a base do operador de identidade. A Figura 6.1b mostra os ícones que representam as operações elementares usadas para computar a base incrementalmente. Usamos estes ícones em diagramas que ilustram graficamente a computação incremental da base de operadores. Tais diagramas serão usados para descrever as representações conhecidas do operador na Linguagem Morfológica.

Figura 6.1: a) Ícone da base do operador de identidade. b) Ícones do ínfimo, supremo, negação, erosão e dilatação, respectivamente, de cima para baixo.

Agora, apresentamos alguns exemplos que ilustram a computação incremental da base.

Exemplo 6.1 *Estudaremos o operador que realiza a extração de bordos internas. Chamemos este operador ψ e adotemos sua representação usual, isto é, $\psi = \iota \wedge \nu \varepsilon_B \iota$, com*

$$B = \begin{array}{c} 1 \\ 111 \\ 1 \end{array}.$$

Computamos incrementalmente a base de ψ a partir da base do operador de identidade:

$$\begin{aligned} B_W(\psi) &= B_W(\iota \wedge \nu \varepsilon_B \iota) \\ &= B_W(\iota) \sqcap B_W(\nu \varepsilon_B \iota) \\ &= B_W(\iota) \sqcap \overline{B}_W(\varepsilon_B \iota) \\ &= \left\{ \begin{bmatrix} 0 & 0 \\ 010 & 111 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 \\ 010 & 011 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 \\ 010 & 110 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 \\ 010 & 111 \\ 0 & 0 \end{bmatrix} \right\}, \end{aligned}$$

$$\text{onde } W = \{o\} \cup (\{o\} \oplus (B \oplus \{o\})) = \begin{array}{c} 1 \\ 111 \\ 1 \end{array}.$$

A Figura 6.2 mostra uma representação gráfica deste procedimento.

□

Figura 6.2: Computação incremental da base de $\psi = \iota \wedge \nu \varepsilon_B \iota$.

Exemplo 6.2 Estudaremos o operador que realiza o esqueleto direcional. Chamaremos este operador ψ e adotaremos sua representação usual, isto é, $\psi = \iota \wedge \nu(\varepsilon_A \iota \wedge \nu \delta_B \iota)$, com

$$A = \begin{array}{ccc} 111 \\ 010 \\ 000 \end{array} \text{ e } B = \begin{array}{ccc} 111 \\ 000 \\ 000 \end{array}.$$

Computaremos incrementalmente a base de ψ a partir da base do operador de identidade:

$$\begin{aligned} B_W(\psi) &= B_W(\iota \wedge \nu(\varepsilon_A \iota \wedge \nu \delta_B \iota)) \\ &= B_W(\iota) \sqcap B_W(\nu(\varepsilon_A \iota \wedge \nu \delta_B \iota)) \\ &= B_W(\iota) \sqcap \overline{B_W}(\varepsilon_A \iota \wedge \nu \delta_B \iota) \\ &= B_W(\iota) \sqcap \overline{B_W(\varepsilon_A \iota)} \sqcap \overline{B_W(\nu \delta_B \iota)} \\ &= B_W(\iota) \sqcap \overline{B_W(\varepsilon_A \iota)} \sqcap \overline{B_W(\delta_B \iota)} \\ &= \left\{ \begin{bmatrix} 000 & 111 \\ \mathbf{1} & \mathbf{1} \\ 001 & 111 \end{bmatrix}, \begin{bmatrix} 000 & 111 \\ \mathbf{1} & \mathbf{1} \\ 010 & 111 \end{bmatrix}, \begin{bmatrix} 000 & 111 \\ \mathbf{1} & \mathbf{1} \\ 100 & 111 \end{bmatrix}, \begin{bmatrix} 000 & 110 \\ \mathbf{1} & \mathbf{1} \\ 000 & 111 \end{bmatrix} \right. \\ &\quad \left. \begin{bmatrix} 000 & 101 \\ \mathbf{1} & \mathbf{1} \\ 000 & 111 \end{bmatrix}, \begin{bmatrix} 000 & 011 \\ \mathbf{1} & \mathbf{1} \\ 000 & 111 \end{bmatrix} \right\}, \end{aligned}$$

Figura 6.3: Computação incremental da base de $\psi = \iota \wedge \nu(\varepsilon_A \iota \wedge \nu \delta_B \iota)$.

$$\text{onde } W = \{o\} \cup (\{\oplus\} \oplus ((A \oplus \{o\}) \cup (B^t \oplus \{o\}))) = \begin{matrix} 111 \\ 1 \\ 111 \end{matrix}.$$

A Figura 6.3 mostra uma representação gráfica deste procedimento.

□

Seja $I = \{1, 2, 3, \dots, n\}$ um conjunto de índices. Agora, apresentamos um resultado que mostra como computar a base de um operador a partir da base de seu dual.

Proposição 6.1 *Se ψ é um operador de conjuntos i.t. e localmente definido dentro de uma janela W , com base $B_W(\psi) = \{[A_i, B_i] : i \in I\}$, então a base de seu operador dual ψ^* é*

$$B_W(\psi^*) = \sqcup \{ \overline{\{[B_i^c, A_i^c] : i \in I\}}.$$

Prova:

$$\psi^* = \nu \psi \nu \text{ (pela definição de } \psi^*)$$

$$\begin{aligned}
&= \nu(\vee \{ \lambda_{[A_i, B_i]}^W : i \in I \}) \nu \text{ (pela sup-decomposição canônica de } \psi) \\
&= \nu(\vee \{ \varepsilon_{A_i} \wedge \nu \delta_{B_i^c} : i \in I \}) \nu \text{ (pela decomposição de } \lambda_{[A, B]}^W) \\
&= \nu(\vee \{ \varepsilon_{A_i} \nu \wedge \nu \delta_{B_i^c} \nu : i \in I \}) \\
&= \nu(\vee \{ \nu \delta_{A_i^t} \wedge \varepsilon_{B_i^c} : i \in I \}) \text{ (pois } \varepsilon_{A_i} \nu = \nu \delta_{A_i^t} \text{ e } \nu \delta_{B_i^c} \nu = \varepsilon_{B_i^c}).
\end{aligned}$$

A partir do isomorfismo de reticulado entre $(\Psi_W, \leq)$ e $(\Pi_W, \leq)$, temos:

$$\begin{aligned}
\mathbf{B}_W(\psi^*) &= \overline{\mathbf{B}_W(\vee \{ \nu \delta_{A_i^t} \wedge \varepsilon_{B_i^c} : i \in I \})} \\
&= \overline{\sqcup \{ \mathbf{B}_W(\nu \delta_{A_i^t} \wedge \varepsilon_{B_i^c}) : i \in I \}}.
\end{aligned}$$

Além disso,

$$\begin{aligned}
\mathbf{B}_W(\nu \delta_{A^t} \wedge \varepsilon_{B^c}) &= \mathbf{B}_W(\nu \delta_{A^t}) \sqcap \mathbf{B}_W(\varepsilon_{B^c}) \\
&= \overline{\{ [\{a\}, W] : a \in A \}} \sqcap \{ [B^c, W] \}
\end{aligned}$$

$$\begin{aligned}
\text{(uma vez que } \mathbf{B}_W(\delta_A) &= \{ [\{a\}, W] : a \in A^t \} \text{ e } \mathbf{B}_W(\varepsilon_B) = \{ [B, W] \}) \\
&= \{ [\emptyset, A^c] \} \sqcap \{ [B^c, W] \}
\end{aligned}$$

(pelo Teorema 3.2)

$$= \{ [B^c, A^c] \}$$

(pelo Teorema 3.1).

Assim,

$$\mathbf{B}_W(\psi^*) = \overline{\sqcup \{ [B_i^c, A_i^c] : i \in I \}}$$

□

Agora, apresentaremos um resultado que mostra como computar a base da composição de dois operadores a partir das bases destes operadores.

Proposição 6.2 *Se ψ_1 e ψ_2 são operadores de conjuntos i.t. e localmente definidos, respectivamente, dentro das janelas W_1 e W_2 e a base de ψ_2 é*

$B_{W_2}(\psi_2) = \{[A_i, B_i] : i \in I\}$, então

$$B_{W_1 \oplus W_2}(\psi_2 \psi_1) = \sqcup \{ B_{W_1 \oplus W_2}(\varepsilon_{A_i} \psi_1) \cap \overline{B}_{W_1 \oplus W_2}(\delta_{B_i^{ct}} \psi_1) : i \in I \}.$$

Prova:

$$\psi_2 \psi_1 = (\vee \{ \lambda_{[A_i, B_i]}^{W_2} : i \in I \}) \psi_1$$

(pela sup-decomposição canônica de ψ_2)

$$= (\vee \{ (\varepsilon_{A_i} \wedge \nu \delta_{B_i^{ct}}) : i \in I \}) \psi_1$$

(pela decomposição de $\lambda_{[A_i, B_i]}^{W_2}$)

$$= \vee \{ \varepsilon_{A_i} \psi_1 \wedge \nu \delta_{B_i^{ct}} \psi_1 : i \in I \}$$

Pelo isomorfismo de reticulado entre $(\Psi_W, \leq)$ e $(\Pi_{W \oplus B}, \leq)$ e o Corolário 5.1, temos:

$$B_{W_1 \oplus W_2}(\psi_2 \psi_1) = \sqcup \{ B_{W_1 \oplus W_2}(\varepsilon_{A_i} \psi_1) \cap \overline{B}_{W_1 \oplus W_2}(\delta_{B_i^{ct}} \psi_1) : i \in I \}.$$

□

Uma aplicação prática de computação incremental da base é a prova automática de equivalência entre operadores morfológicos, uma vez que dois operadores são iguais se e somente se suas bases são iguais. A Figura 6.4 mostra o ícone usado para representar a comparação entre operadores. O próximo exemplo ilustra este procedimento.

Exemplo 6.3 *Sejam os operadores ψ_1 e ψ_2 definidos por $\psi_1 = \delta_B \varepsilon_{B^t}$ e $\psi_2 = \delta_B \varepsilon_B \delta_{B^t}$, respectivamente, com $B = 111$.*

Computando a base de ψ_1 temos:

$$B_{W_1}(\psi_1) = \left\{ \left[\begin{array}{cc} 11100 & , & 11111 \end{array} \right], \left[\begin{array}{cc} 01110 & , & 11111 \end{array} \right], \left[\begin{array}{cc} 00111 & , & 11111 \end{array} \right] \right\},$$

Figura 6.4: Comparação de igualdade entre dois operadores.

com, $W_1 = B^t \oplus (B \oplus \{o\}) = 11111$

Computando a base de ψ_2 , temos:

$$B_{W_2}(\psi_2) = \left\{ \left[\begin{array}{cc} 001110000 & , & 111111111 \end{array} \right], \left[\begin{array}{cc} 000111000 & , & 111111111 \end{array} \right], \right. \\ \left. \left[\begin{array}{cc} 000011100 & , & 111111111 \end{array} \right] \right\},$$

com $W_2 = B^t \oplus (B \oplus (B^t \oplus (B \oplus \{o\}))) = 111111111$

Note que estas bases são equivalentes. A Figure 6.5 mostra um diagrama representando este procedimento. $\square$

Figura 6.5: Verificação de equivalência de ψ_1 e ψ_2 .

Outra aplicação da computação incremental da base é a paralelização da representação de operadores morfológicos. Esta técnica é útil na preparação de programas para MMach's altamente paralelas.

Na paralelização de programas para MMach's podemos escolher entre as duas decomposições canônicas. A Figure 6.6 mostra a estrutura altamente paralela destas decomposições.

Note que o número de operadores sup-geradores usados na sup-decomposição depende da base do operador, entretanto o número de operadores inf-geradores na inf-decomposição depende da base do operador dual. Assim, para escolher a melhor representação paralela devemos comparar a cardinalidade da base do operador com a cardinalidade da base do operador dual.

Figura 6.6: Estruturas de decomposição altamente paralelas. a) Sup-decomposição canônica. b) Inf-decomposição canônica.

O próximo exemplo ilustra o procedimento de paralelização de programas para MMach's.

Exemplo 6.4 Seja o operador ψ definido por $\psi = \iota \wedge \nu(\varepsilon_A \iota \wedge \nu \delta_B \iota)$, com $A = \begin{smallmatrix} & & 111 \\ & 010 & \\ & & 000 \end{smallmatrix}$ e $B = \begin{smallmatrix} & & 111 \\ & 000 & \\ & & 000 \end{smallmatrix}$. Esta representação de ψ é ilustrada na Figura 6.7a. $B_W(\psi)$ foi computada no Exemplo 6.2. Computando a base de ψ^* , temos:

$$B_W(\psi^*) = \left\{ \left[\begin{array}{cc} 000 & 000 \\ \mathbf{0} & \mathbf{1} \\ 111 & 111 \end{array} \right], \left[\begin{array}{cc} 000 & 111 \\ \mathbf{1} & \mathbf{1} \\ 000 & 111 \end{array} \right] \right\},$$

que é usada na inf-decomposição. Esta representação de ψ é ilustrada na Figura 6.7b. Note que esta última representação é mais eficiente do que a primeira, uma vez que ela tem menos níveis de processamento de dados.

□

Figura 6.7: Duas representações do operador ψ . a) Definição de ψ . b) Inf-decomposição de ψ .

a)

b)

Figura 6.8: Duas representações do operador. a) Definição de ψ . b) Inf-decomposição de ψ .

Exemplo 6.5 Seja o operador ψ definido por $\psi = (\iota \wedge \nu_{\varepsilon_B} \iota)$ com $B = \begin{matrix} & 1 \\ 1 & 1 & 1 \\ & 1 \end{matrix}$. Esta representação de ψ é ilustrada na Figura 6.8a. $B_W(\psi)$ foi computada no Exemplo 6.1. Computando a base de ψ^* , temos:

$$B_W(\psi^*) = \left\{ \left[\begin{array}{cc} 0 & 1 \\ 010 & 111 \\ 0 & 1 \end{array} \right], \left[\begin{array}{cc} 0 & 0 \\ 000 & 010 \\ 0 & 0 \end{array} \right] \right\},$$

que é usada na inf-decomposição. Esta representação de ψ é ilustrada na Figura 6.8b. Note que esta última representação é mais eficiente do que a primeira, pois o elemento estruturante da erosão tem cardinalidade menor.

□

Exemplo 6.6 Seja o operador ψ definido por $\psi = (\iota \wedge \nu \varepsilon_B \iota) \vee (\nu \iota \wedge \delta_B \iota)$, com $B = \begin{smallmatrix} & & 1 \\ & 1 & 1 & 1 \\ & & & 1 \end{smallmatrix}$.

Esta representação de ψ é ilustrada na Figura 6.9a. A base de ψ tem vinte elementos. Computando a base de ψ^* , temos:

$$B_W(\psi^*) = \left\{ \left[\begin{array}{cc} 0 & 0 \\ 000 & 000 \\ 0 & 0 \end{array} \right], \left[\begin{array}{cc} 1 & 1 \\ 111 & 111 \\ 1 & 1 \end{array} \right] \right\},$$

que é usada na inf-decomposição. Esta representação de ψ é ilustrada na Figura 6.9b. Note que esta última representação é mais eficiente do que a primeira, uma vez que ela tem menos níveis de processamento de dados.

□

Figura 6.9: Duas representações do operador. a) Definição de ψ . b) Inf-decomposição de ψ .

Capítulo 7

Algoritmos para a Computação da Base

Neste capítulo, apresentamos os principais algoritmos propostos para a computação incremental da base. Todos os algoritmos propostos dependem essencialmente de dois procedimentos: ínfimo de duas coleções de intervalos maximais e complemento de uma coleção de intervalos.

A sintaxe usada para a descrição dos algoritmos é similar à linguagem C. As principais instruções de controle são mostradas na tabela 7.1.

Símbolo	Significado
;	delimita o fim de uma instrução
{ , }	delimita um bloco de instruções
=, ≠, >, ≥, <, ≤	comparações (resultados lógicos)
∀	laço para
while	laço enquanto
if, else	ramificações lógicas
true, false	valores lógicos
←	atribuição

Tabela 7.1: Lista de instruções de controle.

7.1 Estrutura de dados

Nesta seção, apresentamos a estrutura de dados usada.

```

estrutura janela{
    /* especificação de uma janela */
}

estrutura intervalo{
    A ;    /* extremidade esquerda */
    Bc;    /* complemento da extremidade direita */
}

estrutura intervalos{
    /* lista de intervalos */
}

estrutura base{
    intervalos int;    /* conjuntos de intervalos da base */
    janela win;    /* janela correspondente à base */
}

```

7.2 Rotinas Auxiliares

Nesta seção, apresentamos algumas rotinas auxiliares.

- `sup-win(win1, win2, win_sup)`

Computa o supremo de duas janelas

Entrada : "*win1*", janela 1.

"*win2*", janela 2.

Saída: "*win_sup*", o supremo das duas janelas de entrada.

- `transfer_basis_window(basis, win)`

Transfere a representação da base "*basis*" de uma dada janela a uma janela maior "*win*". Depois de transferidos, os intervalos da base serão considerados incluídos no conjunto das partes de "*win*".

Entrada : "*basis*", base.

"*win*", nova janela.

Saída : "*basis*", base sob a nova janela.

- `verify_interval(I)`

Verifica se o intervalo "*I*" é não vazio.

Entrada : "*I*", intervalo.

Saída : 1 se verdadeiro, 0 caso contrario.

- `insert(I, list)`

Insero o intervalo "*I*" da lista "*list*".

Entrada : "*I*", intervalo a inserir.

"*list*", lista de intervalos.

Saída : "*list*", lista de intervalos modificada.

- `cond_insert(I, list)`

Insero o intervalo "*I*" na lista "*list*", se "*I*" não está incluído em algum dos intervalos de "*list*".

Entrada : "*I*", intervalo a inserir.

"*list*", lista de intervalos.

Saída : "*list*", lista de entrada modificada.

- `cond_insert_remove(I, list)`

Se o intervalo "*I*" não está incluído em algum dos intervalos da lista "*list*", então insere "*I*" em "*list*" e tira os intervalos de "*list*" que estão contidos em "*I*".

Entrada : "*I*", intervalo a inserir.

"*list*", lista de intervalos a verificar.

Saída : "*list*", lista de intervalos modificada.

- `remove(I, list)`

Remove o intervalo "*I*" da lista "*list*".

Entrada : "*list*", lista de intervalos.

Saída : "*list*", lista de intervalos modificada.

"*I*", intervalo removido.

7.3 Rotina do Ínfimo

Nesta seção, apresentamos dois algoritmos que computam o ínfimo de duas coleções de intervalos maximais pelo Teorema 3.1. O primeiro algoritmo chamado *inf1* é o seguinte:

- `inf1(basis1, basis2, basis)`

Computa o ínfimo de duas coleções de intervalos maximais.

Entrada : "*basis1*", coleção de intervalos maximais ($\mathcal{X}$).

"*basis2*", coleção de intervalos maximais ($\mathcal{Y}$).

Saída : "*basis*", coleção de intervalos maximais correspondentes ao

ínfimo das duas coleções de entrada ($\mathcal{X} \sqcap \mathcal{Y}$).

Inicialização : $basis.int \leftarrow \emptyset$

Computa o supremo das duas janelas de entrada:

$sup_win(basis1.win, basis2.win, basis.win);$

Transfere a representação das bases de entrada para $basis.win$

$transfer_basis_window(basis1, basis.win);$

$transfer_basis_window(basis2, basis.win);$

Laço principal:

```

 $\forall$  Intervalo  $I1 \in basis1.int$  {
     $\forall$  Interval  $I2 \in basis2.int$  {
         $I \leftarrow I1 \cap I2;$ 
        if  $verify\_interval(I)$  { /* verifica se  $I$  é não vazio */
             $cond\_insert\_remove(I, basis.int);$ 
        }
    }
}
```

Em seguida apresentamos o segundo algoritmo, chamado *inf2*. Neste algoritmo, usamos a lista auxiliar "*list*", que inicialmente está vazia. Os passos iniciais deste algoritmo são os mesmos de *inf1*, então apresentamos apenas o seu laço principal.

- $inf2(basis1, basis2, basis)$

Laço principal

```

     $\forall$  Interval  $I1 \in basis1.int$  {
         $\forall$  Interval  $I2 \in basis2.int$  {
             $I \leftarrow I1 \cap I2$ ;

            if verify_interval( $I$ ) {
                cond_insert_remove( $I$ ,  $list$ );
            }
        }
    }

    while  $list \neq \emptyset$  {
        remove( $I$ ,  $list$ );

        cond_insert_remove( $I$ ,  $basis.int$ );
    }
}
```

Neste segundo algoritmo, a lista de intervalos criada pela interseção de um intervalo pertencente a *basis1* com os intervalos pertencentes a *basis2* é filtrada antes de ser introduzida na lista que guarda o resultado.

7.4 Rotina de Complementação

Nesta seção, apresentamos alguns algoritmos para a computação da coleção de intervalos maximais que representa o complemento de uma coleção de subconjuntos, partindo da representação desta coleção por uma coleção de intervalos (não necessariamente maximais). Estes algoritmos correspondem a realizações computacionais do Teorema 3.2.

Em primeiro lugar, apresentamos alguns resultados adicionais.

Seja $I = \{1, 2, \dots, n\}$ um conjunto de índices. Seja $k \in I$. Suponha que $\mathcal{I}_{k-1}$ é a coleção de intervalos maximais obtida depois de $k - 1$ aplicações recursivas da operação $\sqcap$.

Seja $\mathcal{X} = \{[A_i, B_i] : i \in I\}$ uma coleção de intervalos, pelo Teorema 3.2, temos:

$$\overline{Max(\cup \mathcal{X})} = \sqcap \{Max([A_i, B_i]^c) : i \in I\}.$$

Esta expressão representa uma aplicação recursiva da operação binária $\sqcap$, isto é,

$$\overline{Max(\cup \mathcal{X})} = (([\emptyset, W] \sqcap (Max([A_1, B_1]^c) \sqcap Max([A_2, B_2]^c))) \sqcap Max([A_3, B_3]^c)) \sqcap \dots$$

Sejam $\mathcal{A}_{k-1}$ e $\mathcal{B}_{k-1}$ uma partição binária de $\mathcal{I}_{k-1}$ (i.e., $\mathcal{A}_{k-1} \cup \mathcal{B}_{k-1} = \mathcal{I}_{k-1}$ e $\mathcal{A}_{k-1} \cap \mathcal{B}_{k-1} = \emptyset$) tal que

$$\mathcal{A}_{k-1} = \{[X, Y] \in \mathcal{I}_{k-1} : [X, Y] \cap [A_k, B_k] = \emptyset\}$$

e

$$\mathcal{B}_{k-1} = \{[X, Y] \in \mathcal{I}_{k-1} : [X, Y] \cap [A_k, B_k] \neq \emptyset\},$$

onde $[A_k, B_k]$ é o k -ésimo intervalo em $\mathcal{X}$.

Introduziremos a seguinte notação:

$$F(\mathcal{B}_{k-1}, [A_k, B_k]) = \{[A, B] \cap [C, D] : [A, B] \in \mathcal{B}_{k-1}, [C, D] \in Max([A_k, B_k]^c)\}.$$

Proposição 7.1 Se $[A_k, B_k]$ e $\mathcal{B}_{k-1}$ são, respectivamente, o intervalo e a coleção de intervalos maximais definidos acima, então

$$F(\mathcal{B}_{k-1}, [A_k, B_k])$$

é maximal.

Prova:

Para provar que $F(\mathcal{B}_{k-1}, [A_k, B_k])$ é maximal, devemos provar que para qualquer $[U, V], [U', V'] \in F(\mathcal{B}_{k-1}, [A_k, B_k])$, $[U, V] \not\subseteq [U', V']$ e $[U', V'] \not\subseteq [U, V]$. Provemos que $[U, V] \not\subseteq [U', V']$.

Se $[U, V], [U', V'] \in F(\mathcal{B}_{k-1}, [A_k, B_k])$, então existem $[A, B], [A', B'] \in \mathcal{B}_{k-1}$ e $[C, D], [C', D'] \in \text{Max}([A_k, B_k]^c)$ tais que $[U, V] = [A \cup C, B \cap D]$ e $[U', V'] = [A' \cup C', B' \cap D']$.

Uma vez que $[C, D], [C', D'] \in \text{Max}([A_k, B_k]^c)$, temos:

$$[C, D] = [\emptyset, \{a\}^c], \text{ com } a \in A_k, \text{ ou } [C, D] = [\{b\}, W], \text{ com } b \in B_k^c,$$

e

$$[C', D'] = [\emptyset, \{a'\}^c], \text{ com } a' \in A_k, \text{ ou } [C', D'] = [\{b'\}, W], \text{ com } b' \in B_k^c$$

(pelas Proposições 3.2, 3.3 e 3.4).

Uma vez que $\mathcal{B}_{k-1}$ é maximal e $[A, B], [A', B'] \in \mathcal{B}_{k-1}$, temos:

$$[A, B] \not\subseteq [A', B'] \text{ e } [A', B'] \not\subseteq [A, B] \text{ ou, equivalentemente, } (A' \not\subseteq A \text{ ou } B \not\subseteq B') \text{ e } (A \not\subseteq A' \text{ ou } B' \not\subseteq B).$$

Para provar que $[U, V] \not\subseteq [U', V']$, devemos provar que $\exists X \in [U, V] : X \notin [U', V']$ em todas as possíveis escolhas de pares $([C, D], [C', D'])$ e $([A, B], [A', B'])$:

1) $[C, D] = [\{b\}, W]$, com $b \in B_k^c$, e $[C', D'] = [\{b'\}, W]$, com $b' \in B_k^c$.

a) se $B \not\subseteq B'$, $B \in [A \cup \{b\}, B]$ e $B \notin [A' \cup \{b'\}, B']$;

b) se $A' \not\subseteq A$, $A \cup \{b\} \in [A \cup \{b\}, B]$ e $A \cup \{b\} \notin [A' \cup \{b'\}, B']$,

uma vez que $b, b' \notin A \cup A'$.

2) $[C, D] = [\emptyset, \{a\}^c]$, com $a \in A_k$, e $[C', D'] = [\emptyset, \{a'\}^c]$, com $a' \in A_k$.

a) se $A' \not\subseteq A$, $A \in [A, B \cap \{a\}^c]$ e $A \notin [A', B' \cap \{a'\}^c]$;

b) se $B \not\subseteq B'$, $B \cap \{a\}^c \in [A, B \cap \{a\}^c]$ e $B \cap \{a\}^c \notin [A', B' \cap \{a'\}^c]$,

uma vez que $a, a' \in B \cap B'$.

3) $[C, D] = [\emptyset, \{a\}^c]$, com $a \in A_k$, e $[C', D'] = [\{b'\}, W]$, com $b' \in B_k^c$.

a) se $A' \not\subseteq A$, $A \in [A, B \cap \{a\}^c]$ e $A \notin [A' \cup \{b'\}, B']$;

b) se $B \not\subseteq B'$, $B \cap \{a\}^c \in [A, B \cap \{a\}^c]$ e $B \cap \{a\}^c \notin [A' \cup \{b'\}, B']$,

uma vez que $a \in B \cap B'$.

4) $[C, D] = [\{b\}, W]$, com $b \in B_k^c$, e $[C', D'] = [\emptyset, \{a'\}^c]$, com $a' \in A_k$.

a) se $A' \not\subseteq A$, $A \cup \{b\} \in [A \cup \{b\}, B]$ e $A \cup \{b\} \notin [A', B' \cap \{a\}^c]$,

uma vez que $b' \notin A \cup A'$;

b) se $B \not\subseteq B'$, $B \in [A \cup \{b\}, B]$ e $B \notin [A', B' \cap \{a\}^c]$.

Para provar que em cada caso, também $[U', V'] \not\subseteq [U, V]$, deveríamos usar os mesmos argumentos apenas trocando os papéis de $[U, V]$ e $[U', V']$.

□

Proposição 7.2 *Se $\mathcal{I}_k$, $\mathcal{A}_{k-1}$, $\mathcal{B}_{k-1}$ e $F(\mathcal{B}_{k-1}, [A_k, B_k])$ são as coleções de intervalos definidos acima, então*

$$\mathcal{I}_k = \mathcal{A}_{k-1} \cup \{[A, B] \in \text{Max}(F(\mathcal{B}_{k-1}, [A_k, B_k])) : \nexists [A', B'] \in \mathcal{A}_{k-1}, [A, B] \subseteq [A', B']\}.$$

Prova:

$$\mathcal{I}_k = \mathcal{I}_{k-1} \cap \text{Max}([A_k, B_k]^c)$$

$$= \text{Max}(\{[A, B] \cap [C, D] : [A, B] \in \mathcal{I}_{k-1}, [C, D] \in \text{Max}([A_k, B_k]^c)\})$$

(pela definição de $\cap$)

$$= \text{Max}(\{[A, B] \cap [C, D] : [A, B] \in \mathcal{A}_{k-1} \cup \mathcal{B}_{k-1}, [C, D] \in \text{Max}([A_k, B_k]^c)\})$$

(uma vez que $\mathcal{I}_{k-1} = \mathcal{A}_{k-1} \cup \mathcal{B}_{k-1}$)

$$= \text{Max}(\{[A, B] \cap [C, D] : [A, B] \in \mathcal{A}_{k-1}, [C, D] \in \text{Max}([A_k, B_k]^c)\} \cup$$

$$\{[A, B] \cap [C, D] : [A, B] \in \mathcal{B}_{k-1}, [C, D] \in \text{Max}([A_k, B_k]^c)\})$$

(por uma propriedade de $\cup$)

$$= \text{Max}(\text{Max}(\{[A, B] \cap [C, D] : [A, B] \in \mathcal{A}_{k-1}, [C, D] \in \text{Max}([A_k, B_k]^c)\}) \cup$$

$$\text{Max}(\{[A, B] \cap [C, D] : [A, B] \in \mathcal{B}_{k-1}, [C, D] \in \text{Max}([A_k, B_k]^c)\}))$$

(por uma propriedade de Max)

$$= \text{Max}(\text{Max}(\mathcal{A}_{k-1}) \cup \text{Max}(\{[A, B] \cap [C, D] : [A, B] \in \mathcal{B}_{k-1}, [C, D] \in \text{Max}([A_k, B_k]^c)\}))$$

(uma vez que $\cup \mathcal{A}_{k-1} \subseteq [A_k, B_k]^c$)

$$= \text{Max}(\mathcal{A}_{k-1} \cup \text{Max}(\{[A, B] \cap [C, D] : [A, B] \in \mathcal{B}_{k-1}, [C, D] \in \text{Max}([A_k, B_k]^c)\}))$$

(uma vez que $\mathcal{A}_{k-1}$ é maximal)

$$= \mathcal{A}_{k-1} \cup \{[A, B] \in \text{Max}(\{[A, B] \cap [C, D] : [A, B] \in \mathcal{B}_{k-1}, [C, D] \in \text{Max}([A_k, B_k]^c)\}) :$$

$$\exists [A', B'] \in \mathcal{A}_{k-1}, [A, B] \subseteq A', B'\}$$

(uma vez que $\mathcal{A}_{k-1} \cup \mathcal{B}_{k-1}$ é maximal o os elementos $[A, B] \in \mathcal{B}_{k-1}$ são fragmentados pela operação $[A, B] \cap \text{Max}([A_k, B_k]^c)$).

O resultado segue pela definição de $F(\mathcal{B}_{k-1}, [A_k, B_k])$.

□

Corolário 7.1 Se $\mathcal{I}_k$, $\mathcal{A}_{k-1}$, $\mathcal{B}_{k-1}$ e $F(\mathcal{B}_{k-1}, [A_k, B_k])$ são as coleções de intervalos definidas acima, então

$$\mathcal{I}_k = \mathcal{A}_{k-1} \cup \{[A, B] \in F(\mathcal{B}_{k-1}, [A_k, B_k]) : \exists [A', B'] \in \mathcal{A}_{k-1}, [A, B] \subseteq [A', B']\}.$$

Prova:

Este corolário é uma consequência imediata das Proposições 7.1 e 7.2. □

Devemos notar que a Proposição 7.2 e o Corolário 7.1 constituem novas formas de computar o Teorema 3.2. Os seguintes algoritmos realizam o complemento de uma coleção de intervalos, baseados no Corolário 7.1.

- `complement(basis1, basis2)`

Complementação de uma coleção de intervalos.

Entrada: "*basis1*", $\mathcal{X}$.

Saída : "*basis2*", $Max((\cup \mathcal{X})^c)$

Inicialização : $basis2.int \leftarrow [\emptyset, W]$;

$list1 \leftarrow \emptyset$;

$list2 \leftarrow \emptyset$;

Laço principal:

/* cria $\mathcal{A}_{k-1}$ e $F(\mathcal{B}_{k-1}, [A_k, B_k])$ */

$\forall$ Interval $I1 \in basis1.int$ { /* $I1$ e $[A_k, B_k]$ */

$\forall$ Interval $I2 \in basis2.int$ { /* $basis2$ e $\mathcal{I}_k$ */

`remove( $I2$ ,  $basis2.int$ );`

$I3 = I1 \cap I2$;


```

        if(verify_interval( $I3$ )) { /* incrementa a computação de  $F(\mathcal{B}_{k-1}, [A_k, B_k])$  */
            f( $I2$ ,  $I1$ ,  $list1$ );
        else{
            insert( $I2$ ,  $list2$ );
        }
    }

    /* cria  $\mathcal{I}_k$  */
    basis2.int  $\leftarrow list2$ ;

     $\forall$  interval  $I \in list1$  {
        remove( $I$ ,  $list1$ );
        cond_insert( $I$ , basis2.int));
    }

    /* Inicializa  $list2$  para computar  $\mathcal{I}_{k+1}$  */
    list2  $\leftarrow \emptyset$ ;
}

```

Em seguida, apresentamos a função f .

• $f(I2, I1, list)\{$

Inserir os intervalos de $F(\{I2\}, I1)$ na lista " $list$ ".

Entrada: " $I2$ ", intervalo

" I_1 ", intervalo

" $list$ ", lista de intervalos.

Saída : " $list$ ", lista de intervalos modificada.

$\forall p \notin I_2.A \cup I_2.B^c \{ /* I_2 \cap [\emptyset, \{p\}^c] */$

if($p \in I_1.A$) {

$I.A \leftarrow I_2.A;$

$I.B^c \leftarrow (I_2.B^c \cup \{p\});$

insert($I, list2$);

}

else { /* $I_2 \cap [\{p\}, W]$ */

if($p \in I_1.B^c$) {

$I.A \leftarrow (I_2.A \cup \{p\});$

$I.B^c \leftarrow I_2.B^c;$

insert($I, list2$);

}

}

}

Este algoritmo usa a Proposição 3.2 para computar I_1^c e o fato que $I_1^c \cap I_2 \subseteq I_2$. A representação do intervalo $[A, B]$ pelo par (A, B^c) também melhora o desempenho do algoritmo.

7.5 Avaliação do Desempenho

Nesta seção, apresentamos alguns resultados experimentais obtidos com os algoritmos propostos. Estes experimentos foram feitos em uma estação sun SPARC 2. O desempenho dos algoritmos foi medido em termos do número de comparações entre intervalos e do tempo de execução (medidos em segundos).

Implementamos os algoritmos que realizam o ínfimo de duas coleções maximais de intervalos. A Tabela 7.2 mostra alguns resultados experimentais obtidos pela aplicação destes algoritmos.

As colunas desta tabela correspondem, respectivamente, ao número de intervalos e à cardinalidade da janela para as coleções de entrada e saída, ao número de comparações entre intervalos e ao tempo de execução para cada algoritmo implementado.

Note que em todos os experimentos realizados o algoritmo *inf2* foi muito mais eficiente do que o algoritmo *inf1*.

Implementamos também os algoritmos que realizam a operação de complemento. Estes algoritmos correspondem, respectivamente, a implementação do Teorema 3.2, da Proposição 7.2 e do Corolário 7.1. A Tabela 7.3 mostra alguns resultados experimentais obtidos pela aplicação destes algoritmos.

As colunas da tabela 7.3 correspondem, respectivamente, ao número de intervalos da coleção de entrada, à cardinalidade da janela, ao número de intervalos da coleção de saída, ao número de comparações entre intervalos e ao tempo de execução. Os último três itens são repetidos para cada algoritmo implementado.

Note que, em todos os experimentos realizados, o algoritmo baseado no Corolário 7.1 foi mais eficiente do que os outros algoritmos.

$B_{W_1}(\psi_1)$		$B_{W_2}(\psi_2)$		$B_W(\psi)$		infl		inf2	
Núm. Int.	$ W_1 $	Núm. Int.	$ W_2 $	Núm. Int.	$ W $	Número de Comparação	Tempo (sec.)	Número de Comparação	Tempo (sec.)
808	34	1,240	37	964	37	120,918,299	245.0	27,807,299	102,7
1,240	37	808	34	964	37	575,296,715	1201.0	85,738,988	240.0
147	39	147	39	52	41	843,040	2.7	209,783	1.7
658	31	658	31	2738	41	293,190,777	560.5	26,432,636	74.0
2,143	39	2,143	39	2,766	39	3,724,727,754	5,693.0	219,718,589	720.7

Tabela 7.2: Avaliação de desempenho dos algoritmos que computam o ínfimo.
 $B_{W=W_1 \cup W_2}(\psi = \psi_1 \wedge \psi_2) = B_{W_1}(\psi_1) \sqcap B_{W_2}(\psi_2)$.

$B_W(\psi)$		$ B_W(\psi) $	Teorema 3.2		Proposição 7.2		Corolário 7.1	
Núm. Int.	$ W $		Número de Comparação	Tempo (sec.)	Número de Comparação	Tempo (sec.)	Número de Comparação	Tempo (sec.)
26	9	28	31,293	.3	3,119	.2	1,493	.2
13	20	1561	47,154,062	110.4	9,561,412	21.4	3,098,442	8.1
1561	20	13	29,130,033	72.1	42,002,927	84.6	28,946,001	63.8
11	20	448	3,846,928	8.9	1,184,045	2.6	367,701	1.1
448	20	11	2,610,070	7.3	614,036	1.8	323,488	1.4
9	25	270	4,859,590	12.0	23,272,379	4.7	389,741	1.2
729	45	9	4,087,237	13.8	285,583	1.5	155,695	1.2
9	45	729	87,216,593	199.13	121,240,497	261.7	6,975,297	16.1
337	77	136	114,749,443	320.0	4,477,999	9.7	1,536,591	4.1

Tabela 7.3: Avaliação de desempenho dos algoritmos que computam o complemento.

Capítulo 8

Algoritmos para o Aprendizado PAC

Neste capítulo, estudaremos alguns algoritmos consistentes para o aprendizado de funções Booleanas com um domínio finito W .

Uma primeira escolha de um algoritmo de aprendizado consistente poderia ser o algoritmo clássico de Quine-Mccluskey [11, 21, 20] com "don't cares". Este algoritmo dá a função Booleana mais simples (isto é, aquela precisa do menor conjunto de intervalos para ser representada) que satisfaz as restrições.

Contudo, o desempenho deste algoritmo não é bom em casos onde a cardinalidade de W é grande e o número de exemplos é pequeno. Já que estes casos são os mais comuns no projeto de operadores morfológicos por aprendizado PAC, recentemente foi proposto um algoritmo mais adequado: a divisão incremental de intervalos (ISI, do inglês Incremental Splitting of Intervals).

A seguir lembraremos o algoritmo ISI e apresentaremos uma generalização dele, baseado sobre as operações sobre coleções de intervalos estudados previamente.

8.1 Divisão Incremental de Intervalos

Os estados deste algoritmo são as funções Booleanas aprendidas, que são indexadas pela sequência de exemplos de entrada (sem repetições). Para cada novo exemplo (X_{i+1}, b_{i+1}) que chega, a função Booleana corrente f_i , aprendida até o momento, é atualizada para f_{i+1} . O estado inicial é a função constante igual a 1.

O mecanismo de atualização da função aprendida é o seguinte: i- quando um exemplo negativo $(X, 0)$ chega, busca os intervalos fechados que contêm X , elimina X destes inter-

valos e representa a coleção de conjuntos resultante por uma união de intervalos maximais; ii- quando um exemplo positivo $(X, 1)$ chega nada é feito. Quando o processo de aprendizado é finalizado, escolhe-se uma combinação mínima de intervalos fechados, que cubra todos os exemplos positivos.

A Figura 8.1 ilustra o passo i deste algoritmo. Neste exemplo, W é um conjunto com três pontos. Na Figura 8.1a, temos o estágio inicial. Na Figura 8.1b, temos o primeiro estágio, depois da entrada do exemplo $(000, 0)$. Na Figura 8.1c, temos o segundo estágio, depois da entrada do exemplo $(100, 0)$. Na Figura 8.1d, temos o terceiro estágio, depois da entrada do exemplo $(001, 0)$. Note que a eliminação de um elemento de um intervalo, representado por n variáveis Booleanas, induz a criação de n intervalos, representados por $n - 1$ variáveis Booleanas.

A chegada de um exemplo negativo induz a divisão de um intervalo ou de um conjunto de intervalos. A Figura 8.2 mostra a árvore de intervalos que é construída quando os exemplos da Figura 8.1 são processados. Note que os intervalos $[110, 111]$ e $[101, 111]$ são eliminados porque eles estão incluídos, respectivamente, nos intervalos $[010, 111]$ e $[001, 111]$. O intervalo $[011, 111]$ é eliminado porque está incluído no intervalo $[010, 111]$. Assim, o resultado é a coleção $\{[010, 111], [101, 111]\}$.

Figura 8.1: a) Estágio inicial. b) Primeiro estágio. c) Segundo estágio. d) Terceiro estágio.

Figura 8.2: Árvore de intervalos do algoritmo ISI.

8.2 Divisão Incremental de Intervalos Generalizada

A idéia do *algoritmo de Divisão de Intervalos Generalizada* (G-ISI, do inglês Generalized ISI) é a mesma que a do algoritmo ISI original. A única mudança é que no G-ISI os exemplos negativos são representados por intervalos e a divisão é induzida pela eliminação de um intervalo.

Como no algoritmo ISI, os estágios do G-ISI são as funções Booleanas aprendidas e elas são indexadas pela sequência de exemplos de entrada. Para cada nova coleção de exemplos $([X_{i+1}, Y_{i+1}], b_{i+1})$ que chega, a função Booleana f_i , aprendida até o momento, é atualizada para f_{i+1} . Neste algoritmo, o estágio inicial é também a função constante igual a 1.

O mecanismo de atualização da função aprendida é o seguinte: i- quando um intervalo fechado de exemplos negativos $([X, Y], 0)$ chega, busca os intervalos fechados que tem interseção não vazia com $[X, Y]$, elimina $[X, Y]$ destes intervalos e representa a coleção de conjuntos resultante por uma união de intervalos maximais; ii- quando chegam exemplos positivos nada é feito. Quando o processo de aprendizado é finalizado, escolhe-se uma combinação mínima de intervalos, que cubra todos os exemplos positivos.

Para ilustrar o passo i deste algoritmo usamos a mesma aplicação apresentada antes para ilustrar o algoritmo ISI. Note que o domínio dos exemplos negativos $\{000, 100, 001\}$ pode ser representado pela coleção de intervalos $\{[000, 100], [000, 001]\}$. A Figura 8.3 mostra esta aplicação.

Na Figura 8.1a, temos o estágio inicial. Na Figura 8.3b, temos o primeiro estágio, depois da entrada do intervalo $([000, 100], 0)$. Na Figura 8.3c, temos o segundo estágio, depois da entrada do intervalo $([000, 001], 0)$. A Figura 8.2 mostra a árvore de intervalos que é construída.

Note que o intervalo $[011, 111]$ é eliminado porque está incluído no intervalo $[010, 111]$. Assim, o resultado é a coleção $\{[010, 111], [101, 111]\}$.

8.3 Avaliação de Desempenho

Devemos observar que o passo i do algorithm G-ISI é dado exatamente pela expressão do Corolário 7.1. Como provamos que esta expressão dá uma coleção maximal de intervalos, o algoritmo G-ISI é ótimo em termos do número de intervalos produzidos e, conseqüentemente, é equivalente ao algoritmo de Quine-McKluskey.

Comparando as Figuras 8.2 e 8.4, notamos que, nesta aplicação particular, o algoritmo G-ISI é mais eficiente do que o algoritmo ISI.

Note que, o algoritmo ISI pode ser visto como um caso particular do algoritmo G-ISI. Para isto, os exemplos oferecidos ao G-ISI devem ser representados por intervalos degenerados que contêm um único elemento, por exemplo, o exemplo $(X, 0)$ seria representado por $([X, X], 0)$.

A Tabela 8.1 mostra o resultado de alguns experimentos com o algoritmo G-ISI. Cada experimento foi realizado duas vezes, uma com a representação dos exemplos negativos por intervalos maximais (computados pelo algoritmo Quine-McKluskey sem "don't cares") e outra com sua representação por intervalos degenerados. No primeiro caso, o tempo para a organização dos exemplos negativos pelo algoritmo Quine-McKluskey foi também considerada.

A Tabela 8.1 apresenta a cardinalidade da janela, o número de exemplos positivos e negativos, o número de intervalos maximais necessário para representar os exemplos negativos, a avaliação de desempenho dos algoritmos ISI (isto é, o G-ISI alimentados com intervalos degenerados) e G-ISI, em termos do número de comparações entre intervalos.

Note que em alguns casos o desempenho do G-ISI é melhor que o desempenho do ISI e vice-versa.

Figura 8.3: a) Estágio inicial. b) Primeiro estágio. c) Segundo estágio.

Figura 8.4: Árvore de intervalos do algoritmos G-ISI.

$ W $	Ex. Positivos.	Ex. Negativos .	Intervalo Neg.	Número de Comp.	
				ISI	G-ISI
5	16	15	4	72	6
6	30	30	4	233	6
7	53	55	10	145	822
8	93	84	16	3,888	5,360
9	213	224	32	7,532	31,404

Tabela 8.1: Avaliação de desempenho dos algoritmos ISI e G-ISI.

Capítulo 9

Descrição da Toolbox Implementada

Neste capítulo apresentamos uma breve descrição do software implementado. Este sistema foi projetado e implementado como uma Toolbox do sistema khoros [16]. Os programas usam os mesmos formatos de arquivos da toolbox PAC [24]. Em seguida, descrevemos tais programas.

- vnegbas

Descrição:

Computa a base da composição de um operador genérico i.t. e locamente definido, representado pela sua base, com o operador de negação.

Entrada:

basis-in.bas, base de operador.

Saída:

basis-out.bas, base da composição.

- vintbas

Descrição:

Computa a base do ínfimo de dois operadores genéricos i.t e localmente definidos, representados pelas suas bases.

Entrada:

basis1-in.bas,

basis2-in.bas, bases dos operadores de entrada.

Saída:

basis-out.bas base do ínfimo dos operadores de entrada.

- vunibas

Descrição:

Computa a base do supremo de dois operadores genéricos i.t e localmente definidos, representados pelas suas bases.

Entrada:

basis1-in.bas,
basis2-in.bas, bases dos operadores de entrada.

Saída:

basis-out.bas, base do supremo dos operadores de entrada.

- verobas

Descrição:

Computa a base da composição de um operador genérico i.t. e localmente definido, representado pela sua base, com o operador de erosão, representado pelo seu elemento estruturante.

Entrada:

basis-in.bas, base do operador.
ker.str, elemento estruturante do operador erosão.

Saída:

basis-out.bas, base da composição.

- vdilbas

Descrição:

Computa a base da composição de um operador genérico i.t. e localmente definido, representado pela sua base, com o operador de dilatação, representado pelo seu elemento estruturante.

Entrada:

basis-in.bas, base do operador.
ker.str, elemento estruturante do operador dilatação.

Figura 9.1: Menu de programas implementados.

Saída:
basis-out.bas, base da composição.

A Figura 9.1 mostra o menu de programas da toolbox. Neste caso, `vdilbas` é selecionada.

Figura 9.2: Exemplo do uso da toolbox para computar a base de $\psi = \iota \wedge (\nu \varepsilon_{A\iota} \wedge \delta_{B\iota})$.

A Figura 9.2 exemplifica o uso da toolbox, apresentando um programa visual que executa o cálculo incremental da base do operador $\psi = \iota \wedge (\nu \varepsilon_{A\iota} \wedge \delta_{B\iota})$, com $A =$

$$A = \begin{pmatrix} 111 \\ 000 \\ 000 \end{pmatrix}$$

e $B =$

$$B = \begin{pmatrix} 111 \\ 000 \\ 000 \end{pmatrix}.$$

As caixinhas vwin e vinterv, pertencentes à toolbox PAC são utilizadas para criar a base da identidade. A caixinha vstrset pertencente da toolbox MMach [4] que é utilizada para criar os elementos estruturantes que são as entradas de verobas e vdilbas. Na figura se pode visualizar a base resultante.

Figura 9.3: Uso da toolbox que mostra duas alternativas para computar o ínfimo de dois operadores.

A Figura 9.3 mostra um exemplo de isomorfismo entre os reticulados $(\Psi_W, \leq)$ e $(\Pi_W, \leq)$. A caixa *vunisup*, pertencente à toolbox PAC, computa a união de sup-geradoras sob uma imagem. A caixa *vinf*, pertencente à toolbox MMach, computa o ínfimo de duas imagens.

As entradas deste exemplo são, uma imagem "Imag-in" e duas bases Base-1 e Base-2. Na parte superior da figura, primeiro computamos sobre a imagem de entrada, a união de sup-geradoras correspondente à Base-1 e à Base-2, obtendo, respectivamente, as imagens Imag-1 e Imag-2. Em seguida, computamos o ínfimo destas imagens obtendo Imag-out1. Na parte inferior da figura, apresentamos o outro caminho, primeiro calculamos $\text{Base-1} \sqcap \text{Base-2}$ e, então, aplicamos a união de sup-geradoras (correspondente a esta base) sobre a imagem de entrada, obtendo Imag-out2.

Capítulo 10

Conclusão

Uma representação eficiente de coleções de subconjuntos é dada por suas respectivas coleções de intervalos maximais. Para manipular estas representações, precisamos definir operações sobre coleções de subconjuntos em termos de seus intervalos maximais.

Nesta trabalho, apresentamos alguns algoritmos rápidos para computar operações sobre coleções de intervalos. Mostramos também como aplicar estes algoritmos para resolver dois problemas relacionados com a programação automática de MMach's: a computação incremental da base e o aprendizado de operadores de conjuntos i.t. e localmente definidos.

Os algoritmos propostos baseiam-se em uma sólida teoria matemática que estuda as operações sobre coleções de subconjuntos, representadas por coleções de intervalos. Um fato fundamental nesta teoria é que, para um subconjunto finito W de um conjunto E , equipado com um grupo Abelian $+$, os posets $(\mathcal{P}(\mathcal{P}(W)), \subseteq)$, $(\Pi_W, \leq)$, $(\{0, 1\}^{\mathcal{P}(W)}, \leq)$ e $(\Psi_W, \leq)$ são reticulados completos Booleanos isomórficos.

Como as erosões e dilatações são construídas a partir de operações sobre conjuntos, suas bases são construídas a partir das correspondentes operações sobre intervalos. Como os operadores de conjunto podem ser decompostos em termos de erosões e dilatações, suas bases podem ser computadas a partir das bases destes operadores elementares. Assim, os algoritmos que realizam operações sobre coleções de intervalos constituem o núcleo de um sistema de software para a computação incremental da base de operadores de conjunto.

A computação incremental da base pode ser aplicada na prática para a paralelização de programas de MMach's ou para a prova automática da equivalência entre operadores morfológicos distintos.

Os algoritmos para a simplificação de funções Booleanas com "don't cares" tem sido

usadas com sucesso no projeto automático de operadores de conjuntos. O mais importante algoritmo desta classe é devido a Quine-McKluskey, proposto na década de sessenta. Este algoritmo dá um resultado ótimo em termos do número de intervalos maximais gerados, mas é extremadamente custoso em termos de memória e tempo de processamento. Recentemente, foi proposto o algoritmo ISI para tratar eficientemente os casos em que a cardinalidade da janela é grande e o número de exemplos é relativamente pequeno. Aplicamos as operações sobre intervalos para generalizar o algoritmo ISI. Apresentamos alguns resultados experimentais que mostram que em alguns casos a generalização proposta melhora a eficiência do algoritmo ISI. Apresentamos também, uma prova formal da equivalência (em termos do número intervalos produzidos) entre os algoritmo G-ISI e Quine-McKluskey.

O algoritmo de aprendizado apresentado aqui no contexto de projeto de operadores de conjunto pode ser aplicado também na área de arquitetura de computadores para o projeto de circuitos lógicos.

Outra contribuição deste trabalho foi a dedução das expressões de decomposição canônica para a família de operadores localmente definido e i.t., que generaliza os resultados conhecidos para a família dos operadores i.t.

Os resultados computacionais apresentados neste trabalho foram implementados como uma toolbox para o sistema khoros [16].

Finalmente, este trabalho deve ser a base para o estudo da transformação da representação canônica em outras estruturas de representação que usam um menor número de erosões e dilatações. Este problema, que pode ser entendido como o inverso da computação incremental da base, é extremamente complexo, mas deve ter um papel importante no projeto de um sistema que realize a geração automática de programas eficientes para MMach's.

Bibliografia

- [1] M. Anthony and N. Biggs. *Computational Learning Theory - An Introduction*. Cambridge University Press, 1992.
- [2] G. J. F. Banon and J. Barrera. Morphological filtering for stripping correction of spot images. *Photogrammetria(PRS)*, 43:195–205, 1989.
- [3] G. J. F. Banon and J. Barrera. Minimal representation for translation-invariant set mappings by mathematical morphology. *SIAM J. Appl. Math.*, 51:1782–1798, 1991.
- [4] J. Barrera, G. Banon, and R. Lotufo. A mathematical morphology toolbox for the khoros system. In SPIE, editor, *Image algebra and Morphological Image Processing V*, july 1994.
- [5] J. Barrera and G. J. F. Banon. Expressiveness of the morphological language. In *Image Algebra and Morphological Image Processing III*, volume 1769, pages 264–275, San Diego, California, 1992. SPIE.
- [6] J. Barrera, F. S. C. da Silva, and G. J. F. Banon. Automatic programming of binary morphological machines. In *Image Algebra and Morphological Image Processing*, volume 2300 of *SPIE Proceedings*, pages 229–240, San Diego, 1994.
- [7] J. Barrera, N. S. Tomita, F. S. C. da Silva, and G. J. F. Banon. Automatic programming of binary morphological machines by pac learning. In *Neural, Morphological and Stochastic Methods in Image and Signal Processing*, To appear, San Diego, 1995.
- [8] G. Birkhoff. *Lattice Theory*. American Mathematical Society, 3 edition, 1967.
- [9] J. Goutsias. Morphological analysis of discrete random shapes. *Journal of Mathematical Imaging and Vision*, 2:193–215, 1992.
- [10] C. Gratin. Élaboration d'un logiciel de morphologie mathématique sur micro-ordinateur. Rapport interme, Ecole Nationale Supérieure des Mines de Paris, 1989.
- [11] F. J. Hill and G. R. Peterson. *Introduction to Switching Theory and Logical Design*. John Wiley, 1981.

- [12] R. Jones. Transformation of the basis representation into cascada representations. In *International Workshop on Mathematical Morphology and its Applications to Signal Processing*, pages 239–244, 1993.
- [13] R. Jones. *Basis Algorithms in Mathematical Morphology*. PhD thesis, Phys. Dep. Faculty of Science, Monash University, Australia, 1994.
- [14] J. C. Klein and R. Peyhard. Pimm1, an image processing asic based on mathematical morphological. In Rochester, editor, *IEEE's AASIC Seminar and Exhibit*, pages 25–28, New York, 1989.
- [15] J. C. Klein and J. Serra. The texture analyser. *Journal of Microscopy*, 95(2):343–356, April 1972.
- [16] K. Konstantinides and J. R. Rasure. The khoros software development environment for image and signal processing. *IEEE Transactions on Image Processing*, 3(3):243–252, 1994.
- [17] R. P. Loce. *Morphological Filter Mean-Absolute-Error Representation Theorems and Their Application to Optimal Morphological Filter Design*. PhD thesis, Center of Image Processing - Rochester Institute of Technology, 1993.
- [18] R. P. Loce and E. R. Dougherty. Optimal restoration using the morphological hit-miss transform. *SPIE, San Diego*, 1769:94–105, 1992.
- [19] G. Matheron. *Random Sets and Integral Geometry*. John Wiley, New York, 1975.
- [20] E. J. McCluskey. *Introduction to the Theory of Switching Circuits*. McGraw-Hill, 1965.
- [21] W. V. Quine. The problem of simplifying truth functions. *Am. Math. Monthly*, 59(8):1417–1444, October 1952.
- [22] J. Serra. *Image Analysis and Mathematical Morphology*. Academic Press, London, 1982.
- [23] J. Serra, editor. *Image Analysis and Mathematical Morphology. II: Theoretical Advances*. Academic Press, London, 1988.
- [24] Nina Sumiko Tomita. Progromação automática de máquinas morfólogica binária baseado em aprendizado pac. *Dissertação de mestrado a ser apresentada no curso de Matemática Aplicada do IME-USP*, 1996.
- [25] L. Valiant. A theory of the learnable. *Comm. ACM*, 27:1134–1142, 1984.