**Sistemas de Reescrita de Termos:
Teoria e uma Aplicação.**

Jefferson Zanutto

DISSERTAÇÃO APRESENTADA
AO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA
DA
UNIVERSIDADE DE SÃO PAULO
PARA A
OBTENÇÃO DO GRAU DE
MESTRE EM MATEMÁTICA APLICADA.

Área de Concentração: **Ciência da Computação**

Orientador: Prof. Dr. *Flávio Soares Corrêa da Silva*

Apoio Financeiro: CAPES

Instituto de Matemática e Estatística da Universidade de São Paulo

- São Paulo, maio de 1999 -

Sistemas de Reescrita de Termos: Teoria e uma Aplicação.

Este exemplar corresponde à redação
final da dissertação devidamente corrigida
e defendida por Jefferson Zanutto
e aprovada pela comissão julgadora.

São Paulo, 21 de junho de 1999.

Banca examinadora:

- Prof. Dr. *Flávio Soares Corrêa da Silva* (Presidente)
- Prof. Dr. *Marcelo Finger*
- Prof. Dr. *Wamberto Weber Vasconcelos*

IME-USP.

IME-USP.

UECE.

*“Nunca será um verdadeiro matemático
aquele que não for um pouco poeta.”*

Karl Weierstrass (1815-1897)
(em uma carta a Sophie Kovalewski).

*Aos meus pais, em especial à minha mãe,
pelo carinho, apoio e amor infinito
que dedica diariamente a seus filhos.*

Agradecimentos

A Flávio Soares, pela orientação, paciência e revisão atenta do texto.

A Dri, por sempre ter acreditado que eu seria capaz de concluir este trabalho, fortalecendo-me em seu carinho e dedicação; por ter sido fonte de inspiração para a conclusão desta dissertação.

À minha família, pelo apoio irrestrito, mesmo nos momentos mais difíceis.

A Simone, Robinson e Christian, pelo carinho que nos une.

A Nê, que sempre acreditou que este dia chegaria; por ter mostrado-me que é com dedicação que se colhe os melhores frutos desta vida.

A Munir, pela amizade de tantos anos; pelo convívio harmonioso em nosso lar.

A Nara e Helena pelo carinho e amizade tão legal.

A Flávio Shigueo, por ser uma pessoa incrível com quem aprendi e aprendo muito; por ter estado sempre disposto a ouvir, com enorme paciência, minhas dúvidas; pelas conversas e bate-papos sobre matemática, futebol e por ter oferecido-me valiosa hospedagem no CRUSP, logo que nos conhecemos.

A Sonti, “fala aí lógico!”, por sempre ter ajudado-me a sanar minhas dúvidas sobre lógica.

A Scal, pela amizade e companheirismo.

A Luciano, por ter sido tão prestativo e amigo.

A Juiti, por sempre ter ajudado-me a resolver minhas diferenças com o Latex e pelas preciosas dicas sobre os recursos do Unix.

A Afonso e Serginho, pelos momentos agradáveis que passamos juntos no CRUSP.

A Claude Marché, que me apresentou o sistema Cime; por ter respondido prontamente minhas dúvidas sobre o tema estudado; por ter sido tão prestativo e gentil nos e-mails.

A Andréa Loparic, pelo excelente curso introdutório à Logica, ministrado na FFLCH-USP, e por sempre ter respondido às minhas dúvidas com muita dedicação, por mais simples que elas pudessem parecer.

A Leucio, pela dedicação prestada ao fornecer uma cópia de seu trabalho de mestrado, bem como cópias de diversos artigos da área de sistemas de reescrita.

A Anjolina, por ter fornecido-me uma cópia de sua dissertação e pelas palavras de apoio e incentivo enviadas por e-mail recentemente.

Sumário

Dada uma teoria equacional T , isto é, um conjunto T de equações, uma pergunta que se faz é se os teoremas da teoria T podem, ou não, ser verificados através de um algoritmo que termina.

Uma abordagem à esta questão, muito divulgada a partir da final da década de sessenta com o trabalho de Knuth & Bendix, baseia-se na busca de sistemas de reescrita de termos completos para a teoria T dada, isto é, sistemas de reescrita completos que induzam as mesmas classes de congruência da teoria T .

Muitos têm sido os esforços nas últimas décadas para o desenvolvimento de técnicas de completação de sistemas de reescrita com o intuito de conseguir-se obter sistemas completos para uma variedade cada vez maior de teorias equacionais.

Neste trabalho, estudamos os aspectos básicos de teorias equacionais, incluindo seus sub-problemas como a unificação de termos e a terminação de sistemas de reescrita.

Descrevemos dois algoritmos de completação de sistemas de reescrita, o algoritmo clássico de Knuth & Bendix e o algoritmo de completação módulo teorial equacional, de Peterson & Stickel, generalizando o anterior.

Por fim, apresentamos uma aplicação do uso de sistemas de reescrita para a Morfologia Matemática, uma área da computação com aplicações diretas no campo de processamento de imagens. Tanto a verificação da identidade entre operadores morfológicos quanto a simplificação de tais operadores são problemas de interesse no campo da morfologia matemática.

Apresentamos, para a sub-classe dos operadores morfológicos invariantes por translação e isotônicos para imagens binárias, um método efetivo para a verificação de identidade entre tais operadores morfológicos, que se materializa na existência de um sistema de reescrita completo para a teoria dos reticulados distributivos. Discutimos, por fim, a aplicabilidade dessa abordagem para a simplificação de operadores morfológicos, tanto para o caso de implementações desses operadores em máquinas sequenciais quanto em máquinas paralelas.

Abstract

Given an equational theory T , that is, a set T of equations, an important question that can be posed concerns the existence of an algorithm that has the termination property, and which could be used to deduce all theorems of T .

An approach to that question, well known since the late sixties after a paper by Knuth & Bendix, is based on the search for complete term rewriting systems with respect to the given equation theory T , that is, complete rewriting systems which induce the same congruence classes of theory T .

Much effort has been put on the last few decades to develop new techniques to generate complete term rewriting systems to a diversity of equational theories.

In this text we study the basic aspects of equational theories, including the problems of term unification and termination.

We describe two algorithms for completion of term rewriting systems: the first is the classic one, due to Knuth & Bendix, and the second one is an algorithm for completion modulo equational theories, due to Peterson & Stickel, which generalizes the first one.

We also show an application of term rewriting systems to Mathematical Morphology, a research area of computer science with a variety of applications to image processing. The verification of identity between morphological operators and the simplification of these operators are problems of interest to Mathematical Morphology.

We present, for the subclass of morphological operators that are isotonic and translation-invariant, an effective method to check the identity between such morphological operators, based on the existence of complete term rewriting system for distributive lattices.

Finally, we discuss the applicability of our approach with respect to the simplification of morphological operators, when these operators are implemented in sequential machines as well as in parallel machines.

Índice

Lista de Figuras	iii
Lista de Tabelas	vi
Lista de Algoritmos	vii
Introdução	1
1 Motivação e Objetivos	7
2 Preliminares	11
2.1 Relações	13
2.2 Árvores	18
2.3 Termos e Substituições	22
3 Teorias Equacionais e Sistemas de Reescrita de Termos	29
3.1 Sintaxe das Teorias Equacionais	29
3.2 Alguns Resultados Básicos em Teorias Equacionais	31
3.3 Semântica das Teorias Equacionais	33
3.4 Sistemas de Reescrita de Termos	40
4 Unificação de Termos	47
4.1 Unificação Comutativa	58
4.2 Unificação Associativa	60
4.3 Unificação Associativa-Comutativa	62
4.3.1 Soluções de Equações Diofantinas Homogêneas Lineares	63

4.3.2	Um Algoritmo de Unificação de Termos- AC	75
5	O Problema da Terminação de SRT's	89
5.1	SRT's Não Noetherianos: Uma Condição Necessária	92
5.2	Métodos para Provar a Terminação de SRT's	97
5.3	Terminação Módulo T	124
6	Completação de Sistemas de Reescrita de Termos	129
7	Completação Módulo T de SRT's	147
7.1	Método das Classes de Equivalência	148
7.2	Compatibilidade de SRT's com Teorias Equacionais	159
7.3	Compatibilidade com T para Teorias Associativas-Comutativas	174
7.4	Um Algoritmo de Completação Módulo T de SRT's.	176
7.5	Uma Implementação do Algoritmo de Completação	180
8	Morfologia Matemática	187
9	Verificação de Identidade e Simplificação de Operadores Morfológicos	203
10	Considerações finais	247
10.1	Perspectivas Futuras	249
	Apêndices:	249
A	O Estado da Arte em SRT's	251
B	Passos da Completação do SRT para a Teoria dos Reticulados Distributivos	257
C	Algumas Estatísticas	267
C.1	Exemplo 9.34	267
C.2	Exemplo 9.35	271
C.3	Exemplo 9.36	273
C.4	Exemplo 9.37	275
C.5	Exemplo 9.38	277

Referências Bibliográficas

279

Índice Remissivo

286

Lista de Figuras

1	Discos sobrepostos.	1
2	Estimação da quantidade de discos.	2
2.1	Diagrama representativo de uma árvore.	21
3.1	Confluência de Sistema de Reescrita.	44
3.2	Confluência local.	45
4.1	Diagrama representativo das classes de congruência induzidas por $\equiv_E$	56
4.2	Diagrama de recursão para uma instância do algoritmo UNIF_C.	61
5.1	Representação de um termo s homeomorficamente embutido em um termo t . . .	94
5.2	Representação de dois termos básicos que não são homeomórficos entre si. . . .	95
5.3	Representação de um termo s homeomorficamente embutido em um termo t . . .	95
5.4	Diagrama da recursão para $-(0 + 1) \succeq 0 + 1$	96
6.1	Diamante de Newman.	131
6.2	Confluência local na reescrita de um termo quando os redex's são disjuntos. . . .	132
6.3	A confluência local é garantida quando um redex é ancestral do outro e este outro corresponde a uma posição de variável da regra de reescrita.	133
6.4	As ocorrências de um subtermo são, duas a duas, disjuntas quando o subtermo corresponde a uma variável.	136
6.5	Diagrama detalhado de confluência local quando um redex é ancestral do outro. .	139
6.6	$l_2\theta_2$ não pertence à estrutura de l_1	140
6.7	$l_2\theta_2 = (l_1\theta_1)/v$	141
6.8	Propriedade Church-Rosser.	141

7.1	$s \xrightarrow{(R,T)} t \Leftrightarrow u \xrightarrow{R} u[p \leftarrow r\theta]$	151
7.2	Diagrama ilustrativo sobre a demonstração do teorema 7.16.	154
7.3	Confluência módulo T.	155
7.4	Um modo equivalente de expressar a confluência módulo T.	155
7.5	Church-Rosser módulo T.	156
7.6	Par crítico $\langle [u\sigma]_T, [s[k_1 \leftarrow v]\sigma]_T \rangle$ módulo T.	160
7.7	Convergência do par crítico é garantida quando $k \perp n$	166
7.8	Diagrama representativo da situação 2 ($n \preceq k$).	166
7.9	$r\psi[m_2 \leftarrow t'] = r\psi'$	171
7.10	$l\psi' = l\psi[m_1 \leftarrow t']$	171
7.11	$u[k \leftarrow r_1\theta] = u[n \leftarrow u/n[k/n \leftarrow r_1\theta]]$	173
8.1	Bijecção entre as dilatações e as funções estruturantes.	192
8.2	Soma de Minkowski de dois subconjuntos.	194
8.3	Diferença de Minkowski de dois subconjuntos.	195
8.4	Três exemplos de soma de Minkowski.	197
8.5	Bijecção entre as dilatações invariantes por translação e os conjuntos.	199
8.6	Dilatação de um conjunto por um elemento estruturante.	200
8.7	Dois modos de construir o dilatado.	200
8.8	Erosão de um conjunto por um elemento estruturante.	201
8.9	Modo de construir o erodido.	201
9.1	Decomposição em termos de subconjuntos do quadrado elementar.	206
9.2	Decomposição de um subconjunto em termos de uniões de translações da origem.	207
9.3	Estrutura de árvore do operador $(\delta_A \circ \epsilon_A) \vee \delta_B$	211
9.4	Diagrama de árvore do termo irreduzível obtido pelo exemplo 9.25.	224
9.5	Convergência fica garantida quando o subtermo onde ocorre a superposição corresponde a uma posição de uma variável.	227
A.1	Decidibilidade do problema da palavra de teorias fechadas módulo E, para alguns E's	255

Lista de Tabelas

4.1	Base das soluções não negativas da equação diofantina $2 \cdot x_1 + x_2 + x_3 = 2 \cdot y_1 + y_2$.	78
9.1	Sintaxe da Linguagem Morfológica restrita aos operadores i.t. e isotônicos.	205
9.2	Semântica da Linguagem Morfológica restrita aos operadores i.t. e isotônicos. . .	205
9.3	Complexidades de alguns operadores.	245
9.4	Estatísticas fornecidas pelo sistema Címe para o cálculo de alguns operadores morfológicos canônicos reconstruídos.	246

Lista de Algoritmos

1.1	Algoritmo de unificação sintática de termos.	53
1.2	Algoritmo de unificação comutativa de termos.	59
1.3	Algoritmo de completção de Knuth e Bendix.	143
1.4	Algoritmo de completção módulo T de Peterson e Stickel.	182

Introdução

Segundo Klop [Klo87], o estudo de Sistemas de Reescrita de Termos tem sua origem nos estudos de Lambda Cálculo e Lógica Combinatória, na década de trinta, desenvolvidos inicialmente com o objetivo de formalizar o conceito de funções computáveis.

O estudo de Sistemas de Reescrita de Termos surgiu da motivação de encontrar-se um método para simplificar expressões de modo automático. Segundo Dershowitz e Jouanaud [DJ90], na segunda metade da década de sessenta, Gorn [Gor67] apresentou pela primeira vez um sistema de reescrita de termos na forma de um programa de computador.

Paralelamente, também na década de sessenta, deu-se início ao desenvolvimento de técnicas capazes de solucionar problemas relacionados à análise de imagens, quando a NASA teve a necessidade de melhorar a qualidade de imagens recebidas provenientes de suas sondas espaciais. Esta tecnologia veio mostrar-se mais tarde útil à solução de outros problemas relacionados à análise de imagens, nas áreas de geologia e medicina, entre outras.

Ainda na década de sessenta, um grupo de pesquisa da École Supérieure des Mines de Paris, em Fontainebleau, liderado por Georges Matheron e Jean Serra começou a estudar a Morfologia Matemática, uma nova abordagem à solução de problemas de análise de imagens que se baseava em extrair informações de imagens a partir de transformações de formas.

Como exemplo, suponha que uma imagem seja formada por vários objetos em forma de discos, alguns parcialmente sobrepostos, como mostra a figura 1.

Figura 1: Discos sobrepostos.

Podemos através de um operador morfológico ϕ estimar a quantidade de discos existentes em uma tal imagem. O operador ϕ poderia realizar uma sequência de aplicações de um operador ψ ($\phi = \psi \circ \dots \circ \psi$), que por sua vez realizaria primeiro uma erosão, causando uma “separação” parcial dos discos sobrepostos e em seguida identificaria, através de um procedimento de contagem, o número de componentes conexas existentes na imagem resultante da erosão efetuada. A comparação da quantidade de discos identificados após a execução da n -ésima e $(n-1)$ -ésima erosões poderia ser utilizada no processo de decisão para a realização da estimação final do número de discos existentes. A figura abaixo ilustra o efeito de aplicações sucessivas de um tal operador ψ sobre a imagem apresentada acima, causando a separação dos discos e possibilitando a estimação da quantidade de discos existentes.

Observando a figura abaixo, notamos que, como consequência da segunda aplicação do operador ψ , houve o desaparecimento de um dos discos. A aplicação de uma erosão sobre uma imagem acarretará no desaparecimento desta sempre que a imagem em questão puder ser “vista” como um subconjunto estrito do parâmetro de sondagem (elemento estruturante) usado pela erosão.

Figura 2: Estimação da quantidade de discos.

A Morfologia Matemática pode ser descrita através do uso de uma linguagem formal [BB92, BB94]. Sistemas de Reescrita de Termos utilizam-se da caracterização formal de linguagens, a partir da qual pode-se resolver o problema de simplificação de expressões e verificação de equivalência entre expressões.

As transformações sobre imagens são realizadas pela Morfologia Matemática através de expressões que chamamos de operadores morfológicos. Um pesquisador em Morfologia Matemática pode solucionar um problema através da construção de um operador morfológico ψ_1 , porém o mesmo problema, se apresentado a um outro pesquisador, pode ser solucionado por um outro operador morfológico ψ_2 . Se de fato, ambos ψ_1 e ψ_2 solucionam o problema proposto, qual deles é mais eficiente, isto é, qual possui um *custo computacional* menor?

Podemos ainda estar interessados em saber se esses operadores realmente realizam a mesma tarefa, isto é, se para toda imagem X , se Y_1 é a imagem resultante da aplicação do operador ψ_1 sobre X e Y_2 é a imagem resultante da aplicação de ψ_2 sobre X , então $Y_1 = Y_2$.

Essas perguntas podem ser respondidas a partir da noção de simplificação das expressões morfológicas (operadores morfológicos) envolvidas.

Neste trabalho de mestrado, propõe-se uma releitura da Morfologia Matemática como uma linguagem formal e o estudo das características dessa linguagem sob o prisma de Sistema de Reescrita de Termos. Esperamos com isto que essas características nos revelem propriedades que indicarão possibilidades de tratamento da Morfologia Matemática com técnicas de dedução automática.

A Morfologia Matemática é constituída por uma gramática e uma semântica.

Uma gramática é o conjunto de expressões corretas segundo uma *sintaxe*, que por sua vez é formada por um conjunto de símbolos funcionais, um conjunto de variáveis e um conjunto de regras para a formação de expressões.

Uma semântica caracteriza um *modelo* para uma dada gramática e busca atribuir significados (elementos num universo de discurso) para os símbolos interpretáveis da linguagem.

Veremos que a semântica da Morfologia Matemática dos operadores (termos) invariantes por translação e isotônicos pode ser caracterizada através de uma teoria equacional e que o problema de verificação de equivalência entre operadores morfológicos podem ser tratados de modo sintático assumindo um conjunto regras de inferência que Birkhoff [Bir35] mostrou ser completo para o contexto de teorias equacionais de primeira ordem.

O texto desta dissertação está organizado do seguinte modo:

- Capítulo 1: Apresentamos a motivação que nos levou ao desenvolvimento deste trabalho, bem como os objetivos do mesmo.
- Capítulo 2: Este foi destinado a conter notações e definições que formalizam conceitos preliminares necessários para um bom entendimento do texto. Apenas conceitos de natureza mais geral para este trabalho foram incluídos neste capítulo, de modo que muitas definições e notações, de natureza mais específica a um ou outro capítulo, foram incluídas em capítulos apropriados no decorrer do texto.
- Capítulo 3: Destinamos este capítulo à apresentação dos conceitos básicos sobre teorias equacionais, sua sintaxe e semântica, bem como uma introdução, de forma resumida, a sistemas de reescrita de termos. Procuramos deixar claro a conexão existente entre teorias equacionais e sistemas de reescrita de termos bem como a forte relação entre sistemas de reescrita e a ciência da computação.
- Capítulo 4: Boa parte do texto desta dissertação é destinada a descrever um algoritmo de completção de sistemas de reescrita. Porém, a tarefa de completção de sistemas de reescrita

possui como sub-problema a unificação de termos. Em geral, o problema de unificação de termos é indecidível, entretanto podemos, em alguns casos particulares, descrever algoritmos de unificação corretos e completos, como é o caso da unificação de termos de teorias associativas-comutativas. Muitas teorias equacionais que têm aplicações interessantes, possuem operadores com a característica de serem associativos-comutativos. Como veremos, este também é o caso da aplicação de sistemas de reescrita que apresentamos nos capítulos finais deste texto. Apresentamos então, neste capítulo, conceitos e idéias básicas sobre unificação de termos e descrevemos um algoritmo de unificação para teorias associativas-comutativas.

o Capítulo 5: A terminação é uma das propriedades necessárias para que um sistema de reescrita de termos seja completo. Assim como o problema da unificação, o da terminação também é indecidível para o caso geral, entretanto podemos, em certas circunstâncias, garantir a propriedade de terminação de sistemas de reescrita específicos.

Neste capítulo, apresentamos resultados básicos sobre o problema de terminação de sistemas de reescrita, que nos serão úteis em capítulos posteriores para demonstrarmos a propriedade de terminação dos sistemas de reescrita de nosso interesse.

o Capítulo 6: Apresentamos as idéias básicas sobre a concepção do algoritmo clássico, de Knuth e Bendix [KB70], de completação de sistemas de reescrita de termos. Observando um resultado muito conhecido no campo da álgebra, a saber, o Lema do Diamante, de Newman [New42], Knuth e Bendix conseguem mostrar que a completeza de sistemas de reescrita noetherianos fica garantida quando a convergência dos pares críticos do sistema é assegurada.

Algumas extensões sobre o algoritmo clássico de completação têm sido propostas desde então, entretanto as idéias originais devidas a Knuth e Bendix permanecem até os dias de hoje, pois os novos algoritmos de completação de sistema de reescrita também baseiam-se no paradigma da convergência dos pares críticos, introduzido por estes autores.

Um bom entendimento das idéias e conceitos contidos no algoritmo clássico de completação é um pré-requisito para que se possa compreender os novos algoritmos propostos sobre completação de sistemas de reescrita.

Sendo assim, neste capítulo procuramos apresentar de modo introdutório, porém com o devido rigor, o algoritmo clássico de completação.

o Capítulo 7: Apresentamos um algoritmo de completação módulo T (uma generalização do algoritmo clássico de completação) de Peterson e Stickel [PS81], bem como alguns exemplos de completação de algumas teorias associativas-comutativas, completadas através deste algoritmo. Como veremos, tal algoritmo consegue completar teorias equacionais de nosso interesse, como é o caso da teoria dos reticulados distributivos.

◦ Capítulo 8: Neste trabalho, apresentamos uma aplicação do uso de sistemas de reescrita de termos para a morfologia matemática. Neste capítulo, introduzimos os conceitos básicos sobre a morfologia matemática.

◦ Capítulo 9: Uma aplicação do uso de sistemas de reescrita é apresentada neste capítulo. O problema da palavra em sistemas de reescrita aqui materializa-se no problema da verificação da identidade entre operadores morfológicos. Mostramos que, para a classe restrita de operadores morfológicos invariantes por translação e isotônicos, a verificação da identidade entre tais operadores fica garantida pela existência de um sistema de reescrita completo para a teoria dos reticulados distributivos.

Na abordagem dada neste trabalho, a verificação da identidade entre os operadores morfológicos resulta no cálculo de operadores morfológicos canônicos.

Apresentamos uma noção de complexidade de operadores morfológicos e mostramos que, para a noção de complexidade dada, os operadores canônicos obtidos não são ótimos para todos os casos e, portanto, nem sempre o operador canônico obtido é mais simples (segundo a noção de complexidade dada) que o operador inicialmente fornecido ao algoritmo de reescrita.

Por fim, conjecturamos que os operadores canônicos reconstruídos são ótimos, segundo nossa noção de complexidade, quando consideramos uma implementação paralela de tais operadores.

◦ Capítulo 10: Apresentamos considerações gerais sobre o que foi realizado no decorrer desta dissertação, comentando os resultados e apresentando perspectivas sobre possíveis trabalhos futuros.

◦ Apêndice A: Descrevemos sucintamente o atual estado da arte em sistemas de reescrita de termos.

◦ Apêndice B: Discutimos os passos principais que ocorrem durante a execução do algoritmo de completção visto no capítulo 7 para a teoria dos reticulados distributivos.

◦ Apêndice C: O termos irreduzíveis em alguns exemplos fornecidos no capítulo 9 foram calculados usando-se o software *Cime*. Tal sistema fornece algumas estatísticas contendo informações como o tempo de execução de máquina e quantidade de matchings realizados para o cálculo dos termos irreduzíveis. Neste apêndice, apresentamos as estatísticas dadas pelo sistema *Cime*.

Capítulo 1

Motivação e Objetivos

Sabemos, pela aritmética usual dos números inteiros, que o termo $(-(-a) + (-a)) + a$ tem o mesmo significado que os termos $(-(-a) + (-a)) + a$, $0 + a$ e a . É comum, neste caso, dizer que estes termos são “iguais” e escreve-se usualmente $0 + a = a$, $(-(-a) + (-a)) + a = (-(-a) + (-a)) + a$, etc. Uma das áreas de estudo em ciência da computação tem sido a verificação automática de igualdade entre expressões, com aplicações imediatas em problemas como prova automática de teoremas, programação em lógica e verificação de consistência na especificação de tipos abstratos de dados.

O interesse pelo tratamento automático, também chamado de tratamento efetivo (através de um algoritmo), de problemas da matemática, como a simplificação de termos e demonstração de teoremas, surgiu antes mesmo do surgimento de uma tecnologia que possibilitasse a implementação em máquina de tais idéias.

Voltando ao nosso exemplo, podemos garantir que $0 + a = a$ pois sabemos previamente que estamos admitindo como verdade pré-estabelecida que todo termo da forma $0 + x = x$, onde x é uma variável. Substituindo a a no lugar de x temos $0 + a = a$.

De fato, a aritmética usual dos inteiros satisfaz a estrutura algébrica usual de grupos, onde é válido o esquema E de axiomas equacionais abaixo:

$$A1: 0 + x = x$$

$$A2: x + 0 = x$$

$$A3: -x + x = 0$$

$$A4: (x + y) + z = x + (y + z)$$

E temos, por exemplo, $(-(-a) + (-a)) + a \stackrel{A3}{=} 0 + a \stackrel{A1}{=} a$. Observe que o termo a resultante das aplicações dos axiomas dados acima é justamente aquele *menos complexo*

(ou seja, o mais curto) dentre todos os termos que possuem o mesmo significado do termo $(-(-a) + (-a)) + a$ inicialmente considerado.

Suponha agora que queiramos implementar um programa de computador que faça deduções como a exposta acima. É interessante podermos garantir que toda seqüência de reescrita de termos realizada por este programa seja finita (propriedade noetheriana).

Observe que a simples aplicação das regras (axiomas equacionais) expostas acima não garante que toda seqüência de dedução seja finita, pois poderíamos fazer sucessivas aplicações das regras A3 e A1 sobre o termo avaliado inicialmente:

$$(-(-a) + (-a)) + a \xrightarrow{A3} 0 + a \xrightarrow{A1} a \xrightarrow{A1} 0 + a \xrightarrow{A3} (-(-a) + (-a)) + a \cdots$$

Os Sistemas de Reescrita de Termos tentam garantir que toda seqüência de dedução seja finita dando uma orientabilidade às equações. As deduções são feitas a partir de um conjunto de regras de reescrita gerado inicialmente a partir do conjunto de axiomas considerado, garantindo que a cada reescrita de um termo s o termo resultante t seja menos complexo que s no sentido que existe uma relação de ordem estrita $\succ$ imposta sobre os termos, de modo que se deduzimos t a partir de s então $s \succ t$, ainda garantindo que em toda seqüência S de reescrita exista um termo u em S tal que $u \not\succ u'$, para qualquer u' em S , com $u' \neq u$. Para tal é necessário e suficiente garantir que a relação $\succ$ imposta sobre os termos seja bem-fundada (noetheriana) para todas as derivações de S .

Dependendo da estrutura algébrica considerada, nem sempre é uma tarefa fácil encontrar uma tal relação de ordem noetheriana. Em certos casos tal relação inexiste, por exemplo, se a estrutura algébrica considerada for comutativa.

Outra área de estudo da ciência da computação, aparentemente desconexa da área de Sistemas de Reescrita de Termos, é a Morfologia Matemática, uma teoria desenvolvida a partir da década de sessenta, tendo como motivação e aplicação principal o tratamento de imagens. As áreas de aplicação da Morfologia Matemática são as mais variadas. Na agricultura, pode-se medir o volume de uma safra a partir da análise de imagens retiradas por satélite; cada cultura tem uma “assinatura espectral” própria. Pode-se, assim, desenvolver um operador morfológico que determine os locais onde exista uma mesma cultura. Na engenharia, pode-se verificar se uma placa de circuito impresso foi fabricada corretamente ou se não há nenhuma trilha com ruptura ou fora de especificação. Pode-se desenvolver um operador morfológico que compare uma tal placa com um “gabarito”.

Pode-se realizar a verificação de autenticidade de assinaturas ou então a comparação de impressões digitais. Enfim, a variedade de aplicações da Morfologia Matemática é muito grande.

Dada uma imagem digital, cabe ao pesquisador em Morfologia Matemática procurar um operador morfológico que solucione um problema de seu interesse. O processo de análise

de imagens, por mais diversificado que seja a origem das mesmas, tem sempre algo em comum, que é a extração de informações das imagens. Dependendo do problema dado, um operador morfológico pode solucioná-lo realizando a extração de contorno, segmentação, suavização de formas, contagem, esqueletização ou eliminação de ruídos da imagem dada, ou ainda, através da composição destas diversas técnicas. Banon e Barrera [BB90] mostraram, generalizando um resultado obtido por Serra e Matheron, que qualquer operador morfológico sempre pode ser expresso por um operador “sinônimo” constituído pela composição de supremos e ínfimos de operadores morfológicos elementares: dilatação, erosão, anti-dilatação, anti-erosão.

Barrera e Banon [BB92] mostraram ainda que a Morfologia Matemática pode ser expressa através de uma linguagem formal, uma Linguagem Morfológica. Como consequência, as expressões morfológicas podem ser implementadas em uma máquina abstrata, chamada Máquina Morfológica.

Este fato nos motivou a verificar se as propriedades algébricas da Morfologia Matemática resultam num sistema de reescrita completo. Ao fazermos isso do modo apropriado, pudemos, como veremos, verificar a identidade de operadores morfológicos através de um método efetivo.

Nossos resultados se restringem à sub-classe dos operadores morfológicos invariantes por translação e isotônicos sobre as imagens binárias planas, podendo facilmente ser estendidos para as imagens espaciais.

A verificação de igualdade entre termos através da utilização de Sistemas de Reescrita de Termos está diretamente relacionada com a noção de simplificação de termos, como detalharemos mais adiante. Esta noção, por sua vez, pode ser relacionada no contexto morfológico à busca automática de operadores morfológicos mais eficientes ou, equivalentemente, programas da Máquina Morfológica mais eficientes. A possibilidade ou não de otimização de tais programas via um sistema de reescrita decorre das características algébricas da Morfologia Matemática.

Capítulo 2

Preliminares

Apresentamos, neste capítulo, definições, notações e resultados sobre conceitos matemáticos básicos que serão utilizados no decorrer do texto.

Após o enunciado de cada definição, lema, corolário, teorema ou proposição escrevemos ‘[x]’, onde x representa a referência bibliográfica de onde o respectivo enunciado foi obtido. Na ausência dele, ficará subentendido que o enunciado em questão não foi obtido de nenhum texto específico.

As palavras da linguagem **natural** não pertencentes à língua portuguesa são delimitadas por aspas e escritas em “*itálico*” apenas na sua primeira ocorrência no texto. Palavras que denominam objetos de importância maior para este texto também são apresentadas em *itálico* apenas em sua primeira aparição no texto, como em ‘‘um termo t é chamado de *termo básico* quando ...’’. Citações a frases obtidas de outros autores são escritas em “*itálico e entre aspas*” seguido da citação bibliográfica pertinente. O símbolo ■ denota o término de uma demonstração quando esta for baseada em alguma demonstração já existente para o lema, teorema, corolário ou proposição em questão. Usamos o símbolo ♦ para denotarmos o término de uma demonstração que foi feita pelo autor deste texto e que não foi baseada em nenhuma demonstração prévia. Por vezes, os lemas ou teoremas são subdivididos em itens $i), ii), \dots$, neste caso, denotamos o final da demonstração de um item usando os símbolos □ ou ◇ que distinguem, com o mesmo espírito dos símbolos apresentados nos parágrafos anteriores, a procedência das demonstrações.

Algumas definições deste capítulo foram obtidas em Monteiro [Mon71]. As demais notações, definições e lemas adotadas neste capítulo foram baseadas no texto de Peterson e Stickel [PS81]: estes autores, por sua vez, mencionam que as notações foram originalmente obtidas na Seção 2 de [Ros73].

Seja X um conjunto, então indicaremos por $|X|$ o número de elementos de X .

O conjunto vazio é denotado por $\emptyset$.

O conjunto $\{1, 2, \dots\}$ dos *números inteiros estritamente positivos* é denotado por $\mathbb{Z}_+$.

O conjunto $\{0, 1, 2, \dots\}$ dos *números naturais* é denotado por $\mathbb{N}$.

O símbolo $\in$ denota a relação de pertinência usual da teoria de conjuntos.

Definição 2.1. *Sejam X e Y conjuntos. Então escrevemos:*

a) $X \subseteq Y$ se e somente se $\forall x \in X, x \in Y$;

b) $X \subsetneq Y$ se e somente se $X \neq Y$ e $\forall x \in X, x \in Y$.

Os símbolos $\cup, \cap, -$ denotam as operações usuais entre conjuntos, chamadas *união*, *interseção* e *diferença*, respectivamente.

Definição 2.2. (Conjunto das Partes) *Seja X um conjunto, então o conjunto das partes de X , denotado por $\mathcal{P}(X)$, é o conjunto cujos elementos são os subconjuntos de X . Em outros termos:*

$$Y \in \mathcal{P}(X) \text{ se e somente se } Y \subseteq X$$

[Mon71]

Se $x_1, x_2, \dots, x_k$ são elementos (não necessariamente distintos dois a dois) de um conjunto X , então $(x_1, x_2, \dots, x_k)$ ou $\langle x_1, x_2, \dots, x_k \rangle$ denota a k -*upla ordenada* ou *seqüência finita* ou *palavra* formada pelos elementos $x_1, x_2, \dots, x_k$ e $\{x'_1, x'_2, \dots, x'_{k'}\}$ é o conjunto (logo, pela noção de conjunto, $x'_i \neq x'_j, \forall i, j : 1 \leq i, j \leq k' \text{ e } i \neq j$) que contém tão somente os elementos $x_1, x_2, \dots, x_k$. Isto é, $\forall x_j, 1 \leq j \leq k, \exists x'_i, 1 \leq i \leq k' : x_j = x'_i$ e ainda, $\forall x'_i, 1 \leq i \leq k', \exists x_j, 1 \leq j \leq k : x_j = x'_i$.

Note que $\begin{cases} k' = k & \text{se } x_i \neq x_j, \forall i, j : 1 \leq i, j \leq k \text{ com } i \neq j. \text{ (não há elementos repetidos} \\ & \text{em } \langle x_1, x_2, \dots, x_k \rangle) \\ k' < k & \text{caso contrário.} \end{cases}$

Denotamos uma palavra vazia por $\langle \rangle$. O *comprimento* de uma palavra $w = \langle x_1, x_2, \dots, x_k \rangle$ é k e é denotado por $\|w\|$. Consideramos ainda que x_i é o i -ésimo elemento de w e w_i representa o i -ésimo elemento de w .

Chamamos de *par não ordenado* um conjunto que possui dois e só dois elementos.

Definição 2.3. *Para qualquer palavra $w = \langle x_1, x_2, \dots, x_k \rangle$ e qualquer inteiro $j \geq 0$, definimos:*

$$j : w = \begin{cases} \langle \rangle & \text{se } j = 0; \\ \langle x_1, x_2, \dots, x_j \rangle & \text{se } 1 \leq j \leq k = \|w\|; \\ w & \text{caso contrário, ou seja, se } j > \|w\|. \end{cases}$$

[PS81]

Definição 2.4. (Concatenação de Palavras) A concatenação de duas palavras $\langle x_1, \dots, x_k \rangle$ e $\langle y_1, \dots, y_m \rangle$ é indicada por um ponto e definida como segue:

$$\langle x_1, \dots, x_k \rangle \cdot \langle y_1, \dots, y_m \rangle = \langle x_1, \dots, x_k, y_1, \dots, y_m \rangle.$$

[PS81]

Definição 2.5. Para qualquer conjunto X , X^* é o conjunto de todas as palavras de elementos de X , isto é,

$$X^* = \{ \langle x_1, \dots, x_k \rangle \mid k \geq 0 \text{ e } x_1, \dots, x_k \in X \}.$$

[PS81]

$\mathbb{Z}_+^*$ é o conjunto de todas as palavras constituídas por inteiros estritamente positivos.

Definição 2.6. (Produto Cartesiano) Sejam X e Y dois conjuntos, o produto cartesiano de X por Y é o conjunto denotado $X \times Y$ e cujos elementos são 2-uplas ordenadas tais que $(a, b) \in X \times Y$ se e somente se $a \in X$ e $b \in Y$.

[Mon71]

Seja $X_1, X_2, \dots, X_k$ conjuntos, $(k \geq 2)$. Podemos estender a noção de produto cartesiano definindo um produto de X_1 por X_2 por $\dots$ por X_k como sendo o conjunto denotado por $X_1 \times X_2 \times \dots \times X_k$ e tal que $(x_1, x_2, \dots, x_k) \in X_1 \times X_2 \times \dots \times X_k$ se e somente se $x_1 \in X_1, x_2 \in X_2, \dots, x_k \in X_k$. Quando $X_1 = X_2 = \dots = X_k = X$ denotamos $X_1 \times X_2 \times \dots \times X_k$ por X^k .

2.1 Relações

Definição 2.7. (Relação Binária) Sejam X e Y dois conjuntos e seja $X \times Y$ o produto cartesiano de X por Y . Qualquer subconjunto $\rightarrow$ de $X \times Y$ é denominado relação binária de X em Y . Em particular, se $\rightarrow$ é uma relação de X em X , isto é, se $\rightarrow$ é um subconjunto de $X \times X$, diz-se simplesmente que $\rightarrow$ é uma relação binária sobre X .

[Mon71]

Se $\rightarrow$ é uma relação binária de X em Y , usaremos a notação $a \rightarrow b$ (lê-se: “ a está na relação $\rightarrow$ com o elemento b ” ou simplesmente, “ $a \rightarrow b$ ”) para indicar que $(a, b) \in R$.

A negação de $a \rightarrow b$ será indicada por $a \not\rightarrow b$ (lê-se: “ a não está na relação $\rightarrow$ com o elemento b ” ou simplesmente, “ a não $\rightarrow b$ ”); portando $a \not\rightarrow b$ significa $(a, b) \notin \rightarrow$.

A relação inversa de $\rightarrow$ é denotada por $\rightarrow^{-1}$ ou ainda por $\leftarrow$ e é tal que $a \leftarrow b$ se e somente se $b \rightarrow a$.

Definição 2.8. Se $\rightarrow_1$ é uma relação binária de X em Y e $\rightarrow_2$ uma relação binária de Y em Z , então a composição de $\rightarrow_2$ com $\rightarrow_1$, denotada $\rightarrow_2 \circ \rightarrow_1$, é dada por $\rightarrow_2 \circ \rightarrow_1 = \{(x, y) \mid \exists z : x \rightarrow_1 z \text{ e } z \rightarrow_2 y\}$

[Hue80]

Proposição 2.9. A composição de relações binárias é associativa, isto é, se $\rightarrow_1, \rightarrow_2, \rightarrow_3$ são relações binárias, então $(\rightarrow_1 \circ \rightarrow_2) \circ \rightarrow_3 = \rightarrow_1 \circ (\rightarrow_2 \circ \rightarrow_3)$.

Demonstração: Consequência imediata da definição 2.8. ◆

Definição 2.10. (Potenciação de Relação Binária) Se $\rightarrow$ é uma relação binária sobre X , definimos recursivamente a potência n -ésima de $\rightarrow$ como segue:

$$\rightarrow^n = \begin{cases} \rightarrow \circ \rightarrow^{n-1} & \text{caso } n > 0 \\ \{(x, x) : x \in X\} & \text{caso } n = 0 \quad (\rightarrow^0 \text{ é a base da recursão}) \end{cases}$$

[Hue80]

Um caso particular de relação binária é aquele constituído pelos mapeamentos ou funções.

Definição 2.11. (Mapeamento ou Função) Um mapeamento ou função $F: X \mapsto Y$ é qualquer subconjunto de $X \times Y$ tal que, para cada $x \in X$, existe um único $y \in Y$, tal que $\langle x, y \rangle \in F$.

[PS81]

Definição 2.12. (Operação) Uma operação é uma função $F: X^k \mapsto X$.

O mapeamento $I_X : X \mapsto X$ tal que $I_X(x) = x, \forall x \in X$ é chamado de *mapeamento identidade*. Quando X está claro no contexto, escrevemos apenas I ao invés de I_X .

É usual denotarmos a potenciação n -ésima de uma operação F por F^n ao invés de F^n .

Note que, se F é uma operação sobre X , então pela definição 2.10 e pela definição de mapeamento identidade temos $F^0 = I_X$, ou simplesmente, $F^0 = I$.

Considere uma função $F: X \mapsto Y$. Para cada $x \in X$, o único $y \in Y$ tal que $\langle x, y \rangle \in F$ é denotado por $F(x)$, $F \langle x \rangle$, Fx ou xF , dependendo do contexto. Note que, se F e G são funções tais que $F : X \mapsto Y$ e $G : Y \mapsto Z$, caso seja utilizada a notação pré-fixa, isto é, operando à esquerda, então $xFG = x(G \circ F)$.

Definição 2.13. Seja a função $F: X \mapsto Y$. Definimos:

- a) o domínio de F , $\text{dom } F$, é X ;
- b) a imagem de F , $\text{img } F$, é $\{y \in Y \mid \langle x, y \rangle \in F\}$;

- c) para qualquer $B \subseteq Y$, $F^{-1}(B) = \{x \in X \mid F(x) \in B\}$;
- d) para qualquer $A \subseteq X$, $F(A) = \{F(x) \mid x \in A\}$;
- e) F é injetora se $(\forall x, x' \in X)(x \neq x' \Rightarrow F(x) \neq F(x'))$;
- f) F é sobrejetora se $\text{img } F = Y$;
- g) F é bijetora se ela for injetora e sobrejetora.

[PS81]

Seja a função $F: X \mapsto Y$. Se $B \subseteq Y$ tal que $B = \{y\}$, então denotamos $F^{-1}(\{y\})$ simplesmente por $F^{-1}(y)$.

Se $k \geq 0$ é um número inteiro e $F: X \mapsto X$, então $F^k = I$, o mapeamento identidade, se $k = 0$, and $F^k = F \circ F^{k-1}$ caso contrário.

Definição 2.14. Uma relação binária $\rightarrow$ sobre X é:

- a) reflexiva, se $(\forall x \in X)(x \rightarrow x)$;
- b) simétrica, se $(\forall x, y \in X)(x \rightarrow y \Rightarrow y \rightarrow x)$;
- c) transitiva, se $(\forall x, y, z \in X)((x \rightarrow y \text{ e } y \rightarrow z) \Rightarrow x \rightarrow z)$;
- d) anti-reflexiva, se $(\forall x \in X)(x \not\rightarrow x)$;
- e) anti-simétrica, se $(\forall x, y \in X)(x \rightarrow y \Rightarrow y \not\rightarrow x)$;
- f) total, se $(\forall x, y \in X)$ então pelo menos uma das três situações a seguir ocorre ($x \rightarrow y$ ou $y \rightarrow x$ ou $x = y$);
- g) “noetheriana” ou bem fundada, se não existe seqüência infinita $\langle x_1, x_2, \dots \rangle$ de elementos de X tal que $x_i \rightarrow x_{i+1}$ para todo $i \geq 1$.

[PS81]

Definição 2.15. Seja $\rightarrow$ uma relação binária sobre X . Um elemento x de X é dito irreduzível em relação a $\rightarrow$, ou simplesmente, irreduzível se, $(\nexists y \in X : x \rightarrow y)$; x é redutível caso contrário.

[PS81]

Outro caso particular e importante de relação binária é aquele formado pelas relações de equivalência.

Definição 2.16. (Relação de Equivalência) Diz-se que uma relação binária $\equiv$ sobre um conjunto X é uma relação de equivalência sobre X se e somente se ela for transitiva, reflexiva e simétrica.

[Mon71]

Seja $\equiv$ uma relação de equivalência sobre X e $x \equiv y$, então dizemos que “ x e y são equivalentes segundo $\equiv$ ” ou analogamente, “ x e y são equivalentes módulo $\equiv$ ” ou quando

não houver dúvida sobre a relação de equivalência considerada ou não houver a necessidade de explicitá-la dizemos simplesmente que “ x e y são equivalentes”.

Definição 2.17. (Classe de Equivalência) *Seja X um conjunto e $\equiv$ uma relação de equivalência sobre X . Para todo elemento x de X o conjunto*

$$[x]_{\equiv} = \{y \in X : y \equiv x\}$$

é denominado “classe de equivalência segundo $\equiv$ determinada pelo elemento x ” ou equivalentemente, “classe de equivalência módulo $\equiv$ determinada pelo elemento x ” e este elemento x , por sua vez, é chamado representante da classe de equivalência $[x]_{\equiv}$.

[Mon71]

Note que qualquer relação de equivalência $\equiv$ sobre um conjunto X induz um particionamento de X em subconjuntos disjuntos. O conjunto formado por tais subconjuntos disjuntos é chamado de conjunto quociente e é definido a seguir:

Definição 2.18. (Conjunto Quociente) *Seja $\equiv$ uma relação de equivalência sobre um conjunto X . O conjunto quociente de X por $\equiv$, denotado $X/\equiv$ é definido como segue:*

$$X/\equiv = \{[x]_{\equiv} \mid x \in X\}$$

[PS81]

Denotamos um elemento genérico de um conjunto quociente através do emprego de letras maiúsculas do início do alfabeto romano, como $A, B, C, \dots$.

Proposição 2.19. *Se $P = X/\equiv$ então $\cup P = X$, onde $\cup P$ denota o conjunto formado pela união dos elementos de P ; e ainda, se $A, B \in P, A \neq B$ então $A \cap B = \emptyset$.*

[PS81]

Demonstração: Imediata. ■

Para qualquer função $F : X \mapsto Y$, a relação $=_F$ sobre X definida por $x =_F y$ se e somente se $F(x) = F(y)$ é uma relação de equivalência.

Definição 2.20. (Ordens) *Seja um conjunto X qualquer, então uma:*

a) ordem parcial $\succeq$ sobre X é uma relação binária transitiva, reflexiva e anti-simétrica definida sobre X ;

- b) ordem parcial estrita $\succ$ sobre X é uma relação binária transitiva e anti-reflexiva (e portanto anti-simétrica) definida sobre X ;
- c) ordem total $\succeq$ sobre X é uma relação binária total transitiva, reflexiva e anti-simétrica definida sobre X ;
- d) Uma ordem total estrita $\succ$ sobre X é uma relação binária total transitiva e anti-reflexiva (e portanto anti-simétrica) definida sobre X .

[PS81]

Note que qualquer ordem total é uma ordem parcial e qualquer ordem total estrita é uma ordem parcial estrita.

Definição 2.21. (Conjunto Parcialmente Ordenado) Um conjunto parcialmente ordenado é uma dupla $\langle X, R \rangle$, onde X é um conjunto e R é uma relação de ordem sobre os elementos de X .

Seja $P \subseteq \{\text{reflexiva, transitiva, simétrica}\}$, $P \neq \emptyset$. Dizemos que uma relação *satisfaz* P se ela satisfaz cada elemento de P .

Definição 2.22. (Fecho de uma Relação Binária) O fecho P de uma relação binária $\rightarrow$ sobre X é a interseção de todos os subconjuntos de $X \times X$ que contém $\rightarrow$ e que satisfazem P .

[PS81]

A interseção (fecho), definida acima, contém $\rightarrow$ e, note ainda que, $X \times X$ satisfaz qualquer um dos possíveis P , logo esta interseção é não vazia (assumindo que X seja um conjunto não vazio).

Escrevemos fecho reflexivo-transitivo ao invés de fecho $\{\text{reflexivo, transitivo}\}$, analogamente para as outras possibilidades de P . Denotamos o fecho reflexivo de $\rightarrow$ por $\overrightarrow{\rightarrow}$, o fecho transitivo por $\overset{+}{\rightarrow}$, o fecho transitivo-reflexivo por $\overset{*}{\rightarrow}$ e o fecho transitivo-reflexivo-simétrico por $\overset{*}{\leftrightarrow}$.

Proposição 2.23. Seja $\rightarrow$ uma relação binária sobre X e sejam $x, y \in X$ e ainda $x_1, x_2, \dots, x_k \in X$, então:

- i) [fecho reflexivo : $\overrightarrow{\rightarrow}$] $\overrightarrow{\rightarrow} = \rightarrow \cup \{(x, x) : x \in X\}$;
- ii) [fecho simétrico : $\leftrightarrow$] $\leftrightarrow = (\leftarrow \cup \rightarrow)$;
- iii) [fecho transitivo : $\overset{+}{\rightarrow}$] $x \overset{+}{\rightarrow} y$ se e somente se existe uma seqüência finita $\langle x_1, \dots, x_k \rangle$, $k \geq 2$, de elementos de X tais que $x_1 = x$, $x_k = y$, e se $1 \leq i < k$, então $x_i \rightarrow x_{i+1}$;
- iv) [fecho transitivo-reflexivo : $\overset{*}{\rightarrow}$] $x \overset{*}{\rightarrow} y$ se e somente se existe uma seqüência finita $\langle x_1, \dots, x_k \rangle$,

$k \geq 1$, tais que $x_1 = x$, $x_k = y$, e se $1 \leq i < k$, então $x_i \xrightarrow{+} x_{i+1}$;

v) [fecho transitivo-reflexivo-simétrico : $\leftrightarrow$] $x \leftrightarrow y$ se e somente se existe uma seqüência finita $\langle x_1, \dots, x_k \rangle$, $k \geq 1$, tais que $x_1 = x$, $x_k = y$, e se $1 \leq i < k$, então $x_i \xrightarrow{+} x_{i+1}$ ou $x_i \xrightarrow{-} x_{i+1}$.

[PS81, DJ90]

Demonstração: Imediata. ■

Em algumas situações haverá a necessidade de se explicitar as seqüências utilizadas. Nestes casos, escreveremos $x \xrightarrow{+} y \langle x_1, \dots, x_k \rangle$ para indicar que $x_1 = x$, $x_k = y$, $k \geq 2$, e se $1 \leq i < k$, então $x_i \rightarrow x_{i+1}$. Os outros casos, $x \xrightarrow{*} y$ e $x \leftrightarrow y$, são análogos.

Definição 2.24. (Forma Normal) *Seja X um conjunto e $\rightarrow$ uma relação binária sobre X . Se $x, y \in X$, então x é uma forma normal de y em relação a $\rightarrow$, ou simplesmente, x é uma forma normal de y se e somente se $y \xrightarrow{*} x$ e x for irredutível. Neste caso, dizemos também que x é uma forma normal em relação a $\rightarrow$, ou simplesmente, x é uma forma normal.*

[PS81]

Se $\succ$ é uma ordem parcial, então $\succeq$ denota o conjunto $\succ \cup =$ (isto é, $\succeq$ é o fecho reflexivo de $\succ$); $s \prec t$ denota $t \succ s$ e $s \preceq t$ significa que $t \succ s$ ou $t = s$.

Definição 2.25. *Uma relação binária $\rightarrow$ sobre X é:*

- a) *confluente, se para qualquer x, y_1, y_2 em X , sempre que $x \xrightarrow{*} y_1$ e $x \xrightarrow{*} y_2$ então $y_1 \xrightarrow{*} z$ e $y_2 \xrightarrow{*} z$ para algum z em X .*
- b) *localmente confluente, se para qualquer x, y_1, y_2 em X , sempre que $x \rightarrow y_1$ e $x \rightarrow y_2$ então $y_1 \xrightarrow{*} z$ e $y_2 \xrightarrow{*} z$ para algum z em X .*

[PS81]

2.2 Árvores

Esta seção é reservada para descrevermos propriedades básicas das estruturas tipo árvore, pois estas serão largamente utilizadas no decorrer do texto.

Definição 2.26. *Sejam $m, n \in \mathbb{Z}_+^*$. Definimos então:*

- a) $\text{Pai}(n) = (\|n\| - 1) : n$ (para $n \neq \langle \rangle$) (pai);
- b) $\text{IE}(n) = \text{Pai}(n) \cdot \langle n_{\|n\|} - 1 \rangle$ (para $n_{\|n\|} \neq 1$) (irmão à esquerda);

- c) $m \preceq n$ sse $\|m\|: n = m$ (ancestral);
 d) $m \prec n$ sse $m \preceq n$ e $m \neq n$ (ancestral estrito);
 e) $m \perp n$ sse $m \not\preceq n$ e $n \not\preceq m$ (independência).

[PS81]

Note que, Pai e IE são mapeamentos, onde $Pai: \mathbb{Z}_+^* - \{<>\} \mapsto \mathbb{Z}_+^*$ e $IE: \mathbb{Z}_+^* - \{n \in \mathbb{Z}_+ \mid n_{\|n\|} \neq 1\} \mapsto \mathbb{Z}_+^*$.

Definição 2.27. Dizemos que um conjunto $M \subsetneq \mathbb{Z}_+^*$ é independente se seus elementos são independentes dois a dois.

[PS81]

Definição 2.28. Um Domínio de Árvore é qualquer subconjunto finito não vazio D de $\mathbb{Z}_+^*$ fechado sobre as funções pai e irmão à esquerda. Isto é, $Pai(D) \subseteq D$ e $IE(D) \subseteq D$.

[PS81]

Definição 2.29. (Árvore) Seja V um conjunto qualquer. Uma árvore de elementos de V é qualquer função $R: D \mapsto V$, onde D é um domínio de árvore. Os elementos de D são chamados de nós da árvore R . O conjunto de todas as árvores de elementos de V é denotado por V_* . Note, em particular, que se R é uma árvore e n é um nó em $\text{dom } R$, então Rn é o valor da árvore na posição n , isto é, o “símbolo” localizado no nó n .

[PS81]

Seja $R \in V_*$, $n \in \text{dom } R$. Dizemos que n é uma folha de R se e somente $\text{dom } R \cap Pai^{-1}(n) = \emptyset$. Observe que se precisa, de fato, considerar a interseção com $\text{dom } R$, pois se $n = \langle x_1, \dots, x_k \rangle \in \mathbb{Z}_+^*$ então $Pai^{-1}(n) = \{\langle x_1, \dots, x_k, 1 \rangle, \langle x_1, \dots, x_k, 2 \rangle, \dots\} \neq \emptyset$.

Definição 2.30. Sejam $R, S \in V_*$ e $n \in \text{dom } R$. Definimos então:

- a) $R/n = \{\langle m, x \rangle \mid \langle n \cdot m, x \rangle \in R\}$ (sub-árvore de R localizada na posição n),
 b) $R[n \leftarrow S] = \{\langle m, x \rangle \mid \langle m, x \rangle \in R \text{ e } (n \not\preceq m)\} \cup \{\langle n \cdot p, y \rangle \mid \langle p, y \rangle \in S\}$
 (troca-se por S a sub-árvore de R localizada na posição n).

[PS81]

Note que R/n é uma função.

Lema 2.31. Sejam $m, n, p \in \mathbb{Z}_+^*$. Então:

- i) $m \preceq n$ sse $(\exists p \in \mathbb{Z}_+^*)(m \cdot p = n)$;
 ii) se $m \preceq n$ e $n \preceq p$, então $m \preceq p$ (transitividade);

- iii) se $m \preceq \text{Pai}(n)$, então $m \preceq n$;
- iv) se $m \preceq n$ e $m \perp p$, então $n \perp p$;
- v) se $n \neq \langle \rangle$, então $\text{Pai}(m \cdot n) = m \cdot \text{Pai}(n)$.

[PS81]

Lema 2.32. *Sejam $R, S, T \in V_*$; $m, n \in \text{dom } R$; $p \in \text{dom } S$; $m \perp n$. Então as seguintes proposições são verdadeiras:*

- i) $R/n = S$ sse $R = R[n \leftarrow S]$;
- ii) $R[n \leftarrow S]/(n \cdot p) = S/p$ (embedding);
- iii) $R[n \leftarrow S[p \leftarrow T]] = R[n \leftarrow S][n \cdot p \leftarrow T]$ (associatividade);
- iv) $R[n \leftarrow S]/m = R/m$ (persistência);
- v) $R[n \leftarrow S][m \leftarrow T] = R[m \leftarrow T][n \leftarrow S]$ (comutatividade);
- vi) $m \in \text{dom } R/n$ sse $n \cdot m \in \text{dom } R$;
- vii) Se $n \in \text{dom } R$ e $m \in \text{dom } R/n$, então $(R/n)/m = R/(n \cdot m)$;
- viii) Se $M \subseteq \text{dom } R$, M um conjunto independente, então podemos definir:

$$R[M \leftarrow S] = \begin{cases} R[m_1 \leftarrow S] \cdots [m_k \leftarrow S] & \text{caso } M \neq \emptyset, \text{ onde } (m_1, \dots, m_k) \text{ é uma lista contendo} \\ & \text{tão somente os elementos de } M, \text{ a ordem dos elementos} \\ & \text{na lista é irrelevante;} \\ R & \text{caso contrário.} \end{cases}$$

(ix) Qualquer conjunto F de folhas, $F \subseteq R$, é um conjunto independente.

[PS81]

Definição 2.33. *Sejam $m, n, p \in \mathbb{Z}_+^*$. O quociente p de n por m é denotado por n/m e é definido do seguinte modo:*

$$n/m = p \quad \text{se e somente se} \quad n = m \cdot p.$$

[PS81]

Lema 2.34. *Suponha $m, n \in \mathbb{Z}_+^*$, $m \preceq n$, e $k \preceq n/m$. Então:*

- i) $m \cdot (n/m) = n$
- ii) $(n/m)/k = n/(m \cdot k)$

[PS81]

Lema 2.35. *Sejam $R, S, T \in V_*$; $m, n \in \text{dom } R$; $m \preceq n$. Então:*

- i) $R/n = (R/m)/(n/m)$ (cancelamento);
- ii) $R[n \leftarrow S]/m = (R/m)[n/m \leftarrow S]$ (distributividade);
- iii) $R[n \leftarrow S][m \leftarrow T] = R[m \leftarrow T]$ (dominação).

[PS81]

Lema 2.36. *Sejam $R \in V_*$; $u \cdot v \in \text{dom } R$, então $R/(u \cdot v) = (R/u)/v$ (associativa).*

[PS81]

Definição 2.37. *Seja $a \in V$. Definimos uma função $\bar{a}:(V_*)^* \mapsto V_*$ como segue. Suponha $\langle R_1, \dots, R_k \rangle \in (V_*)^*$; então:*

$$\bar{a}\langle R_1, \dots, R_k \rangle = \{ \langle \langle \rangle, a \rangle \} \cup \bigcup_{1 \leq j \leq k} \{ \langle \langle j \rangle \cdot m, y \rangle \mid \langle m, y \rangle \in R_j \}.$$

Isto será chamado de “a árvore com raiz a e sub-árvores $R_1, \dots, R_k$ ”. Em particular, $\bar{a}\langle \rangle$ (que abreviaremos por $\bar{a}$) é a árvore com raiz a e nenhum outro nó.

[PS81]

Exemplo 2.38. *Considere a árvore $R = \bar{t}(\bar{e}(\bar{t}, \bar{h}), \bar{b}, \bar{s}(\bar{b}))$. A figura 2.1 ilustra este exemplo de árvore.*

Figura 2.1: Diagrama representativo de uma árvore.

[PS81]

Lema 2.39. *Suponha $a \in V$; $S, R_1, \dots, R_k \in V_*$; e $m \in \text{dom } R_j$ para algum j tal que $1 \leq j \leq k$. Então:*

$$\bar{a}(R_1, \dots, R_k)[\langle j \rangle \cdot m \leftarrow S] = \bar{a}(R_1, \dots, R_j[m \leftarrow S], \dots, R_k).$$

[PS81]

2.3 Termos e Substituições

Para a criação das árvores, que expressam os termos, consideramos dois conjuntos, a saber, um conjunto arbitrário infinito e enumerável $\mathcal{V}$ de *símbolos variáveis*, ou simplesmente *variáveis* e um conjunto arbitrário finito $\mathcal{F}$ de *símbolos funcionais* ou simplesmente *operadores*. Dependendo do contexto, escrevemos Σ ao invés de $\mathcal{F}$ e dizemos que Σ é um *alfabeto*.

Denotamos as variáveis pelas letras minúsculas da parte final do alfabeto romano como v, x, y, z, w e por símbolos como $v', x', y', z', w', v_1, x_1, y_1, z_1, w_1, v_2, x_2, y_2, z_2, w_2, \dots$

Impomos a seguinte restrição, $\mathcal{V} \cap \mathcal{F} = \emptyset$.

Definimos ainda uma função $Arid : \mathcal{V} \cup \mathcal{F} \mapsto \mathcal{P}(\mathbb{N})$, tal que $Arid(v) = \{0\}$ se $v \in \mathcal{V}$ e para qualquer símbolo funcional f em $\mathcal{F}$, $Arid(f) = \{x_1, x_2, \dots, x_i, \dots\}$, onde cada $x_{i(i \geq 1)}$ é uma *aridade* ou *grau* válido para f . Se $Arid(f) = \{x_1\}$, isto é, se $|Arid(f)| = 1$, diremos que f possui aridade x_1 , e dizemos que tais símbolos possuem *aridade fixa*, caso contrário diremos que o símbolo funcional possui *aridade variável*.

Se $\mathcal{F}$ for tal que só possua símbolos de aridades fixas então, neste caso, escrevemos simplesmente $Arid(f_i) = x_i$ para qualquer f_i em $\mathcal{F}$, ao invés de $Arid(f_i) = \{x_i\}$.

Denotamos os símbolos funcionais com aridade estritamente positiva (> 0) pelas letras minúsculas da parte intermediária do alfabeto romano como f, g, h e por símbolos como $f', g', h', f_1, g_1, h_1, f_2, g_2, h_2, \dots$

Denotamos os símbolos funcionais com aridade zero pelas letras minúsculas do início do alfabeto romano como a, b, c, d , por símbolos como $a', b', c', d', a_1, b_1, c_1, d_1, a_2, b_2, c_2, d_2, \dots$, ou através de numerais $0, 1, 2, 3, \dots$

Vamos impor que, se $|Arid(f)| > 1$, então $Arid(f) = \mathbb{N} - \{0, 1\}$.

Qualquer variável em $\mathcal{V}$ possui aridade zero, isto é, $\forall x \in \mathcal{V}, Arid(x) = \{0\}$.

Os símbolos funcionais em $\mathcal{F}$ que possuem aridade zero são chamados de *constantes*.

Definição 2.40. (Termo) Um termo é um elemento do conjunto $\mathcal{T}(\mathcal{F} \cup \mathcal{V})$ definido abaixo:

$$\mathcal{T}(\mathcal{F} \cup \mathcal{V}) = \{t \in (\mathcal{F} \cup \mathcal{V})_* : \forall n \in \text{dom } t, |\text{dom } t \cap \text{Pai}^{-1}(n)| \in Arid(tn)\}.$$

[PS81]

Se t é um termo e n é um nó em $\text{dom } t$, neste caso diremos que n é uma *posição* em t .

Sejam s, t, r termos tais que $s = t[n \leftarrow r]$, neste caso, dizemos que t é chamado de *contexto de aplicação*, ou simplesmente, *contexto*.

Denotamos os termos pelas letras minúsculas da segunda metade do alfabeto romano como s, t, u, l, r e por símbolos como $s', t', u', l', r', s_1, t_1, u_1, l_1, r_1, s_2, t_2, u_2, l_2, r_2, \dots$

Dizemos que $\mathcal{T}(\mathcal{F} \cup \mathcal{V})$ é o conjunto formado pelos termos construídos a partir dos

símbolos funcionais em $\mathcal{F}$ e das variáveis em $\mathcal{V}$.

Um termo t é chamado de *termo básico* (do inglês: “ground term”) quando $t \in \mathcal{F}_*$. Indicamos por $\mathcal{T}(\mathcal{F})$ o conjunto formado por todos os termos em $\mathcal{F}_*$, isto é, $\mathcal{T}(\mathcal{F})$ é o conjunto formado pelos termos construídos a partir dos símbolos funcionais de $\mathcal{F}$.

Chamamos de *termos livres* os termos que possuem pelo menos uma variável.

Note que $\mathcal{T}(\mathcal{F} \cup \mathcal{V})$ contém tanto termos básicos quanto termos livres (quando $\mathcal{V} \neq \emptyset$).

Muitos autores referem-se aos termos básicos como *termos fechados* e aos termos livres como *termos abertos*.

Nesta dissertação faremos distinção entre “termo básico” e “termo livre” quando necessário. Sendo assim, quando numa frase aparecer a palavra “termo” isoladamente, isto significa que o “termo” em questão tanto pode ser um termo básico quanto um termo livre.

No decorrer do texto escreveremos $\mathcal{T}$ ao invés de $\mathcal{T}(\mathcal{F} \cup \mathcal{V})$ quando não houver a necessidade de especificarmos quais são os elementos dos conjuntos $\mathcal{F}$ e $\mathcal{V}$ considerados.

No decorrer do texto não será feita distinção entre a árvore $\{\langle \rangle, s \rangle\}$ e o símbolo s . Logo, se dissermos que uma árvore t “é uma constante”, significa que t é da forma $\{\langle \rangle, c \rangle\}$ e c é uma constante. Com o mesmo espírito, se n for uma folha de uma árvore t , então t/n e tn serão considerados idênticos (observe que t/n é uma função enquanto que tn é a imagem da função t quando é aplicado n em t).

Observe que, como consequência da definição 2.28, qualquer termo t possui uma quantidade finita de símbolos funcionais e variáveis.

Definição 2.41. Um termo t é dito *linear* quando $(\forall x \in \mathcal{V})$ se $t/n = t/m = x$ então $(n = m)$. [PS81]

Definição 2.42. O Domínio Estrito de um termo t , denotado $sdom\ t$, é definido como segue:

$$sdom\ t = \{n \in dom\ t \mid t/n \notin \mathcal{V}\}$$

[PS81]

Definição 2.43. (Subtermo) Um subtermo de um termo t é uma subárvore de t .

[PS81]

Note, então, que todo subtermo é também um termo, pois toda subárvore é também, por definição, uma árvore (veja definição 2.30).

Dizemos que t' é *próprio* de t se t' é subtermo de t e $t' \neq t$.

Seja t um termo tal que $dom\ t \neq \{\langle \rangle\}$, $t/\langle \rangle = f$, $k \in Arid(f)$, $t/\langle 1 \rangle = t_1$, $t/\langle 2 \rangle = t_2$, $\dots$, $t/\langle k \rangle = t_k$, então nestas condições temos que $t = \bar{f}(t_1, \dots, t_k)$. Omitiremos, daqui por diante, a barra e escreveremos simplesmente $t = f(t_1, \dots, t_k)$. Dizemos que cada $t_{i(1 \leq i \leq k)}$ é

um *operando* de t e ainda, dizemos que t_i é o i -ésimo operando de t .

Seja s um termo tal que $s < > = f$. Sendo assim, para expressarmos que um termo t é subtermo de s , usamos, por vezes, a notação $f(\dots t \dots)$.

Definição 2.44. (Congruência) *Uma relação de equivalência $\equiv$ sobre um conjunto de termos $\mathcal{T}$ é uma congruência se $f(s_1, \dots, s_n) \equiv f(t_1, \dots, t_n)$ sempre que $s_i \equiv t_i$ para todo $1 \leq i \leq n$*
[DJ90]

Se $\equiv$ é uma congruência sobre um conjunto de termos $\mathcal{T}$ e s e t são termos em $\mathcal{T}$ e são tais que $s \equiv t$, dizemos que s e t são *congruentes módulo $\equiv$* , ou ainda, quando $\mathcal{T}$ estiver claro no contexto, diremos simplesmente que, s e t são congruentes.

Definição 2.45. (Substituição) *Uma substituição é um mapeamento $\theta : \mathcal{T} \mapsto \mathcal{T}$ tal que:*

$$\theta(t) = \begin{cases} t & \text{se } t \text{ for uma constante;} \\ f(\theta(t_1), \dots, \theta(t_k)) & \text{se } t \in \mathcal{T}, t < > = f, k \in \text{Arid}(f) \text{ e } k \neq 0, \text{ e } t = f(t_1, \dots, t_k). \end{cases}$$

[PS81]

Logo, uma substituição fica determinada a partir de seus valores sobre as variáveis.

Denotamos as substituições por letras gregas $\alpha, \beta, \gamma, \sigma, \theta, \rho, \mu, \eta, \psi, \dots$ e pelos símbolos $\alpha', \beta', \gamma', \sigma', \theta', \rho', \mu', \eta', \psi', \dots$.

É usual escrever $t\theta$ ao invés de $\theta(t)$. Seguiremos esta notação usual, a pré-fixa, para as substituições no decorrer do texto. Sendo assim, se θ e ϕ forem substituições, então a notação $\theta\phi$ tem o mesmo significado de $\phi \circ \theta$, onde $\circ$ é a composição usual (da direita para a esquerda) de funções (veja definição 2.8). Segue-se que, se θ, ϕ e ψ forem substituições e t um termo, então $\psi(\theta\phi) = (\psi\theta)\phi$ e $t(\theta\phi) = (t\theta)\phi$.

Se t é um termo e θ uma substituição, dizemos que $t\theta$ é o termo *instanciado* por θ .

Definição 2.46. (Permutação) *Uma permutação é uma substituição Φ tal que $\forall x \in \mathcal{V}, x\Phi \in \mathcal{V}$ (ou seja, Φ apenas muda “nomes” de variáveis).*

[dAG87]

Definição 2.47. *Sejam t e s termos e θ uma substituição, então definimos:*

- a) $\text{Var}(t) = \{v \in \mathcal{V} \mid v \in \text{img } t\};$
- b) $\text{Var}(< t, s >) = \text{Var}(t) \cup \text{Var}(s);$
- c) $\text{Lvar}(\theta) = \{v \in \mathcal{V} \mid v\theta \neq v\} \quad (\text{domínio de } \theta);$
- d) $\text{Rvar}(\theta) = \bigcup_{x \in \text{Lvar}(\theta)} \text{Var}(x\theta) \quad (\text{imagem de } \theta);$
- e) $\text{Var}(\theta) = \text{Lvar}(\theta) \cup \text{Rvar}(\theta);$
- f) $\text{Const}(t) = \{c \in \mathcal{F}, \text{ onde } c \text{ é uma constante, tal que } c \in \text{img } t\};$

$$g) \text{Var}(\{t, s\}) = \text{Var}(t) \cup \text{Var}(s).$$

[PS81]

Observe:

Se $Lvar(\theta)$ for finito, então θ fica determinado por $Lvar(\theta)$;

$t\theta = t$ se e somente se $\text{Var}(t) \cap \text{Var}(\theta) = \emptyset$;

$\text{Var}(t\theta) \subseteq (\text{Var}(t) - Lvar(\theta)) \cup Rvar(\theta)$.

Como as substituições são aplicadas sobre termos e estes são finitos (isto é, eles possuem uma quantidade finita de símbolos funcionais e variáveis), então para cada substituição θ , a cardinalidade de $Lvar(\theta)$ é finita e se $y \in \mathcal{V} - Lvar(\theta)$ então $y\theta = y$. Conseqüentemente, podemos considerar cada substituição θ como sendo um conjunto finito de pares $\{(x, x\theta) : x \in Lvar(\theta)\}$.

Dizemos que cada par $(x, x\theta)$ de uma substituição θ é uma *atribuição* e costumamos denotá-la por $x \leftarrow x\theta$. Portanto, se $X = Lvar(\theta) = \{x_1, \dots, x_k\}$ denotamos θ como $\theta = \{x_1 \leftarrow t_1, \dots, x_k \leftarrow t_k\}$, onde $x_j\theta = t_j$, $1 \leq j \leq k$, e dizemos que θ *está determinada por* X . Dizemos que θ é uma substituição em $\mathcal{T}$ quando $t_1, t_2, \dots, t_k \in \mathcal{T}$.

Denotamos por $\mathcal{S}$ o conjunto de todas as substituições em $\mathcal{T}$.

Notação: Sejam $t, s \in \mathcal{T}$ e $x \in \text{Var}(t)$. Seja ainda, $N = \{n \in \text{dom } t \mid tn = x\}$. Então sob estas condições denotamos $t[n_1 \leftarrow s][n_2 \leftarrow s] \cdots [n_k \leftarrow s]$ por $t[N \leftarrow s]$, onde $N = \{n_1, n_2, \dots, n_k\}$.

Sobre a notação acima, note que $N \subseteq \text{dom } t$ e N é um conjunto independente (veja definição 2.27).

Lema 2.48. *Seja t um termo e θ uma substituição. Suponha que $n \in \text{dom } t\theta$. Então, ou $n \in \text{sdom } t$ ou existem nós m e p tais que $n = m \cdot p$, $m \in \text{dom } t$, $tm \in \mathcal{V}$, e $p \in \text{dom } (tm)\theta$. No primeiro caso,*

$$(i) \ t\theta/n = (t/n)\theta;$$

no segundo caso,

$$(ii) \ t\theta/n = (tm)\theta/p.$$

[PS81]

Lema 2.49. *Sejam s e t termos, $u \in \text{dom } t$, e θ uma substituição qualquer. Então:*

$$(i) \ u \in \text{dom } t\theta;$$

$$(ii) \ (t/u)\theta = (t\theta)/u;$$

$$(iii) \ t[u \leftarrow s]\theta = t\theta[u \leftarrow s\theta];$$

Além disso, se não houver $m \in \text{dom } t$ tal que $m \perp u$ e $tm \in Lvar(\theta)$, então

$$(iv) \ t[u \leftarrow s]\theta = t[u \leftarrow s\theta].$$

[PS81]

As demonstrações a seguir foram baseadas em L. de Azevedo Guerra [dAG87].

Demonstração de ii): Por indução sobre u , segundo $\prec$.

Demonstração da base da indução ($u = \langle \rangle$): É claro que $t\theta / \langle \rangle = (t / \langle \rangle)\theta = t\theta$.

Suponha que vale para todo $v \prec u$.

Seja $u = v \cdot \langle i \rangle$, onde $i \in \mathbb{Z}_+$, $v \in \text{dom } t$.

Pela lei do cancelamento (lema 2.35), $t\theta / u = (t\theta / v) / \langle i \rangle$.

Pela hipótese de indução, $t\theta / v = (t / v)\theta$.

Então temos: $t\theta / u = (t / v)\theta / \langle i \rangle$.

Mas t / v é da forma $f(t_1, \dots, t_n)$, onde

$$(t / v)\theta = f(t_1, \dots, t_n)\theta = f(t_1\theta, \dots, t_n\theta)$$

e portanto, $(t / v)\theta / \langle i \rangle = t_i\theta$.

$$\text{Logo, } t\theta / u = (t / v)\theta / \langle i \rangle = \underbrace{((t / v) / \langle i \rangle)}_{t_i} \theta \stackrel{\text{lei cancel.}}{=} (t / \underbrace{v \cdot \langle i \rangle}_u) \theta = (t / u) \theta.$$

□

Demonstração de iii): Seja $\prec_d$ a relação binária definida por:

$$u \prec_d v \text{ se e somente se } \exists w \in \mathbb{Z}_+^* \text{ tal que } v = w \cdot u$$

Usaremos indução em u segundo $\prec_d$.

O Lema vale para a base da indução ($u = \langle \rangle$) pois: $t\theta[\langle \rangle \leftarrow s] = s\theta = t[\langle \rangle \leftarrow s]\theta$

Seja $u = i \cdot v$, onde $i \in \mathbb{Z}_+^*$, $v \in \mathbb{Z}_+^*$.

$$t\theta[u \leftarrow s\theta] = t\theta[i \cdot v \leftarrow s\theta] = f(t_1, \dots, t_{i-1}, t_i[v \leftarrow s\theta], t_{i+1}, \dots, t_n)\theta =$$

$$f(t_1\theta, \dots, t_i\theta[v \leftarrow s\theta], \dots, t_n\theta) \stackrel{\text{hipótese indução}}{=} f(t_1\theta, \dots, t_i[v \leftarrow s]\theta, \dots, t_n\theta) =$$

$$f(t_1, \dots, t_{i-1}, t_i[v \leftarrow s], t_{i+1}, \dots, t_n)\theta = t[i \cdot v \leftarrow s]\theta = t[u \leftarrow s]\theta.$$

■

Corolário 2.50. i) Se $t / u = t' \notin X$, então $t\theta / u = t'\theta$.

Temos, $\text{dom } t\theta = \text{dom } t \cup \bigcup_{t/u \in X} \{u \cdot v \mid v \in \text{dom } (t / u)\theta\}$, logo:

ii) Se $t / u = x \in X$, então $t\theta / (u \cdot v) = x\theta / v$, $\forall v \in \text{dom } x\theta$.

[dAG87]

As demonstrações a seguir foram baseadas em L. de Azevedo Guerra [dAG87].

Demonstração de i): $t\theta / u \stackrel{\text{lema 2.49}}{=} (t / u)\theta = t'\theta$.

□

Demonstração de ii): $t\theta / (u \cdot v) \stackrel{\text{associat. lema 2.36}}{=} (t\theta / u) / v \stackrel{\text{lema 2.49}}{=} \underbrace{(t / u)}_x \theta / v = x\theta / v$, $\forall v \in \text{dom } x\theta$.

■

Lema 2.51. Suponha que θ e ϕ sejam substituições.

i) se $Rvar(\theta) \cap Lvar(\phi) = \emptyset$, $Rvar(\phi) \cap Lvar(\theta) = \emptyset$, e $Lvar(\theta) \cap Lvar(\phi) = \emptyset$, então $\theta\phi = \phi\theta$;

ii) se t é um termo, $Rvar(\theta) \cap Lvar(\phi) = \emptyset$, e $Var(t) \subseteq Lvar(\theta)$, então $t\theta\phi = t\theta$.

[PS81]

Definição 2.52. Suponha que s e t sejam termos. Dizemos que s é uma instância de t se existir uma substituição θ tal que $s = t\theta$ e, neste caso, dizemos ainda que t é uma generalização de s e que θ realiza um “matching” de t em s . Dizemos que s e t são unificáveis se existir uma substituição θ tal que $s\theta = t\Phi\theta$, onde Φ é uma permutação tal que $Var(s) \cap Var(t\Phi) = \emptyset$. Neste caso, dizemos que θ é uma unificadora dos termos s e t .

[dAG87]

Se σ e θ são substituições, então escrevemos $\sigma = \theta$ para indicarmos que $x\sigma = x\theta$ para qualquer x em $\mathcal{V}$.

Definição 2.53. Uma unificadora θ de t_1 e t_2 é dita unificadora mais geral (u.m.g.) de t_1 e t_2 se e somente se para toda unificadora θ' de t_1 e t_2 existe uma substituição σ tal que $\theta' = \theta\sigma$.

[dAG87]

Como Guerra [dAG87] observa, a unificadora mais geral de dois termos é única, a menos de permutações. Se θ e σ são unificadoras mais gerais de s e t então há permutações Φ e Φ' tais que $\theta = \sigma\Phi$ e $\sigma = \theta\Phi'$ e mais, $\forall x \in \mathcal{V}, x\Phi'\Phi = x\Phi\Phi' = x$, (isto é, Φ e Φ' são inversas entre si). Diremos que θ e σ são *variantes* da u.m.g. de t e s .

O termo resultante pela unificação de dois termos através de uma unificadora é chamado de *termo unificador*. Se t é um termo unificador mais geral dos termos s_1 e s_2 então qualquer termo t' obtido a partir de t , pela simples renomeação das variáveis em t , também é um termo unificador mais geral de s_1 e s_2 . Em outras palavras, t' é o termo que se obtém pela aplicação de uma variante de uma u.m.g. de s_1 e s_2 sobre s_1 ou s_2 .

Definição 2.54. Suponha que θ e ϕ sejam substituições tais que $Lvar(\theta) \cap Lvar(\phi) = \emptyset$. Seja v uma variável qualquer, sendo assim, definimos $\theta \cup \phi$ como uma substituição tal que:

- a) $v(\theta \cup \phi) = v\theta$ se $v \in Lvar(\theta)$;
- b) $v(\theta \cup \phi) = v\phi$ se $v \in Lvar(\phi)$;
- c) $v(\theta \cup \phi) = v$ caso contrário.

[PS81]

Uma consequência imediata da definição 2.54 é que se s e t são termos, θ e ϕ substituições tais que $Lvar(\theta) \cap Lvar(\phi) = \emptyset$, $Var(s) \cap Lvar(\phi) = \emptyset$, $Lvar(\theta) \cap Rvar(\theta) = \emptyset$ e $s\theta = t\phi$,

então $s(\theta \cup \phi) = t(\theta \cup \phi)$. Conseqüentemente, s e t são unificáveis.

Capítulo 3

Teorias Equacionais e Sistemas de Reescrita de Termos

A seguir, apresentamos um sumário sobre teorias equacionais, veremos que, no caso das teorias equacionais, as noções de consequência semântica e sintática se coincidem. Mostraremos, nos capítulos posteriores, algoritmos para o calculo dos teoremas de teorias equacionais particulares, uma vez que este problema é indecidível para o caso geral. Tais algoritmos são baseados na noção, que descreveremos neste capítulo, de sistemas de reescrita de termos.

3.1 Sintaxe das Teorias Equacionais

Definição 3.1. (Equação) *Dado um conjunto $\mathcal{T}$ de termos, uma equação é um par não ordenado $\{s, t\}$ de termos em $\mathcal{T}$, denotada $s = t$.*

[DJ90]

Pela definição 3.1, tanto $s = t$ quanto $t = s$ denotam exatamente o mesmo par não ordenado $\{s, t\}$ de termos. Uma equação $s = t$ é dita *linear* quando os termos s e t são lineares.

Definição 3.2. (Instância de uma Equação) *Seja $s = t$ uma equação e θ uma substituição, então $s\theta = t\theta$ é uma instância da equação $s = t$.*

[dO95]

Se E é um conjunto de equações, escrevemos $\mathcal{T}_E$ para indicar o conjunto $\mathcal{T}(\Sigma \cup \mathcal{V})$ onde Σ é o menor alfabeto que contém os símbolos funcionais existentes nos termos presentes

nas equações em E e $\mathcal{V}$ é um conjunto infinito e enumerável de variáveis e tal que contém as variáveis existentes nas equações de E .

Definição 3.3. (Axioma Equacional) *Seja E um conjunto de equações, então definimos a relação binária $\leftrightarrow_E$ sobre $\mathcal{T}_E$ como segue. O par $(s, t) \in \leftrightarrow_E$ se e somente se existir uma equação $l = r \in E$, um nó $n \in \text{dom } s$, e uma substituição σ tal que:*

$$s/n = l\sigma \quad e \quad t = s[n \leftarrow r\sigma].$$

Os elementos de $\leftrightarrow_E$ são chamados de axiomas equacionais da teoria equacional de E . Nestas circunstâncias, escrevemos $s \leftrightarrow_E t(l = r, n, \sigma)$ ou simplesmente $s \leftrightarrow_E t$.

[PS81]

O que a definição 3.3 nos diz é que $s \leftrightarrow_E t$ se s tem um subtermo s' que é uma instância de um dos lados de uma equação $l = r$ em E e t é o resultado da troca de s' com a correspondente instância do outro lado da equação $l = r$.

Corolário 3.4. (Regra de Inferência - Simetria) *Se $s \leftrightarrow_E t$ então $t \leftrightarrow_E s$. Podemos expressar que $\leftrightarrow_E$ é simétrica através do seguinte diagrama:*

$$\boxed{\frac{t \leftrightarrow_E s}{s \leftrightarrow_E t} \quad [\text{simetria}]}$$

Demonstração: $s \leftrightarrow_E t(l = r, n, \sigma)$ quando $s/n = l\sigma$ e $t = s[n \leftarrow r\sigma]$ e, conseqüentemente, $t/n = r\sigma$ e $s = t[n \leftarrow l\sigma]$ ou seja $t \leftrightarrow_E s(l = r, n, \sigma)$ e portanto a relação $\leftrightarrow_E$ é simétrica. ■

Definição 3.5. (Teoria Equacional) *A teoria equacional de um conjunto E de equações, denotada $Th(E)$ é o fecho transitivo-reflexivo de $\leftrightarrow_E$, isto é, $Th(E) = \overset{*}{\leftrightarrow}_E$.*

[DJ90]

Por vezes, no decorrer do texto, preferiremos adotar o símbolo $\equiv$ para o fecho transitivo-reflexivo de $\leftrightarrow_E$. Sendo assim, $\overset{*}{\leftrightarrow}_E \stackrel{\text{notação}}{=} \equiv$.

Pela definição de fecho transitivo-reflexivo, podemos entender que a relação $\overset{*}{\leftrightarrow}_E$ é o conjunto obtido a partir de $\leftrightarrow_E$ e assumindo como regras de inferência a leis da transitividade e reflexividade expressas abaixo de forma esquematizada:

$\frac{}{s \xrightarrow[E]{*} s}$	[reflexividade]
$\frac{t \xrightarrow[E]{*} s \quad s \xrightarrow[E]{*} u}{t \xrightarrow[E]{*} u}$	[transitividade]

Como $\xrightarrow[E]{*}$ é simétrica, então $\xrightarrow[E]{*}$ é uma relação de equivalência sobre $\mathcal{T}_E$.

Temos ainda, pelo lema 3.6 abaixo, que $\xrightarrow[E]{*}$ é a menor relação de congruência que contém $\xrightarrow[E]{*}$.

Lema 3.6. *O fecho transitivo-reflexivo $\xrightarrow[E]{*}$ de $\xrightarrow[E]{*}$ é a menor congruência que contém $\xrightarrow[E]{*}$.* [DJ90]

Sendo assim, $s \xrightarrow[E]{*} t$ se e somente se s e t são congruentes módulo $\xrightarrow[E]{*}$. Podemos ainda neste caso, por simplicidade, dizer que s e t são congruentes módulo E , ou ainda, congruentes- E . Também escrevemos $[s]_E$ ao invés de $[s]_{\xrightarrow[E]{*}}$ para denotarmos a classe de congruência módulo E de um termo s e ainda escrevemos $\mathcal{T}/E$ ao invés de $\mathcal{T}_E/\xrightarrow[E]{*}$ para denotarmos o conjunto formado por todas as classes de congruências módulo E .

Definição 3.7. (Teorema) *Uma equação $s = t$ é um teorema de um conjunto E de equações se e somente se o par (s, t) for elemento de $Th(E)$, o que denotamos por $E \vdash s = t$.* [DJ90]

Definição 3.8. (E-Derivação) *Uma derivação em E é qualquer seqüência $s_0 \xrightarrow[E]{*} s_1 \xrightarrow[E]{*} \cdots \xrightarrow[E]{*} s_i \xrightarrow[E]{*} \cdots$ de aplicações de axiomas equacionais de E .* [DJ90]

Definição 3.9. (E-Prova) *Uma prova em E de que uma equação $s = t$ é um teorema de E é uma derivação em E finita e justificada $s = s_0 \xrightarrow[E]{*} s_1 \xrightarrow[E]{*} \cdots \xrightarrow[E]{*} s_i \xrightarrow[E]{*} \cdots s_{n-1} \xrightarrow[E]{*} s_n = t$ ($n \geq 0$), onde cada passo $s_i \xrightarrow[E]{*} s_{i+1}$ é justificado através da referência a uma equação $l = r$ em E , uma posição p em s_i e uma substituição σ , tal que $s_i/p = l\sigma$ e $s_{i+1} = s_i[p \leftarrow r\sigma]$.* [DJ90]

3.2 Alguns Resultados Básicos em Teorias Equacionais

Lema 3.10. *Seja E um conjunto de equações. Se $u = v \in E$ então $u \equiv_E v$.*

Demonstração: Como consequência imediata da definição 3.3, se $u = v \in E$ então $u \xleftrightarrow{E} v$, e portanto $u \equiv_E v$ pois $\xleftrightarrow{E} \subseteq \equiv_E$. ■

Note ainda que, como consequência imediata da definição 3.3, temos que, se $v \in \mathcal{V}$ e $t \in \mathcal{T}_E$, não implica que $v \equiv_E t$.

Observe também que não necessariamente tem-se $\xleftrightarrow{E} \subseteq \equiv_E$. Para ver isto tome $E = \{a = a\}$, logo $\mathcal{T}_E = \{a, x, y, \dots\}$ e conseqüentemente $\xleftrightarrow{E} = \{(a, a)\}$ e $\xleftrightarrow{E}^* = \{(a, a)\}$, sendo assim temos, em particular para este caso, que $\xleftrightarrow{E} = \equiv_E$.

Corolário 3.11. (Regra de Inferência: Instanciação) *Sejam $s, t \in \mathcal{T}$, θ uma substituição, e E um conjunto de equações. Se $s \equiv_E t$, então $s\theta \equiv_E t\theta$. Podemos expressar este lema através do seguinte diagrama:*

$$\boxed{\frac{t \xleftrightarrow{E}^* s}{t\theta \xleftrightarrow{E}^* s\theta} \quad [\text{instanciação}]}$$

[PS81]

Demonstração: Suponha primeiramente que $s \xleftrightarrow{E} t(u = v, n, \phi)$. Então, $s/n = u\phi$ e $t = s[n \leftarrow v\phi]$. Pelo lema 2.49, $n \in \text{dom } s\theta, s\theta/n = (s/n)\theta = u\phi\theta$, e $t\theta = s[n \leftarrow v\phi]\theta = s\theta[n \leftarrow v\phi\theta]$. Então, $s\theta \xleftrightarrow{E} t\theta(u = v), n, \phi\theta$.

Agora, suponha que $s = t_1 \xleftrightarrow{E} t_2 \xleftrightarrow{E} \dots \xleftrightarrow{E} t_k = t$. Então pelo que acabamos de demonstrar, $t_j\theta \xleftrightarrow{E} t_{j+1}\theta$ se $1 \leq j < k$. Então $s\theta \equiv_E t\theta$. ■

Lema 3.12. *Sejam t um termo e $\{n_1, \dots, n_k\}$ um conjunto independente de nós em $\text{dom } t$. Se $t/n_i \equiv_E s_i$ para $1 \leq i \leq k$, então $t \equiv_E t[n_1 \leftarrow s_1] \dots [n_k \leftarrow s_k]$.*

[PS81]

Demonstração: É suficiente demonstrarmos que, se $n \in \text{dom } t$ e $t/n \equiv_E s$, então $t \equiv_E t[n \leftarrow s]$.

Suponha, primeiramente, que $t/n \xleftrightarrow{E} s(u = v, m, \theta)$. Então, $(t/n)/m = u\theta$ e $s = (t/n)[m \leftarrow v\theta]$.

Pela proposição vii) do lema 2.32, $t/(n \cdot m) = u\theta$. Pelas proposições i) e iii) do lema 2.32,

$t[n \leftarrow s] = t[n \leftarrow (t/n)[m \leftarrow v\theta]] = t[n \cdot m \leftarrow v\theta]$. Então $\boxed{t \xleftrightarrow{E} t[n \leftarrow s](u = v, n \cdot m, \theta)}^\dagger$.

Suponha que $t/n \equiv_E s < t_1, \dots, t_k >$. Seja $t'_j = t[n \leftarrow t_j]$ para $1 \leq j \leq k$.

Observe que $t'_j = t[n \leftarrow t_j] \xleftrightarrow{E} t[n \leftarrow t_{j+1}] = t'_{j+1}$, pois, por hipótese, $t_j \xleftrightarrow{E} t_{j+1}$, mas $t_j = t'_j/n$.

Logo, pelo que acabamos de demonstrar ($\dagger$), temos que $t'_j \xleftrightarrow{E} t'_j[n \leftarrow t_{j+1}] = t[n \leftarrow t_j][n \leftarrow t_{j+1}] = t[n \leftarrow t_{j+1}] = t'_{j+1}$. E, de fato, $t'_j \xleftrightarrow{E} t'_{j+1}$ para $1 \leq j < k$, logo $t'_j \equiv_E t'_{j+1}$ para $1 \leq j < k$ e pela transitividade da $\equiv_E$, temos que $t'_1 \equiv_E t'_k$; isto é, $t \equiv_E t[n \leftarrow s]$. ■

Corolário 3.13. (Regra de Inferência: Contexto de Aplicação) *Sejam $u, v, t \in \mathcal{T}$ e $n \in \text{dom } t$. Se $u \stackrel{E}{=} v$ então $t[n \leftarrow u] \stackrel{E}{=} t[n \leftarrow v]$. Podemos expressar este corolário através do seguinte diagrama:*

$$\boxed{\frac{t \stackrel{*}{\underset{E}{\rightleftharpoons}} s}{f(\dots t \dots) \stackrel{*}{\underset{E}{\rightleftharpoons}} f(\dots s \dots)} \quad [\text{contexto de aplicação}]$$

Demonstração: Conseqüência imediata do lema 3.12. ■

Podemos entender uma teoria equacional $Th(E)$ como sendo o conjunto obtido a partir das equações de E , ou seja, se $s = t \in E$ então $s = t \in Th(E)$, e assumindo como regras de inferência as 5 regras apresentadas acima: a simetria, reflexividade, transitividade, instanciação e contexto de aplicação.

Uma teoria equacional E , ou simplesmente, uma teoria E é dita *básica* ou *fechada* se as equações de E são formadas por termos básicos.

3.3 Semântica das Teorias Equacionais

A completeza do conjunto formado pelas 5 regras de inferência apresentadas na seção anterior foi demonstrada por Birkhoff [Bir35]. Porém, antes de enunciarmos o teorema de Birkhoff sobre a completeza das teorias equacionais, precisamos introduzir os conceitos de significado de um termo e modelos de teorias equacionais. Entretanto, para formalizarmos os conceitos de significado de um termo e modelos de teorias equacionais precisamos antes apresentar o conceito de Σ -álgebra.

Definição 3.14. (Realização ou Interpretação) *Seja Σ um alfabeto e seja A um conjunto. A interpretação em A de um símbolo f de aridade n , pertencente em Σ , denotada $\mathcal{I}_A[f]$ ou simplesmente $\mathcal{I}[f]$, quando estiver claro o conjunto A considerado, é uma operação $f_A : A^n \mapsto A$.* [dO95]

Definição 3.15. (Σ_A – álgebra) *Seja Σ um alfabeto e A um conjunto não vazio, denominado base, então uma Σ – álgebra A , denotada por Σ_A – álgebra, é dada por:*

$$\Sigma_A - \text{álgebra} = \{f_A : \text{para cada símbolo } f \text{ em } \Sigma, f_A \text{ é a interpretação em } A \text{ do símbolo } f\}$$

[dO95]

O conjunto base A de uma Σ_A – álgebra é também costumeiramente chamado de “carrier” (do francês, que significa “pedreiro”), *suporte*, *universo de discurso* ou, ainda, simplesmente *universo* da álgebra. Os conceitos a seguir foram baseados em [Grä79, dA91]. Por vezes não estamos interessados em explicitar o universo da Σ_A – álgebra e nestes casos escrevemos simplesmente Σ – álgebra. É comum escrever “seja Σ uma álgebra...” quando queremos referir-nos à uma Σ – álgebra. Dizemos que uma Σ – álgebra é *finita* ou *infinita* se a cardinalidade de seu universo for finita ou infinita, respectivamente. Quando a cardinalidade de Σ é finita e a aridade dos símbolos funcionais em Σ está sub-entendida no contexto então, neste caso, costuma-se representar uma Σ – álgebra através de uma $(k+1)$ –upla $\langle A; f_{1_A}, f_{2_A}, \dots, f_{k_A} \rangle$ ou por simplicidade $\langle A; f_1, f_2, \dots, f_k \rangle$, onde A é o universo da Σ – álgebra e $\Sigma = \{f_1, f_2, \dots, f_k\}$.

Algumas Σ – álgebras são “mais interessantes” que outras por expressarem características de problemas práticos de interesse da matemática:

- Um *semigrupo* é qualquer álgebra que possa ser representada através de um par $\langle S; \cdot \rangle$, onde $\cdot$ é uma operação binária e a interpretação dada ao símbolo funcional $\cdot$ satisfaz a identidade abaixo, a qual expressa a lei associativa:

$$x \cdot (y \cdot z) = (x \cdot y) \cdot z, \quad \forall x, y, z \in S.$$

- Um *monóide* é qualquer álgebra que possa ser representada através de uma tripla $\langle M; \cdot, 1 \rangle$, onde $\cdot$ é uma operação binária, 1 é uma operação constante, isto é, $1_M : \mapsto M$, e as seguintes identidades verificam-se:

$$\begin{aligned} x \cdot (y \cdot z) &= (x \cdot y) \cdot z, & \forall x, y, z \in M; \\ 1 \cdot x &= x, & \forall x \in M; \\ x \cdot 1 &= x, & \forall x \in M. \end{aligned}$$

É usual simplesmente dizer que um monóide é um semigrupo $\langle M; \cdot \rangle$ possuindo um elemento neutro 1 , ou seja, um elemento 1 em M tal que:

$$1 \cdot x = x \cdot 1 = x, \quad \forall x \in M.$$

- Um *grupo* é qualquer álgebra que possa ser representada através de uma quádrupla $\langle G; \cdot, 1, {}^{-1} \rangle$, onde $\cdot$ é uma operação binária, ${}^{-1}$ é uma operação unária, 1 é uma operação constante e as seguintes identidades verificam-se:

$$\begin{aligned} x \cdot (y \cdot z) &= (x \cdot y) \cdot z, & \forall x, y, z \in G; \\ 1 \cdot x &= x, & \forall x \in G; \\ x \cdot 1 &= x, & \forall x \in G; \\ x \cdot x^{-1} &= 1, & \forall x \in G; \\ x^{-1} \cdot x &= 1, & \forall x \in G. \end{aligned}$$

- Um grupo $\langle AG; \cdot, 1, {}^{-1} \rangle$ é dito *abeliano* ou *comutativo* se, adicionalmente, for válida a seguinte identidade:

$$x \cdot y = y \cdot x, \quad \forall x, y \in G.$$

- Um *anel comutativo* é qualquer álgebra que possa ser representada através de uma terna $\langle BR; +, * \rangle$, onde $+$ e $*$ são operações binárias costumeiramente chamadas de *adição* e *multiplicação*, respectivamente, e são tais que:

1. a operação de adição $+$ define uma estrutura de grupo comutativo sobre BR ;
2. a operação de multiplicação $*$ define uma estrutura de semigrupo sobre BR .

e as seguintes identidades verificam-se:

$$x * (y + z) = (x * y) + (x * z), \quad \forall x, y, z \in BR \text{ (distributiva à direita de } * \text{ em relação à } +);$$

$$(y + z) * x = (y * x) + (z * x), \quad \forall x, y, z \in BR \text{ (distributiva à esquerda de } * \text{ em relação à } +).$$

- Um *reticulado* é qualquer álgebra que possa ser representada através de uma terna $\langle R; \vee, \wedge \rangle$ onde $\vee$ e $\wedge$ são operações binárias e as seguintes identidades verificam-se:

$$x \vee (y \vee z) = (x \vee y) \vee z, \quad \forall x, y, z \in R \quad (\text{lei associativa});$$

$$x \wedge (y \wedge z) = (x \wedge y) \wedge z, \quad \forall x, y, z \in R \quad (\text{lei associativa});$$

$$x \vee y = y \vee x, \quad \forall x, y \in R \quad (\text{lei comutativa});$$

$$x \wedge y = y \wedge x, \quad \forall x, y \in R \quad (\text{lei comutativa});$$

$$x \vee x = x, \quad \forall x \in R \quad (\text{idempotência});$$

$$x \wedge x = x, \quad \forall x \in R \quad (\text{idempotência});$$

$$x \wedge (x \vee y) = x, \quad \forall x, y \in R \quad (\text{absorção});$$

$$x \vee (x \wedge y) = x, \quad \forall x, y \in R \quad (\text{absorção}).$$

- Um reticulado é dito *distributivo* se, adicionalmente, forem válidas as seguintes identidades:

$$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z), \quad \forall x, y, z \in R \quad (\text{distributividade});$$

$$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z), \quad \forall x, y, z \in R \quad (\text{distributividade}).$$

- Uma *álgebra de Boole* é qualquer álgebra que possa ser representada através de uma 6-upla $\langle B; \vee, \wedge, ', 0, 1 \rangle$, onde $\vee$ e $\wedge$ são operações binárias, $'$ é unária, 0 e 1 são constantes e as seguintes identidades verificam-se:

$$\begin{array}{lll}
x \vee (y \vee z) = (x \vee y) \vee z, & \forall x, y, z \in B & (\text{associativa}); \\
x \wedge (y \wedge z) = (x \wedge y) \wedge z, & \forall x, y, z \in B & (\text{associativa}); \\
x \vee y = y \vee x, & \forall x, y \in B & (\text{comutativa}); \\
x \wedge y = y \wedge x, & \forall x, y \in B & (\text{comutativa}); \\
x \vee x = x, & \forall x \in B & (\text{idempotência}); \\
x \wedge x = x, & \forall x \in B & (\text{idempotência}); \\
x \wedge (x \vee y) = x, & \forall x, y \in B & (\text{absorção}); \\
x \vee (x \wedge y) = x, & \forall x, y \in B & (\text{absorção}); \\
x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z), & \forall x, y, z \in B & (\text{distributiva}); \\
x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z), & \forall x, y, z \in B & (\text{distributiva}); \\
x \wedge 0 = 0, & \forall x \in B; & \\
x \vee 1 = 1, & \forall x \in B; & \\
x \wedge x' = 0, & \forall x \in B; & \\
x \vee x' = 1, & \forall x \in B. &
\end{array}$$

É usual simplesmente dizer que uma álgebra de Boole é um reticulado distributivo $\langle B; \vee, \wedge \rangle$ mais uma operação unária $'$ e duas constantes 0 e 1 tais que satisfazem adicionalmente as seguintes identidades:

$$\begin{array}{ll}
x \wedge 0 = 0, & \forall x \in B; \\
x \vee 1 = 1, & \forall x \in B; \\
x \wedge x' = 0, & \forall x \in B; \\
x \vee x' = 1, & \forall x \in B.
\end{array}$$

Costuma-se também dizer que uma álgebra de Boole é um *reticulado distributivo complementado*. Os conceito sobre anel booleano, dado abaixo, foi obtido em [Mar96a].

- Um *anel booleano* é qualquer álgebra que possa ser representada através de uma 6-upla $\langle BR; +, *, -, 0, 1 \rangle$, onde $+$ e $*$ são operações binárias, $-$ é unária, 0 e 1 são constantes e que as seguintes identidades verificam-se:

$$\begin{array}{ll}
x * (y + z) = (x * y) + (x * z), & \forall x, y, z \in BR; \\
x + 0 = x, & \forall x \in BR; \\
x + x = 0, & \forall x \in BR; \\
x * 1 = x, & \forall x \in BR; \\
x * x = x, & \forall x \in BR; \\
x * 0 = 0, & \forall x \in BR;
\end{array}$$

$$-x = x, \quad \forall x \in BR.$$

O conceito sobre corpo finito, dado abaixo, foi obtido em [Mar96b].

- Um *corpo finito* para um número primo p é qualquer álgebra que possa ser representada através de uma 7-upla $\langle FF; *, +, -, 0, 1, p \rangle$, onde $+$ e $*$ são operações binárias, $-$ é unária, 0 e 1 são constantes e p é um número primo e que as seguintes identidades verificam-se:

$$\begin{aligned} x + 0 &= x, & \forall x \in FF; \\ x * 1 &= x, & \forall x \in FF; \\ x * 0 &= 0, & \forall x \in FF; \\ -x &= (p - 1) * x, & \forall x \in FF; \\ x * (y + z) &= (x * y) + (x * z), & \forall x, y, z \in FF; \\ p * x &= 0, & \forall x \in FF. \end{aligned}$$

Definição 3.16. (Atribuição ou Valoração) Uma atribuição é uma função $v : \mathcal{V} \mapsto A$, onde $\mathcal{V}$ é um conjunto formado por variáveis e A é um conjunto base.

[dO95]

Definição 3.17. (Significado de um Termo) Seja v uma atribuição, então o significado de um termo t em relação à v , denotado $\| t \|^v$, é definido recursivamente como segue:

i) Se t for uma variável, então $\| t \|^v = v(t)$.

ii) Caso contrário, t será da forma $f(t_1, \dots, t_n)$, com $n \geq 0$, e $\| t \|^v = \mathcal{I}_A[f](\| t_1 \|^v, \dots, \| t_n \|^v)$.

[dO95]

Definição 3.18. (Modelo de uma $Th(E)$) Dizemos que uma Σ_A – álgebra é um modelo de $Th(E)$ se e somente se para toda equação $s = t$ em E e para toda atribuição $v : \mathcal{V} \mapsto A$, o significado de s e t for idêntico, isto é, $\| s \|^v = \| t \|^v$.

[dO95]

Por simplicidade, dizemos ‘modelo de E ’ ao invés de ‘modelo de $Th(E)$ ’. Sendo assim, escrevemos $E \models s = t$ quando, qualquer modelo de E também for modelo de $\{s = t\}$. Tomando como ponto de partida um exemplo apresentado por Dershowitz [DJ90] apresentamos o seguinte exemplo:

Exemplo 3.19. Seja $\mathcal{V} = \{x, y, z, \dots\}$ um conjunto de variáveis. Seja $\Sigma = \{\cdot, e, I\}$ um conjunto de símbolos funcionais com aridades $\text{Arid}(e) = 0, \text{Arid}(I) = 1$ e $\text{Arid}(\cdot) = 2$. Considere o conjunto E de equações dado por:

$E = \{(x \cdot y) \cdot z = x \cdot (y \cdot z), e \cdot x = x, x \cdot e = x, I(x) = I(I(I(x)))\}$. Vamos, agora, apresentar interpretações de tal modo que a álgebra associada às interpretações dadas seja um modelo para a teoria equacional $Th(E)$. Vamos, então, “construir” uma Σ_G -álgebra e mostrar que ela é um modelo para $Th(E)$. Considere o grupo $\langle G; \cdot, e, I \rangle$, onde $Arid(e) = 0$, $Arid(I) = 1$ e $Arid(\cdot) = 2$ e o conjunto universo $G = \{\text{preto, branco, neutro}\}$.

Especifiquemos as seguintes interpretações:

$$\begin{aligned} \mathcal{I}_G[I] &= \{(\text{branco}, \text{preto}), (\text{preto}, \text{branco}), (\text{neutro}, \text{neutro})\} \\ \mathcal{I}_G[\cdot] &= \{((\text{preto}, \text{preto}), \text{branco}), ((\text{branco}, \text{branco}), \text{branco}), \\ &\quad ((\text{preto}, \text{branco}), \text{preto}), ((\text{branco}, \text{preto}), \text{preto}), \\ &\quad ((\text{neutro}, \text{branco}), \text{branco}), ((\text{neutro}, \text{preto}), \text{preto}), \\ &\quad ((\text{branco}, \text{neutro}), \text{branco}), ((\text{preto}, \text{neutro}), \text{preto}), \\ &\quad ((\text{neutro}, \text{neutro}), \text{neutro})\} \\ \mathcal{I}_G[e] &: \mapsto \{\text{neutro}\} \end{aligned}$$

Doravante, para facilitarmos a construção do texto, escrevemos apenas B , P e N ao invés de branco, preto e neutro. Seja $R = \{(w, N), \forall w \in \mathcal{V} : w \neq x, w \neq y, w \neq z\}$. Considere as seguintes atribuições:

$v_1 = \{(x, B), (y, B), (z, B)\} \cup R$	$v_2 = \{(x, B), (y, B), (z, P)\} \cup R$	$v_3 = \{(x, B), (y, P), (z, B)\} \cup R$
$v_4 = \{(x, B), (y, P), (z, P)\} \cup R$	$v_5 = \{(x, P), (y, B), (z, B)\} \cup R$	$v_6 = \{(x, P), (y, B), (z, P)\} \cup R$
$v_7 = \{(x, P), (y, P), (z, B)\} \cup R$	$v_8 = \{(x, P), (y, P), (z, P)\} \cup R$	$v_9 = \{(x, B), (y, B), (z, N)\} \cup R$
$v_{10} = \{(x, B), (y, P), (z, N)\} \cup R$	$v_{11} = \{(x, P), (y, B), (z, N)\} \cup R$	$v_{12} = \{(x, P), (y, P), (z, N)\} \cup R$
$v_{13} = \{(x, B), (y, N), (z, B)\} \cup R$	$v_{14} = \{(x, B), (y, N), (z, P)\} \cup R$	$v_{15} = \{(x, P), (y, N), (z, B)\} \cup R$
$v_{16} = \{(x, P), (y, N), (z, P)\} \cup R$	$v_{17} = \{(x, N), (y, B), (z, B)\} \cup R$	$v_{18} = \{(x, N), (y, B), (z, P)\} \cup R$
$v_{19} = \{(x, N), (y, P), (z, B)\} \cup R$	$v_{20} = \{(x, N), (y, P), (z, P)\} \cup R$	$v_{21} = \{(x, B), (y, N), (z, N)\} \cup R$
$v_{22} = \{(x, P), (y, N), (z, N)\} \cup R$	$v_{23} = \{(x, N), (y, B), (z, N)\} \cup R$	$v_{24} = \{(x, N), (y, P), (z, N)\} \cup R$
$v_{25} = \{(x, N), (y, N), (z, B)\} \cup R$	$v_{26} = \{(x, N), (y, N), (z, P)\} \cup R$	$v_{27} = \{(x, N), (y, N), (z, N)\} \cup R$

Note que acabamos de expressar todas as possíveis atribuições para o subconjunto $V = \{x, y, z\} \subseteq \mathcal{V}$ de variáveis pertencentes às equações de E . Precisamos agora mostrar que a Σ_G -álgebra construída (que, de fato, é um grupo) é um modelo para E . Para isto, precisamos mostrar que, para cada equação $s = t$ de E , $\|s\|^{v_i} = \|t\|^{v_i}$ para todo $1 \leq i \leq 27$.

- Para a equação $(x \cdot y) \cdot z = x \cdot (y \cdot z)$ temos:

$$\| (x \cdot y) \cdot z \|^{v_1} = \mathcal{I}_G[\cdot](\| x \cdot y \|^{v_1}, \| z \|^{v_1}) = \mathcal{I}_G[\cdot](\mathcal{I}_G[\cdot](\| x \|^{v_1}, \| y \|^{v_1}), \| z \|^{v_1}) = \mathcal{I}_G[\cdot](\mathcal{I}_G[\cdot](B, B), B) = \mathcal{I}_G[\cdot](B, B) = B.$$

$$\| x \cdot (y \cdot z) \|^{v_1} = \mathcal{I}_G[\cdot](\| x \|^{v_1}, \| y \cdot z \|^{v_1}) = \mathcal{I}_G[\cdot](\| x \|^{v_1}, \mathcal{I}_G[\cdot](\| y \|^{v_1}, \| z \|^{v_1})) = \mathcal{I}_G[\cdot](B, \mathcal{I}_G[\cdot](B, B)) = \mathcal{I}_G[\cdot](B, B) = B.$$

$$\| (x \cdot y) \cdot z \|^{v_2} = \mathcal{I}_G[\cdot](\| x \cdot y \|^{v_2}, \| z \|^{v_2}) = \mathcal{I}_G[\cdot](\mathcal{I}_G[\cdot](\| x \|^{v_2}, \| y \|^{v_2}), \| z \|^{v_2}) = \mathcal{I}_G[\cdot](\mathcal{I}_G[\cdot](B, B), P) = \mathcal{I}_G[\cdot](B, P) = P.$$

$$\| x \cdot (y \cdot z) \|^{v_2} = \mathcal{I}_G[\cdot](\| x \|^{v_2}, \| y \cdot z \|^{v_2}) = \mathcal{I}_G[\cdot](\| x \|^{v_2}, \mathcal{I}_G[\cdot](\| y \|^{v_2}, \| z \|^{v_2})) = \mathcal{I}_G[\cdot](B, \mathcal{I}_G[\cdot](B, P)) = \mathcal{I}_G[\cdot](B, P) = P.$$

analogamente, para $v_3, \dots, v_{27}$.

- Para a equação $(I(x) = I(I(I(x))))$ temos:

$$\| I(x) \|^{v_1} = \mathcal{I}_G[I](\| x \|^{v_1}) = \mathcal{I}_G[I](B) = P.$$

$$\| I(I(I(x))) \|^{v_1} = \mathcal{I}_G[I](\| I(I(x)) \|^{v_1}) = \mathcal{I}_G[I](\mathcal{I}_G[I](\| I(x) \|^{v_1})) = \mathcal{I}_G[I](\mathcal{I}_G[I](\mathcal{I}_G[I](\| x \|^{v_1}))) = \mathcal{I}_G[I](\mathcal{I}_G[I](\mathcal{I}_G[I](B))) = \mathcal{I}_G[I](\mathcal{I}_G[I](P)) = \mathcal{I}_G[I](B) = P.$$

analogamente, para $v_2, \dots, v_{27}$.

- Para a equação $e \cdot x = x$ temos:

$$\| e \cdot x \|^{v_1} = \mathcal{I}_G[\cdot](\| e \|^{v_1}, \| x \|^{v_1}) = \mathcal{I}_G[\cdot](N, B) = B.$$

$$\| x \|^{v_1} = B.$$

analogamente, para $v_2, \dots, v_{27}$.

- Para a equação $x = e \cdot x$ é o caso simétrico da equação acima.

Portanto, a Σ_G – álgebra que especificamos neste exemplo é, de fato, um modelo para a teoria equacional $Th(E)$, ou seja, $\langle G; \cdot, e, I \rangle$ é um modelo para $Th(E)$.

Teorema 3.20. (Completeza - Birkhoff [Bir35]) *Seja E um conjunto de equações, então $E \vdash s = t$ se e somente se $E \models s = t$.*

[dO95]

Convém realçar que alguns autores separam a noção de completeza expressa pelo teorema 3.20 em duas partes:

se $E \models s = t$ então $E \vdash s = t$ (*completeza*);

se $E \vdash s = t$ então $E \models s = t$ (*correção*).

$E \models s = t$ é um conceito semântico: escrever $E \models s = t$ significa dizer que $s = t$ é uma consequência lógica de E , isto é, $s = t$ se verifica em qualquer modelo para E , ou equivalentemente, significa dizer que s e t possuem exatamente o mesmo significado, para qualquer modelo de E . $E \vdash s = t$, entretanto, é um conceito formal dedutivo.

É importante notar que, no exemplo 3.19, dado acima, vimos que a Σ_G – álgebra especificada é um modelo para E . O que fizemos para chegar a esta conclusão foi verificar que a Σ_G – álgebra é um grupo, uma vez que as equações de E “expressam” as identidades válidas para qualquer Σ – álgebra que possua a estrutura de grupo. Consequentemente, qualquer grupo $\langle G'; \cdot, I \rangle$ é um modelo para E , além disso, se t e s são termos quaisquer, construídos a partir dos símbolos $\cdot, I$ e símbolos variáveis, se $\| t \|^\nu = \| s \|^\nu$ para qualquer atribuição $\nu : \mathcal{V} \mapsto G$, então $\| t \|^\nu = \| s \|^\nu$ para qualquer atribuição $\nu : \mathcal{V} \mapsto G'$, para qualquer grupo $\langle G'; \cdot, I \rangle$, uma vez que, o grupo $\langle G; \cdot, I \rangle$ satisfaz todas as identidades de grupo e somente elas. Portanto, pelas observações feitas acima e o Teorema de completeza de Birkhoff [Bir35] podemos garantir a equivalência entre os teoremas de E e as identidades que se verificam em $\langle G; \cdot, I \rangle$. Por fim, note que, no Teorema de completeza de Birkhoff, as noções de dedução e consequência semântica se assemelham. Entretanto, deve-se tomar o devido cuidado para poder-se aplicar corretamente este teorema. Note que não poderíamos chegar à conclusão dada no parágrafo acima simplesmente pelo Teorema de completeza: o Teorema de Birkhoff não nos garante que, se M é modelo para E , então toda consequência semântica de M é teorema de E . Isto só é possível afirmar quando a consequência semântica de M também for consequência semântica para qualquer outro modelo de E .

3.4 Sistemas de Reescrita de Termos

Pode-se entender os sistemas de reescrita de termos como sendo conjuntos formados a partir de “equações orientadas” (pares ordenados de termos) de tal modo que se o par (s, t) é elemento de um sistema de reescrita R então impomos que (t, s) não esteja em R , isto é, os sistemas de reescrita devem ser anti-simétricos. Seja $E = \{t_1 = s_1, t_2 = s_2, \dots, t_i = s_i, \dots\}$. Um sistema de reescrita $R = \{t_1 \rightarrow s_1, t_2 \rightarrow s_2, \dots, t_i \rightarrow s_i, \dots\}$ construído a partir de E pode ser visto como um “empobrecimento” de E .

Os conceitos e resultados apresentados para teorias equacionais são praticamente os mesmos para sistemas de reescrita de termos, exceto aqueles que utilizam de algum modo a

propriedade da simetria. Entretanto, por uma questão de clareza do texto, vamos formalizar os conceitos e resultados básicos para sistemas de reescrita como já fizemos para teorias equacionais. Algumas vezes, doravante, abreviamos “sistema de reescrita de termos” por “SRT”. As próximas 3 definições são baseadas em [DJ90, PS81].

Definição 3.21. (Regra de Reescrita) *Dado um conjunto $\mathcal{T}$ de termos, uma regra de reescrita é um par ordenado (s, t) de termos em $\mathcal{T}$, denotada $s \rightarrow t$ e tal que $\text{Var}(s) \supseteq \text{Var}(t)$.*

Uma regra de reescrita $s \rightarrow t$ é dita *linear à esquerda* se o termo s é linear, $s \rightarrow t$ é dita *linear à direita* se o termo t é linear; por fim, uma regra de reescrita é dita *linear* se ela for linear à esquerda e linear à direita.

Veremos no capítulo 5 que para garantirmos que um SRT R seja noetheriano precisamos mostrar que existe uma ordem noetheriana $\succ$ sobre os termos de $\mathcal{T}_R$ tal que $l \succ r$ para toda regra de reescrita $l \rightarrow r$ em R e que ainda é necessário garantir sua estabilidade sobre os termos de R , com respeito a estrutura dos termos, ou seja, é necessário que se $t' \succ t''$ então $t'[x \leftarrow t] \succ t''[x \leftarrow t]$.

Se $l \rightarrow r \in R$ e x é uma variável tal que $x \in \text{Var}(r)$ e $x \notin \text{Var}(l)$ então, neste caso, não há como obter-se uma relação de ordem $\succ$ estável sobre os termos de R , uma vez que a variável x existente em r e não existente em l pode ser substituída por um termo suficientemente complexo a ponto de tornar qualquer relação de ordem não estável.

Definição 3.22. (Instância de uma Regra de Reescrita) *Seja $s \rightarrow t$ uma regra de reescrita e θ uma substituição, então $s\theta \rightarrow t\theta$ é uma instância da regra de reescrita $s \rightarrow t$.*

Se R é um sistema de reescrita de termos, escrevemos $\mathcal{T}_R$ para indicar o conjunto $\mathcal{T}(\Sigma \cup \mathcal{V})$ onde Σ é o menor alfabeto que contém os símbolos funcionais existentes nos termos presentes nas regras de reescrita em R e $\mathcal{V}$ é um conjunto infinito e enumerável de variáveis e tal que contém as variáveis existentes nas regras de reescrita de R .

Definição 3.23. (Axioma de Reescrita) *Seja R um sistema de reescrita de termos, então definimos a relação de reescrita $\rightarrow_R$ sobre $\mathcal{T}_R \times \mathcal{T}_R$ como se segue. O par $(s, t) \in \rightarrow_R$ se e somente se existir uma regra de reescrita $l \rightarrow r \in R$, uma posição p em s e uma substituição σ tal que:*

$$s/p = l\sigma \quad e \quad t = s[p \leftarrow r\sigma]$$

Os elementos de $\rightarrow_R$ são chamados de axiomas de reescrita de R . Nestas circunstâncias escrevemos $s \rightarrow_R t(l \rightarrow r, p, \sigma)$ ou simplesmente $s \rightarrow_R t$.

Em outras palavras, para qualquer relação de reescrita $\rightarrow_R$, se $s \rightarrow_R t$ então $u[p \leftarrow s\sigma] \rightarrow_R u[p \leftarrow t\sigma]$, para todos os termos s, t em $\mathcal{T}_R$, contexto u , uma posição p em u , e substituição σ .

Conseqüentemente, dizemos que toda relação de reescrita $\rightarrow_R$ possui a propriedade de ser *estável* e *compatível*, onde:

1. se $s \rightarrow_R t$ então $s\theta \rightarrow_R t\theta$ (estabilidade);
2. se $s \rightarrow_R t$ então $u[p \leftarrow s] \rightarrow_R u[p \leftarrow t]$ (compatibilidade).

Escrevemos $s \xrightarrow[p]{R} t$ para indicar que (s, t) está em $\rightarrow_R$ e ainda que t foi obtido a partir de s pela aplicação de alguma regra de R na posição p de s . Daqui por diante, a menos que se diga o contrário, sempre que aparecer o símbolo $\rightarrow$ entende-se que o mesmo é uma relação de reescrita. Note que, enquanto $\xrightarrow{E}$ é simétrica, $\xrightarrow{R}$ é anti-simétrica. Um subtermo s/p sobre o qual uma reescrita ocorre é chamado “*redex*” (do inglês: *reducible expression*). Note que s será irreduzível caso s não tenha redex, pois deste modo não haverá termo t tal que $s \xrightarrow{R} t$.

Definição 3.24. (Teoria de Reescrita) A teoria de reescrita de um sistema de reescrita R , denotada $Th(R)$ é o fecho transitivo-reflexivo de $\xrightarrow{R}$, isto é, $Th(R) = \xrightarrow{*}_R$.

[DJ90]

Como o que foi feito para teorias equacionais, podemos entender que a relação $\xrightarrow{*}_R$ é obtida a partir de $\xrightarrow{R}$, isto é $\xrightarrow{R} \subseteq \xrightarrow{*}_R$, e assumindo como regras de inferência a transitividade e a reflexividade, esquematizadas abaixo:

$\frac{}{s \xrightarrow{*}_R s}$	[reflexividade]
$\frac{t \xrightarrow{*}_R s \quad s \xrightarrow{*}_R u}{t \xrightarrow{*}_R u}$	[transitividade]

Podemos mostrar de modo análogo ao feito para teorias equacionais que $\xrightarrow{*}_R$ é fechada em relação a instanciação e contexto de aplicação, isto é:

$\frac{t \xrightarrow{*}_R s}{t\theta \xrightarrow{*}_R s\theta}$	[instanciação]
$\frac{t \xrightarrow{*}_R s}{f(\dots t \dots) \xrightarrow{*}_R f(\dots s \dots)}$	[contexto de aplicação]

Em teorias equacionais, $\stackrel{*}{\leftrightarrow}_E$ é uma relação de equivalência sobre $\mathcal{T}$ e, ainda mais, é a menor congruência que contém $\stackrel{*}{\leftrightarrow}_E$. Em teorias de reescrita, entretanto, $\stackrel{*}{\rightarrow}_R$ não se constitui numa relação de equivalência, pois $\rightarrow_R$ é anti-simétrica. Entretanto temos:

Lema 3.25. *O fecho transitivo-reflexivo-simétrico $\stackrel{*}{\leftrightarrow}_R$ de $\rightarrow_R$ é a menor congruência que contém $\rightarrow_R$.*

[DJ90]

Observe que, tanto $Th(R)$ quanto $\rightarrow_R$ são ambas relações binárias sobre R e mais, $\rightarrow_R \subseteq Th(R)$. Dizemos que um sistema de reescrita R é uma *interpretação* de um conjunto de equações E se para todo par $(a, b) \in Th(R)$ então $E \vdash a = b$. Em particular, um sistema de reescrita R será uma interpretação de um conjunto E de equações se, para toda equação $a = b \in E$, $a \rightarrow b \in R$ ou $b \rightarrow a \in R$. Note que, se R é uma interpretação de E então $E \vdash a = b$ não implica, necessariamente, que $(a, b) \in Th(R)$. Note, que se $E \neq \emptyset$, $E = \{s_1 = t_1, \dots, s_i = t_i, \dots\}$ e $R = \{s_1 \rightarrow t_1, \dots, s_i \rightarrow t_i, \dots\}$, então $\stackrel{*}{\leftrightarrow}_E = \rightarrow_R \cup \rightarrow_R^{-1} \stackrel{\text{notaq.}}{=} \rightarrow_R \cup \leftarrow_R \stackrel{\text{notaq.}}{=} \stackrel{*}{\leftrightarrow}_R$. Logo, $\stackrel{*}{\leftrightarrow}_E = \stackrel{*}{\leftrightarrow}_R$ e $\stackrel{*}{\leftrightarrow}_E \supseteq \stackrel{*}{\leftrightarrow}_R$.

Definição 3.26. (R-Derivação) *Uma derivação em R ou uma R -derivação é qualquer seqüência (finita ou infinita) $t_0 \rightarrow_R t_1 \rightarrow_R \dots \rightarrow_R t_i \rightarrow_R \dots$ de aplicações de regras de reescrita de R .*

[DJ90]

Definição 3.27. (R-Prova) *Seja R uma interpretação de um conjunto E de equações. Uma prova em R ou uma R -prova de que uma equação $s = t$ é um teorema de E é uma derivação finita justificada $s = s_0 \rightarrow_R s_1 \rightarrow_R \dots \rightarrow_R s_i \rightarrow_R \dots s_{n-1} \rightarrow_R s_n = t$ ($n \geq 0$), onde cada passo $s_i \rightarrow_R s_{i+1}$ é justificado através da referência a uma regra de reescrita $l \rightarrow r$ em R , uma posição p em s_i e uma substituição σ , tal que $s_i/p = l\sigma$ e $s_{i+1} = s_i[p \leftarrow r\sigma]$.*

[DJ90]

Definição 3.28. (Sistema de Reescrita Confluyente) *Um sistema de reescrita R é dito confluyente se e somente se $\forall a, b, c \in \mathcal{T}_R, \exists d \in \mathcal{T}_R : (c \stackrel{*}{\rightarrow}_R a \stackrel{*}{\rightarrow}_R b \Rightarrow c \stackrel{*}{\rightarrow}_R d \stackrel{*}{\rightarrow}_R b)$.*

[DJ90]

Ou seja, R é confluyente se e somente se $\rightarrow_R$ é confluyente. A figura 3.1 apresenta por meio de um diagrama a propriedade de confluência.

Dado um SRT R e dois termos t_1 e t_2 em $\mathcal{T}_R$, dizemos que um par de termos $\langle t_1, t_2 \rangle$ *converge* ou é *convergente* e denotamos por $t_1 \downarrow_R t_2$ se e somente se existir um termo t tal que $t_1 \stackrel{*}{\rightarrow}_R t$ e $t_2 \stackrel{*}{\rightarrow}_R t$. Quando um par de termos não é convergente, ele é chamado de *divergente*.

Dizemos que um termo t é *confluyente* se, sempre que $t \stackrel{*}{\rightarrow}_R s_1$ e $t \stackrel{*}{\rightarrow}_R s_2$ então $s_1 \downarrow_R t_2$.

Figura 3.1: Confluência de Sistema de Reescrita.

Definição 3.29. (Sistema de Reescrita Noetheriano) Um sistema de reescrita R é dito noetheriano se qualquer seqüência $s = s_0 \xrightarrow{R} s_1 \cdots \xrightarrow{R} s_i \xrightarrow{R} \cdots$ de reescrita de termos é finita, isto é, $\forall s \in \mathcal{T}_R \exists t \in \mathcal{T}_R$ tal que $s \xrightarrow{*R} t$ e $\nexists q \in \mathcal{T}_R$ tal que $t \xrightarrow{R} q$.

[DJ90]

Ou seja, um sistema de reescrita R é noetheriano se e somente se $\xrightarrow{R}$ é noetheriana. $\xrightarrow{R}$ é noetheriana se e somente se qualquer R -derivação é finita. Uma R -derivação é finita se e somente se todo termo t possui uma forma normal (irredutível) em relação a $\xrightarrow{R}$.

Uma forma normal de um termo t em relação a $\xrightarrow{R}$ é denotada $R(t)$. Por vezes, quando o sistema de reescrita considerado estiver subtendido no contexto, podemos denotar uma forma normal de um termo t escrevendo apenas t^* . Quando um sistema de reescrita é noetheriano então costuma-se dizer que o sistema de reescrita em questão possui a propriedade noetheriana ou, equivalentemente, que ele possui a propriedade de terminação.

Definição 3.30. (Sistema de Reescrita Completo) Um sistema de reescrita é dito completo ou canônico se ele for confluyente e noetheriano.

[DJ90]

Quando um sistema de reescrita é completo, os termos irredutíveis são costumeiramente chamados de *termos canônicos*. Ainda, se t' é a forma normal do termo t , então costuma-se dizer que t' é a *forma canônica* de t .

Definição 3.31. (Confluência Local) Dizemos que um SRT R é localmente confluyente quando para quaisquer termos t, u, s em $\mathcal{T}_R$ tais que $t \xrightarrow{R} u$ e $t \xrightarrow{R} s$ então $u \xrightarrow{R} s$.

[DJ90]

Ou seja, R é localmente confluyente se e somente se $\xrightarrow{R}$ é localmente confluyente.

Dizemos que um termo t converge localmente ou ainda, que t é localmente convergente

se, sempre que $t \xrightarrow[R]{*} s_1$ e $t \xrightarrow[R]{*} s_2$, então $s_1 \xrightarrow[R]{*} s_2$.

Figura 3.2: Confluência local.

Capítulo 4

Unificação de Termos

Diversas áreas da ciência da computação estudam problemas que envolvem noções sobre unificação e matching. Raulefs et al. [RSSU79] apresentam e comentam algumas destas áreas, tais como Banco de Dados Relacionais, Recuperação de Informações, Visão Computacional, Computação Algébrica, Prova Automática de Teoremas e Linguagens de Programação. Na área de prova automática de teoremas, temos os algoritmos de unificação de termos, que constituem parte essencial da maioria dos provadores de teoremas.

O primeiro algoritmo a solucionar o problema de unificação de termos do cálculo de predicados de primeira ordem (linguagens de primeira ordem) deve-se a Herbrand [Her30], ainda no início da década de trinta, quando ele apresentou um algoritmo para o cálculo de unificadoras mais gerais. Herbrand desenvolveu um algoritmo que substitui termos no lugar das variáveis universalmente quantificadas dos termos fornecidos ao algoritmo, de tal modo que, os termos instanciados tornam-se idênticos. Entretanto, foi apenas com Robinson [Rob65] em meados de sessenta, que a tese de doutorado de Herbrand [Her30] ganhou grande significância no campo da dedução automática de teoremas, quando Robinson aplicou os resultados de Herbrand ao seu método de Resolução [Rob65]. O processo de unificação também é parte central do mecanismo de geração de pares críticos desenvolvido por Knuth e Bendix [KB70], no final dos anos 60, para a solução do *problema da palavra* (do inglês “*word problem*”), isto é, a verificação de equivalência entre termos no contexto de linguagens de primeira ordem.

“A existência de uma unificadora, é mostrado por [Huet 73], ser indecidível para linguagens de terceira ordem, enquanto que a indecidibilidade do problema de unificação de termos para linguagens de segunda ordem foi mostrada por [Goldfarb 81]”, Fages e Huet [FH83].

Há textos na literatura, como Huet e Oppen [HO80], que separam o problema da palavra em duas categorias: *problema da palavra* e *problema uniforme da palavra*. A primeira refere-se a termos que não possuem variáveis como subtermos, isto é, refere-se apenas aos ter-

mos básicos, já a segunda aplica-se não apenas a termos sem variáveis mas também a termos com variáveis. Em geral, quando um texto não faz tal distinção, considera-se que o problema da palavra abrange tanto termos básicos quanto termos com variáveis.

Foi depois do trabalho de Knuth e Bendix que o estudo de Sistemas de Reescrita de Termos obteve seu lugar, designado como tal, no campo da dedução automática.

Facilmente pode-se estender o algoritmo de unificação ordinária para o tratamento de operadores comutativos. Entretanto, foi apenas no final da década de setenta, que Stickel [Sti81] (esta referência foi publicada no início dos anos oitenta, esta referência cita uma versão revisada do artigo apresentado no final da década de setenta) apresentou um algoritmo que solucionou o problema de unificação de termos na presença de operadores associativos-comutativos, ou ainda, associativos-comutativos com unidade. Stickel notou que o processo de unificação, nestes casos, está relacionado com a solução de outro problema, no campo da Álgebra, o da solução de equações diofantinas homogêneas lineares, sendo que o conjunto de todas as soluções não negativas de uma tal equação forma um monóide comutativo em $\mathbb{N}^m$ (onde, m é o número de variáveis distintas presentes na equação diofantina homogênea linear que caracteriza o problema de unificação de dois termos especificados).

Clifford e Preston [CP67] mostraram, no final dos anos sessenta, que um monóide comutativo em $\mathbb{N}^m$ pode ser gerado através de uma base finita, enquanto que Huet [Hue78] foi o primeiro, no final dos setenta, a descrever efetivamente um algoritmo que calculasse a base finita de um monóide comutativo em $\mathbb{N}^m$.

Neste capítulo, descrevemos o algoritmo de unificação de Stickel [Sti81], enquanto que no capítulo 7 veremos como este algoritmo de unificação pode ser utilizado pelo algoritmo de completção de Peterson e Stickel [PS81]. Veremos posteriormente, no capítulo 9, como este algoritmo de completção pode ser utilizado para completar um sistema de reescrita de termos especificado que caracteriza a Morfologia Matemática dos operadores invariantes por translação e isotônicos sobre subconjuntos como uma estrutura algébrica.

A seguir generalizamos os conceitos apresentados pelas definições 2.52 e 2.53 para o caso $E \supseteq \emptyset$.

Os conceitos apresentados nos parágrafos a seguir foram obtidos em Fages e Huet [FH83].

Seja E um conjunto finito, eventualmente vazio, de equações.

Dizemos que duas substituições σ e ρ são *congruentes* – E quando $\forall x \in \mathcal{V}, x\sigma \equiv_E x\rho$, e neste caso, escrevemos por abuso de notação, $\sigma \equiv_E \rho$:

- i) Note que, pela definição 2.44 de congruência, se $t_1, t_2 \in \mathcal{T}_E$ são tais que $t_1 \equiv_E t_2$ então $f(\dots, t_1, \dots) \equiv_E f(\dots, t_2, \dots)$. Logo, se $\forall x \in \mathcal{V} \ x\sigma \equiv_E x\rho$ então $t\sigma \equiv_E t\rho \ \forall t \in \mathcal{T}_E$.
- ii) Trivialmente temos: se $\forall t \in \mathcal{T}_E, t\sigma \equiv_E t\rho$ então $x\sigma \equiv_E x\rho \ \forall x \in \mathcal{V}$, pois $\mathcal{V} \subseteq \mathcal{T}_E$.

Logo, por i) e ii) temos: $\forall x \in \mathcal{V} \ x\sigma \stackrel{V}{\equiv}_E x\rho$ se e somente se $\forall t \in \mathcal{T}_E \ t\sigma \stackrel{V}{\equiv}_E t\rho$.

Se V é um subconjunto qualquer de $\mathcal{V}$ então escrevemos: $\sigma \stackrel{V}{\equiv}_E \rho$ se e somente se $\forall x \in V \ x\sigma \stackrel{V}{\equiv}_E x\rho$, e, portanto, $\forall x \in V \ x\sigma \stackrel{V}{\equiv}_E x\rho$ se e somente se $\forall t \in \mathcal{T}_E : \text{Var}(t) \subseteq V, \ t\sigma \stackrel{V}{\equiv}_E t\rho$. É evidente que, se $\sigma \stackrel{V}{\equiv}_E \rho$, então $\sigma \stackrel{V'}{\equiv}_E \rho, \ \forall V' \subseteq V$. Particularmente, se $\sigma \stackrel{V}{\equiv}_E \rho$ então $\sigma \stackrel{V}{\equiv}_E \rho \ \forall V \subseteq \mathcal{V}$.

Definição 4.1. *Sejam s, t termos e E um conjunto de equações. Dizemos que s é uma instância- E de t , e t é uma generalização- E de s , se existir uma substituição θ tal que $t\theta \stackrel{V}{\equiv}_E s$. Dizemos, neste caso, que θ realiza um “matching- E ” de t em s , ou ainda, dizemos que t admite um matching- E com s através de θ . Chamamos de instanciação- E o processo de encontrar uma tal θ .*

[Sti81]

Definição 4.2. *Denotamos $\sigma \stackrel{V}{\leq}_E \rho$ se, e somente se, existir substituição η tal que $\sigma\eta \stackrel{V}{\equiv}_E \rho$ e dizemos neste caso que σ é mais geral que ρ em E em relação a V ou, equivalentemente, dizemos que ρ é mais específica que σ em E em relação a V . Também chamamos de instanciação- E o processo de encontrar uma tal η .*

[FH83]

Denotamos $\sigma \stackrel{V}{\leq}_E \rho$ se e somente se $\sigma \stackrel{V}{\leq}_E \rho$ e $\rho \not\stackrel{V}{\leq}_E \sigma$, ou seja, $\stackrel{V}{\leq}_E = \stackrel{V}{\leq}_E - \stackrel{V}{\geq}_E$.

No decorrer do texto, vamos omitir V quando $V = \mathcal{V}$ e, neste caso, escrevemos simplesmente $\leq, <$ e $\equiv$ ao invés de $\stackrel{V}{\leq}, \stackrel{V}{<}$ e $\stackrel{V}{\equiv}$ respectivamente. De modo análogo, omitimos E quando $E = \emptyset$ e, neste caso, escrevemos apenas $\stackrel{V}{\leq}, \stackrel{V}{<}$ e $\stackrel{V}{\equiv}$ ao invés de $\stackrel{V}{\leq}_E, \stackrel{V}{<}_E$ e $\stackrel{V}{\equiv}_E$ respectivamente. Finalmente, quando $V = \mathcal{V}$ e $E = \emptyset$, então escremos $\leq, <$ e $=$ ao invés de $\stackrel{V}{\leq}_E, \stackrel{V}{<}_E$ e $\stackrel{V}{\equiv}_E$ respectivamente.

Definição 4.3. *Sejam s, t termos e E um conjunto de equações. Dizemos que s e t são unificáveis- E se existir uma substituição θ tal que $s\theta \stackrel{V}{\equiv}_E t\Phi\theta$, onde Φ é uma permutação tal que $\text{Var}(s) \cap \text{Var}(t\Phi) = \emptyset$, e neste último caso dizemos que θ é uma unificadora- E dos termos s e t .*

[PS81]

Note que, quando E for o conjunto vazio, $s \stackrel{V}{\equiv}_E t$ se e somente se s e t forem idênticos, ou seja, $s = t$.

Exemplo 4.4. *Os termos $f(a, x)$ e $f(y, b)$ com x, y sendo variáveis e a, b constantes, são unificáveis através da substituição $\sigma = \{x \leftarrow b, y \leftarrow a\}$, pois $f(a, x)\sigma = f(a, b) = f(y, b)\sigma$.*

[Sti86]

É relevante citar que alguns autores, como Baader e Siekmann [BS94], definem unificadora- E de dois termos s e t como sendo uma substituição θ tal que $s\theta \equiv_E t\theta$. Sendo assim, note que matching- E não é um caso particular de unificação- E . Considere os termos x e $f(x)$: x admite um matching com $f(x)$ através da substituição $\sigma = \{x \leftarrow f(x)\}$, pois assim, temos $x\sigma = f(x)$. Entretanto, x e $f(x)$ não são unificáveis.

Pela maneira como definimos unificação- E neste texto, também é verdade que matching- E não é um caso particular de unificação- E . Para ver isto consideremos novamente os termos x e $f(x)$ e seja a permutação Φ uma permutação tal que $Var(x\Phi) \cap Var(f(x)) = \emptyset$; tome $\Phi = \{x \leftarrow y\}$ e seja $\theta = \{y \leftarrow f(x), x \leftarrow f(x)\}$. Sendo assim, $x\theta \equiv_E f(x)$, logo $f(x)$ é uma instância- E de x , ou equivalentemente, θ realiza um matching- E de x em $f(x)$. Entretanto, θ não é uma unificadora- E de x e $f(x)$ pois $x\Phi\theta = f(x)$ e $f(x)\theta = f(f(x))$, mas não necessariamente temos $f(x) \equiv_E f(f(x))$.

Porém, em certas situações, podemos garantir que uma substituição θ realizando um matching- E de um termo s em um termo t , também realize uma unificação- E de s com t .

Situação 1: se s e t forem termos, tais que s seja um termo básico, então neste caso, o problema de unificação- E de s e t é exatamente igual ao problema de realizar um matching- E de t em s .

Situação 2: se $Lvar(\theta) \cap Var(t) = \emptyset$, então neste caso, o problema de matching- E de um termo s em t coincide com o da unificação- E destes termos.

Definição 4.5. Denotamos $U_E(s, t)$ o conjunto de todas as unificadoras- E de s e t , isto é:

$$U_E(s, t) = \{\sigma \in \mathcal{S} : t\sigma \equiv_E s\Phi\sigma\}.$$

[FH83]

Definição 4.6. Uma unificadora- E ρ de s e t é independente em relação a V de uma unificadora- E σ de s e t se e somente se $\rho \not\leq_E^V \sigma$ e $\sigma \not\leq_E^V \rho$.

[Sti86]

Se ρ é independente em relação a V de σ e $V = \mathcal{V}$, então dizemos simplesmente que ρ é independente de σ .

Definição 4.7. Seja σ uma unificadora- E de um par (s, t) de termos. Dizemos que σ é uma unificadora- E mais geral de (s, t) em relação a V , quando qualquer outra unificadora- E ρ do par (s, t) ou for independente de σ em relação a V ou $\sigma \leq_E^V \rho$.

[Sti86]

Dizemos simplesmente que σ é unificadora- E mais geral de (s, t) quando adotamos $V = \mathcal{V}$.

Note que, se dois termos s e t são unificáveis- E , então a cardinalidade de $U_E(s, t)$ é infinita. Entretanto, estamos interessados num mecanismo automático (algoritmo) que nos forneça apenas um subconjunto S de $U_E(s, t)$ formado pelas unificadoras- E de s e t , digamos σ' s, tais que qualquer outra unificadora- E de s e t que não esteja em S possa ser obtida a partir de alguma das σ' s por instanciação- E .

Definição 4.8. *Seja $V = Var(s) \cup Var(t)$ e W um conjunto finito de variáveis protegidas tal que $W \supseteq V$. S é um Conjunto Completo de Unificadoras- E de s e t em relação a W se, e somente se:*

- i) $\forall \sigma \in S, \quad Lvar(\sigma) \subseteq V \text{ e } Rvar(\sigma) \cap W = \emptyset$ (proteção de W);
- ii) $\sigma \subseteq U_E(s, t)$ (correção);
- iii) $\forall \rho \in U_E(s, t) \quad \exists \sigma \in S : \sigma \stackrel{V}{\leq}_E \rho$ (completeza);

Ainda mais, se S satisfizer a seguinte condição iv) abaixo, dizemos que S é o Conjunto Completo e Mínimo de Unificadoras- E de s e t em relação a W :

- iv) $\forall \sigma, \sigma' \in S \quad \sigma \neq \sigma' \Rightarrow \sigma \not\stackrel{V}{\leq}_E \sigma'$ (mínimo).

[FH83]

Abreviamos Conjunto Completo de Unificadoras- E (respectivamente, Conjunto Completo e Mínimo de Unificadoras- E) por CSU_E (respectivamente, μCSU_E).

A condição $Rvar(\sigma) \cap W = \emptyset$ imposta acima, pela definição 4.8, pode parecer muito forte, entretanto, considere a seguinte situação proposta por Fages e Huet [FH83], em que $E = \emptyset$, $s = x$ e $t = y$, são termos a serem unificados e $W = V = \{x, y\}$. A unificadora mais geral de (s, t) é $\rho = \{x \leftarrow y\}$, mas observe que ρ não forma um CSU de s e t em relação a W , pois $Rvar(\rho) \cap V \neq \emptyset$. Entretanto, a seguinte substituição $\sigma = \{x \leftarrow z, y \leftarrow z\}$ é uma unificadora de s e t e satisfaz a condição i) da definição 4.8 e ainda $\sigma \stackrel{V}{\leq} \rho$. Ora, mas ρ é uma unificadora mais geral, logo $\rho \leq \sigma$ e portanto, $\rho \stackrel{V}{\leq} \sigma$. Mas, pode-se verificar ainda que $\rho < \sigma$, logo $\rho \leq \sigma$ e $\sigma \not\leq \rho$.

Fages e Huet [FH83] observam que o problema é que o conceito de unificadora mais geral é muito forte para ser generalizado e que basta considerarmos as unificadoras mais gerais em relação o conjunto V das variáveis do par de termos que se quer unificar.

Fages e Huet observam que se ρ é uma unificadora- E de um par (s, t) de termos, tal que $Rvar(\rho) \cap V \neq \emptyset$, então sempre existirá unificadora- E σ , com $Rvar(\sigma) \cap V = \emptyset$, para o mesmo par (s, t) de termos, tal que $\sigma \stackrel{V}{\leq}_E \rho$.

Realmente, não há necessidade de considerarmos $W \neq V$ para obtermos um CSU_E , mas

em certos algoritmos é necessário realizar a unificação entre subtermos e, neste caso, há a necessidade de separar as novas variáveis introduzidas pelas unificadoras das variáveis do contexto de aplicação. Este é o caso, por exemplo, da generalização do algoritmo de completção de Knuth e Bendix, proposta por Peterson e Stickel que veremos no Capítulo 7. Fages e Huet [FH83] demonstram ainda que a existência de um CSU_E não implica obrigatoriamente na existência de um μCSU_E e ainda provam que quando existe um μCSU_E este é único.

Dizemos que um par (σ_1, σ_2) de substituições é unificável se e somente se existir alguma substituição β tal que $\sigma_1\beta = \sigma_2\beta$.

A seguir, descrevemos o algoritmo clássico de unificação ($E = \emptyset$), também citado na literatura como algoritmo ordinário de unificação. Tal algoritmo consiste em “varrer” da esquerda para a direita os dois termos s e t que queremos unificar, procurando pela primeira diferença existente entre os dois termos. Em particular, para a descrição do algoritmo ordinário de unificação vamos considerar a definição de unificação dada por Baader e Siekmann [BS94], isto é, vamos considerar que se θ é uma unificadora de t_1 e t_2 , então $t_1\theta = t_2\theta$. Fazemos isto para que possamos observar o que pode ocorrer (Situação 3) quando permitimos que $\text{Var}(s) \cap \text{Var}(t) \neq \emptyset$. Sendo assim, no algoritmo ordinário de unificação podemos ter duas situações:

Situação 1: Se um dos dois subtermos localizados na posição onde ocorre tal diferença for uma variável, digamos v , e o outro subtermo for tal que não contenha v , então este último subtermo é atribuído à v e tal atribuição é incluída na substituição que está sendo construída para formar uma unificadora mais geral do par de termos (s, t) e s, t são instanciados por esta nova atribuição e o processo continua.

Situação 2: Caso nenhum dos dois subtermos sejam variáveis, então o processo de unificação dos dois termos falha.

Situação 3: Caso um dos subtermos seja uma variável, digamos z , e o outro subtermo seja distinto z mas seja tal que contenha z como subtermo, então o processo de unificação dos dois termos s e t falha. Esta verificação é necessária para garantir que o algoritmo termine, pois suponha que queiramos unificar os termos x e $f(x)$ (ou ainda, qualquer outro termo distinto de x e que contenha x). Logo de início, o algoritmo encontraria uma diferença entre os termos, pois $x \neq f$, e se neste caso fosse permitido a seguinte atribuição $x \leftarrow f(x)$ os termos $f(x)$ e $f(f(x))$ seriam obtidos e o processo de unificação continuaria indefinidamente.

Se após todas as diferenças existentes terem sido verificadas e em nenhuma das diferenças encontradas tenha ocorrido a *Situação 2* ou a *Situação 3* descritas acima, então o processo de unificação termina com sucesso, devolvendo uma substituição σ que é uma unificadora mais

geral do par de termos (s, t) .

O algoritmo ordinário de unificação constrói a substituição (unificadora) incrementalmente, acrescentando novas atribuições à variáveis e à corrente substituição e instanciando-a com as novas atribuições, enquanto também cria novas instâncias dos termos originais que estão sendo unificadas. No pior caso, o algoritmo ordinário de unificação tem um custo exponencial em relação ao comprimento dos termos que estão sendo unificados.

A seguir, apresentamos o algoritmo de unificação de um modo mais computacional. A implementação a seguir, assume que os termos a serem unificados não admitem símbolos funcionais de aridade variável, entretanto, pode-se, facilmente, alterar esta implementação para que ela possa admitir símbolos funcionais de aridade variável. Doravante, quando num algoritmo, escrevermos $x:=y$, significará que o conteúdo de y é atribuído a x .

Algoritmo 1.1 Algoritmo de unificação sintática de termos.

UNIF(s, t): % s, t são argumentos passados por referência.

% As variáveis de programa s, t e σ estão definidas fora de UNIF. Ao término da execução de UNIF, σ conterá a unificadora mais geral, caso haja, dos termos fornecidos ao algoritmo. Em s teremos, ao final da execução, o termo unificador mais geral.

Sejam s, t os termos que se quer unificar.

$n := \langle \rangle$.

enquanto $n \in \text{dom } s$ **faça**

começa

se $sn \neq tn$ **então**

se UNIF_NÚCLEO(s, n, t, n) é *falso* **então retorna falso**;

$n := \text{próximo nó de } s$. % Busca do tipo: pai, filho à esquerda, filho à direita.

termina

retorna verdadeiro

UNIF_NÚCLEO(s, n_1, t, n_2): % s, t são argumentos passados por referência.

se sn_1 é uma variável **ou** tn_2 é uma variável **então**

começa

se sn_1 é uma variável **então** $\text{subtermo} := t/n_2$; $\text{var} := sn_1$

c.contrário $\text{subtermo} := s/n_1$; $\text{var} := tn_2$.

$p := <1>$; continua := *sim*.

enquanto $p \in \text{dom subtermo}$ e continua = *sim* **faça**

se $\text{var} = (\text{subtermo})p$ **então** continua := *não*

c.contrário $p := \text{próximo nó de subtermo}$. % Busca do tipo: pai, filho à esquerda, filho à direita.

se continua = *sim* **então**

$\sigma := \sigma\{\text{var} \leftarrow \text{subtermo}\} \cup \{\text{var} \leftarrow \text{subtermo}\}; s := s\sigma; t := t\sigma$; **retorna verdadeiro**.

c.contrário $\sigma := \emptyset$; **retorna falso**.

termina

c.contrário $\sigma := \emptyset$; **retorna falso**. % o algoritmo termina e os termos não são unificáveis.

Exemplo 4.9. Suponha que queiramos unificar os termos $s = f(u, h(w, w), w, j(y, y))$ e $t = f(g(v, v), v, i(x, x), x)$; abaixo apresentamos as etapas intermediárias, a cada diferença encontrada entre os termos, da construção da unificadora mais geral do par (s, t) de termos:

$$s = f(\overset{\downarrow}{u}, h(w, w), w, j(y, y)) \quad e \quad t = f(\overset{\downarrow}{g}(v, v), v, i(x, x), x)$$

$$\Rightarrow \begin{aligned} \sigma &= \sigma_0 = \{u \leftarrow g(v, v)\} \\ s_1 &= s\sigma_0 = f(g(v, v), h(w, w), w, j(y, y)) \\ t_1 &= t\sigma_0 = f(g(v, v), v, i(x, x), x) \end{aligned}$$

$$s_1 = f(g(v, v), \overset{\downarrow}{h}(w, w), w, j(y, y)) \quad e \quad t_1 = f(g(v, v), \overset{\downarrow}{v}, i(x, x), x)$$

$$\Rightarrow \begin{aligned} \sigma_1 &= \{v \leftarrow h(w, w)\} \\ \sigma &= \sigma\sigma_1 \cup \sigma_1 = \{u \leftarrow g(h(w, w), h(w, w)), h(w, w)\} \cup \{v \leftarrow h(w, w)\} \\ s_2 &= s_1\sigma_1 = f(g(h(w, w), h(w, w)), h(w, w), w, j(y, y)) \\ t_2 &= t_1\sigma_1 = f(g(h(w, w), h(w, w)), h(w, w), i(x, x), x) \end{aligned}$$

$$s_2 = f(g(h(w, w), h(w, w)), h(w, w), \overset{\downarrow}{w}, j(y, y)) \quad e \quad t_2 = f(g(h(w, w), h(w, w)), h(w, w), \overset{\downarrow}{i}(x, x), x)$$

$$\Rightarrow \begin{aligned} \sigma_2 &= \{w \leftarrow i(x, x)\} \\ \sigma &= \sigma\sigma_2 \cup \sigma_2 = \{u \leftarrow g(h(i(x, x), i(x, x)), h(i(x, x), i(x, x))), v \leftarrow h(i(x, x), i(x, x))\} \cup \\ &\quad \{w \leftarrow i(x, x)\} \\ s_3 &= s_2\sigma_2 = f(g(h(i(x, x), i(x, x)), h(i(x, x), i(x, x))), h(i(x, x), i(x, x)), i(x, x), j(y, y)) \\ t_3 &= t_2\sigma_2 = f(g(h(i(x, x), i(x, x)), h(i(x, x), i(x, x))), h(i(x, x), i(x, x)), i(x, x), x) \end{aligned}$$

$$s_3 = f(g(h(i(x, x), i(x, x)), \dots, i(x, x), \overset{\downarrow}{j}(y, y))) \quad e \quad t_3 = f(g(h(i(x, x), i(x, x)), \dots, i(x, x), \overset{\downarrow}{x}))$$

$$\begin{aligned}
\sigma_3 &= \{x \leftarrow j(y, y)\} \\
\sigma &= \sigma\sigma_3 \cup \sigma_3 = \{u \leftarrow g(h(i(j(y, y), j(y, y)), i(j(y, y), j(y, y))), h(i(j(y, y), j(y, y)), i(j(y, y), j(y, y)))), \\
&\quad v \leftarrow h(i(j(y, y), j(y, y)), i(j(y, y), j(y, y)))\} \cup \{w \leftarrow i(x, x)\} \\
\Rightarrow s_4 &= s_3\sigma_3 = f(g(h(i(j(y, y), j(y, y)), i(j(y, y), j(y, y))), h(i(j(y, y), j(y, y)), i(j(y, y), j(y, y)))), \\
&\quad h(i(j(y, y), j(y, y)), i(j(y, y), j(y, y))), i(j(y, y), j(y, y)), j(y, y)) \\
t_4 &= t_3\sigma_3 = f(g(h(i(j(y, y), j(y, y)), i(j(y, y), j(y, y))), h(i(j(y, y), j(y, y)), i(j(y, y), j(y, y)))), \\
&\quad h(i(j(y, y), j(y, y)), i(j(y, y), j(y, y))), i(j(y, y), j(y, y)), j(y, y))
\end{aligned}$$

O algoritmo obtém σ como uma unificadora mais geral do par (s, t) de termos. Note que $s\sigma = s_4 = t_4 = t\sigma$.

[Sti86]

Acabamos de exibir um algoritmo que decide se dois termos são ou não unificáveis. Pode-se ver que também existe um algoritmo que decide se um termo s é ou não uma instância de um termo t e, caso seja, encontre tal substituição θ tal que $s = t\theta$.

Pelo lema 4.10 a seguir, não é difícil ver que existe um algoritmo similar para instâncias- E , contanto que, para um dado termo s , $[s]_E$ seja finita e exista um algoritmo para gerá-la, isto é, obter todos os seus elementos.

Lema 4.10. *Seja E um conjunto de equações e sejam $s, t \in \mathcal{T}_E$. As seguintes proposições são equivalentes:*

- (1) s é uma instância- E de t .
- (2) Existe $s' \in [s]_E$ tal que s' é uma instância de t .

[PS81]

Demonstração: Trivial, uma vez que $\equiv_E$ é uma relação de equivalência, logo $s \equiv_E t\theta$ se e somente se $s, t\theta \in [s]_E$, sendo assim basta tomarmos $s' = t\theta$. ■

Sob estas circunstâncias, isto é, se as classes de congruência induzidas por E forem finitas e houver um algoritmo para gerá-las, então haverá também um algoritmo para testar se $s \equiv_E t$, bastando para isto gerar $[t]_E$ e testar se $s \in [t]_E$.

Infelizmente não podemos testar se dois termos são unificáveis- E através de um processo tão simples. Poderia-se pensar que a unificabilidade- E de s e t fosse equivalente à existência de $s' \in [s]_E$ e $t' \in [t]_E$ tal que s' e t' fossem unificáveis, entretanto o exemplo 4.11 baseado em Peterson e Stickel [PS81] mostra que isto é falso.

Exemplo 4.11. Tomemos $E = \{f(c) = g(c)\}$ e $x, y \in \mathcal{V}$. Então temos $\mathcal{T}_E = \{c, x, y, f(x), f(y), g(x), g(y), f(f \cdots f(x) \cdots), f(f \cdots f(y) \cdots), g(g \cdots g(x) \cdots), g(g \cdots g(y) \cdots), f(g(x)), f(g(y)), g(f(x)), g(f(y)), f(f \cdots f(g(g \cdots g(x) \cdots)) \cdots), f(f \cdots f(g(g \cdots g(y) \cdots)) \cdots), g(g \cdots g(f(f \cdots f(x) \cdots)) \cdots), g(g \cdots g(f(f \cdots f(y) \cdots)) \cdots), \dots, f(c), g(c), f(f \cdots f(c) \cdots), g(g \cdots g(c) \cdots), f(g(c)), g(f(c)), f(f \cdots f(g(g \cdots g(c) \cdots)) \cdots), g(g \cdots g(f(f \cdots f(c) \cdots)) \cdots), \dots\}$.

Pelo conjunto E de equações escolhido, temos que:

$$\begin{aligned} \leftrightarrow_E = \{ & \langle f(c), g(c) \rangle, \langle f^2(c), f(g(c)) \rangle, \dots, \langle f^k(c), f^{k-1}(g(c)) \rangle, \dots, \langle g(f(c)), g^2(c) \rangle, \dots, \\ & \langle g^{k-1}(f(c)), g^k(c) \rangle, \dots, \langle g(c), f(c) \rangle, \langle f(g(c)), f^2(c) \rangle, \dots, \langle f^{k-1}(g(c)), f^k(c) \rangle, \dots, \\ & \langle g^2(c), g(f(c)) \rangle, \dots, \langle g^k(c), g^{k-1}(f(c)) \rangle, \dots \} \end{aligned}$$

Mas como $\equiv_E$ é o fecho transitivo-reflexivo da relação $\leftrightarrow_E$ temos as seguintes classes de congruência módulo E , representadas pela figura 4.1:

Figura 4.1: Diagrama representativo das classes de congruência induzidas por $\equiv_E$.

Tomemos $s = f(x)$ e $t = g(y)$, logo tanto s como t não pertencem a nenhuma classe de congruência induzida por $\equiv_E$. Entretanto s e t são unificáveis- E , pois basta tomarmos $\theta = \{x \leftarrow c, y \leftarrow c\}$ para obtermos $s\theta = f(c) \equiv_E g(c) = t\theta$. Isto é, para verificar se dois termos, digamos s e t , são unificáveis- E não é suficiente verificar-se a existência de $s' \in [s]_E$ e $t' \in [t]_E$ tais que s' e t' sejam unificáveis. Note ainda que, pelo corolário 3.11 se $s \equiv_E t \Rightarrow s\theta \equiv_E t\theta, \forall \theta$, onde θ é uma substituição. Observe que não vale, necessariamente, a “volta” ($\Leftarrow$), isto é, $s\theta \equiv_E t\theta \not\Rightarrow s \equiv_E t, \forall \theta$.

[PS81]

Muitas vezes pode-se querer unificar termos que possuem símbolos funcionais (operadores) comutativos, associativos ou associativos-comutativos, estes últimos frequentemente chamados na literatura de *operadores-AC*. Igualmente, um *termo-AC* é aquele que possui algum operador-AC. Uma das dificuldades em unificar-se dois termos-AC que possuam como símbolo raiz um mesmo operador-AC (é claro que, se os termos a serem unificados possuem em suas raízes operadores diferentes, então eles não são unificáveis) é que há várias representações para um mesmo termo. Por exemplo, os termos $f(a, f(b, c))$, $f(a, f(c, b))$, $f(b, f(a, c))$, $f(f(a, c), b)$, ... são todos congruentes-AC se f for associativo-comutativo, isto é, se f admite os seguintes esquemas de axiomas $f(f(x, y), z) = f(x, f(y, z))$ e $f(x, y) = f(y, x)$, $\forall x, y, z$ símbolos variáveis.

Basicamente, há duas abordagens para a unificação de termos-AC. A primeira delas, e bem menos eficiente, consiste em expressar a propriedade associativa-comutativa de um operador de modo axiomático, como exibido no parágrafo anterior. Deste modo, o algoritmo de unificação utilizado é o clássico, que acabamos de descrever de modo sucinto, sendo necessário chamá-lo várias vezes para que se possa unificar dois termos-AC, onde cada chamada ao algoritmo de unificação consiste em considerar as diversas representações possíveis dos termos envolvidos no processo de unificação. Este método torna-se ineficiente, pois o espaço de busca para a unificação torna-se demasiadamente extenso sendo necessário considerarmos todas as permutações, no que diz respeito à comutatividade, para cada representação distinta de um termo, segundo à associatividade. Ainda, segundo Stickel [Sti86], outro fato que torna tal método ineficiente, é que, em geral, obtêm-se unificadoras que não são as mais gerais, além do fato de em certas circunstâncias poderemos obter um conjunto infinito de unificadoras, fazendo com que o algoritmo, na prática, não termine.

A segunda maneira de tratar a unificação de termos-AC é incorporar a associatividade-comutatividade dentro do próprio processo de unificação, e não mais como regras integrantes de um conjunto de axiomas. Este método de abordar o problema de unificação na presença de operadores-AC tem-se mostrado muito mais eficiente que o primeiro, pois desta maneira garante-se que o algoritmo de unificação-AC termine, seja correto e completo. Uma discussão ampla sobre os méritos de incorporar-se a propriedade associativa-comutativa dentro do algoritmo de unificação, ao invés de considerá-la como esquemas de axiomas, externos ao processo de unificação, pode ser encontrada em Plotkin [Plot72].

O próximo lema nos mostra que, em certas circunstâncias, o problema de matching- E é um caso particular de unificação- E , mesmo quando os dois termos envolvidos na unificação não forem termos básicos.

Lema 4.12. *Sejam s, t termos-AC com $\text{Var}(s) \cap \text{Var}(t) = \emptyset$ e σ uma substituição tal que $s\sigma \stackrel{AC}{=} t$ e ainda $L\text{var}(\sigma) \cap R\text{var}(\sigma) = \emptyset$, então σ é uma unificadora-AC do par (s, t) de termos-AC, isto é, $s\sigma \stackrel{AC}{=} t\sigma$.*

Demonstração:

Note que as congruências- AC satisfazem a seguinte propriedade: se $t_1 \overline{\overline{AC}} t_2$ então $Var(t_1) = Var(t_2)$. Suponha por absurdo que $t \not\overline{\overline{AC}} t\sigma$, logo $\exists v \in Var(t) : v \in Lvar(\sigma)$. Mas, por hipótese, $s\sigma \overline{\overline{AC}} t$, logo $v \in Var(s\sigma)$ e portanto ou $v \notin Lvar(\sigma)$, que é uma contradição, ou $v \in Lvar(\sigma)$ e neste caso também temos uma contradição. Portanto, $t \overline{\overline{AC}} t\sigma$ e, conseqüentemente, $s\sigma \overline{\overline{AC}} t\sigma$. ♦

Sabemos que o problema de matching de dois termos- AC é $\mathcal{NP}$ -completo [BKN85]. Informalmente, dizer que o problema de matching- E é $\mathcal{NP}$ significa que não se conhece um algoritmo capaz de solucionar este problema em tempo polinomial e ainda, dada qualquer solução potencial para o problema pode-se verificar, em tempo polinomial, se a solução dada é correta ou não. Além disso, matching- E é $\mathcal{NP}$ -completo, isto é, este é um problema $\mathcal{NP}$ e ainda todo problema $\mathcal{NP}$ se reduz polinomialmente ao problema de matching- E ; como conseqüência, se um algoritmo polinomial for exibido para a solução do problema de matching- E , então teríamos $\mathcal{P} = \mathcal{NP}$.

Note que os CSU_E (veja definição 4.8) são formados por unificadoras, digamos σ' s, que satisfazem a condição imposta pelo lema 4.12, isto é, $Lvar(\sigma) \cap Rvar(\sigma) = \emptyset$. Logo, pelo lema 4.12 o processo de encontrar uma unificadora- AC que componha o CSU_E de um par de termos- AC possui, pelo menos, a mesma dificuldade do problema de matching- AC .

4.1 Unificação Comutativa

Podemos alterar, sem muita dificuldade, o algoritmo clássico de unificação de termos para que o mesmo possa resolver o problema de unificação de termos que possuam operadores comutativos. Por exemplo, se f é um operador comutativo, quando tentarmos unificar os termos $f(s_1, s_2)$ e $f(t_1, t_2)$ será necessário tentar unificar os argumentos s_1 com t_1 e s_2 com t_2 como no caso clássico de unificação, mas também será necessário tentar unificar s_1 com t_2 e s_2 com t_1 . Esta simples modificação do algoritmo clássico de unificação garante um novo algoritmo que é completo para o caso de operadores comutativos, veja Stickel [Sti86].

Podemos observar que o algoritmo de unificação comutativa possui uma complexidade muito maior que o algoritmo clássico, visto que precisamos considerar nesta extensão as permutações dos subtermos envolvidos, decorrentes da presença de operadores comutativos. Outra observação importante é que a unificação comutativa pode gerar um conjunto não unitário de unificadoras mais gerais, ao contrário do caso clássico.

Por exemplo, caso queiramos unificar os termos $s = f(x, y)$ e $t = f(a, b)$, onde f é comutativo, obtemos as unificadoras mais gerais $\{x \leftarrow a, y \leftarrow b\}$ e $\{x \leftarrow b, y \leftarrow a\}$ dos termos s e t . A seguir, apresentamos o algoritmo de unificação comutativa de uma forma mais computa-

cional.

Algoritmo 1.2 Algoritmo de unificação comutativa de termos.

UNIF_C(s, n_1, t, n_2): % s, t são argumentos passados por referência.

% σ, t, s são variáveis de programa declaradas fora de UNIF_C. Ao termino da execução de UNIF_C, σ conterá uma unificadora mais geral, caso haja, dos termos fornecidos. s será, ao final da execução do algoritmo, o termo unificador mais geral dos termos inicialmente fornecidos para serem unificados. Caso os termos comutativos a serem unificados possuírem duas unificadoras mais gerais, este algoritmo devolverá apenas uma delas. Facilmente pode-se modificar este algoritmo para que ele retorne as duas unificadoras mais gerais, caso elas existam.

Sejam s, t os termos que se quer unificar.

Inicialmente, UNIF_C dever ser chamado com $n_1 := \langle \rangle$ e $n_2 := \langle \rangle$.

Sejam s', t', aux, i variáveis auxiliares do programa.

```

se ( $s/n_1$ ) $\langle \rangle = (t/n_2)\langle \rangle$  então
  se ( $s/n_1$ ) $\langle \rangle$  é comutativo então
    começa
       $s' := s, t' := t$ ;
       $aux := UNIF\_C(s, n_1 \cdot \langle 1 \rangle, t, n_2 \cdot \langle 1 \rangle)$ ;
      se  $aux = verdadeiro$  então
        começa
           $aux := UNIF\_C(s, n_1 \cdot \langle 2 \rangle, t, n_2 \cdot \langle 2 \rangle)$ ;
          se  $aux = verdadeiro$  então retorna verdadeiro
        c.contrário
          começa
             $s := s'; t := t'$ .
             $aux := UNIF\_C(s, n_1 \cdot \langle 1 \rangle, t, n_2 \cdot \langle 2 \rangle)$ ;
            se  $aux = verdadeiro$  então
              começa
                 $aux := UNIF\_C(s, n_1 \cdot \langle 2 \rangle, t, n_2 \cdot \langle 1 \rangle)$ ;
                se  $aux = verdadeiro$  então retorna verdadeiro
              c.contrário  $s := s'; t := t'$ ; retorna falso.
  
```



```

      termina.
    termina.
  termina.
termina.
c.contrário
  se  $s/n_1$  é uma constante então retorna verdadeiro
c.contrário
  começa
     $i := 1$ ;
    enquanto  $n \leq \text{Arid}((s/n_1) < >)$  faça
      começa
        se  $\text{UNIF\_C}(s, n_1 \cdot i, t, n_2 \cdot i) = \text{falso}$  então retorna falso;
         $i := i + 1$ ;
      termina.
    retorna verdadeiro.
  termina.
c.contrário
  retorna  $\text{UNIF\_NÚCLEO}(s, n_1, t, n_2)$ .

```

A seguir, apresentamos um exemplo de execução da implementação dada acima ao algoritmo de unificação comutativa.

Exemplo 4.13. *Sejam os termos $s = f(x, g(y))$ e $t = f(g(z), b)$ que se quer unificar, onde f é um símbolo funcional comutativo, g é um símbolo ordinário com $\text{Arid}(g) = 1$, b é uma contante e x e y são variáveis. A figura 4.2 apresenta o diagrama de recursão do algoritmo UNIF_C (1.2), descrito acima, para a instância s e t dada.*

4.2 Unificação Associativa

O processo de unificar dois termos associativos é mais difícil que o caso comutativo. Há situações, neste caso, onde a cardinalidade do conjunto completo de unificadoras (CSU) é infinito, como por exemplo na unificação dos termos $f(a, x)$ e $f(x, a)$, sendo f um símbolo funcional associativo, as unificadoras mais gerais são $\{x \leftarrow a\}, \{x \leftarrow f(a, a)\}, x \leftarrow f(a, f(a, a)), \dots\}$.

No caso de símbolos funcionais associativos, é útil não fazer distinção de representações como $f(x, f(y, z))$ e $f(f(x, y), z)$, mas sim representar ambos os termos como $f(x, y, z)$, tornando desta forma f um símbolo funcional de aridade não fixa (aridade variável).

Observe que para unificar os termos $s = f(x, y)$ e $t = f(a, b, c)$ não basta, como no

Figura 4.2: Diagrama de recursão para uma instância do algoritmo UNIF-C.

algoritmo clássico de unificação, as atribuições $x \leftarrow a$ e $y \leftarrow b$. Uma vez que os termos associativos não possuem aridade fixa, é necessário primeiro tentar as atribuições $x \leftarrow a$ e $x \leftarrow f(a, u)$, onde u é uma variável nova, isto é, u não pode ser subtermo de s e nem de t e u não pode ter sido utilizado em nenhuma atribuição pertencente à substituição que está sendo criada para ser uma unificadora do par (s, t) de termos.

O problema agora consiste em tentarmos unificar $f(a, y)$ e $f(a, u, y)$ com $f(a, b, c)$. Para unificar $f(a, y)$ com $f(a, b, c)$ é necessário fazer $y \leftarrow b$ e $y \leftarrow f(b, v)$ e conseqüentemente resolver o seguinte sub-problema que consiste de tentarmos unificar os termos $f(a, b)$ e $f(a, b, v)$ com o termo $f(a, b, c)$, mas $f(a, b)$ não é unificável com $f(a, b, c)$, logo $y \leftarrow b$ é descartada, agora $f(a, b, v)$ e $f(a, b, c)$ são unificáveis através da unificadora $\{v \leftarrow c\}$. Obtemos então uma unifi-

cadora $\{x \leftarrow a, y \leftarrow f(b, c)\}$ para o par $(f(x, y), f(a, b, c))$ de termos.

Falta ainda resolver o segundo sub-problema, que consiste em tentarmos unificar o termo $f(a, u, y)$ com o termo $f(a, b, c)$, sendo assim fazemos as seguintes atribuições $u \leftarrow b$ e $u \leftarrow f(b, w)$. Precisamos agora tentar unificar os termos $f(a, b, y)$ e $f(a, b, w, y)$ com $f(a, b, c)$. Para unificar $f(a, b, y)$ com $f(a, b, c)$ basta fazer $y \leftarrow c$, entretanto $f(a, b, w, a)$ e $f(a, b, c)$ não são unificáveis, conseqüentemente a atribuição $u \leftarrow f(b, w)$ é descartada e obtemos a segunda e última unificadora $\{x \leftarrow f(a, b), y \leftarrow c\}$ para o par $(f(x, y), f(a, b, c))$ de termos.

Nos parágrafos acima procuramos dar algumas idéias de como funciona o algoritmo de unificação associativa descrito em Stickel [Sti86]. Este algoritmo de unificação de termos associativos é correto e completo, entretanto este algoritmo não termina quando $|U_{(s,t)}| = +\infty$ como já mencionamos (por exemplo, se $s = f(a, x)$ e $t = f(x, a)$ onde f é um operador associativo).

O algoritmo de unificação de termos associativos pode ainda não terminar em certos casos onde o par de termos associativos não forem unificáveis, como exemplo basta tomarmos $s = f(a, x)$ e $t = f(x, b)$, onde f é associativo, e o algoritmo tentará encontrar alguma unificadora a partir das sucessivas atribuições $x \leftarrow a, x \leftarrow f(a, u_1), x \leftarrow f(a, a, u_2), \dots$

Conforme citado por Stickel [Sti86], em 1977 Makanin apresentou uma prova da decidibilidade da unificação associativa para os casos restritos a termos associativos $f(x_1, \dots, x_i, \dots, x_n)$ onde cada x_i pode ser ou uma variável ou uma constante; segundo Stickel [Sti86], Jaffar, em 1985, apresentou um algoritmo *completo* e *Mínimo* para o caso particular de unificação associativa abordado por Makanin.

Em 1992, Baader e Schulz [BS92] demonstram a decidibilidade da unificação associativa para o caso geral.

4.3 Unificação Associativa-Comutativa

Como já citamos no início deste capítulo, as soluções positivas de equações diofantinas homogêneas (isto é, cada termo da equação possui o mesmo grau) lineares são parte importante dentro dos algoritmos de unificação sobre termos que contêm símbolos funcionais associativos-comutativos. A seguir descreveremos o algoritmo de Huet [Hue78] que calcula a base de soluções não negativas destas equações. Este algoritmo é por sua vez, parte essencial do algoritmo de unificação de termos-AC proposto por Stickel [Sti81] que descrevemos na subseção 4.3.2.

4.3.1 Soluções de Equações Diofantinas Homogêneas Lineares

Seja $a_1, \dots, a_m, b_1, \dots, b_n$ números inteiros estritamente positivos (≥ 0). Vamos descrever o algoritmo de Huet [Hue78] que resolve o seguinte esquema de equação,

$$\sum_{i=1}^m a_i \cdot x_i = \sum_{j=1}^n b_j \cdot y_j, \quad (4.1)$$

para soluções inteiras não negativas. O conjunto de todas as soluções deste tipo (soluções inteiras não negativas) para uma equação do tipo (4.1) forma um monóide comutativo em $\mathbb{N}^{m+n}$. A partir de uma base finita, formada por soluções mínimas, pode-se gerar o conjunto formado por todas soluções não negativas através de combinações lineares aditivas da base [CP67]. Dizemos que uma solução não negativa (x, y) para uma equação da forma (4.1) é *mínima* se e somente se

i) ela não for trivial, isto é, algum x_i da solução deve ser diferente de zero;

ii) não existir uma outra solução não negativa (x', y') não trivial e diferente de (x, y) tal que $(x', y') \preceq (x, y)$

(onde $(x', y') \preceq (x, y)$ se e somente se para todo $\forall i, 1 \leq i \leq m$ e $\forall j, 1 \leq j \leq n$, $x'_i \leq x_i$ e $y'_j \leq y_j$, sendo $\leq$ o símbolo usual de menor ou igual da aritmética).

A base de soluções não negativas de uma equação da forma (4.1) consiste de todas as soluções mínimas. Descrevemos agora o algoritmo desenvolvido por Huet [Hue78] que gera esta base.

Sejam,

$$\max a = \max\{a_i | 1 \leq i \leq m\} \text{ e } \max b = \max\{b_i | 1 \leq i \leq n\}.$$

Veremos agora que é possível garantir o seguinte: se uma solução (x, y) é mínima, então seu valor, ou seja, os valores dos x_i 's e dos y_j 's, são garantidamente menores ou iguais a um certo valor. É isto o que o lema 4.14 nos mostra.

Lema 4.14. *Seja $(x, y) \in \mathbb{N}^m \times \mathbb{N}^n$ uma solução mínima para uma equação da forma (4.1). Então:*

$$\forall i \leq m, x_i \leq \max b \text{ e } \forall j \leq n, y_j \leq \max a$$

[Hue78]

Demonstração: Pela simetria do problema, vamos exibir uma prova apenas para a primeira metade do lema.

Suponha que (x, y) seja uma solução para uma equação da forma (4.1) tal que $\exists k \leq m$, $x_k > \max b$. Vamos mostrar que se isto ocorrer, então (x, y) não será uma solução mínima. Temos que:

$$\max b \cdot \sum_{j=1}^n y_j \geq \sum_{j=1}^n b_j \cdot y_j = \sum_{i=1}^m a_i \cdot x_i \geq a_k \cdot x_k$$

como por hipótese, $\exists k \leq m : x_k > \max b$, então:

$$a_k \cdot x_k > a_k \cdot \max b$$

e portanto,

$$\sum_{j=1}^n y_j > a_k. \quad (1)$$

Considere agora $u = (u_1, \dots, u_n) \in \mathbb{N}^n$ tal que:

$$\forall i \leq n, u_i \leq y_i \quad (2)$$

$$\text{e} \\ 0 < \sum_{j=1}^n u_j \leq a_k. \quad (3)$$

Como consequência de (1) existe uma cadeia crescente (segundo a ordem $\preceq$), composta por a_k elementos distintos de tais u 's em $\mathbb{N}^n$. Logo, existe um tal $u = (u_1, \dots, u_n)$ para o qual verificamos o seguinte:

$$\sum_{j=1}^n b_j \cdot u_j \equiv 0 \pmod{a_k}. \quad (4)$$

Suponha que um tal u não existisse, então existiria dois elementos distintos nesta cadeia crescente de tais u 's, chamemo-los de v e w , tais que:

$$\sum_{j=1}^n b_j \cdot v_j \equiv \sum_{j=1}^n b_j \cdot w_j \pmod{a_k}$$

Note que v e w são elementos da cadeia crescente de u 's considerada, logo $v < w$ ou $w < v$, suponha, sem perda de generalidades que $w < v$, logo:

$$\sum_{j=1}^n b_j \cdot v_j - \sum_{j=1}^n b_j \cdot w_j \equiv 0 \pmod{a_k}$$

ou seja:

$$b_1 \cdot v_1 + b_2 \cdot v_2 + \dots + b_n \cdot v_n - v_1 \cdot w_1 - v_2 \cdot w_2 - \dots - v_n \cdot w_n \equiv 0 \pmod{a_k}$$

$$b_1 \cdot (v_1 - w_1) + b_2 \cdot (v_2 - w_2) + \dots + b_n \cdot (v_n - w_n) \equiv 0 \pmod{a_k}$$

Mas $u = v - w$, é um elemento da cadeia crescente de u 's. Contradição!

A contradição surgiu do fato de supormos que não existe u na cadeia crescente de u 's tal que $\sum_{j=1}^n b_j \cdot u_j \equiv 0 \pmod{a_k}$.

Considere então um tal u que satisfaça (2), (3) e (4), então temos:

$$\sum_{j=1}^n b_j \cdot u_j \leq \max b \cdot \sum_{j=1}^n u_j \leq \max b \cdot a_k < a_k \cdot x_k \quad (5)$$

e agora usando (4) e (5) obtemos:

$$\sum_{j=1}^n b_j \cdot u_j = a_k \cdot z_k \quad \text{para algum } z_k \text{ tal que } 0 < z_k < x_k,$$

o que nos mostra que (z, u) é uma solução não trivial e que a solução (x, y) é estritamente maior que (z, u) , com $\forall i \neq k, z_i = 0$ e $z_k < x_k$. ■

O lema 4.14 acima nos fornece uma condição de necessidade para que as soluções não negativas de uma equação da forma (4.1) sejam mínimas.

Consideremos agora, $d_{ij} = \text{mmc}(a_i, b_j)/a_i$ e $e_{ij} = \text{mmc}(a_i, b_j)/b_j$, para $1 \leq i \leq m$ e $1 \leq j \leq n$.

Seja $S_{ij} = (x, y) \in \mathbb{N}^m \times \mathbb{N}^n$ tal que $x_i = d_{ij}$, $y_j = e_{ij}$ e todas as outras coordenadas com valor zero. Os S_{ij} 's são soluções especiais para as equações da forma (4.1).

Lema 4.15. *Seja $a \cdot x = b \cdot y$ uma equação com $a, b \in \mathbb{Z}_+$ e x, y variáveis que podem assumir apenas valores inteiros estritamente positivos. Então (x, y) , onde $x = \frac{\text{mmc}(a, b)}{a}$ e $y = \frac{\text{mmc}(a, b)}{b}$ é solução da equação e ainda, não existe outra solução (não trivial) (x', y') para a mesma equação tal que $x' < x$ e $y' < y$.*

Demonstração: Se $x = \frac{\text{mmc}(a, b)}{a}$ e $y = \frac{\text{mmc}(a, b)}{b}$, então (x, y) é solução para a equação $a \cdot x = b \cdot y$.

Resta-nos mostrar que, qualquer outra solução não trivial (x', y') para a equação dada é tal que $x' > x$ e $y' > y$.

Observe que toda solução (x', y') será da forma $x' = \frac{(b \cdot y')}{a}$ e $y' = \frac{(a \cdot x')}{b}$, conseqüentemente para terminarmos a demonstração basta mostrarmos que $b \cdot y' \geq \text{mmc}(a, b)$ e $a \cdot x' \geq \text{mmc}(a, b)$.

Pela simetria do problema basta mostrarmos que $b \cdot y' \geq \text{mmc}(a, b)$.

Sejam $a = p_1^{i_1} \cdot \dots \cdot p_k^{i_k}$ e $b = p_1^{j_1} \cdot \dots \cdot p_k^{j_k}$, onde $i_1, \dots, i_k, j_1, \dots, j_k \geq 0$, as respectivas expansões em fatores primos de a e b .

Sendo assim, $x' = \frac{p_1^{j_1} \cdot \dots \cdot p_k^{j_k}}{p_1^{i_1} \cdot \dots \cdot p_k^{i_k}} \cdot y'$.

Mas, x' e y' são inteiros estritamente positivos, logo

$$y' \geq \prod_{l=1}^k p_l^{r_l}, \text{ onde } r_l = \begin{cases} i_l - j_l & \text{caso } i_l \geq j_l \\ 0 & \text{c.c.} \end{cases}$$

Portanto,

$$b \cdot y' \geq p_1^{j_1} \cdot \dots \cdot p_k^{j_k} \cdot \prod_{l=1}^k p_l^{r_l} = \prod_{d=1}^k p_d^{e_d}, \text{ onde } e_d = \max\{i_d, j_d\}.$$

Por outro lado,

$$\prod_{d=1}^k p_d^{e_d} = \text{mmc}(a, b)$$

e portanto $b \cdot y' \geq \text{mmc}(a, b)$. ♦

Corolário 4.16. *As soluções especiais S_{ij} são mínimas.*

Demonstração: Conseqüência imediata do lema 4.15. ♦

Isto nos dá um segundo modo de calcularmos um limite superior para as soluções mínimas de equações da forma (4.1). Isto é o que mostra o lema 4.17.

Lema 4.17. *Seja $(x, y) \in \mathbb{N}^m \times \mathbb{N}^n$ uma solução para uma equação da forma (4.1) tal que $\exists i \leq m, \exists j \leq n$ tais que $x_i \geq d_{ij}$ e $y_j \geq e_{ij}$. Então $(x, y) \succeq S_{ij}$.*

[Hue78]

Demonstração: Imediata, conseqüência direta da definição de S_{ij} e da definição da relação de ordem $\succeq$. ■

O lema 4.17 nos fornece uma condição de suficiência para que as soluções não sejam mínimas. Note que pela definição de solução mínima temos que se $(x, y) \succeq S_{ij}$ então (x, y) não é mínima.

O algoritmo que descreveremos agora gera a base de soluções não negativas para qualquer equação da forma (4.1) através do método de “backtracking” usando como condição de parada os limites dados pelos lemas 4.14 e 4.17. As soluções candidatas a serem mínimas são geradas em ordem lexicográfica.

Mais precisamente, assuma que valores $x_1, \dots, x_k \in \mathbb{N}$ foram escolhidos, para algum $k \leq m$. Definimos então, para cada $j \leq n$, o seguinte:

$$E_j^k = \{e_{ij} | \exists i \leq k, x_i \geq d_{ij}\}.$$

Computamos o vetor $\max y$ dos valores máximos de y considerados como segue:

$$\max y_j^k(x_1, \dots, x_k) = \begin{cases} \min(E_j^k) & \text{se } E_j^k \neq \emptyset, \\ \max a & \text{caso contrário} \end{cases} \quad (j \leq n)$$

Teorema 4.18. *Seja (x, y) uma solução mínima para uma equação da forma (4.1) tal que $(x, y) \notin \{S_{ij}\}$. Então $\forall k \leq m$ temos:*

$$(i) \ x_i \leq \max b, \quad \forall i \leq k;$$

$$(ii) \ \sum_{i=1}^k a_i \cdot x_i \leq \sum_{j=1}^n b_j \cdot \max y_j^k(x_1, \dots, x_k);$$

e $\forall l \leq n$, temos (c) e (d) como seguem:

$$(iii) \ y_j \leq \max y_j^m(x_1, \dots, x_m), \quad \forall j \leq l;$$

$$(iv) \ \sum_{j=1}^l b_j \cdot y_j \leq \sum_{i=1}^m a_i \cdot x_i.$$

[Hue81]

Demonstração: O item (i) é consequência imediata do lema 4.14.

A demonstração do item (iv) também é trivial e segue do fato de (x, y) ser uma solução.

Vamos agora apresentar uma demonstração para o item (iii):

Caso 1: $E_j^m \neq \emptyset$.

Suponha, por absurdo, que para algum $l \leq n$, exista $j \leq l$ tal que $y_j > \max y_j^m(x_1, \dots, x_m)$.

Mas, por hipótese, $E_j^m \neq \emptyset$, neste caso existe $\min(E_j^m)$. Seja $i' \leq m$ tal que $e_{i'j} = \min(E_j^m)$.

Conseqüentemente, pela definição de E_j^m temos que $x_{i'} \geq d_{i'j}$.

Logo $\max y_j^m(x_1, \dots, x_m) \stackrel{\text{def.}}{=} \min(E_j^m) = e_{i'j}$.

Logo, $y_j > e_{i'j}$, então como $x_{i'} \geq d_{i'j}$ e $y_j > e_{i'j} \stackrel{\text{lema 4.17}}{\implies} (x, y) \succeq S_{ij}$.

Mas por hipótese, $(x, y) \notin \{S_{ij}\}$, isto é, $\nexists (x', y') \in \{S_{ij}\}$ tal que $(x, y) = (x', y')$.

Logo, pela definição de solução mínima temos que (x, y) não é uma solução mínima (pois

$(x, y) \succeq S_{ij}$ e $(x, y) \notin \{S_{ij}\}$. Contradição ! Pois por hipótese (x, y) é solução mínima.

Portanto, $\forall l \leq n$, vale:

$$y_j \leq \max y_j^m(x_1, \dots, x_m) \quad \forall j \leq l.$$

Caso 2: $E_j^m = \emptyset$.

Neste caso, $\max y_j^m(x_1, \dots, x_k) \stackrel{def.}{=} \max a$ e, pelo lema 4.14, $y_j \leq \max a$ pois (x, y) é uma solução mínima.

Portanto, $\forall l \leq n$ vale:

$$y_j \leq \max y_j^m(x_1, \dots, x_m) \quad \forall j \leq l.$$

Ainda resta-nos demonstrar o item (ii), o que faremos a seguir:

Caso 1: $E_j^k \neq \emptyset$.

$$\begin{aligned} \sum_{i=1}^k a_i \cdot x_i &\leq \sum_{i=1}^m a_i \cdot x_i = \sum_{j=1}^n b_j \cdot y_j \quad (\text{pois } (x, y) \text{ é uma solução}). \\ &\leq \sum_{j=1}^n b_j \cdot \max y_j^m(x_1, \dots, x_m) \quad (\text{pois vale (iii)}). \\ &\leq \sum_{j=1}^n b_j \cdot \max y_j^k(x_1, \dots, x_k) \quad (\text{pois } E_j^k \subseteq E_j^m). \end{aligned}$$

Caso 2: $E_j^k = \emptyset$.

$$\sum_{i=1}^k a_i \cdot x_i \leq \sum_{i=1}^m a_i \cdot x_i = \sum_{j=1}^n b_j \cdot y_j \quad \text{pois } (x, y) \text{ é uma solução.}$$

Mas, por hipótese, (x, y) é solução mínima. Logo, pelo lema 4.14 $y_j \leq \max a$ e portanto:

$$\sum_{j=1}^n b_j \cdot y_j \leq \sum_{j=1}^n b_j \cdot \max a = \sum_{j=1}^n b_j \cdot \max y_j^k(x_1, \dots, x_k) \text{ pois } E_j^k = \emptyset.$$

■

O algoritmo consiste em gerar todas as soluções potenciais $(x_m, \dots, x_1, y_n, \dots, y_1)$ em ordem lexicográfica, partindo da solução trivial $(0, \dots, 0)$ não a incluindo. Os valores dos x 's são progressivamente gerados (segundo a ordem lexicográfica) e seus valores sendo limitados

pelas condições (i) e (ii) do teorema 4.18. Para cada m -upla $(x_1, \dots, x_m)$ gerada que satisfaça as condições (i) e (ii) do teorema 4.18, gera-se uma n -upla $(y_1, \dots, y_n)$ satisfazendo as condições (iii) e (iv) do mesmo teorema. Assim, como as condições (i) e (ii) são condições necessárias para que os x 's escolhidos possam fazer parte de uma solução mínima, as condições (c) e (d) fazem o papel análogo, mas agora para os y 's escolhidos. Uma vez escolhido uma m -upla de x 's e uma n -upla de y 's que satisfaçam o teorema 4.18, isto significa que tais x 's e tais y 's são candidatos a serem soluções mínimas da equação da forma (4.1) considerada. O teorema apresentado não garante que a “solução” encontrada seja de fato mínima, uma vez que, tal teorema apresenta apenas uma condição necessária para a existência de qualquer solução mínima (não pertencente ao conjunto $\{S_{ij}\}$) de uma equação da forma (4.1), e ainda, as “soluções” (x', y') construídas deste modo e que satisfaçam as condições (i) a (iv) do teorema 4.18 não são necessariamente soluções para a equação da forma (4.1) considerada. Precisamos então verificar:

- i) se de fato os x 's e y 's encontrados formam uma solução para a equação da forma (4.1).
- ii) se já há alguma solução mínima encontrada que seja menor ou igual (segundo a ordem $\preceq$) a que estamos testando pois, caso haja, significa que a solução que estamos testando não é mínima, sendo portanto descartada.

O algoritmo prossegue, por backtraking, tentando encontrar soluções mínimas para a equação da forma (4.1). Quando não houver mais seqüências de x 's e y 's que satisfaçam o teorema dado, então o algoritmo inclui no conjunto de soluções mínimas encontrado as soluções mínimas especiais S_{ij} 's que eventualmente ainda não foram incluídas na base de soluções.

• Observação: quando uma “solução” para uma equação da forma (4.1) é encontrada e é tal que satisfaz i) e ii) acima então tal solução é de fato mínima, pois qualquer outra solução gerada mais adiante pelo algoritmo ou terá uma ordem lexicográfica maior que esta (e portanto não será menor ou igual segundo a ordem $\preceq$), ou será um dos S_{ij} 's e portanto não poderá ser menor ou igual a esta

(segundo a ordem $\preceq$) pela própria construção que define as soluções S_{ij} 's. Entretanto, as soluções S_{ij} 's são mínimas. Com isto concluímos que o algoritmo de Huet [Hue78] é correto.

Exemplo 4.19. Considere a equação diofantina homogênea linear $2 \cdot x_1 + x_2 = y_1 + y_2 + 2 \cdot y_3$. Vamos simular a execução do algoritmo de Huet para o cálculo da base de soluções não negativas da equação dada:

Para a equação $2 \cdot x_1 + x_2 = y_1 + y_2 + 2 \cdot y_3$ (*) temos: $m = 2$; $n = 3$; $a_1 = 2$; $a_2 = 1$; $b_1 = 1$; $b_2 = 1$; $b_3 = 2$; $\max a = \max b = 2$; e, conseqüentemente:

$$\begin{array}{cccccc} d_{11} = 1 & d_{12} = 1 & d_{13} = 1 & d_{21} = 1 & d_{22} = 1 & d_{23} = 2 \\ e_{11} = 2 & e_{12} = 2 & e_{13} = 1 & e_{21} = 1 & e_{22} = 1 & e_{23} = 1 \end{array}$$

◇ $k = 1$

$$x = \begin{array}{cc} 2 & 1 \\ \boxed{0} & \boxed{1} \\ & \uparrow \\ & k \end{array}$$

$$E_1^{k=1} = \{2\}; E_2^1 = \{2\}; E_3^1 = \{1\}$$

$$\max y_1^{k=1}(x_1) = 2; \max y_2^1(x_1) = 2; \max y_3^1(x_1) = 1.$$

$$E_1^{m=2} = \{2\}; E_2^m = \{2\}; E_3^m = \{1\}$$

$$\max y_1^{m=2}(x_1, x_2) = 2; \max y_2^m(x_1, x_2) = 2; \max y_3^m(x_1, x_2) = 1.$$

(a) $x_1 = 1 < 2 = \max b$ *ok!*

(b) $\sum_{i=1}^k a_i \cdot x_i = 2 < 6 = 1 \cdot 2 + 1 \cdot 2 + 2 \cdot 1 = \sum_{j=1}^n b_j \cdot \max y_j^{k=1}(x_1)$ *ok!*

◇ $l = 1$

$$y = \begin{array}{ccc} 3 & 2 & 1 \\ \boxed{0} & \boxed{0} & \boxed{1} \\ & \uparrow \\ & l \end{array}$$

(c) $y_1 = 1 < 2 = \max y_1^m(x_1, x_2); y_2 = y_3 = 0$ *ok!*

(d) $\sum_{j=1}^l b_j \cdot y_j = 1 < 2 = \sum_{i=1}^m a_i \cdot x_i$ *ok!*

mas (x, y) não é solução de $()$!*

$$y = \begin{array}{ccc} 3 & 2 & 1 \\ \boxed{0} & \boxed{0} & \boxed{2} \\ & \uparrow \\ & l \end{array}$$

(c) $y_1 = 2 = \max y_1^m(x_1, x_2)$ *ok!*

(d) $\sum_{j=1}^l b_j \cdot y_j = 2 = \sum_{i=1}^m a_i \cdot x_i$ *ok!*

(x, y) é solução da equação $(*)$, e ainda $(x, y) = S_{11}$.

Base:

x_2	x_1	y_3	y_2	y_1	
0	1	0	0	2	$\leftarrow S_{11}$

 $\diamond l = 2$

$$y = \begin{array}{c} 3 \quad 2 \quad 1 \\ \boxed{0 \mid 1 \mid 0} \\ \uparrow \\ l \end{array}$$

(c) $y_1 = 0$ e $y_2 = 1$ ok!(d) $\sum_{j=1}^l b_j \cdot y_j = 1$ ok!mas (x, y) não é solução de $(*)$!

$$y = \begin{array}{c} 3 \quad 2 \quad 1 \\ \boxed{0 \mid 1 \mid 1} \\ \uparrow \\ l \end{array}$$

(c) $y_1 = y_2 = 1$ ok!(d) $\sum_{j=1}^l b_j \cdot y_j = 2 = \sum_{i=1}^m a_i \cdot x_i$ ok!

(x, y) é solução de $(*)$ e (x, y) não é maior (segundo a ordem $\preceq$) que nenhuma outra solução mínima já encontrada. Portanto, (x, y) é mínima. (veja observação, pg. 69)

Base:

x_2	x_1	y_3	y_2	y_1	
0	1	0	0	2	$\leftarrow S_{11}$
0	1	0	1	1	

$$y = \begin{array}{c} 3 \quad 2 \quad 1 \\ \boxed{0 \mid 2 \mid 0} \\ \uparrow \\ l \end{array}$$

(c) $y_1 = 0$; $y_2 = 2 = \max y_2^m(x_1, x_2)$ ok!

$$(d) \sum_{j=1}^l b_j \cdot y_j = 2 = \sum_{i=1}^m a_i \cdot x_i \quad ok!$$

(x, y) é solução de $(*)$, e ainda $(x, y) = S_{12}$.

Base:

x_2	x_1	y_3	y_2	y_1	
0	1	0	0	2	$\leftarrow S_{11}$
0	1	0	1	1	
0	1	0	2	0	$\leftarrow S_{12}$

◇ $l = 3$

$$y = \begin{array}{ccc} & 3 & 2 & 1 \\ \boxed{1} & \boxed{0} & \boxed{0} \\ \uparrow & & \\ l & & \end{array}$$

$$(c) y_1 = y_2 = 0; y_3 = 1 \quad ok!$$

$$(d) \sum_{j=1}^l b_j \cdot y_j = 2 = \sum_{i=1}^m a_i \cdot x_i \quad ok!$$

(x, y) é solução de $(*)$, e ainda $(x, y) = S_{13}$.

Base:

x_2	x_1	y_3	y_2	y_1	
0	1	0	0	2	$\leftarrow S_{11}$
0	1	0	1	1	
0	1	0	2	0	$\leftarrow S_{12}$
0	1	1	0	0	$\leftarrow S_{13}$

$$x = \begin{array}{cc} & 2 & 1 \\ \boxed{0} & \boxed{2} \\ & \uparrow & \\ & k & \end{array}$$

Observe que não há alterações nos conjuntos $E_{1 \leq j \leq 3}^1$ e, conseqüentemente, não há alterações nos valores de $\max_{1 \leq j \leq 3}^{k=1}(x_1)$.

$$(a) x_1 = 2 = \max b \quad ok!$$

$$(b) \sum_{i=1}^k a_i \cdot x_i = 4 < 6 = 1 \cdot 2 + 1 \cdot 2 + 2 \cdot 1 = \sum_{j=1}^n b_j \cdot \max_{1 \leq j \leq 3}^{k=1}(x_1) \quad ok!$$

◇ $l = 2$

$$y = \begin{array}{ccc} 3 & 2 & 1 \\ \boxed{0} & \boxed{1} & \boxed{0} \\ & \uparrow & \\ & l & \end{array}$$

valem (c) e (d) mas (x, y) não é solução de $(*)$!

◇ $k = 2$

$$x = \begin{array}{cc} 2 & 1 \\ \boxed{1} & \boxed{0} \\ & \uparrow \\ & k \end{array}$$

$$E_1^{k=2=m} = \{1\}; E_2^2 = \{1\}; E_3^2 = \emptyset$$

$$\max y_1^{k=2=m}(x_1, x_2) = 1; \max y_2^2(x_1, x_2) = 1; \max y_3^1(x_1, x_2) = \max a = 2.$$

$$(a) x_1 = 0; x_2 = 1 < 2 = \max b \quad ok!$$

$$(b) \sum_{i=1}^k a_i \cdot x_i = 1 < 6 = 1 \cdot 1 + 1 \cdot 1 + 2 \cdot 2 = \sum_{j=1}^n b_j \cdot \max y_j^{k=2}(x_1, x_2) \quad ok!$$

◇ $l = 1$

$$y = \begin{array}{ccc} 3 & 2 & 1 \\ \boxed{0} & \boxed{0} & \boxed{1} \\ & & \uparrow \\ & & l \end{array}$$

valem (c) e (d), e ainda $(x, y) = S_{21}$ e portanto é mínima!

Base:

x_2	x_1	y_3	y_2	y_1	
0	1	0	0	2	$\leftarrow S_{11}$
0	1	0	1	1	
0	1	0	2	0	$\leftarrow S_{12}$
0	1	1	0	0	$\leftarrow S_{13}$
1	0	0	0	1	$\leftarrow S_{21}$

◇ $l = 2$

$$y = \begin{array}{ccc} 3 & 2 & 1 \\ \boxed{0} & \boxed{1} & \boxed{0} \\ \uparrow & & \\ l & & \end{array}$$

valem (c) e (d), e ainda $(x, y) = S_{22}$ e portanto é mínima!

Base:

x_2	x_1	y_3	y_2	y_1	
0	1	0	0	2	$\leftarrow S_{11}$
0	1	0	1	1	
0	1	0	2	0	$\leftarrow S_{12}$
0	1	1	0	0	$\leftarrow S_{13}$
1	0	0	0	1	$\leftarrow S_{21}$
1	0	0	1	0	$\leftarrow S_{22}$

◇ $l = 3$

$$y = \begin{array}{ccc} 3 & 2 & 1 \\ \boxed{1} & \boxed{0} & \boxed{0} \\ \uparrow & & \\ l & & \end{array}$$

(c) $y_1 = y_2 = 0$; $y_3 = 1$ ok!

(d) $\sum_{j=1}^l b_j \cdot y_j = 2 > 1 = \sum_{i=1}^m a_i \cdot x_i$ (e portanto não satisfaz o teorema 4.18)

$$x = \begin{array}{cc} 2 & 1 \\ \boxed{2} & \boxed{0} \\ \uparrow & \\ k & \end{array}$$

$$E_1^{k=2=m} = \{1\}; E_2^2 = \{1\}; E_3^2 = \{1\}$$

$$\max y_1^{k=2=m}(x_1, x_2) = 1; \max y_2^2(x_1, x_2) = 1; \max y_3^1(x_1, x_2) = 1.$$

(a) $x_1 = 0$; $x_2 = 2 = \max b$ ok!

(b) $\sum_{i=1}^k a_i \cdot x_i = 2 < 4 = 1 \cdot 1 + 1 \cdot 1 + 2 \cdot 1 = \sum_{j=1}^n b_j \cdot \max y_j^{k=2}(x_1, x_2)$ ok!

◇ $l = 3$

$$y = \begin{array}{ccc} & 3 & 2 & 1 \\ \boxed{1} & \boxed{0} & \boxed{0} \\ \uparrow & & \\ l & & \end{array}$$

$$(c) \ y_1 = y_2 = 0; \ y_3 = 1 \quad ok!$$

$$(d) \ \sum_{j=1}^l b_j \cdot y_j = 2 = \sum_{i=1}^m a_i \cdot x_i \quad ok!$$

(x, y) é solução de $(*)$; e ainda $(x, y) = S_{23}$, portanto é mínima!

Portanto a base das soluções inteiras não negativas (e não trivial) da equação $(*)$ é:

Base:

x_2	x_1	y_3	y_2	y_1	
0	1	0	0	2	$\leftarrow S_{11}$
0	1	0	1	1	
0	1	0	2	0	$\leftarrow S_{12}$
0	1	1	0	0	$\leftarrow S_{13}$
1	0	0	0	1	$\leftarrow S_{21}$
1	0	0	1	0	$\leftarrow S_{22}$
2	0	1	0	0	$\leftarrow S_{23}$

...

Na realidade, há outros algoritmos mais eficientes que o desenvolvido por Huet e que calculam a base finita das soluções de equações diofantinas homogêneas lineares [BS94].

4.3.2 Um Algoritmo de Unificação de Termos-AC

“No caso da unificação de termos-AC, sabemos que há um número finito de unificadoras e é possível obter um algoritmo de unificação que termine, seja correto, completo e mínimo” Stickel [Sti86].

Nesta subseção, descreveremos o algoritmo de unificação-AC, de Stickel [Sti81]. Até o final deste capítulo vamos admitir, sem perda de generalidades, que todo termo-AC possui como raiz um operador-AC. Os termos-AC serão representados em notação pré-fixa, considerando-os admitirem um número variável (≥ 2) de argumentos.

Stickel [Sti81] demonstra que, se dois termos-AC, sejam eles s e t , forem tais que $s < > = t < >$ e $s < > = s < >$ for um operador-AC, então seus argumentos em comum podem ser cancelados

aos pares antes de iniciar-se a tentativa de unificá-los. Por exemplo, o problema de unificar os termos $f(x, x, y, a, b, c)$ e $f(b, b, b, c, z)$ onde f é um operador-AC, pode ser trocado pelo problema equivalente de unificação dos termos $f(x, x, y, a)$ e $f(b, b, z)$ pois em ambos os casos os conjuntos completos das unificadoras-AC em relação a $W = \{x, y, z\}$ são os mesmos. O teorema 4.20 abaixo, demonstra que este fato é verdade para qualquer par de termos-AC.

Por simplicidade de notação, ao invés de separarmos os argumentos dos termos por vírgulas, eles serão separados por justaposição. Por exemplo, $f(t_1, t_2, \dots, t_n) \stackrel{\text{notac.}}{=} f(t_1 t_2 \dots t_n)$.

Teorema 4.20. *Seja f um operador-AC e sejam $s_1, \dots, s_m, t_1, \dots, t_n$ termos com $s_i = t_j$, para algum $1 \leq i \leq m$ e para algum $1 \leq j \leq n$. Seja θ uma unificadora-AC de $f(s_1 \dots s_m)$ e $f(t_1 \dots t_n)$ e seja σ uma unificadora-AC de $f(s_1 \dots s_{i-1} s_{i+1} \dots s_m)$ e $f(t_1 \dots t_{j-1} t_{j+1} \dots t_n)$. Então:*

- (i) θ é uma unificadora-AC de $f(s_1 \dots s_{i-1} s_{i+1} \dots s_m)$ e $f(t_1 \dots t_{j-1} t_{j+1} \dots t_n)$, e
- (ii) σ é uma unificadora-AC de $f(s_1 \dots s_m)$ e $f(t_1 \dots t_n)$.

[Sti81]

Demonstração de (i):

$$\begin{aligned} f(s_i \theta f(s_1 \dots s_{i-1} s_{i+1} \dots s_m) \theta) &\stackrel{\text{def.}}{=} f(s_i f(s_1 \dots s_{i-1} s_{i+1} \dots s_m) \theta) \stackrel{\text{notac.}}{=} f(s_i s_1 \dots s_m) \theta \stackrel{\text{AC}}{=} \\ f(s_1 \dots s_m) \theta &\stackrel{\text{hip.}}{=} f(t_1 \dots t_n) \theta \stackrel{\text{AC}}{=} f(t_j t_1 \dots t_{j-1} t_{j+1} \dots t_n) \theta \stackrel{\text{def.}}{=} f(t_j \theta f(t_1 \dots t_{j-1} t_{j+1} \dots t_n) \theta), \end{aligned}$$

mas por hipótese $s_i \theta = t_j \theta$. Pelo lema 3.6, a relação $\stackrel{\text{AC}}{=}$ é uma congruência, logo como consequência da definição 2.44 de congruência, temos:

$$f(s_1 \dots s_{i-1} s_{i+1} \dots s_m) \theta \stackrel{\text{AC}}{=} f(t_1 \dots t_{j-1} t_{j+1} \dots t_n) \theta,$$

e portanto θ é uma unificadora-AC de $f(s_1 \dots s_{i-1} s_{i+1} \dots s_m)$ e $f(t_1 \dots t_{j-1} t_{j+1} \dots t_n)$. □

Demonstração de (ii):

$$f(s_1 \dots s_{i-1} s_{i+1} \dots s_m) \sigma \stackrel{\text{hip.}}{=} f(t_1 \dots t_{j-1} t_{j+1} \dots t_n) \sigma,$$

e $s_i \sigma = t_j \sigma$.

Por hipótese, $s_i \sigma = t_j \sigma$ e $f(s_1 \dots s_{i-1} s_{i+1} \dots s_m) \sigma \stackrel{\text{AC}}{=} f(t_1 \dots t_{j-1} t_{j+1} \dots t_n) \sigma$. Logo, por definição de congruência, $f(s_i \sigma f(s_1 \dots s_{i-1} s_{i+1} \dots s_m) \sigma) \stackrel{\text{AC}}{=} f(t_j \sigma f(t_1 \dots t_{j-1} t_{j+1} \dots t_n) \sigma)$. Mas,

$$\begin{aligned}
f(s_i \sigma f(s_1 \dots s_{i-1} s_{i+1} \dots s_m) \sigma) &\stackrel{def.}{=} f(s_i f(s_1 \dots s_{i-1} s_{i+1} \dots s_m)) \sigma \stackrel{notas.}{\stackrel{AC}{=}} f(s_i s_1 \dots s_{i-1} s_{i+1} \dots s_m) \sigma \stackrel{AC}{=} \\
&f(s_1 \dots s_m) \sigma \\
\text{e} \\
f(t_j \sigma f(t_1 \dots t_{j-1} t_{j+1} \dots t_n) \sigma) &\stackrel{def.}{=} f(t_j f(t_1 \dots t_{j-1} t_{j+1} \dots t_n)) \sigma \stackrel{notas.}{\stackrel{AC}{=}} f(t_j t_1 \dots t_{j-1} t_{j+1} \dots t_n) \sigma \stackrel{AC}{=} \\
&f(t_1 \dots t_n) \sigma
\end{aligned}$$

E, portanto, como toda congruência é uma relação de equivalência, temos $f(s_1 \dots s_m) \sigma \stackrel{AC}{=} f(t_1 \dots t_n) \sigma$.

Portanto σ é uma unificadora de $f(s_1 \dots s_m)$ e $f(t_1 \dots t_n)$. ■

Boa parte do algoritmo de Stickel consiste num algoritmo de unificação (que termina, é completo, correto e mínimo) de termos-*AC* que possuem apenas variáveis como argumentos. Primeiro vamos apresentar o algoritmo de Stickel para o caso restrito dos termos-*AC* que possuam como argumentos apenas variáveis, depois o algoritmo será exposto para o caso geral e apresentamos a demonstração dada por Stickel [Sti81] para a correção, completeza e terminação do algoritmo para o caso restrito para termos-*AC* que possuem apenas variáveis como argumentos.

Caso Restrito: Apenas Variáveis como Argumentos

Conceitualmente, o resultado de unificar termos-*AC* que possuam apenas variáveis como argumentos consiste na atribuição de alguma seqüência de termos a cada variável dos termos-*AC* considerados.

Seja f o símbolo funcional-*AC* dos dois termos a serem unificados e que possuem apenas variáveis como argumentos. A cada variável dos termos-*AC* é atribuído um termo t_i (cujo símbolo funcional não é f) ou é atribuído um termo-*AC* $f(t_1^{n^1} \dots t_m^{n^m})$ (com n^i ocorrências dos termo t_i como argumentos de f). Para que tal substituição seja uma unificadora é necessário apenas que o número de ocorrências de cada termo t_i usado em qualquer uma das atribuições seja o mesmo em cada um dos dois termos-*AC* (que se quer unificar) instanciados pelas atribuições. Por exemplo, $\{x_1 \leftarrow f(bb), x_2 \leftarrow f(ab), x_3 \leftarrow a, y_1 \leftarrow b, y_2 \leftarrow f(aabbb)\}$ é uma unificadora de $f(x_1 x_1 x_2 x_3)$ e $f(y_1 y_1 y_2)$ uma vez que há 2 a 's e 5 b 's nas instanciações de ambos, $f(x_1 x_1 x_2 x_3)$ e $f(y_1 y_1 y_2)$, e o termo-*AC* unificado por esta substituição é $f(aabbbbb)$.

A cada par de termos-*AC*, que possuem o mesmo símbolo funcional na raiz e que possuem apenas variáveis como argumentos, podemos associar uma simples equação para representar o número de multiplicidade de variáveis em cada um dos termos-*AC*. Por exemplo, a equação $2 \cdot x_1 + x_2 + x_3 = 2 \cdot y_1 + y_2$ está associada ao par de termos-*AC* $f(x_1 x_1 x_2 x_3)$ e $f(y_1 y_1 y_2)$.

Soluções inteiras não negativas de tais equações podem ser usadas para representarem as unificadoras de um par de termos-*AC* considerados. As soluções devem ser compostas por números inteiros não negativos uma vez que a cada variável deve ser atribuído um número não negativo de ocorrências de um dado termo (e evidentemente uma quantidade inteira de termos).

Com o intuito de gerar todas as soluções referentes ao problema de unificação de dois termos-*AC* é necessário que sejamos capazes de representar todas as soluções da equação obtida a partir dos termos-*AC* que se quer unificar. Como vimos na subseção 4.3.1, qualquer equação diofantina homogênea linear, como aquela vista acima ($2 \cdot x_1 + x_2 + x_3 = 2 \cdot y_1 + y_2$), possui uma base finita de soluções positivas, e através de combinações lineares aditivas da base pode-se obter qualquer solução não negativa da equação dada.

Considere a equação diofantina homogênea linear $2 \cdot x_1 + x_2 + x_3 = 2 \cdot y_1 + y_2$ dada no exemplo 4.19. A base de soluções não negativas desta equação está apresentada na tabela 4.1:

Número da solução	x_1	x_2	x_3	y_1	y_2	$2 \cdot x_1 + x_2 + x_3$	$2 \cdot y_1 + y_2$	
1	0	0	1	0	1	1	1	z_1
2	0	0	2	1	0	2	2	z_2
3	0	1	0	0	1	1	1	z_3
4	0	1	1	1	0	2	2	z_4
5	0	2	0	1	0	2	2	z_5
6	1	0	0	0	2	2	2	z_6
7	1	0	0	1	0	2	2	z_7

Tabela 4.1: Base das soluções não negativas da equação diofantina $2 \cdot x_1 + x_2 + x_3 = 2 \cdot y_1 + y_2$.

Note que cada linha da tabela acima representa uma solução parcial do problema de unificação dos termos-*AC* $f(x_1x_2x_3)$ e $f(y_1y_2)$. Por exemplo, a solução parcial de número 5 da tabela acima nos diz que devemos substituir x_2 por duas ocorrências da variável z_5 enquanto que a variável y_1 deve ser substituída por uma ocorrência de z_5 , isto é, devemos fazer $x_2 \leftarrow f(z_5, z_5)$ e $y_1 \leftarrow z_5$, deste modo o número de ocorrências de z_5 em cada um dos termos-*AC* instanciados, $f(x_1x_2z_5z_5x_3)$ e $f(z_5z_5y_2)$, será o mesmo.

Entretanto, observe que tomando apenas a solução de número 5 da tabela 4.1 os dois termos-*AC* não se unificam. Para que possamos unificá-los é necessário compormos as soluções dadas pelo conjunto base de soluções (tabela 4.1). Como já citamos na subseção 4.3.1, toda solução não negativa de uma equação diofantina homogênea linear é uma combinação linear aditiva de sua base, em particular, para a equação dada ($2 \cdot x_1 + x_2 + x_3 = 2 \cdot y_1 + y_2$) temos

que toda solução S é da forma:

$$S = (z_1 \cdot (0, 0, 1) + z_2 \cdot (0, 0, 2) + z_3 \cdot (0, 1, 0) + z_4 \cdot (0, 1, 1) + z_5 \cdot (0, 2, 0) + z_6 \cdot (1, 0, 0) + z_7 \cdot (1, 0, 0), z_1 \cdot (0, 1) + z_2 \cdot (1, 0) + z_3 \cdot (0, 1) + z_4 \cdot (1, 0) + z_5 \cdot (1, 0) + z_6 \cdot (0, 2) + z_7 \cdot (1, 0)) = ((z_6 + z_7, z_3 + z_4 + 2 \cdot z_5, z_1 + 2 \cdot z_2 + z_4), (z_2 + z_4 + z_5 + z_7, z_1 + z_3 + 2 \cdot z_6))$$

ou simplesmente,

$x_1 = z_6 + z_7, x_2 = z_2 + z_4 + 2 \cdot z_5, x_3 = z_1 + 2 \cdot z_3 + z_4, y_1 = z_3 + z_4 + z_5 + z_7, y_2 = z_1 + z_2 + 2 \cdot z_6$ para qualquer z_i ($1 \leq i \leq 7$) ≥ 0 . Evidentemente, se $z_i = 0$ para todo $1 \leq i \leq 7$, teremos a solução trivial da equação.

Mas, como já observamos no parágrafo acima, nem toda solução não negativa de uma equação diofantina homogênea linear é uma solução do problema de unificação de termos-AC a partir dos quais a equação foi deduzida. Há então um subconjunto do conjunto das soluções não negativas destas equações que não é útil para a solução do problema de unificação-AC, desconsideramos este subconjunto impondo a seguinte restrição:

Restrição 1: A cada variável dos termos-AC originais a serem unificados deve-se atribuir pelo menos um termo em seu valor, ou seja, não pode haver variável que não possua atribuição de algum termo em seu valor.

Para a tabela 4.1, há um total de $2^7 = 128$ formas diferentes de compor as 7 soluções parciais apresentadas, de modo que cada solução da tabela 4.1 apareça no máximo uma única vez em cada composição. Dentre estas 128 soluções há 69 delas onde todas as variáveis $(x_1, x_2, x_3, y_1, y_2)$ possuem pelo menos uma atribuição de algum termo em seu valor (59 soluções são eliminadas pela Restrição 1).

A seguir apresentamos de um modo mais conciso o algoritmo de unificação de termos-AC para o caso restrito que foi discutido nesta subsubseção:

0. Fornece-se os dois termos-AC a serem unificados, digamos s e t .
1. Elimina-se os argumentos comuns a s e t .
2. Forma-se uma equação diofantina homogênea linear a partir dos dois termos-AC que se quer unificar. Os coeficientes de cada variável desta equação é igual a quantidade de vezes que a variável correspondente aparece no termo-AC.
3. Obtêm-se a base das soluções não negativas da equação.
4. Associa-se a cada solução da base uma nova variável não existente nos termos-AC a serem unificados.
5. Considere o conjunto onde cada elemento é uma solução da equação, solução esta obtida através de uma combinação linear aditiva de soluções da base (sendo que cada solução da base não deve aparecer mais que uma vez em cada combinação).
6. Dentre todas as soluções obtidas no passo 5 considere apenas aquelas que possuam todas as variáveis atribuídas a algum termo (Restrição 1).

7. Para cada solução da equação obtida no passo 6 constrói-se uma respectiva unificadora-AC, que será uma substituição que irá atribuir as novas variáveis (criadas em cada linha de solução parcial apresentada pela base das soluções não negativas da equação) à cada uma das variáveis existentes nos termos-AC a serem unificados. A quantidade de vezes que uma nova variável será atribuída a uma dada variável original, digamos v , dos termos-AC a serem unificados será justamente o coeficiente da variável nova, apresentado na componente da solução S da equação, relativa à variável v dos termos-AC originais.

Exemplo 4.21. *Sejam s, t termos, tais que $s = f(x_1x_1x_2x_3)$ e $t = f(y_1y_1y_2)$ onde f é um operador-AC e x_1, x_2, x_3, y_1, y_2 variáveis. Seja $W = \text{Var}(s) \cup \text{Var}(t)$.*

Será que s e t são unificáveis, isto é, será que existe substituição σ tal que $s\sigma \stackrel{AC}{=} t\sigma$?

Como já vimos, tal resposta reside na busca de soluções não negativas de uma equação diofantina homogênea linear que represente o par (s, t) de termos. Esta equação é muito fácil de ser determinada, e neste exemplo ela é $(2 \cdot x_1 + x_2 + x_3 = 2 \cdot y_1 + y_2)$.

A base das soluções não negativas desta equação está expressa na tabela 4.1.

Qualquer composição (combinação linear aditiva) desta base que pertença ao subconjunto das 69 composições citadas no penúltimo parágrafo da página anterior expressa uma substituição que pertence ao conjunto completo de unificadoras-AC dos termos s e t em relação a W .

Tomemos como exemplo 3 dessas 69 composições, são elas: $\{2, 3, 6\}$, $\{1, 2, 3, 6\}$ e $\{4, 6\}$, onde cada elemento destes conjuntos diz respeito ao número da solução (veja a tabela 4.1) que faz parte da composição considerada. Abaixo temos as três unificadoras-AC em relação a W , a saber, σ_1, σ_2 e σ_3 associadas, respectivamente, às três composições apresentadas acima:

$$\begin{aligned}\sigma_1 &= \{x_1 \leftarrow z_6, x_2 \leftarrow z_3, x_3 \leftarrow f(z_2z_2), y_1 \leftarrow z_2, y_2 \leftarrow f(z_3z_6z_6)\}, \\ \sigma_2 &= \{x_1 \leftarrow z_6, x_2 \leftarrow z_3, x_3 \leftarrow f(z_1z_2z_2), y_1 \leftarrow z_2, y_2 \leftarrow f(z_1z_3z_6z_6)\}, \\ \sigma_3 &= \{x_1 \leftarrow z_6, x_2 \leftarrow z_4, x_3 \leftarrow z_4, y_1 \leftarrow z_4, y_2 \leftarrow f(z_6z_6)\}.\end{aligned}$$

Para a unificadora σ_3 pertencente do conjunto completo de unificadoras-AC em relação a W de s e t temos que $s\sigma_3 = f(z_6z_6z_4z_4)$ e $t\sigma_3 = f(z_4z_4f(z_6z_6)) \stackrel{\text{notas}}{\stackrel{AC}{=}} f(z_4z_4z_6z_6)$, sendo $f(z_6z_6z_4z_4) \stackrel{AC}{=} f(z_4z_4z_6z_6)$.

Analogamente, pode-se verificar a validade de σ_1 e σ_2 quanto unificadoras pertencentes ao conjunto completo de unificadoras-AC dos termos s e t em relação a W .

[Sti81]

O Caso Geral

O caso geral consiste de termos-AC com um operador-AC na raiz e com argumentos quaisquer, isto é, argumentos que podem conter operadores ordinários (não associativos, não comutativos) ou operadores-AC. A possibilidade da presença de operadores ordinários é necessária

para que possamos tratar dos aspectos não associativos-comutativos da unificação.

Quando tenta-se unificar dois termos-AC, dois novos termos, chamados de *abstrações das variáveis* ou simplesmente *abstrações*, são criados a partir dos termos originais que se quer unificar. Estes novos termos-AC possuem apenas variáveis como argumentos e são criados simplesmente trocando cada argumento dos termos-AC originais por variáveis novas (não utilizadas pelos termos-AC originais). Estes dois novos termos-AC são, então, generalizações dos dois termos-AC originais.

Exemplo 4.22. *Sejam $s = f(xxya)$ e $t = f(bbz)$ dois termos-AC a serem unificados, onde x, y, z são variáveis e a, b são constantes. Então, $s' = f(x_1x_1x_2x_3)$ e $t' = f(y_1y_1y_2)$ são, respectivamente, abstrações de s e t , sendo portanto, generalizações de s e t , pois basta tomarmos a substituição $\sigma = \{x_1 \leftarrow x, x_2 \leftarrow y, x_3 \leftarrow a, y_1 \leftarrow b, y_2 \leftarrow z\}$ e verificaremos que, de fato, $t'\sigma = t$ e $s'\sigma = s$.*

[Sti81]

Considere que queiramos unificar os dois termos-AC, s e t , apresentados no exemplo 4.22 acima. Usando o algoritmo de unificação para o caso restrito descrito na subseção 4.3.2 anterior, unifica-se (se possível) os dois novos termos-AC, s' e t' , generalizações de s e t . Note que a unificação de s' e t' já foi apresentada na subseção anterior (veja o exemplo 4.21) e vimos que, neste caso, há um total de 69 unificadoras que compõem o conjunto completo de unificadoras-AC de s' e t' em relação a $W' = \{x_1, x_2, x_3, y_1, y_2\}$.

Para que possamos obter o conjunto completo das unificadoras-AC dos termos s e t em relação a $W = \{x, y, z\}$ é necessário unificar, para cada variável que foi substituída nos termos-AC originais, o valor na substituição com o valor nas unificadoras das generalizações.

Exemplo 4.23. *Queremos unificar os termos $s = f(xxya)$ e $t = f(bbz)$. Seja $W = \{x, y, z\}$. Para tal obtemos generalizações $s' = f(x_1x_1x_2x_3)$ e $t' = f(y_1y_1y_2)$ de s e t , respectivamente. Seja $W' = \{x_1, x_2, x_3, y_1, y_2\} \cup W$.*

Temos $\sigma = \{x_1 \leftarrow x, x_2 \leftarrow y, x_3 \leftarrow a, y_1 \leftarrow b, y_2 \leftarrow z\}$ que é a substituição que instancia as generalizações s' e t' aos termos originais s e t , pois $s'\sigma = s$ e $t'\sigma = t$. Como já vimos no exemplo 4.21, s' e t' possuem 69 unificadoras candidatas a comporem o conjunto completo das unificadoras-AC de s e t em relação a W .

Para obter-se o conjunto completo das unificadoras-AC de s e t em relação a W precisamos, para cada unificadora-AC β de s' e t' em relação a W' , unificar os elementos de β com os respectivos elementos da substituição σ .

Tomemos, como exemplo a seguinte unificadora $\beta = \{x_1 \leftarrow z_6, x_2 \leftarrow z_3, x_3 \leftarrow f(z_2z_2), y_1 \leftarrow z_2, y_2 \leftarrow f(z_3z_6z_6)\}$ de s' e t' que é obtida pela seguinte composição da base $\{6, 3, 2\}$. Vamos agora tentar “unificá-la” com σ .

Para x_1 :

Em σ temos $x_1 \leftarrow x$, enquanto que em β temos $x_1 \leftarrow z_6$. “Unificando” estas duas atribuições obtemos $x \leftarrow z_6$.

Para x_2 :

Em σ temos $x_2 \leftarrow y$, em β temos $x_2 \leftarrow z_3$, “unificando-as” obtemos $y \leftarrow z_3$.

Para x_3 :

Em σ temos $x_3 \leftarrow a$, em β temos $x_3 \leftarrow f(z_2 z_2)$. Aqui obtemos o seguinte problema, a é uma constante, sendo assim torna-se impossível sua substituição no termo-AC original s por $f(z_2 z_2)$ (pois a é uma constante!). Analogamente, a mesma observação pode ser feita para y_1 . Isto torna a tentativa de unificação de σ e β impossível, entretanto como já vimos, ainda há mais 68 unificadoras-AC em CSU de s' e t' em relação a W' que precisam ser testadas, com o intuito de gerarmos unificadoras-AC em CSU para os termos originais s e t em relação a W . Entretanto, o fato de σ possuir as atribuições $x_3 \leftarrow a$ e $y_1 \leftarrow b$, faz com que 63 dentre as 69 unificadoras-AC de s' e t' tornam-se impossíveis de serem unificadas com σ .

Restrição 2: De fato, toda e qualquer unificadora de s' e t' que atribuírem a x_3 um valor que não seja uma variável, isto é, qualquer valor do tipo $g(\dots)$ (eventualmente $g = f$) pode ser excluída imediatamente da tentativa de unificá-la com σ , pois seria impossível “unificar” algo do tipo $x_3 \leftarrow g(\dots)$ com $x_3 \leftarrow a$.

Precisamos então, tentar unificar σ com apenas 6 unificadoras-AC restantes. Esta simples observação faz com que o algoritmo de unificação economize muito tempo, pois agora há apenas 6 casos para verificar-se a existência de unificadoras-AC em CSU para os termos originais s e t em relação a W , obtidas a partir das seguintes composições (combinações lineares aditivas) dos elementos da base das soluções não negativas da equação $2 \cdot x_1 + x_2 + x_3 = 2 \cdot y_1 + y_2$ (veja tabela 4.1):

- (1) $\{4, 6\}$
- (2) $\{3, 4, 6\}$
- (3) $\{1, 5, 6\}$
- (4) $\{1, 3, 5, 6\}$
- (5) $\{1, 3, 7\}$
- (6) $\{1, 3, 6, 7\}$

Abaixo apresentamos as 6 unificadoras-AC em CSU de s' e t' em relação a W' associadas às 6 composições apresentadas acima:

- (1) $\{x_1 \leftarrow z_6, \quad x_2 \leftarrow z_4, \quad x_3 \leftarrow z_4, \quad y_1 \leftarrow z_4, \quad y_2 \leftarrow f(z_6 z_6)\}.$
- (2) $\{x_1 \leftarrow z_6, \quad x_2 \leftarrow f(z_3 z_4), \quad x_3 \leftarrow z_4, \quad y_1 \leftarrow z_4, \quad y_2 \leftarrow f(z_3 z_6 z_6)\}.$
- (3) $\{x_1 \leftarrow z_6, \quad x_2 \leftarrow f(z_5 z_5), \quad x_3 \leftarrow z_1, \quad y_1 \leftarrow z_5, \quad y_2 \leftarrow f(z_1 z_6 z_6)\}.$
- (4) $\{x_1 \leftarrow z_6, \quad x_2 \leftarrow f(z_3 z_5 z_5), \quad x_3 \leftarrow z_1, \quad y_1 \leftarrow z_5, \quad y_2 \leftarrow f(z_1 z_3 z_6 z_6)\}.$
- (5) $\{x_1 \leftarrow z_7, \quad x_2 \leftarrow z_3, \quad x_3 \leftarrow z_1, \quad y_1 \leftarrow z_7, \quad y_2 \leftarrow f(z_1, z_3)\}.$
- (6) $\{x_1 \leftarrow f(z_6 z_7), \quad x_2 \leftarrow z_3, \quad x_3 \leftarrow z_1, \quad y_1 \leftarrow z_7, \quad y_2 \leftarrow f(z_1 z_3 z_6 z_6)\}.$

Agora, unificando cada uma das 6 unificadoras-AC em CSU de s' e t' em relação a W' , expostas acima, com

$\sigma = \{x_1 \leftarrow x, x_2 \leftarrow y, x_3 \leftarrow a, y_1 \leftarrow b, y_2 \leftarrow z\}$ obtemos o que segue:

- (1*) não resulta em uma unificadora-AC pois $z_4 \leftarrow a$ e $z_4 \leftarrow b$ não são “unificáveis”.
- (2*) não resulta em uma unificadora-AC pois, novamente, $z_4 \leftarrow a$ e $z_4 \leftarrow b$ não são “unificáveis”.
- (3*) $\{x \leftarrow z_6, \quad y \leftarrow f(bb), \quad z \leftarrow f(az_6 z_6)\} \quad (= \{y \leftarrow f(bb), z \leftarrow f(axx)\}).$
- (4*) $\{x \leftarrow z_6, \quad y \leftarrow f(bbz_3), \quad z \leftarrow f(az_3 z_6 z_6)\} \quad (= \{y \leftarrow f(bbz_3), z \leftarrow f(az_3 xx)\}).$
- (5*) $\{x \leftarrow b, \quad y \leftarrow z_3, \quad z \leftarrow f(az_3)\} \quad (= \{x \leftarrow b, z \leftarrow f(ay)\}).$
- (6*) $\{x \leftarrow f(bz_6), \quad y \leftarrow z_3, \quad z \leftarrow f(az_3 z_6 z_6)\} \quad (= \{x \leftarrow f(bz_6), z \leftarrow f(ayz_6 z_6)\}).$

As quatro unificadoras-AC acima (3*), (4*), (5*) e (6*) constituem o conjunto completo de unificadoras-AC em relação a W para $s = f(xxya)$ e $t = f(bbz)$.

Restrição 3: Note que, como x_3 e y_1 da unificadora-AC σ do caso restrito correspondem a a e b , respectivamente, e a e b não são unificáveis, pois são constantes distintas, então qualquer composição incluindo a solução de número 4 (veja tabela 4.1) da base das soluções não negativas da equação $2 \cdot x_1 + x_2 + x_3 = 2 \cdot y_1 + y_2$ pode ser excluída de consideração, pois uma tal composição (como em (1) e (2) deste exemplo) iria requerer a unificação de a com b , o que é impossível.

[Sti81]

A restrição 1, como vimos, é verificada durante o processo de geração das unificadoras-AC para o caso restrito (**passo 1**). Já, as restrições 2 e 3 podem ser realizadas após o processo de produção das unificadoras-AC do caso restrito, ou seja, serem realizadas durante o processo de produção das unificadoras-AC do caso geral (**passo 2**). Entretanto é muito mais eficiente se estas duas últimas restrições forem incorporadas ao passo 1. No exemplo dado, se incorporarmos as restrições 2 e 3 ao passo 1, desconsideraríamos imediatamente as soluções de número 2 e 4 da tabela 4.1, respectivamente. Tais restrições, reduziriam o conjunto inicial de 128 composições

possíveis de virem a tornarem-se unificadoras no caso geral, para apenas 4 (e não mais para 69), já no passo 1. As 4 combinações produzidas seriam justamente, as unificadoras-AC (3), (4), (5) e (6) do caso restrito apresentadas acima, cada uma delas tendo uma unificadora-AC correspondente para o caso geral.

A seguir, descrevemos, de um modo mais conciso, o algoritmo de unificação-AC proposto por Stickel [Sti81].

0. Fornecer os dois termos-AC a serem unificados.
1. Construir as generalizações (abstrações) dos dois termos-AC que se quer unificar. Faz-se isto trocando cada argumento por novas variáveis. Obtem-se conseqüentemente uma substituição, digamos σ , que instancia as generalizações aos termos-AC originais.
2. Usar o algoritmo do caso restrito para gerar as unificadoras em CSU das generalizações dos dois termos-AC originais, construídas em 1. Pode-se, e é recomendado, por ganho em eficiência, incorporar no algoritmo do caso restrito as restrições 2 e 3.
3. Unificar a substituição σ com cada uma das unificadoras-AC obtidas no item 2, obtendo assim um conjunto completo de unificadoras-AC para os dois termos originais em relação a W.

Teorema 4.24. *O algoritmo de unificação para termos-AC cujos subtermos são apenas variáveis, termina é correto (não retorna substituições que não sejam unificadoras) e é completo (toda unificadora é uma instância-AC de alguma unificadora retornada pelo algoritmo).*

[Sti81]

Demonstração:

O algoritmo termina: É fácil ver que o algoritmo, como descrevemos acima, termina, uma vez que ele realiza um número finito de operações sobre um conjunto finito de soluções inteiras positivas (base) geradas a partir da equação correspondente aos dois termos-AC que se quer unificar.

O algoritmo é correto: Basta observar que cada solução da equação (que satisfaça a Restrição 1), derivada dos dois termos-AC que se quer unificar, causa a introdução em cada um dos dois termos-AC instanciados de um mesmo número de ocorrências para cada nova variável e portanto tornando-os unificados.

O algoritmo é completo: Agora precisamos mostrar que toda unificadora dos dois termos-AC é uma instância-AC de alguma unificadora devolvida pelo algoritmo. Observe que, se σ e σ' são unificadoras pertencentes ao conjunto completo de unificadoras-AC retornado pelo algoritmo de Stickel, então $Lvar(\sigma) = Lvar(\sigma')$ e portanto, neste caso, dizer que σ é mais geral que σ' em E sobre V , para qualquer $V \subseteq \mathcal{V}$, é equivalente a dizer que σ' é uma instância- E de σ . Ou seja, neste caso, $\sigma \leq_V^E \sigma'$ se e somente se σ' for uma instância- E de σ , para qualquer $V \subseteq \mathcal{V}$. Seja f o operador-AC dos termos-AC que se quer unificar. Qualquer unificadora deve atribuir a cada variável um termo da forma t_i (tal que $t_i|_0 \neq f$) ou um termo $f(t_1^{n_1} \cdots t_m^{n_m})$ (com n_i ocorrências

do termo t_i como argumentos de f), chamemos tal unificadora hipotética de unificadora α . Seja k a cardinalidade do conjunto formado por tais termos t_i em quaisquer soluções do problema de unificação de um par de termos que possuam apenas variáveis como argumentos. Os dois termos-AC instanciados devem ter o mesmo número de ocorrências de cada um destes k termos como argumentos de f . Isto é, $a_1 \cdot c_{i1} + \dots + a_m \cdot c_{im} = b_1 \cdot d_{i1} + \dots + b_n \cdot d_{in}$ ($1 \leq i \leq k$), onde m é o número de variáveis distintas que ocorrem no primeiro termo-AC, n é o número de variáveis distintas que ocorrem no segundo termo-AC, a_j é o número de ocorrências da j -ésima variável no primeiro termo-AC, b_j é o número de ocorrências da j -ésima variável no segundo termo-AC, c_{ij} é o número de ocorrências do termo i na variável j no primeiro termo-AC, e d_{ij} é o número de ocorrências do termo i na variável j no segundo termo-AC.

Cada $n+m$ -upla $(c_{i1}, \dots, c_{im}, d_{i1}, \dots, d_{in})$, para cada $1 \leq i \leq k$, é uma solução (não necessariamente uma unificadora) para a equação $a_1 \cdot x_1 + \dots + a_m \cdot x_m = b_1 \cdot y_1 + \dots + b_n \cdot y_n$ (*) que corresponde ao par de termos-AC que se quer unificar. Mas esta é uma equação diofantina homogênea linear e sabemos que para cada equação deste tipo existe uma base finita que gera todas as soluções não negativas da equação através de combinações lineares aditivas dos elementos da base [CP67]. Logo, tal $m+n$ -upla também pode ser expressa através de uma combinação linear aditiva da base das soluções não negativas da equação acima.

Considere a unificadora (chamemo-la σ) correspondente à soma de todas aquelas soluções pertencentes à base da equação que são necessárias à formação de qualquer uma das $n+m$ -uplas $(c_{i1}, \dots, c_{im}, d_{i1}, \dots, d_{in})$. A unificadora hipotética (unificadora α) considerada é uma instância desta unificadora σ retornada pelo algoritmo.

Seja S_j a j -ésima solução da base da equação (*), com cada S_j sendo uma $m+n$ -upla da base. Seja r o número de soluções que pertencem à base. Para cada solução $(c_{i1}, \dots, c_{im}, d_{i1}, \dots, d_{in})$ da equação (*) seja $n_{q_1} \cdot S_{q_1} + n_{q_2} \cdot S_{q_2} + \dots + n_{q_i} \cdot S_{q_i} + \dots + n_{q_l} \cdot S_{q_l}$ a combinação linear aditiva da base que a gera, onde $n_1, \dots, n_s$ são inteiros estritamente positivos, $s \leq r$ e ainda $q_1 \leq i \leq s \in [1..r]$.

Inclua no valor da nova variável associada à cada uma destas soluções da base uma quantidade de ocorrências do termo i igual ao coeficiente da solução na soma aditiva ponderada. Isto é, se $z_1, \dots, z_r$ são as novas variáveis, então faça $z_{q_1} \leftarrow f(\underbrace{t_i \dots t_i}_{n_{q_1} \text{ vezes}}), z_{q_2} \leftarrow f(\underbrace{t_i \dots t_i}_{n_{q_2} \text{ vezes}}), \dots, z_{q_l} \leftarrow$

$$f(\underbrace{t_i \dots t_i}_{n_{q_l} \text{ vezes}}).$$

Isto irá resultar justamente na atribuição de c_{ij} ocorrências do termo i à cada variável j do primeiro termo-AC e d_{ij} ocorrências do termo i à cada variável j do segundo termo-AC.

Faça isto para cada um dos k termos da solução considerada para a equação (*).

Esta atribuição dos termos t_i 's da unificadora hipotética considerada, para as novas variáveis associadas às soluções da equação geradas no processo de unificação resulta no número

correto c_{ij} ou d_{ij} de cada termo a ser atribuído para cada variável dos dois termos-AC originais.

Logo, qualquer solução para a unificação de dois termos-AC que possuam apenas variáveis como argumentos é uma instância de uma unificadora retornada, ou seja, o algoritmo é completo. ■

Para exemplificar a linha de raciocínio da demonstração apresentada acima, quanto à completeza do algoritmo, considere a equação $2 \cdot x_1 + x_2 + x_3 = 2 \cdot y_1 + y_2$ ^(*) referente aos termos-AC $s = f(x_1, x_1, x_2, x_3)$ e

$t = f(y_1, y_1, y_2)$ que queremos unificar. A tabela 4.1 apresenta a base das soluções não negativas da equação (*).

Considere a unificadora $\alpha = \{x_1 \leftarrow f(t_1, t_1, t_2, t_2, t_2), x_2 \leftarrow t_3, x_3 \leftarrow t_4, y_1 \leftarrow f(t_1, t_1, t_2), y_2 \leftarrow f(t_2, t_2, t_2, t_2, t_3, t_4)\}$. De fato, α é uma unificadora-AC do par (s, t) de termos pois $f(x_1, x_1, x_2, x_3)\alpha = f(t_1, t_1, t_2, t_2, t_2, t_1, t_1, t_2, t_2, t_2, t_3, t_4) \stackrel{AC}{=} f(t_1, t_1, t_2, t_1, t_1, t_2, t_2, t_2, t_2, t_2, t_3, t_4) = f(y_1, y_1, y_2)\alpha$.

• Para o termo t_1 temos: $(\underbrace{2}_{x_1}, \underbrace{0}_{x_2}, \underbrace{0}_{x_3}, \underbrace{2}_{y_1}, \underbrace{0}_{y_2})$ que é obrigatoriamente uma solução da equação (*) e, portanto, pode ser obtida como alguma combinação aditiva da base, a saber: $2 \cdot (1, 0, 0, 1, 0)$.

Considere a seguinte substituição $\sigma_1 = \{x_1 \leftarrow z_7, y_1 \leftarrow z_7\}$ e atribua a nova variável z_7 uma quantidade de ocorrências do termo t_1 igual ao coeficiente (neste caso “2”) da combinação linear aditiva da base que obtém a quintupla $(2, 0, 0, 2, 0)$. Para isto, seja $\theta_1 = \{z_7 \leftarrow f(t_1, t_1)\}$.

• Para o termo t_2 temos: $(3, 0, 0, 1, 4) = 2 \cdot (1, 0, 0, 0, 2) + 1 \cdot (1, 0, 0, 1, 0)$,
 $\sigma_2 = \{x_1 \leftarrow f(z_6, z_7), y_1 \leftarrow z_7, y_2 \leftarrow f(z_6, z_6)\}$ e $\theta_2 = \{z_6 \leftarrow f(t_2, t_2), z_7 \leftarrow t_2\}$.

• Para o termo t_3 temos: $(0, 1, 0, 0, 1)$ que já é a própria solução de número 3 da base das soluções não negativas da equação diofantina (*) (veja tabela 4.1).

$\sigma_3 = \{x_2 \leftarrow z_3, y_2 \leftarrow t_3\}$ e $\theta_3 = \{z_3 \leftarrow t_3\}$.

• Para o termo t_4 temos: $(0, 0, 1, 0, 1)$ que é a solução de número 1 da base da equação (*).
 $\sigma_4 = \{x_3 \leftarrow z_1, y_2 \leftarrow z_1\}$ e $\theta_4 = \{z_1 \leftarrow t_4\}$.

A solução $(\underbrace{0, 0, 1, 0, 1}_{\text{solução 1 da base}}) + (\underbrace{0, 1, 0, 0, 1}_{\text{solução 3 da base}}) + (\underbrace{1, 0, 0, 0, 2}_{\text{solução 6 da base}}) + (\underbrace{1, 0, 0, 1, 0}_{\text{solução 7 da base}}) = (2, 1, 1, 1, 4)$ corresponde a seguinte unificadora-AC σ retornada pelo algoritmo:

$$\begin{array}{lcl}
\sigma_1 = \{ & x_1 \leftarrow z_7, & y_1 \leftarrow z_7 \\
\sigma_2 = \{ & x_1 \leftarrow f(z_6), & y_2 \leftarrow f(z_6, z_6) \\
\sigma_3 = \{ & & x_2 \leftarrow z_3, \quad y_2 \leftarrow z_3 \\
\sigma_4 = \{ & & x_3 \leftarrow z_1, \quad y_2 \leftarrow z_1 \\
\hline
\sigma = \{ & x_1 \leftarrow f(z_6, z_7), \quad x_2 \leftarrow z_3, \quad x_3 \leftarrow z_1, \quad y_1 \leftarrow z_7, \quad y_2 \leftarrow f(z_1, z_3, z_6, z_6) & \}
\end{array}$$

Agora, note que α é uma instância-AC de σ pois basta tomarmos a seguinte substituição θ :

$$\begin{array}{lcl}
\theta_1 = \{ & & z_7 \leftarrow f(t_1, t_1) \\
\theta_2 = \{ & & z_6 \leftarrow f(t_2, t_2) \quad z_7 \leftarrow t_2 \\
\theta_3 = \{ & & z_3 \leftarrow t_3 \\
\theta_4 = \{ & z_1 \leftarrow t_4 & \\
\hline
\theta = \{ & z_1 \leftarrow t_4 \quad z_3 \leftarrow t_3 \quad z_6 \leftarrow f(t_2, t_2) \quad z_7 \leftarrow t_2 & \}
\end{array}$$

e temos:

$$\begin{aligned}
& \{x_1 \leftarrow f(z_6, z_7)\theta, x_2 \leftarrow z_3\theta, x_3 \leftarrow z_1\theta, y_1 \leftarrow z_7\theta, y_2 \leftarrow f(z_1, z_3, z_6, z_6)\theta\} = \\
& \{x_1 \leftarrow f(t_2, t_2, t_1, t_1, t_2), x_2 \leftarrow t_3, x_3 \leftarrow t_4, y_1 \leftarrow f(t_1, t_1, t_2), y_2 \leftarrow f(t_4, t_3, t_2, t_2, t_2, t_2)\} \stackrel{\text{AC}}{=} \alpha.
\end{aligned}$$

A demonstração que o algoritmo termina, é correto e completo para o caso geral é mais difícil. Stickel [Sti81] apresenta apenas uma prova para uma subclasse do caso geral de termos, que inclui o caso dos termos que possuem apenas variáveis como argumentos.

Teorema 4.25. *O algoritmo de unificação-AC termina, é correto e é completo para termos que são compostos por funções associativas-comutativas e funções ordinárias tais que qualquer variável que seja um argumento de uma função associativa-comutativa não seja argumento de nenhum outro símbolo funcional.*

[Sti81]

Entretanto, F. Fages [Fag84], utilizando noções de λ -cálculo, descreveu o algoritmo da Stickel na linguagem funcional *ML* e exibiu uma demonstração para a terminação, correção e completeza do algoritmo de Stickel para o caso geral, definindo graus de complexidade para a unificação de subtermos dos termos-AC envolvidos e associando a estas complexidades uma ordem noetheriana para a prova da terminação.

O algoritmo descrito por Stickel [Sti81] gera um conjunto completo de unificadoras-AC em relação a W , mas não necessariamente um conjunto completo e mínimo. Entretanto, como observam Fages e Huet [FH83], o algoritmo termina, significando que o conjunto completo unificadoras-AC é sempre finito, logo para obter-se o conjunto completo mínimo das unificadoras-AC basta aplicarmos um procedimento de matching-AC sobre as unificadoras-AC devolvidas

pelo algoritmo, eliminando assim aquelas que, eventualmente, são instâncias-*AC* de alguma unificadora-*AC* devolvida pelo algoritmo.

Capítulo 5

O Problema da Terminação de SRT's

Apresentamos agora os conceitos básicos sobre o problema de terminação de sistemas de reescrita de termos. A propriedade de terminação de um SRT é muito importante pois ela garante a existência de uma forma normal para cada termo do SRT. Entretanto, a propriedade de terminação de um SRT não garante que cada termo do SRT possua uma única forma normal. Computacionalmente isto reflete-se na existência de um algoritmo que encontra as formas normais dos termos e, conseqüentemente, pela própria definição de forma normal de um termo, cada forma normal é encontrada após uma quantidade finita de passos do algoritmo. Outro papel importante da propriedade de terminação de SRT's é que esta, associada à propriedade de confluência, garante que cada termo de um SRT possua exatamente uma única forma normal. Um SRT que possua estas duas propriedades em geral é muito útil pois nos permite verificar o *problema da palavra* que consiste em, dados dois termos s e t , saber se s e t possuem um mesmo significado, ou seja, se s e t pertencem a uma mesma classe de equivalência induzida por uma definição de uma relação binária entre os termos.

Dizemos que Um SRT possui a propriedade de terminação, ou ainda, que um SRT termina se e somente se ele for *noetheriano*. É comum encontrarmos artigos onde os autores referem-se a SRT's *noetherianos* como SRT's que *terminam uniformemente*, ou ainda, como sendo sistemas que possuem a propriedade de *terminação forte* (do inglês “*strongly terminating*”). Cabe realçarmos que todas estas denominações possuem significados equivalentes. Vejamos um exemplo simples e intuitivo apresentado por Dershowitz [Der87].

Exemplo 5.1. *Considere o seguinte exemplo informal de um SRT:*

$$\begin{aligned} \text{branco, vermelho} &\rightarrow \text{vermelho, branco} \\ \text{azul, vermelho} &\rightarrow \text{vermelho, azul} \\ \text{azul, branco} &\rightarrow \text{branco, azul}. \end{aligned}$$

Este SRT é como um jogo o qual, dada uma seqüência de listas de cores numa ordem qualquer, cada qual possuindo uma das cores acima, ele as ordena, colocando as listas vermelhas à esquerda, as listas azuis à direita e as listas brancas ao centro.

É muito fácil ver, neste exemplo, que este SRT termina para qualquer seqüência finita das cores vermelho, branco e azul dada inicialmente. Também é muito fácil ver que ele sempre pára quando todas as listas vermelhas estiverem à esquerda das brancas e estas últimas por sua vez estiverem à esquerda das azuis.

No decorrer deste capítulo fica entendido que as variáveis serão representadas por letras do final do alfabeto romano: $\dots, x, y, z, w$. As constantes serão representadas pelas letras do início do alfabeto: $a, b, c, \dots$ e também pelos numerais: $0, 1, 2, \dots$

Considere agora o seguinte exemplo apresentado por Dershowitz [Der82].

Exemplo 5.2. *Este SRT reescreve fórmulas do cálculo proposicional para suas formas normais disjuntivas:*

$$\neg\neg x \rightarrow x \quad (1)$$

$$\neg(x \vee y) \rightarrow (\neg x \wedge \neg y) \quad (2)$$

$$\neg(x \wedge y) \rightarrow (\neg x \vee \neg y) \quad (3)$$

$$x \wedge (y \vee z) \rightarrow (x \wedge y) \vee (x \wedge z) \quad (4)$$

$$(y \vee z) \wedge x \rightarrow (y \wedge x) \vee (z \wedge x) \quad (5)$$

Um exemplo de reescrita de um termo para sua forma normal disjuntiva é:

$$a \wedge \neg\neg(b \vee c) \rightarrow_{(2)} a \wedge \neg(\neg b \wedge \neg c) \rightarrow_{(3)} a \wedge (\neg\neg b \vee \neg\neg c) \rightarrow_{(1)} a \wedge (b \vee \neg\neg c) \rightarrow_{(1)} a \wedge (b \vee c) \rightarrow_{(4)} (a \wedge b) \vee (a \wedge c).$$

Observe que, neste último exemplo, já não podemos dizer que é fácil ver que tal SRT termina. Note que a regra (1) faz com que os termos diminuam de tamanho quando ela é aplicada, entretanto as demais regras tornam os termos cada vez maiores.

Uma característica do SRT dado acima é que ele não é confluyente. Observe que, partindo inicialmente do termo $\neg(a \wedge (b \vee c))$, e aplicando as regras de reescrita dadas no sistema de reescrita do exemplo acima, podemos chegar a duas formas normais disjuntivas. São elas:

$$\neg a \vee (\neg b \wedge \neg c) \quad \text{e} \quad (((\neg a \wedge \neg a) \vee (\neg a \wedge \neg c)) \vee ((\neg b \wedge \neg a) \vee (\neg b \wedge \neg c))).$$

De fato, sabe-se que este SRT em particular é noetheriano e podemos provar isto através do uso da noção de *ordem de caminho recursivo* a ser visto mais adiante.

No caso geral, a propriedade de terminação de SRT é indecidível, tendo sido demons-

trado por Huet e Lankford [HL78]. Dershowitz [Der87] também apresenta uma demonstração para o problema da terminação de SRT's, enunciada através do seguinte teorema:

Teorema 5.3. (Indecidibilidade da Terminação de SRT's) *A verificação da propriedade da terminação em SRT's é indecidível, mesmo se eles tiverem apenas duas regras de reescrita.*
[Der87]

O que o teorema 5.3 diz é que não adianta tentar criar um algoritmo para decidir sobre a propriedade de terminação de SRT's em geral, pois tal algoritmo é impossível de ser criado. Costuma-se dizer então, por conta deste fato, que o problema de verificação da terminação de SRT's (no caso geral) não é computável. Dershowitz exhibe uma prova para o teorema 5.3 que se baseia na seguinte idéia: Um problema é computável se e somente se podemos resolvê-lo simulando-o através de uma máquina de Turing que pára uniformemente; Dershowitz mostra que para cada máquina de Turing $\mathcal{M}$ existe um SRT, com apenas duas regras de reescrita, “equivalente” a $\mathcal{M}$. Mas, como o problema de verificar se uma máquina de Turing pára uniformemente (isto é, no caso geral) é indecidível, então o problema da verificação de parada uniforme (problema da terminação para o caso geral) de SRT's também é indecidível.

Ora, esta premissa de equivalência entre computabilidade e máquinas de Turing é justamente a Tese de Church-Turing. Na verdade, não existe ainda uma prova para a Tese de Church-Turing, entretanto a grande maioria dos matemáticos e cientistas da computação aceitam esta Tese e hoje muitas das demonstrações aceitas sobre computabilidade são feitas através da simulação de máquinas de Turing.

A questão de verificar se um problema não pode ser tratado computacionalmente teve grande aumento de interesse a partir das idéias equivalentes e independentes apresentadas em meados da década de 30 por Alonzo Church (tratou a questão da computabilidade através do formalismo matemático para funções, o λ -Cálculo, que veio a tornar-se a base para a criação da linguagem de programação LISP), Alan Turing (estudou a questão da computabilidade através da criação de uma máquina abstrata: a máquina de Turing) e Stephen Kleene (estudou o mesmo problema através da definição de uma classe de objetos chamados de *funções recursivas gerais*). Não pretendemos aqui estabelecer as bases teóricas do formalismo sobre algoritmos e computabilidade, pois não é este o objetivo deste texto. Entretanto, o leitor mais interessado no assunto pode recorrer a textos básicos sobre o tema, por exemplo [Men66, cap.5] ou ainda [Har92, parte III, cap.8].

Apesar de sabermos que não existe um algoritmo que possa decidir se um SRT qualquer termina, podemos criar mecanismos para “tentar” verificar se um SRT específico é noetheriano ou não. Para que possamos provar que um SRT R é noetheriano, é necessário levarmos em consideração todas as diversas maneiras possíveis de seqüências de reescritas em R , isto para todos os termos de $\mathcal{T}_R$, geradas através de um processo que é não determinístico (no que diz

respeito às escolhas das regras e dos subtermos dos termos sobre os quais aplicamos alguma regra de reescrita). Para uma lista sobre várias técnicas desenvolvidas para o tratamento da terminação de SRT's veja Dershowitz [Der87, pg. 70]. Uma boa referência sobre o assunto pode ser encontrada em Oliveira [dO95].

Nas próxima duas subseções, apresentamos técnicas usadas na tentativa de caracterizar a terminação ou não de SRT's particulares. Muitas das técnicas que apresentamos baseiam-se na noção de *ordens de simplificação*. Veremos que ordens deste tipo preservam a seguinte condição: qualquer termo que for sintaticamente mais simples que outro termo é também menor que este segundo. Veremos que, como uma consequência do Teorema de Kruskal (5.14), qualquer SRT que não seja noetheriano deve ter a propriedade de ser *auto-embutido*, isto é, ele deve permitir que algum termo s menos simples do que um termo t possa ser derivado a partir de t . A maioria das ordens que se tem pesquisado e utilizado para a prova de terminação em SRT são de fato ordens de simplificação e em geral, o uso desta noção de simplificação dos termos torna as provas de terminação mais simples.

5.1 SRT's Não Noetherianos: Uma Condição Necessária

Apresentamos a seguir uma condição de suficiência, a derivação com laço, para que um SRT não termine e depois apresentamos uma condição necessária (teorema 5.16) para que um SRT não seja noetheriano. Como consequência imediata, teremos uma condição de suficiência (o SRT não ser auto-embutido) para que um SRT termine.

Exemplo 5.4. *É trivial que o SRT abaixo possui a propriedade de terminação:*

$$- - x \rightarrow x$$

[Der87]

Exemplo 5.5. *É igualmente trivial o fato que o seguinte SRT não é noetheriano:*

$$-x \rightarrow - - -x$$

[Der87]

Lema 5.6. *Um SRT R termina (para todos os termos) se e somente se ele termina para qualquer derivação que se inicia com um termo que seja um instância do lado esquerdo de alguma regra em R .*

[Der87]

Definição 5.7. Dizemos que a derivação $t_1 \xrightarrow{R} t_2 \xrightarrow{R} \cdots \xrightarrow{R} t_j \xrightarrow{R} \cdots \xrightarrow{R} t_k \xrightarrow{R} \cdots$ é uma derivação com laço se t_j é um subtermo (próprio ou não) de t_k para algum $j < k$.

[Der87]

É claro que um SRT que permita uma derivação com laço não é um sistema que possui a propriedade de terminação. Entretanto, o próximo exemplo nos mostra que um SRT que não termina não precisa necessariamente admitir alguma derivação com laço.

Exemplo 5.8. Não há como gerar uma derivação com laço para o SRT abaixo:

$$-(x + y) \rightarrow (- - x + y) + y$$

Entretanto, podemos gerar a seguinte derivação infinita, iniciando-se com um termo que é uma instância do lado esquerdo, $-(x + y)$, da regra de reescrita fornecida:

$$\begin{aligned} -((- - 0 + 1) + 1) &\xrightarrow{R} (- - (- - 0 + 1) + 1) + 1 \\ &\xrightarrow{R} (-((- - - - 0 + 1) + 1) + 1) + 1 \\ &\xrightarrow{R} (((- - (- - - - 0 + 1) + 1) + 1) + 1) + 1 \\ &\xrightarrow{R} \cdots \end{aligned}$$

[Der87]

O exemplo acima mostra-nos que a derivação com laço não é uma condição necessária para que um SRT não possua a propriedade de terminação. A derivação com laço, no entanto, é uma condição suficiente para que um SRT não seja noetheriano. Vamos agora apresentar uma condição mais fraca (uma condição necessária) para a caracterização da não terminação de SRT's.

Definição 5.9. (Relação Homeomórfica Embutida) A relação homeomórfica embutida $\supseteq$ sobre um conjunto $\mathcal{T}(\mathcal{F} \cup \mathcal{V})$ de termos é definida recursivamente como segue:

$$s = f(s_1, s_2, \dots, s_m) \supseteq g(t_1, t_2, \dots, t_n) = t$$

se ocorrer um dos seguintes casos:

$$s_i \supseteq t \text{ para algum } i = 1, \dots, m$$

ou

$$f = g \text{ e } s_{i_j} \supseteq t_j \text{ para todo } j = 1, \dots, n,$$

onde $1 \leq i_1 < i_2 < \cdots < i_n \leq m$ e f, g pertencem a $\mathcal{F} \cup \mathcal{V}$.

[Der87]

A definição de relação homeomórfica embutida engloba a noção de “simplicidade sintática” no sentido que $t \supseteq s$ (neste caso dizemos que s está homeomorficamente embutido em t) se e somente se podemos obter s a partir de t “apagando” símbolos funcionais e variáveis em t . Na verdade, a definição acima motivou-se pelo fato de podermos “enxergar” os termos como árvores ordenadas: procedendo assim torna-se muito claro o que significa dizer que um termo está homeomorficamente embutido em outro. Vejamos alguns exemplos:

Exemplo 5.10. O termo $s = 0 + 1$ está homeomorficamente embutido em $t = -(0 + 1)$. Isto é, $t \supseteq s$. Vamos “olhá-los” como árvores ordenadas através da figura 5.1. As arestas em negrito expressam a sobreposição da “estrutura” do termo s sobre t . Observe que o termo s pode ser obtido a partir de t se “encaixarmos” s sobre t e eliminarmos as arestas de t que não pertençam a esta sobreposição:

Figura 5.1: Representação de um termo s homeomorficamente embutido em um termo t .

Observe que, segundo a definição dada, para que um termo s esteja embutido em um termo t não basta apenas que a “estrutura” de árvore de s possa ser sobreposta sobre t , como mostramos no exemplo, acima, mas também é necessário que possamos obter s a partir da remoção de símbolos funcionais e variáveis de t . O exemplo abaixo mostra um caso de dois termos que não são homeomórficos entre si.

Exemplo 5.11. O termo $s = 1 + 2$ não está homeomorficamente embutido em relação a $t = -(0 + 2)$. Observe que é impossível reconstruir s a partir de t pela remoção de símbolos funcionais e variáveis de t (Veja figura 5.2).

No próximo exemplo apresentamos dois termos homeomórficos entre si. Entretanto, é um caso mais geral do que o apresentado pelo exemplo 5.10.

Exemplo 5.12. $t = -((b * a) + (c * b)) * y + x * c \supseteq (-a * y) + x = s$. Veja o diagrama em árvore apresentado pela figura 5.3. Os nós da árvore que expressa o termo t que estão em negrito representam os símbolos que ocorrem em s de tal modo que, os símbolos que não estão em negrito, quando apagados, produzirão uma estrutura de árvore resultante coincidirá com o termo s .

Figura 5.2: Representação de dois termos básicos que não são homeomórficos entre si.

Figura 5.3: Representação de um termo s homeomorficamente embutido em um termo t .

No próximo exemplo, simulamos a recursividade dada pela definição 5.9.

Exemplo 5.13. *Sejam os termos $s = -(0+1)$ e $t = 0+1$. Vamos aplicar a definição 5.9 para verificarmos se t está ou não homeomorficamente embutido em s . Isto é, queremos mostrar que $s \supseteq t$. A figura 5.4 mostra a simulação da recursividade imposta aos termos s e t pela definição de $\supseteq$ ($s \supseteq t$).*

Dershowitz [Der82] estabelece uma condição suficiente para a caracterização da propriedade de terminação em SRT's, ele faz isto partindo do seguinte Teorema de Kruskal:

Teorema 5.14. (Kruskal, 1954, 1960) *Se $\mathcal{F}$ for um conjunto finito de símbolos funcionais, então qualquer seqüência infinita de termos $t_1, t_2, \dots$, com cada t_i em $\mathcal{T}(\mathcal{F})$ contém dois termos t_j e t_k , ($j < k$) tal que $t_k \supseteq t_j$.*

[Der87]

Figura 5.4: Diagrama da recursão para $-(0 + 1) \geq 0 + 1$.

Definição 5.15. Uma derivação $t_1 \xrightarrow{R} t_2 \xrightarrow{R} \cdots \xrightarrow{R} t_j \xrightarrow{R} \cdots \xrightarrow{R} t_k \xrightarrow{R} \cdots$ é auto-embutida se $t_k \geq t_j$ para algum j e algum k tal que $j < k$. Diremos que um sistema de reescrita é auto-embutido se ele permitir uma derivação auto-embutida.

[Der87]

Teorema 5.16. (Dershowitz, [Der82]) Se um SRT básico e finito (isto é, um SRT com uma quantidade finita de regras de reescrita, onde cada regra de reescrita é formada por termos básicos) não tiver a propriedade de terminação, então ele será auto-embutido.

[Der87]

Demonstração: Se um sistema de reescrita de termos R não termina, então, por definição, existe pelo menos uma derivação infinita $t_1 \xrightarrow{R} t_2 \xrightarrow{R} \cdots$. Pelo fato de R ser, por hipótese, um sistema básico e finito, então deve obrigatoriamente haver apenas um número finito de diferentes símbolos de $\mathcal{F}$ que aparecem na derivação, logo pelo Teorema de Kruskal (5.14) existem j e k ,

$(j < k)$, tal que $t_k \succeq t_j$. ■

O exemplo abaixo dado em [Der87] ilustra o fato que um SRT pode ser ao mesmo tempo um sistema auto-embutido e *noetheriano*.

Exemplo 5.17. *Seja R o seguinte SRT formado pela única regra de reescrita:*

$$f(f(x)) \rightarrow f(g(f(x)))$$

*Tomemos $s = f(f(x))$ e $t = f(g(f(x)))$. Observe que $s \xrightarrow{R} t$ e $t \succeq s$, logo R é auto-embutido. Entretanto, é fácil ver que R é *noetheriano*.*

[Der87]

Pelo que vimos até aqui, para provarmos a terminação de qualquer SRT R , bastaria tentarmos encontrar um algoritmo capaz de provar que R não seja auto-embutido. Entretanto, Plaisted [Pla85] provou que esta condição de suficiência para caracterizar-se a terminação de SRT também é indecidível.

Teorema 5.18. *O problema de verificar se um sistema de reescrita de termos (finito) é auto-embutido é indecidível.*

[Pla85]

O teorema 5.18 de Plaisted diz ser impossível criar um algoritmo para verificar se um SRT (finito) qualquer é auto-embutido. Porém, é claro que, dado um SRT (finito) R qualquer, pode-se “tentar” descobrir se R é auto-embutido pela geração de derivações até que se encontre alguma derivação auto-embutida. É nesse sentido que também podemos dizer que o problema de verificar-se se um SRT é auto-embutido é semi-decidível.

5.2 Métodos para Provar a Terminação de SRT's

Já citamos, no teorema 5.3, que no caso geral, o problema de verificar a terminação de SRT's é indecidível. O que vamos apresentar aqui são alguns métodos para “tentar” provar a terminação de SRT's específicos.

Através da noção de relação de ordem parcial *noetheriana*, Manna e Ness propuseram o seguinte método, enunciado através do teorema abaixo, para provar-se a terminação de SRT's.

Teorema 5.19. (Manna e Ness, 1969) *Um SRT R sobre um conjunto de termos $\mathcal{T}$ termina se e somente se existir uma relação de ordem parcial noetheriana $\succ$ sobre os termos de $\mathcal{T}$ tal que:*

$$t \xrightarrow{R} u \text{ implica } t \succ u$$

para todos os termos t e u em $\mathcal{T}$.

[Der87]

Ou seja, o teorema de Manna e Ness nos diz que um SRT R termina se sua relação de reescrita $\xrightarrow{R}$ estiver contida na relação de ordem parcial noetheriana $\succ$.

Definição 5.20. (Ordem Lexicográfica) *Se $\succ_i$ é uma ordem parcial sobre um conjunto S_i , para cada $i \in [1..n]$, então sua combinação lexicográfica $(\succ_1, \succ_2, \dots, \succ_n)$ é definida como segue:*

$$(a_1, a_2, \dots, a_n)(\succ_1, \succ_2, \dots, \succ_n)(b_1, b_2, \dots, b_n)$$

se existir $i \in [1..n]$ tal que para todo $j \in [1..i-1]$ tem-se:

$$a_j \succsim_j b_j \quad \text{e} \quad a_i \succ_i b_i$$

onde $\succsim_j$ denota o fecho reflexivo de $\succ_j$.

[Hen91]

Note que qualquer combinação lexicográfica sobre um conjunto $S_1 \times S_2 \times \dots \times S_n$ é uma ordem parcial sobre $S_1 \times S_2 \times \dots \times S_n$.

Lema 5.21. *Se as ordens parciais $\succ_i$ são bem fundadas para todo $i \in [1..n]$, então a combinação lexicográfica $(\succ_1, \succ_2, \dots, \succ_n)$ é bem fundada.*

[Hen91]

O exemplo a seguir é baseado num outro exemplo apresentado por Dershowitz [Der87].

Exemplo 5.22. *No exemplo 5.1, onde o alfabeto é:*

$$\Sigma = \{\text{branco}, \text{vermelho}, \text{azul}\}$$

podemos mostrar que, de fato, R termina se considerarmos a seguinte relação de ordem parcial noetheriana $\succ$ sobre as cores:

$$\succ = \{(\text{branco}, \text{vermelho}), (\text{azul}, \text{branco}), (\text{azul}, \text{vermelho})\}$$

Por facilidade de escrita vamos nos referir às cores pelas suas iniciais. Então, a relação de ordem lexicográfica $\succ^n$ induzida por $\succ$ é *nætheriana*, onde n é fixo e indica que apenas as “palavras” em Σ^* de tamanho n são consideradas. Sendo assim, a título de ilustração, se $n = 2$ as palavras de comprimento 2 em Σ^* são: $AA, AB, AV, BA, BB, BV, VA, VB$ e VV e a ordem lexicográfica $\succ^2$ induzida por $\succ$ é:

$$\succ^2 = \{(VA, VV), (VB, VV), (AV, VV), (BV, VV), \dots, (AB, AV), (AA, AV), (AA, AB)\}$$

A ordem $\succ^2$ é compatível com as regras do SRT pois a regra 1 é $BV \rightarrow VB$ e temos $BV \succ^2 VB$; a regra 2 é $AV \rightarrow VA$ e temos $AV \succ^2 VA$; por último, a regra 3 é $AB \rightarrow BA$ e temos $AB \succ^2 BA$.

Logo R termina, pois para qualquer palavra t de tamanho n temos uma ordem lexicográfica *nætheriana* $\succ^n$, induzida por $\succ$, tal que:

$$t \xrightarrow{R} u \text{ implica } t \succ^n u$$

pois, dizer que $t \xrightarrow{R} u$ significa que:

$$t = a_1 a_2 \dots a_i a_{i+1} \dots a_n \quad e \quad u = b_1 b_2 \dots b_i b_{i+1} \dots b_n$$

onde, $a_j = b_j$ para todo $1 \leq j < i$ e $i+1 \leq j \leq n$ e ainda, $a_i a_{i+1} = BV$ ou $a_i a_{i+1} = AV$ ou $a_i a_{i+1} = AB$ e então obtemos $b_i b_{i+1} = VB$ ou $b_i b_{i+1} = VA$ ou $b_i b_{i+1} = BA$ respectivamente, pela simples aplicação de uma das três regras do SRT R considerado.

E, portanto, pela própria definição de ordem lexicográfica, é fácil ver que:

$$t = a_1 a_2 \dots a_i a_{i+1} \dots a_n \succ^n a_1 a_2 \dots b_i b_{i+1} \dots a_n = u$$

Observe o que fizemos neste exemplo: Foi dado um SRT R (aquele do exemplo 5.1) e queríamos mostrar (e de fato mostramos) que R termina. Usamos uma ordem $\succ^n$ *nætheriana* (uma ordem que, em particular, é uma ordem lexicográfica) induzida pela ordem parcial $\succ$ *nætheriana* e mostramos que esta ordem satisfaz o Teorema 5.19, isto é, que $t \xrightarrow{R} u$ implica $t \succ^n u$. Portanto, pelo teorema 5.19, R termina.

Kamin e Levy reformulam o teorema 5.19. Levando em consideração a estrutura dos termos, eles apresentam o seguinte corolário:

Corolário 5.23. (Kamin e Levy, 1980) *Um SRT R sobre um conjunto de termos $\mathcal{T}$ termina se e somente se existir uma relação de ordem parcial *nætheriana* $\succ$ sobre os termos de $\mathcal{T}$ tal que, para cada regra $l \rightarrow r$ em R , tem-se que:*

$$i) \quad l\theta \succ r\theta$$

onde, θ é qualquer substituição em $\mathcal{T}$ cujo domínio são as variáveis da regra $l \rightarrow r$ e tal que

$$ii) \ t \succ u \text{ e } t \xrightarrow{R} u \text{ implica } f(\dots t \dots) \succ f(\dots u \dots)$$

para todos os termos em $\mathcal{T}$

[Der87]

O exemplo abaixo foi baseado num outro exemplo apresentado por Dershowitz [Der87].

Exemplo 5.24. Tomemos o seguinte SRT R formado por apenas uma única regra de reescrita:

$$f(f(x)) \rightarrow f(g(f(x)))$$

Queremos provar que R termina. Vamos, então, definir a seguinte relação de ordem parcial estrita $\succ$ sobre os termos de $\mathcal{T}$:

- $t \succ s$ se e somente se a quantidade de símbolos “f” adjacentes em t for estritamente maior que a quantidade de símbolos “f” adjacentes em s .

Alguns comentários são necessários, neste ponto. Primeiro, observamos que, de fato, $\succ$, especificada acima, é uma relação de ordem parcial estrita, pois trivialmente ela satisfaz as propriedades transitiva e anti-simétrica. Segundo, $\succ$ é também ω -nætheriana, fato este também trivialmente verificável. Agora, como vamos mostrar que, de fato:

$$f(f(x))\theta \succ f(g(f(x)))\theta$$

para qualquer substituição θ com domínio $\{x\}$, uma vez que há uma quantidade infinita de tais θ 's?

A resposta a esta pergunta é a seguinte: não há prova “construtiva”, no sentido que é impossível exibir cada θ , uma a uma, para verificar se cada θ satisfaz a condição acima, pois há infinitas θ 's. Como consequência, é comum tentar-se apresentar provas “não construtivas” (no sentido dado acima) para verificar-se a terminação de SRT's específicos. Sendo assim, observe que, neste nosso exemplo, qualquer θ de interesse para provar-se a terminação terá uma das seguintes formas:

$$\theta = \{x \leftarrow a\}, \text{ onde } a \text{ é uma constante qualquer.}$$

ou

$$\theta = \{x \leftarrow y\}, \text{ onde } y \text{ é uma variável qualquer.}$$

ou

$$\theta = \{x \leftarrow f(\dots)\}$$

ou

$$\theta = \{x \leftarrow h(\dots)\}, \text{ onde } h \text{ é um símbolo funcional qualquer tal que } h \neq f. \text{ Eventualmente } h = g.$$

É fácil ver que em qualquer um dos quatro casos acima temos garantida a condição i) do corolário 5.23:

$$f(f(x))\theta \succ f(g(f(x)))\theta$$

onde $\succ$ é a nossa ordem especificada no início do exercício. Também é muito fácil ver que nossa ordem adotada também satisfaz a condição ii) do corolário 5.23. Conseqüentemente, temos que $R = \{f(f(x)) \rightarrow f(g(f(x)))\}$ termina.

Definição 5.25. (Ordem Monotônica) Uma ordem $\succ$ parcial sobre $\mathcal{T}$ é dita monotônica quando:

$$\text{se } s \succ t \text{ então } f(\dots s \dots) \succ f(\dots t \dots)$$

[Der87]

Observe que a ordem adotada no exemplo 5.24 não é monotônica pois $g(f(f(a))) \succ f(a)$ e, entretanto, $f(g(f(f(a)))) \not\succ f(f(a))$. Tanto o teorema 5.19 quanto o corolário 5.23 não impõem que a monotonicidade seja uma condição necessária à relação de ordem noetheriana utilizada. Como consequência, eles precisam considerar um número infinito de possíveis reescritas $t \xrightarrow{R} u$ nas provas de terminação. Lankford (1977) elimina este problema incorporando a noção de monotonicidade à ordem usada para provar-se a terminação dos SRT's.

Teorema 5.26. (Lankford, 1977) Um SRT R sobre um conjunto de termos $\mathcal{T}$ termina se e somente se existir uma ordem parcial noetheriana monotônica $\succ$ sobre $\mathcal{T}$ tal que, para cada regra $l \rightarrow r$ em R ,

$$l\theta \succ r\theta$$

onde θ é qualquer substituição em $\mathcal{T}$ para as variáveis da regra $l \rightarrow r$.

[Der87]

Definição 5.27. (Ordem Parcial Estrita Estável) Uma ordem parcial estrita $\succ$ sobre $\mathcal{T}$ é dita estável se e somente se para quaisquer termos s e t em $\mathcal{T}$ e qualquer substituição θ em $\mathcal{T}$ tem-se:

$$\text{se } s \succ t \text{ então } s\theta \succ t\theta$$

[dO95]

Costuma-se dizer que uma ordem parcial estrita estável é uma ordem parcial estrita que possui a propriedade de estabilidade. Nos textos em inglês é comumente denominada por “full invariance order”.

O teorema de Lankford 5.26 acima elimina a problemática inerente aos teorema 5.19 e corolário 5.23, entretanto, ainda é necessário requerer algum meio para testar-se a desigualdade ($\succ$) para todas as substituições que, em geral, são infinitas (a menos quando $\mathcal{F}$ possua apenas símbolos funcionais de aridade zero, isto é, constantes e em quantidade finita e o conjunto dos termos $\mathcal{T}$ considerado for restrito ao subconjunto de termos básicos $\mathcal{T}(\mathcal{F})$; ou ainda se, $\mathcal{F} \cup \mathcal{V}$ possuir apenas uma quantidade finita de elementos, quando o conjunto $\mathcal{T}$ considerado incluir termos livres, isto é: $\mathcal{T} = \mathcal{T}(\mathcal{F} \cup \mathcal{V})$).

Podemos eliminar este último problema se, ao invés de procurarmos ordens monotônicas e noetherianas sobre termos, procurarmos por ordens estáveis, monotônicas e noetherianas sobre termos. Como consequência, para provar que um SRT R termina basta provar que para cada regra $l \rightarrow r$ em R tem-se $l \succ r$, onde $\succ$ é uma ordem parcial estável, monotônica e noetheriana sobre termos. Plaisted [Pla85] sintetiza estas observações do seguinte modo:

Definição 5.28. (Ordem de Terminação) *Uma ordem de terminação $\succ$ sobre $\mathcal{T}$ é uma ordem parcial, monotônica, estável e noetheriana sobre $\mathcal{T}$.*

[Pla85]

Corolário 5.29. *Sejam $\succ$ uma ordem de terminação e s, t termos, tais que $s \succ t$, então $Var(s) \supseteq Var(t)$.*

Demonstração: Considere, por absurdo, que exista variável x tal que $x \in Var(t)$ e $x \notin Var(s)$. Seja a substituição $\gamma = \{x \leftarrow s\}$. Então, $s = s\gamma$ e $t\gamma$ é da forma $f(\dots s \dots)$, onde $f = t \langle \rangle$, isto é, f é a raiz de t . Mas, $s = s\gamma \succ t\gamma$ pela propriedade de estabilidade de $\succ$. Agora, pela propriedade monotônica de $\succ$ temos a existência de uma sequência infinita:

$$f(\dots s \dots) \succ f(\dots f(\dots s \dots) \dots) \succ f(\dots f(\dots f(\dots s \dots) \dots) \dots) \succ \dots$$

o que é uma contradição, pois por hipótese, $\succ$ é noetheriana. ♦

Teorema 5.30. (Plaisted) *Sejam R um SRT e $\succ$ uma ordem de terminação sobre um conjunto de termos $\mathcal{T}$, então R termina sobre os termos de $\mathcal{T}$ se para toda regra $l \rightarrow r$ em R tem-se que $l \succ r$.*

[Pla85]

Exemplo 5.31. *Considere o seguinte SRT:*

$$R = \{f(g(x)) \rightarrow g(f(x))\},$$

Estamos considerando que f e g são símbolos funcionais unários. Queremos encontrar uma ordem parcial monotônica $\mathfrak{n}\mathfrak{a}\mathfrak{e}\mathfrak{t}\mathfrak{h}\mathfrak{e}\mathfrak{r}\mathfrak{i}\mathfrak{a}\mathfrak{n}\mathfrak{a}$ e estável $\succ$ sobre os termos de $\mathcal{T}_R$, de tal modo que possamos aplicar o teorema 5.30 para provarmos que R termina para os termos de $\mathcal{T}_R$. Vamos especificar então $\succ$ do seguinte modo:

- Dois termos são incomparáveis (segundo $\succ$) se algum deles possui alguma variável que o outro não possua, caso contrário $s \succ t$ se uma das três condições ocorrer:
 - i) a quantidade de símbolos funcionais em s , denotada por $\text{Func}(s)$, for estritamente maior que a quantidade de símbolos funcionais em t , $\text{Func}(t)$; isto é: $\text{Func}(s) > \text{Func}(t)$
 - ii) $\text{Func}(s) = \text{Func}(t)$ e o símbolo raiz de s for “ f ”, isto é, $s \langle \rangle = f$, enquanto que $t \langle \rangle = g$;
 - iii) $s \langle \rangle = t \langle \rangle$, mas o operando em s for (recursivamente) “maior” (segundo $\succ$) que o operando em t .

É fácil ver que, de fato, $\succ$ é uma ordem de terminação e que $f(g(x)) \succ g(f(x))$. Portanto pelo teorema 5.30, R termina.

Definição 5.32. (Função de Terminação) Uma função de terminação τ com domínio num conjunto de termos $\mathcal{T}$ e contra-domínio num conjunto parcialmente ordenado $(\mathcal{W}, \succ)$, onde $\succ$ é uma ordem parcial $\mathfrak{n}\mathfrak{a}\mathfrak{e}\mathfrak{t}\mathfrak{h}\mathfrak{e}\mathfrak{r}\mathfrak{i}\mathfrak{a}\mathfrak{n}\mathfrak{a}$ sobre os elementos de $\mathcal{W}$, isto é,

$$\tau : \mathcal{T} \mapsto \mathcal{W}$$

é composta por:

- i) um conjunto de funções $f_\tau : \mathcal{W}^n \mapsto \mathcal{W}$, uma para cada símbolo funcional f de aridade n

união com

- ii) um conjunto de funções constantes $x_\tau : \mapsto \mathcal{W}$, uma para cada variável x em $\mathcal{V}$

e é definida indutivamente por: $\tau(f(t_1, \dots, t_n)) = f_\tau(\tau(t_1), \dots, \tau(t_n))$

para cada termo $f(t_1, \dots, t_n)$ em $\mathcal{T}$ e tal que:

$$w \succ w' \text{ implica } f_{\tau}(\dots w \dots) \succ f_{\tau}(\dots w' \dots) \quad (\text{monotonicidade})$$

para todo w, w' em $\mathcal{W}$ e f em $\mathcal{F} \cup \mathcal{V}$.

[Der87]

O teorema abaixo, com base na definição de função de terminação, nos dá uma condição necessária e suficiente para a terminação de SRT's.

Teorema 5.33. (Lankford,1975a) *Um SRT R sobre um conjunto de termos $\mathcal{T}$ termina se e somente se existir um conjunto parcialmente ordenado $(\mathcal{W}, \succ)$, onde $\succ$ é uma ordem parcial ω -noetheriana sobre os elementos de $\mathcal{W}$, e existir uma função de terminação $\tau : \mathcal{T} \mapsto \mathcal{W}$, tal que:*

$$\tau(l\sigma) \succ \tau(r\sigma)$$

para cada regra $l \rightarrow r$ em R e qualquer substituição σ em $\mathcal{T}$ para as variáveis da regra $l \rightarrow r$.

[Der87]

Iturriaga (1967) propôs interpretações exponenciais às funções de terminação, enquanto que Lankford (1975) desenvolveu interpretações polinomiais às funções de terminação. A proposta de Lankford consiste em associar a cada símbolo funcional f um polinômio de grau menor ou igual a aridade de f , de tal modo que cada termo seja associado a um número inteiro não negativo, desta forma pode-se garantir que a interpretação seja ω -noetheriana, pois a ordem estrita $>$ usual dos naturais é ω -noetheriana. Para isto, os polinômios escolhidos numa interpretação polinomial devem ser tais que seus coeficientes, escolhidos adequadamente, garantam a propriedade monotônica da relação $\succ$ em $\mathcal{W}$.

Observe que, nas interpretações polinomiais dadas à função de terminação τ , temos:

- $\succ = >$, onde $>$ é a relação de ordem estrita usual dos naturais;
- $\mathcal{W} \subseteq \mathbb{N}$. Note que $\mathcal{W} = \{\tau(t) : t \in \mathcal{T}\}$

Exemplo 5.34. (Interpretação Polinomial) *Seja $\mathcal{T}$ o conjunto dos termos construídos a partir das constantes 0 e 1, dos símbolos funcionais binários $*$ e $\times$ e das variáveis $x, y, z, \dots$. Para mostrarmos que o seguinte SRT:*

$$\begin{aligned} x \times (y * z) &\rightarrow (x \times y) * (x \times z) \\ (y * z) \times x &\rightarrow (y \times x) * (z \times x) \\ (x * y) * z &\rightarrow x * (y * z) \end{aligned}$$

termina para todos os termos de $\mathcal{T}$, usamos a seguinte interpretação polinomial para a função de terminação τ , composta pelas seguintes funções:

$$\begin{aligned}
\times_{\tau} : \mathbb{N}^2 &\mapsto \mathbb{N} \\
*_\tau : \mathbb{N}^2 &\mapsto \mathbb{N} \\
0_{\tau} : &\mapsto \mathbb{N} \\
1_{\tau} : &\mapsto \mathbb{N} \\
x_{\tau} : &\mapsto \mathbb{N}
\end{aligned}$$

e especificadas, respectivamente, por:

$$\begin{aligned}
\times_{\tau}(x, y) &= x_{\tau} \cdot y_{\tau} \\
*_\tau(x, y) &= 2 \cdot x_{\tau} + y_{\tau} + 1 \\
0_{\tau} &= 2 \\
1_{\tau} &= 2 \\
x_{\tau} &= 2 \quad \text{para qualquer } x \text{ em } \mathcal{V}
\end{aligned}$$

É fácil ver que esta interpretação polinomial garante a propriedade monotônica, pois $\tau(0) = 0_{\tau} = 2$, $\tau(1) = 1_{\tau} = 2$ e $\tau(x) = x_{\tau} = 2$, para qualquer x em $\mathcal{V}$, logo $\tau(t)$ é estritamente positivo (ainda mais, $\tau(t) \geq 2$) para qualquer t em $\mathcal{T}$, ou seja, de fato, esta interpretação polinomial é uma função de terminação. Sendo assim,

- para a primeira regra de R e qualquer substituição σ temos:

$$\begin{aligned}
\tau\left((x \times (y * z))\sigma\right) &= \tau\left(\times(x\sigma, *(y\sigma, z\sigma))\right) = \times_{\tau}\left(\tau(x\sigma), \tau(* (y\sigma, z\sigma))\right) = \\
&= \times_{\tau}\left(\tau(x\sigma), *_\tau(\tau(y\sigma), \tau(z\sigma))\right) = \tau(x\sigma) \cdot *_\tau(\tau(y\sigma), \tau(z\sigma)) = \tau(x\sigma) \cdot (2 \cdot \tau(y\sigma) + \tau(z\sigma) + 1) = \\
&= 2 \cdot \tau(x\sigma) \cdot \tau(y\sigma) + \tau(x\sigma) \cdot \tau(z\sigma) + \tau(x\sigma)
\end{aligned}$$

e

$$\begin{aligned}
\tau\left(((x \times y) * (x \times z))\sigma\right) &= \tau\left(*\left(\times(x\sigma, y\sigma), \times(x\sigma, z\sigma)\right)\right) = *_\tau\left(\tau(\times(x\sigma)), \tau(\times(x\sigma, z\sigma))\right) = \\
&= *_\tau\left(\times_{\tau}(\tau(x\sigma), \tau(y\sigma)), \times_{\tau}(\tau(x\sigma), \tau(z\sigma))\right) = 2 \cdot \times_{\tau}(\tau(x\sigma)) + \times_{\tau}(\tau(x\sigma), \tau(z\sigma)) + 1 = \\
&= 2 \cdot \tau(x\sigma) \cdot \tau(y\sigma) + \tau(x\sigma) \cdot \tau(z\sigma) + 1
\end{aligned}$$

Ora, só relembrando, queremos que $\tau\left((x \times (y * z))\sigma\right) \succ \tau\left(((x \times y) * (x \times z))\sigma\right)$. τ é uma interpretação polinomial, sendo assim, $\mathcal{W} \subseteq \mathbb{N}$ e $\succ$ é a relação “maior que” usual dos números naturais. Sendo assim, queremos que $\tau\left((x \times (y * z))\sigma\right) > \tau\left(((x \times y) * (x \times z))\sigma\right)$ ou equivalentemente, $\tau\left((x \times (y * z))\sigma\right) - \tau\left(((x \times y) * (x \times z))\sigma\right) > 0$.

Temos que, $\tau\left((x \times (y * z))\sigma\right) - \tau\left(((x \times y) * (x \times z))\sigma\right) = \tau(x\sigma) - 1$.

Mas, de fato, $\tau(x\sigma) - 1 > 0$, pois se x for uma constante então, por definição de τ , teremos que $\tau(x\sigma) = \tau(x) = 2$, caso contrário, $\tau(x\sigma) \geq 2$ e portanto, $\tau\left((x \times (y * x))\sigma\right) > \tau\left(((x \times y) * (x \times x))\sigma\right)$.

$(x \times z))\sigma$).

- para a segunda regra de R , a análise é similar à que acabamos de fazer para a primeira regra.
- para a terceira regra de R , temos:

$$\begin{aligned} \tau\left(((x * y) * z)\sigma\right) &= \tau((x\sigma * y\sigma) * z\sigma) = \tau\left(*\left(*\left(x\sigma, y\sigma\right), z\sigma\right)\right) = *_\tau\left(\tau\left(*\left(x\sigma, y\sigma\right)\right), \tau(z\sigma)\right) = \\ & *_\tau\left(*_\tau\left(\tau(x\sigma), \tau(y\sigma)\right), \tau(z\sigma)\right) = 2 \cdot \left(*_\tau\left(\tau(x\sigma), \tau(y\sigma)\right)\right) + \tau(z\sigma) + 1 = \\ 2 \cdot (2 \cdot \tau(x\sigma) + \tau(y\sigma) + 1) + \tau(z\sigma) + 1 &= 4 \cdot \tau(x\sigma) + 2 \cdot \tau(y\sigma) + 2 + \tau(z\sigma) + 1 = 4 \cdot \tau(x\sigma) + 2 \cdot \tau(y\sigma) + \tau(z\sigma) + 3 \end{aligned}$$

e

$$\begin{aligned} \tau\left((x * (y * z))\sigma\right) &= \tau(x\sigma * (y\sigma * z\sigma)) = \tau\left(*\left(x\sigma, *(y\sigma, z\sigma)\right)\right) = *_\tau\left(\tau(x\sigma), \tau\left(*\left(y\sigma, z\sigma\right)\right)\right) = \\ & *_\tau\left(\tau(x\sigma), *_\tau\left(\tau(y\sigma), \tau(z\sigma)\right)\right) = 2 \cdot \tau(x\sigma) + *_\tau\left(\tau(y\sigma), \tau(z\sigma)\right) + 1 = \\ 2 \cdot \tau(x\sigma) + 2 \cdot \tau(y\sigma) + \tau(z\sigma) + 1 + 1 &= 2 \cdot \tau(x\sigma) + 2 \cdot \tau(y\sigma) + \tau(z\sigma) + 2 \end{aligned}$$

Agora, $\tau\left(((x * y) * z)\sigma\right) - \tau\left((x * (y * z))\sigma\right) = 2 \cdot \tau(x\sigma) + 1 > 0$, pois $\tau(x\sigma) \geq 2$.

Portanto, pelo teorema 5.33, R termina.

“A interpretação polinomial define uma classe de ordens de terminação, pois a relação $>$ (“maior que” usual dos inteiros) é *nætheriana* sobre os inteiros positivos e a estabilidade é garantida pela restrição de monotonicidade dos polinômios. Entretanto, não é fácil escolher o polinômio que será suficiente para mostrar a terminação de um dado SRT. Além disso, nem todo SRT se adequa a essa técnica” Plaisted [Pla85].

Lema 5.35. Dado um alfabeto (finito) Σ e uma ordem total $\preceq$ sobre Σ , então existe um maior elemento f em Σ tal que:

$$\nexists f' \in \Sigma, \text{ tal que } f \preceq f' \text{ e } f \neq f'$$

ainda, f é único.

Demonstração: Σ é finito, logo $\preceq$ é finito. $\preceq$ é total, logo existe pelo menos um maior elemento f em Σ . Ainda mais, f é único. Suponha por absurdo que existam f e f' , $f \neq f'$, tais que ambos sejam maiores elementos de Σ segundo alguma ordem total $\preceq$. Ora, mas isto só seria possível caso $f \not\preceq f'$ e também $f' \not\preceq f$ o que é impossível, pois $\preceq$ é total. ♦

Knuth e Bendix [KB70] propuseram um método para a prova de terminação de SRT's baseado na atribuição de “pesos” aos termos. A atribuição dos pesos aos termos é dada através

da especificação de uma função peso, denotada w (do inglês: “weight”), tal que:

$$w : \mathcal{T} \cup \mathcal{F} \cup \mathcal{V} \mapsto \mathbb{N}$$

e definida por:

$$w(f(t_1, \dots, t_n)) = w(f) + w(t_1) + w(t_2) + \dots + w(t_n)$$

Ou seja, o peso de um termo fica determinado a partir da atribuição de pesos aos símbolos funcionais e variáveis. Segundo Knuth e Bendix [KB70] as especificações de funções peso devem satisfazer as seguintes condições:

- i) $w(c) > 0$, para qualquer c tal que $\text{Arid}(c) = 0$;
- ii) $w(x) = \min\{w(c) | \text{Arid}(c) = 0\}$, $\forall x \in \mathcal{V}$. Logo $w(x) > 0$, $\forall x \in \mathcal{V}$.
- iii) Fixada uma ordem total estrita $\succ$, chamada *ordem de precedência*, sobre os símbolos funcionais tem-se que:
 - Se f é um operador unário e:
 - se f não for o maior elemento, segundo $\succ$, então $w(f)$ tem de ser estritamente maior que 0.
 - se f for o maior elemento, segundo $\succ$, então $w(f)$ tem de ser maior ou igual a 0.

Seja t um termo, então $n(x, t)$ denota a quantidade de ocorrências da variável x em t .

A partir das noções de ordem de precedência e de interpretação linear para as funções de terminação, Knuth e Bendix [KB70] definem uma ordem estrita, chamada de *ordem de Knuth e Bendix*, como segue:

Definição 5.36. (Ordem de Knuth e Bendix) A ordem estrita $>_{KB}$ de Knuth e Bendix sobre os termos em $\mathcal{T}$, considerando uma ordem de precedência $\succ$ sobre $\mathcal{F}$ e uma função peso w é definida como segue:

Para quaisquer termos t, s em $\mathcal{T}$, temos que $t >_{KB} s$ se e somente se um dos dois casos abaixo ocorrer:

Caso 1: $w(t) > w(s)$ e $n(x, t) \geq n(x, s)$, $\forall x \in \mathcal{V}$

Caso 2: $w(t) = w(s)$ e $n(x, t) = n(x, s)$, $\forall x \in \mathcal{V}$

Caso 2.1: t for da forma $f \cdots f(x)$, $s = x$, $x \in \mathcal{V}$ e f for o maior elemento de $\mathcal{F}$ segundo a ordem de precedência $\succ$ adotada.

ou

Caso 2.2: t for da forma $f(t_1, \dots, t_n)$, s for da forma $g(u_1, \dots, u_n)$ e um dos seguintes subcasos ocorrer:

Caso 2.2.1: $f \succ g$

Caso 2.2.2: $f = g$ e $t_1 = u_1, \dots, t_{i-1} = u_{i-1}$, $t_i >_{KB} u_i$ para algum $1 \leq i \leq n$.

[Der87]

Knuth e Bendix [KB70] provaram que a ordem estrita dada pela definição 5.36 acima é noetheriana e monotônica.

Exemplo 5.37. (Ordem de Knuth e Bendix) *Considere o seguinte conjunto de símbolos funcionais $\mathcal{F} = \{e, i, *\}$, onde $\text{Arid}(e) = 0$, $\text{Arid}(i) = 1$ e $\text{Arid}(*) = 2$.*

Seja R o seguinte SRT:

$$\begin{aligned} (x * y) * z &\rightarrow x * (y * z) \\ i(x * y) &\rightarrow i(x) * i(y) \\ i(i(x)) &\rightarrow x \end{aligned}$$

Considere a seguinte ordem de precedência $\succ$ sobre $\mathcal{F}$:

$$\succ = \{(i, *), (*, e), (i, e)\}$$

logo:

$$i \succ * \succ e$$

e portanto, neste caso, i é o maior elemento da ordem de precedência adotada. Vamos agora especificar uma interpretação linear w à função de terminação, do seguinte modo:

$$w(e) = 1, w(*) = 1 \text{ e } w(i) = 0$$

Vamos mostrar que R termina e usaremos para esta prova a ordem estrita $>_{KB}$ de Knuth e Bendix. Só lembrando que, neste nosso exemplo, temos que $w(x) = 1$ para qualquer variável x em $\mathcal{V}$, pois:

$$w(x) \stackrel{\text{def.}}{=} \min\{w(c) \mid \text{Arid}(c) = 0\} \quad \forall x \in \mathcal{V}$$

Precisamos mostrar que para cada regra $l \rightarrow r$ em R e qualquer substituição σ vale $l\sigma >_{KB} r\sigma$.

- *Para a primeira regra em R , temos: $l = (x * y) * z$ e $r = x * (y * z)$. Sendo assim,*

$$\begin{aligned} w(l\sigma) &= w((x\sigma * y\sigma) * z\sigma) \stackrel{\text{not.}}{=} w\left(*\left(*\left(x\sigma, y\sigma\right), z\sigma\right)\right) = w(*) + w(*\left(x\sigma, y\sigma\right)) + w(z\sigma) = \\ &= w(*) + w(*) + w(x\sigma) + w(y\sigma) + w(z\sigma) \end{aligned}$$

e

$$\begin{aligned} w(r\sigma) &= w(x\sigma * (y\sigma * z\sigma)) \stackrel{\text{not.}}{=} w\left(*\left(x\sigma, *(y\sigma, z\sigma)\right)\right) = w(*) + w(x\sigma) + w(*\left(y\sigma, z\sigma\right)) = \\ &= w(*) + w(x\sigma) + w(*) + w(y\sigma) + w(z\sigma) \end{aligned}$$

Bem, $w(l\sigma) = w(r\sigma)$, $n(x, l\sigma) = n(x, r\sigma)$, $\forall x \in \mathcal{V}$ e $l\sigma <> = r\sigma <> = *$, logo pelo Caso 2.2.2 da definição 5.36 basta-nos mostrar que $l\sigma / <1> = *(x\sigma, y\sigma) >_{KB} x\sigma = r\sigma / <1>$. De fato, $w(*\left(x\sigma, y\sigma\right)) = w(*) + w(x\sigma) + w(y\sigma) > w(x\sigma)$. Portanto, segundo a definição 5.36, Caso 2.2.2, temos: $(x\sigma * y\sigma) * z\sigma >_{KB} x\sigma * (y\sigma * z\sigma)$.

- Para a segunda regra em R , temos $l = i(x * y)$ e $r = i(y) * i(x)$. Sendo assim,

$$\begin{aligned} w(l\sigma) &= w(i(x\sigma * y\sigma)) \stackrel{\text{not.}}{=} w\left(i\left(*\left(x\sigma, y\sigma\right)\right)\right) = w(i) + w(*\left(x\sigma, y\sigma\right)) = \\ &= w(i) + w(*) + w(x\sigma) + w(y\sigma) \end{aligned}$$

e

$$\begin{aligned} w(r\sigma) &= w(i(y\sigma) * i(x\sigma)) \stackrel{\text{not.}}{=} w\left(*\left(i(y\sigma), i(x\sigma)\right)\right) = w(*) + w(i(y\sigma)) + w(i(x\sigma)) = \\ &= w(*) + w(i) + w(y\sigma) + w(i) + w(x\sigma) \end{aligned}$$

Bem, como $w(i) = 0$ então $w(l\sigma) = w(r\sigma)$. $n(x, l\sigma) = n(x, r\sigma)$, $\forall x \in \mathcal{V}$, mas $l\sigma <> = i \succ * = r\sigma <>$, logo pela definição 5.36, Caso 2.2.1, temos: $i(x\sigma * y\sigma) >_{KB} i(y\sigma) * i(x\sigma)$.

- Para a terceira regra em R , temos: $l = i(i(x))$ e $r = x$. Sendo assim,

$$w(l\sigma) = w\left(i\left(i(x\sigma)\right)\right) = w(i) + w(i) + w(x\sigma) = 0 + 0 + w(x\sigma) = w(x\sigma)$$

e

$$w(r\sigma) = w(x\sigma)$$

Bem, $w(l\sigma) = w(r\sigma)$, $n(x, l\sigma) = n(x, r\sigma)$, $\forall x \in \mathcal{V}$, i é o maior elemento segundo a ordem de precedência $\succ$ especificada neste exemplo. Logo, segundo a definição 5.36, Caso 2.1, temos: $i(i(x\sigma)) >_{KB} x\sigma$.

Portanto, o SRT R termina.

[Wec92]

Para demonstrar a terminação de SRT's não é necessário encontrar uma relação de ordem parcial que seja noetheriana sobre todos os termos de todas as derivações possíveis, basta

apenas garantir que a ordem usada seja *noetheriana* para todos os termos de cada derivação possível. Exemplificando melhor, seja R um SRT (finito) que admita apenas as seqüências de reescritas $S_1, S_2, \dots$, cada uma delas sendo finita, ou seja, R é *noetheriano*:

$$\begin{aligned} S_1 &= (s_{11} \xrightarrow{R} s_{12} \xrightarrow{R} \cdots \xrightarrow{R} s_{1n_1}) \\ S_2 &= (s_{21} \xrightarrow{R} s_{22} \xrightarrow{R} \cdots \xrightarrow{R} s_{2n_2}) \\ &\vdots \end{aligned}$$

onde cada s_{jn_j} $j \geq 1$ é um termo irreduzível. Então, para demonstrarmos que R termina, basta exibirmos uma ordem parcial estrita $\succ$ sobre o conjunto de termos de R tal que: $\succ$ seja total e *noetheriana* para os conjunto dos termos de S_1 , e $\succ$ seja total e *noetheriana* para o conjunto de termos de S_2 , e assim por diante, para todos os S_i , $i \geq 1$ existentes. Diremos, neste caso, que $\succ$ é uma *ordem noetheriana para derivações*. Não é necessário que $\succ$ seja *noetheriana* para o conjunto formado pela união dos termos de cada S_i .

Definição 5.38. (Ordem de Simplificação - Dershowitz, 1982a) *Uma ordem parcial estrita monotônica $\succ$ é uma ordem de simplificação para um conjunto de termos $\mathcal{T}$ se ela possuir as propriedades de subtermo e remoção, respectivamente abaixo:*

$$\begin{aligned} f(\cdots t \cdots) &\succ t && \text{(propriedade de subtermo)} \\ f(\cdots t \cdots) &\succ f(\cdots \cdots) && \text{(propriedade de remoção)} \end{aligned}$$

[Der87]

Se o conjunto de símbolos funcionais considerado não possuir símbolos com aridade variável então, neste caso, uma ordem parcial estrita monotônica será uma ordem de simplificação caso ela possuir a propriedade de subtermo.

Proposição 5.39. *Qualquer ordem total estrita monotônica $\succ$ sobre um conjunto de termos $\mathcal{T}(\mathcal{F} \cup \mathcal{V})$, onde $\mathcal{F}$ é finito e possui apenas símbolos funcionais com aridade fixa é *noetheriana* se e somente se $\succ$ possuir a propriedade de subtermo.*

[Der87]

A proposição 5.39 nos mostra que em muitos casos, as ordens usadas para provar a terminação de SRT's são, de fato, ordens de simplificação.

Lema 5.40. (Embedding Lemma) *Sejam s e t termos em $\mathcal{T}$. Se $s \leq t$, então $s \preceq t$ em qualquer ordem de simplificação $\succ$.*

[Der82]

O teorema 5.41, a seguir, nos mostra como as ordens de simplificação podem ser usadas para a prova de terminação de SRT's. A vantagem neste método é que não precisamos mais provar que a ordem estrita usada seja noetheriana.

Teorema 5.41. (Dershowitz, 1982a) *Um SRT R sobre um conjunto de termos $\mathcal{T}$ termina, se existir uma ordem de simplificação $\succ$ sobre $\mathcal{T}$ tal que*

$$l\theta \succ r\theta$$

para cada regra $l \rightarrow r$ em R e qualquer substituição θ sobre as variáveis da regra $l \rightarrow r$

[Der82]

Demonstração: Primeiro vamos mostrar que $s \xrightarrow{R} t$ implica $s \succ t$.

Note que, se $s \xrightarrow{R} t$ então existe $s' = s/p$ para algum $p \in \text{doms}$ e existem substituição θ e regra $l \rightarrow r$ em R tais que:

$s' = l\theta$ e $t = s[p \leftarrow r\theta]$, isto é, $l\theta$ e $r\theta$ são subtermos de s e t , respectivamente; mas por hipótese $l\theta \succ r\theta$, logo como $\succ$ é uma ordem de simplificação então $\succ$ é monotônica e portanto $f(\dots l\theta \dots) \succ f(\dots r\theta \dots)$ e conseqüentemente, em particular temos $s \succ t$.

Portanto, acabamos de mostrar que $s \xrightarrow{R} t$ implica $s \succ t$.

Agora, suponha por absurdo, que R não termina. Então, pelo teorema 5.16, existe uma derivação $t_j \xrightarrow{R} \dots \xrightarrow{R} t_k$ ($j < k$) tal que $t_j \leq t_k$ e, pelo lema 5.40, $t_j \preceq t_k$, onde $\succ$ é a ordem de simplificação considerada no enunciado.

Mas, por hipótese $l\theta \succ r\theta$ para toda regra $l \rightarrow r$ de R e qualquer substituição θ . Como vimos $s \xrightarrow{R} t$ implica $s \succ t$, logo $t_j \succ \dots \succ t_k$ e pela transitividade de $\succ$ temos $t_j \succ t_k$. Mas $t_j \neq t_k$ pois $\succ$ é não-reflexiva, então obtemos que $t_j \prec t_k$ e $t_j \succ t_k$ que é uma contradição pela anti-simetria de $\succ$.

Portanto, R termina. ■

O próximo teorema 5.42 no entanto nos apresenta uma dificuldade no uso de ordens de simplificação para provar-se terminação de SRT's.

Teorema 5.42. (Dershowitz, 1982a) *Qualquer ordem estrita $\succ$ é total, monotônica e noetheriana para derivações se e somente se ela for uma ordem de simplificação.*

[Der87]

Isto é, de fato, toda e qualquer ordem de simplificação além de ser monotônica é também total.

A dificuldade implícita no parágrafo anterior deve-se ao fato de nem todo SRT que termina “admitir” uma ordem total monotônica sobre suas regras. O exemplo 5.43, a seguir, ilustra este fato:

Exemplo 5.43. *Considere o seguinte SRT nœtheriano R :*

$$\begin{aligned} f(a) &\rightarrow f(b) \\ g(b) &\rightarrow g(a) \end{aligned}$$

Apesar de R ser nœtheriano, ele não admite uma ordem total estrita monotônica sobre suas regras. Vejamos porquê: Seja $\succ$ uma ordem total estrita monotônica qualquer, logo $a \succ b$ ou $b \succ a$. Suponha que $a \succ b$, então $g(a) \succ g(b)$ pois $\succ$ é, por hipótese, monotônica e portanto $g(b) \rightarrow g(a)$ não seria uma redução. Caso contrário, se $b \succ a$ teríamos, com raciocínio análogo ao acima, que $f(a) \rightarrow f(b)$ não seria uma redução.

[Der87]

Definição 5.44. (Pré-ordem) *Uma pré-ordem $\succsim$ é uma relação binária transitiva e reflexiva definida sobre os elementos de um conjunto $\mathcal{L}$.*

[Der87]

Observe que, se $\succsim$ for uma pré-ordem sobre um conjunto $\mathcal{L}$, então pode-se, eventualmente, ocorrer $a \succsim b$ e $b \succsim a$, com $a \neq b$, a, b em $\mathcal{L}$.

Definição 5.45. *A parte estrita $\succ$ de uma pré-ordem $\succsim$ é definida como sendo o conjunto formado pela subtração do conjunto $\succsim$ ao conjunto $\succsim$, isto é:*

$$\succ = \succsim - \succsim$$

[Der87]

Note que a parte estrita $\succ$ de uma pré-ordem $\succsim$ é uma ordem parcial estrita.

Definição 5.46. (Pré-ordem Nœtheriana) *Dizemos que uma pré-ordem $\succsim$ é nœtheriana sempre que sua parte estrita $\succ$ (que é uma ordem parcial estrita) for nœtheriana.*

[Der87]

Note que, a parte estrita $\succ$ de uma pré-ordem $\succsim$ sobre um conjunto X é nœtheriana se e somente se toda sequência infinita $s_1 \succsim s_2 \succsim s_3 \succsim \dots$ de elementos de X contém um par $s_j \succ s_k$ para algum $j < k$.

Dershowitz [Der87] mostra que podemos usar pré-ordens (do inglês - “quasi-orderings”) para provarmos a terminação de SRT's.

Definição 5.47. (Pré-ordem Monotônica) Uma pré-ordem $\succsim$ sobre um conjunto de termos $\mathcal{T}$ é monotônica se e somente se

$$t \succsim u \text{ implica } f(\dots t \dots) \succsim f(\dots u \dots)$$

para todos os termos em $\mathcal{T}$.

[Der87]

Teorema 5.48. Um SRT R sobre um conjunto de termos $\mathcal{T}$ termina se e somente se existir uma pré-ordem $\succsim$ nætheriana que possua a propriedade de subtermo:

$$f(\dots t \dots) \succsim t$$

e tal que

$$l\theta \succ r\theta$$

para cada regra $l \rightarrow r$ em R e qualquer substituição θ sobre as variáveis da regra $l \rightarrow r$ e tal que:

$$t \succsim u \text{ e } t \xrightarrow{R} u \text{ implica } f(\dots t \dots) \succsim f(\dots u \dots)$$

para todos os termos em $\mathcal{T}$.

[Der87]

Exemplo 5.49. Seja R o seguinte SRT, formado por apenas uma única regra de reescrita:

$$(x \times y) \times z \rightarrow x \times (y \times z)$$

Denominemos $l = (x \times y) \times z$ e $r = x \times (y \times z)$, assim sendo temos que $R = \{l \rightarrow r\}$.

Vamos provar, aplicando o teorema 5.48, que R termina para os termos básicos. Para provarmos isto, vamos especificar a seguinte pré-ordem $\succsim$:

Sejam t, u termos quaisquer em $\mathcal{T}_R$, então:

- $t \succsim u$ se o tamanho $|t|$ de t (isto é, o número de símbolos funcionais em t pois t é um termo básico) for estritamente maior que o tamanho $|u|$ de u .

ou

- $t \succsim u$ se $|t| = |u|$ mas $|t|/|<1>| \geq |u|/|<1>|$. Isto é, se o tamanho de t for igual ao tamanho de u , mas o tamanho do primeiro operando de t for maior ou igual ao tamanho do primeiro operando de u .

Primeiro: $\succsim$ é, de fato, uma pré-ordem, pois ela é transitiva e reflexiva.

Segundo: $\succsim$ é *noetheriana*, pois $t \succ u$ se e somente se $|t| > |u|$; e $>$ é *noetheriana* para os números naturais. ($>$ é a parte estrita de $\succsim$).

Terceiro: É fácil ver que $\succsim$ possui a propriedade de subtermo.

Quarto: $|l\theta| = |(x\theta \times y\theta) \times z\theta| = |x\theta \times (y\theta \times z\theta)| = |r\theta|$. Mas o primeiro operando de $l\theta$ é $l\theta / <1> = x\theta \times y\theta$, e o primeiro operando de $r\theta$ é $r\theta / <1> = x\theta$. Temos $|l\theta / <1>| = |x\theta \times y\theta| = 1 + |x\theta| + |y\theta|$ e $|r\theta / <1>| = |x\theta|$. Logo, $|l\theta / <1>| > |r\theta / <1>|$ e conseqüentemente $l\theta \succ r\theta$ para qualquer substituição θ .

Quinto: Resta-nos mostrar que:

$$t \succsim u \text{ e } t \xrightarrow{R} u \text{ implica } \times(\dots t \dots) \succsim \times(\dots u \dots)$$

para todos os termos $\mathcal{T}_R$. Note, sempre que $t \xrightarrow{R} u$, temos $|t| = |u|$, pois $|l\theta| = |r\theta|$ para qualquer substituição θ . Observe ainda que, neste exemplo, apenas um dos dois casos abaixo pode ocorrer:

a) t é subtermo (próprio ou não) do segundo operando de $\times(\dots t \dots)$ e u é subtermo (próprio ou não) do segundo operando de $\times(\dots u \dots)$.

ou

b) t é subtermo (próprio ou não) do primeiro operando de $\times(\dots t \dots)$ e u é subtermo (próprio ou não) do primeiro operando de $\times(\dots u \dots)$.

Em ambos os casos, temos:

$$\begin{aligned} |\times(\dots t \dots)| &= |\times(\dots u \dots)| \quad (\text{lembre-se que } |t| = |u|) \\ &\quad e \\ |\times(\dots t \dots) / <1>| &= |\times(\dots u \dots) / <1>| \end{aligned}$$

logo,

$$\times(\dots t \dots) \succsim \times(\dots u \dots)$$

Portanto, pelo teorema 5.48, R termina.

[Der87]

A noção de pré-ordem é menos restritiva que a noção de ordem, isto é, se C_1 é o conjunto de todas as ordens e C_2 é o conjunto de todas as pré-ordens, então $C_1 \subsetneq C_2$. Dershowitz [Der87] apresenta uma “generalização” da ordem de Knuth-Bendix $>_{KB}$ através da noção de pré-ordem.

Definição 5.50 ($>_{KB}$ Generalizada). *Seja $\succsim$ uma pré-ordem sobre um conjunto $\mathcal{F} \cup \mathcal{V}$ de símbolos funcionais de aridade fixa e variáveis e seja $\succsim_{\mathcal{T}}$ uma pré-ordem sobre o conjunto de termos $\mathcal{T}$ sobre $\mathcal{F} \cup \mathcal{V}$. A ordem de Knuth-Bendix generalizada $\succsim_{kbo}$ sobre o conjunto dos termos $\mathcal{T}$ é definida recursivamente como segue:*

$$s = f(s_1, \dots, s_m) \succsim_{kbo} g(t_1, \dots, t_n) = t$$

se

$$s \succ_{\mathcal{T}} t$$

ou, caso contrário,

$$s \approx_{\mathcal{T}} t \quad e \quad f \succ_{\mathcal{F} \cup \mathcal{V}} g$$

ou, caso contrário,

$$s \approx_{\mathcal{T}} t, f \approx_{\mathcal{F} \cup \mathcal{V}} g \quad e \quad (s_1, \dots, s_m) \succ_{kbo}^* (t_1, \dots, t_n)$$

onde $\succ_{kbo}^*$ é a ordem lexicográfica induzida por $\succsim_{kbo}$.

[Der87]

Definição 5.51. (Dershowitz,1982a) *Uma pré-ordem monotônica $\succsim$ é uma pré-ordem de simplificação para um conjunto de termos $\mathcal{T}$ se ela possui a propriedade de subtermo:*

$$f(\dots t \dots) \succsim t$$

e a propriedade de remoção

$$f(\dots t \dots) \succsim f(\dots \dots)$$

para todos os termos em $\mathcal{T}$.

[Der87]

Observe que a propriedade de remoção só se aplica a termos cujo símbolo funcional, localizado na raiz do termo, possua aridade variável.

Teorema 5.52. *Se $\succsim_{\mathcal{T}}$ é uma pré-ordem de simplificação sobre um conjunto $\mathcal{T}$ de termos sobre um conjunto $\mathcal{F} \cup \mathcal{V}$ de símbolos funcionais de aridade fixa e variáveis, tal que $f(\dots t \dots) \approx_{\mathcal{T}} t$ pode eventualmente ocorrer apenas quando f for unário e f for o elemento maximal em relação à uma pré-ordem $\succsim_{\mathcal{F} \cup \mathcal{V}}$ sobre $\mathcal{F} \cup \mathcal{V}$. (isto é, $f \succsim_{\mathcal{F} \cup \mathcal{V}} g$ para qualquer símbolo g em $\mathcal{F} \cup \mathcal{V}$), então $\succ_{kbo}$ é uma ordem de simplificação sobre o conjunto dos termos $\mathcal{T}$.*

[Der87]

Exemplo 5.53. Considere o seguinte SRT R :

$$\begin{aligned} --x &\rightarrow x \\ -(x+y) &\rightarrow ---x \times ---y \\ -(x \times y) &\rightarrow ---x + ---y \end{aligned}$$

Queremos especificar uma $\succ_{kbo}$ de modo que esta seja uma ordem de simplificação sobre os termos em $\mathcal{T}_R$ e tal que possamos aplicar o teorema 5.41. Observe que o teorema 5.52 nos fornece uma condição suficiente para que $\succ_{kbo}$ seja uma ordem de simplificação sobre os termos de $\mathcal{T}_R$. Vamos especificar a seguinte pré-ordem $\lesssim_{\mathcal{T}}$ sobre os termos de $\mathcal{T}_R$:

- $s \lesssim_{\mathcal{T}} t$ se e somente se o número de ocorrências de símbolos funcionais, exceto o símbolo menos “-” em s for maior ou igual ao número de ocorrências de símbolos funcionais, exceto o menos “-” em t .

Note que, de fato, $\lesssim_{\mathcal{T}}$ assim especificada, é uma pré-ordem.

Especifiquemos agora a seguinte pré-ordem $\lesssim_{\mathcal{F} \cup \mathcal{V}}$ sobre os símbolos de $\mathcal{F} \cup \mathcal{V}$ de R :

- $- \succ_{\mathcal{F} \cup \mathcal{V}} g$ para todo g em $\mathcal{F} \cup \mathcal{V}$.

Algumas observações:

- A pré-ordem $\lesssim_{\mathcal{T}}$ especificada acima é uma pré-ordem de simplificação pois ela é monotônica e possui a propriedade de subtermo.

- Neste nosso exemplo, não nos importa a verificação da propriedade de remoção, pois, por hipótese, $\mathcal{F}_R$ possui apenas símbolos funcionais de aridade fixa.

Por último, para podermos aplicar o teorema 5.52 precisamos verificar se, de fato, $f(\dots t \dots) \approx_{\mathcal{T}} t$ pode (eventualmente) ocorrer apenas quando f for unário e f for o elemento maximal em $\lesssim_{\mathcal{F} \cup \mathcal{V}}$.

(Lembre-se que, $\approx_{\mathcal{T}} = \lesssim_{\mathcal{T}} \cap \gtrsim_{\mathcal{T}}$).

Suponha que f não seja unário:

- Se $f = +$, então precisamos mostrar que $+(\dots t \dots) \not\approx_{\mathcal{T}} t$:
Mas, trivialmente, $t \not\lesssim_{\mathcal{T}} +(\dots t \dots)$, logo $+(\dots t \dots) \not\approx_{\mathcal{T}} t$.
- Se $f = \times$, então precisamos mostrar que $\times(\dots t \dots) \not\approx_{\mathcal{T}} t$:
Mas, trivialmente, $t \not\lesssim_{\mathcal{T}} \times(\dots t \dots)$, logo $\times(\dots t \dots) \not\approx_{\mathcal{T}} t$.

Logo pelo teorema 5.52, $\succ_{kbo}$ especificada neste exemplo, é uma ordem de simplificação sobre os termos em $\mathcal{T}_R$. Lembre-se que a $\succ_{kbo}$ especificada neste exemplo é, em particular, pelo teorema 5.52, uma ordem de simplificação, logo, para qualquer substituição θ de variáveis, temos:

- $(- - x)\theta \succ_{kbo} x\theta$, pela propriedade de subtermo de $\succ_{kbo}$.
- $-(x + y)\theta \succ_{kbo} (- - -x \times - - -y)\theta$ pois:
 $-(x + y)\theta \approx_{\mathcal{T}} (- - -x \times - - -y)\theta \quad e \quad - \succ_{\mathcal{F} \cup \mathcal{V}} \times$.
- $-(x \times y)\theta \succ_{kbo} (- - -x + - - -y)\theta$ pois:
 $-(x \times y)\theta \approx_{\mathcal{T}} (- - -x + - - -y)\theta \quad e \quad - \succ_{\mathcal{F} \cup \mathcal{V}} +$

Portanto, pelo teorema 5.41, R termina.

[Der87]

Definição 5.54. Se $\succ$ é uma ordem parcial sobre um conjunto S , então sua extensão $\succ_{set}$ para conjuntos é definida como segue:

$$M \succ_{set} N$$

se

$$M \neq N \text{ e para todo } s \in N - M \text{ existir } s' \in M - N \text{ tal que } s' \succ s.$$

[Hen91]

Proposição 5.55. Se $\succ$ é uma ordem $\aleph$ -theriana, então sua extensão $\succ_{set}$ para conjuntos é $\aleph$ -theriana sobre conjuntos finitos.

[Hen91]

Definição 5.56. (Multi-Conjuntos) Um multi-conjunto sobre um conjunto S é um mapeamento $\mathcal{M}$ de S em $\mathbb{N}$.

[Hen91]

Dizemos que x é elemento de um multi-conjunto $\mathcal{M}$, denotado $x \in_{mul} \mathcal{M}$, se $\mathcal{M}(x) > 0$.

Exemplo 5.57. Considere os multi-conjuntos $\mathcal{M}_1$ e $\mathcal{M}_2$ abaixo, ambos com domínio sendo o conjunto dos números naturais:

$$\mathcal{M}_1 = \{(1, 3), (4, 2), (2, 1)\} \cup \{(x, 0)\} \quad \forall x \in \mathbb{N} - \{1, 4, 2\}$$

$$\mathcal{M}_2 = \{(2, 4), (1, 2), (4, 1)\} \cup \{(x, 0)\} \quad \forall x \in \mathbb{N} - \{1, 4, 2\}$$

É comum recorrer-se a um abuso de notação para representar-se multi-conjuntos. Para representar-se $\mathcal{M}_1$ e $\mathcal{M}_2$, por exemplo, costuma-se escrever simplesmente $\mathcal{M}_1 \stackrel{\text{notação}}{=} \{1, 1, 1, 4, 4, 2\}$ e $\mathcal{M}_2 \stackrel{\text{notação}}{=} \{2, 2, 2, 2, 1, 1, 4\}$.

Observe que a ordem em que aparecem os elementos em um multi-conjunto é irrelevante. Por exemplo, um multi-conjunto $\mathcal{M} = \{1, 1, 3\} = \{1, 3, 1\} = \{3, 1, 1\}$. Por vezes, na literatura inglesa, multi-conjuntos são denominados “bags”.

Definição 5.58. (União, Interseção e Diferença entre Multi-Conjuntos) *Sejam $\mathcal{M}_1$ e $\mathcal{M}_2$ multi-conjuntos quaisquer. Definiremos a união, interseção e diferença entre $\mathcal{M}_1$ e $\mathcal{M}_2$, respectivamente, como segue:*

- $(\mathcal{M}_1 \cup \mathcal{M}_2)(x) = \mathcal{M}_1(x) + \mathcal{M}_2(x)$
- $(\mathcal{M}_1 \cap \mathcal{M}_2)(x) = \min\{\mathcal{M}_1(x), \mathcal{M}_2(x)\}$
- $(\mathcal{M}_1 - \mathcal{M}_2)(x) = \begin{cases} \mathcal{M}_1(x) - \mathcal{M}_2(x) & \text{se } \mathcal{M}_1(x) - \mathcal{M}_2(x) \geq 0, \\ 0 & \text{caso contrário.} \end{cases}$

[Hen91]

Definição 5.59. *Se $\succ$ é uma ordem parcial sobre os elementos de um conjunto S , então sua extensão $\succ_{mul}$ para multi-conjuntos é definida como segue:*

$M \succ_{mul} N$ se $M \neq N$ e para todo $s \in_{mul} N - M$ existir $s' \in_{mul} M - N$ tal que $s' \succ s$.

[Hen91]

Proposição 5.60. *Se $\succ$ é uma ordem naetheriana, então sua extensão $\succ_{mul}$ para multi-conjuntos é naetheriana para multi-conjuntos finitos.*

[Hen91]

Com base nas noções de multi-conjunto e ordem parcial extendida a multi-conjuntos, Dershowitz [Der82] define uma ordem parcial estrita a qual ele denominou “ordem de caminho recursivo” e mostra que ela é um caso particular das ordens de simplificação e, conseqüentemente, pode-se usá-la para tentar-se provar a terminação de SRT's.

Definição 5.61. (Ordem de Caminho Recursivo) *Seja $\succ$ uma ordem parcial estrita sobre um conjunto de símbolos funcionais e variáveis $\mathcal{F} \cup \mathcal{V}$. A ordem de caminho recursivo $\succ^*$ sobre o conjunto $\mathcal{T}$ de termos sobre $\mathcal{F} \cup \mathcal{V}$ é definida recursivamente como segue:*

$$s = f(s_1, \dots, s_m) \succ^* g(t_1, \dots, t_n) = t$$

se e somente se

- i) $f = g$ e $\{s_1, \dots, s_m\} \succ_{mul}^* \{t_1, \dots, t_n\}$,
ou,
ii) $f \succ g$ e $\{s\} \succ_{mul}^* \{t_1, \dots, t_n\}$,
ou,
iii) $f \not\succeq g$ e $\{s_1, \dots, s_m\} \succ_{mul}^* \{t\}$,
ou,
iv) $f \not\succeq g$ e $\{s_1, \dots, s_m\} = \{t\}$, [Der82]

Exemplo 5.62. Consideremos o conjunto de símbolos funcionais como sendo o conjunto dos números naturais, $\mathcal{F} = \mathbb{N}$. Consideremos ainda a relação de ordem parcial estrita $>$ usual dos naturais. Vamos mostrar que:

$$s = \begin{array}{c} 3 \\ | \\ 2 \\ / \quad \backslash \\ 3 \quad 1 \\ | \quad / \quad \backslash \\ 0 \quad 0 \quad 3 \\ \quad \quad \quad | \\ \quad \quad \quad 0 \end{array} >^* \begin{array}{c} 3 \\ | \\ 1 \\ / \quad \backslash \\ 2 \quad 2 \\ / \quad \backslash \quad / \quad \backslash \\ 3 \quad 0 \quad 3 \quad 3 \\ | \quad | \quad | \quad | \\ 0 \quad 0 \quad 0 \quad 0 \end{array} = t$$

$s < > = 3 = t < >$, logo pela definição 5.61 temos de mostrar a validade do seguinte predicado:

$$M = \left\{ \begin{array}{c} 2 \\ / \quad \backslash \\ 3 \quad 1 \\ | \quad / \quad \backslash \\ 0 \quad 0 \quad 3 \\ \quad \quad \quad | \\ \quad \quad \quad 0 \end{array} \right\} >_{mul}^* \left\{ \begin{array}{c} 1 \\ / \quad \backslash \\ 2 \quad 2 \\ / \quad \backslash \quad / \quad \backslash \\ 3 \quad 0 \quad 3 \quad 3 \\ | \quad | \quad | \quad | \\ 0 \quad 0 \quad 0 \quad 0 \end{array} \right\} = N$$

Temos $N - M = N$ e $M - N = M$, logo, segundo a definição 5.59, temos de verificar se:

$$s / < 1 > = \begin{array}{c} 2 \\ / \quad \backslash \\ 3 \quad 1 \\ | \quad / \quad \backslash \\ 0 \quad 0 \quad 3 \\ \quad \quad \quad | \\ \quad \quad \quad 0 \end{array} >^* \begin{array}{c} 1 \\ / \quad \backslash \\ 2 \quad 2 \\ / \quad \backslash \quad / \quad \backslash \\ 3 \quad 0 \quad 3 \quad 3 \\ | \quad | \quad | \quad | \\ 0 \quad 0 \quad 0 \quad 0 \end{array} = t / < 1 >$$

Como $s < 1 > = 2 > 1 = t < 1 >$, então pela definição 5.61 precisamos mostrar que:

$$M_1 = \left\{ \begin{array}{c} 2 \\ / \quad \backslash \\ 3 \quad 1 \\ | \quad / \quad \backslash \\ 0 \quad 0 \quad 3 \\ \quad \quad \quad | \\ \quad \quad \quad 0 \end{array} \right\} >_{mul}^* \left\{ \begin{array}{c} 2 \\ / \quad \backslash \\ 3 \quad 0 \\ | \quad | \\ 0 \quad 0 \end{array}, \begin{array}{c} 2 \\ / \quad \backslash \\ 3 \quad 3 \\ | \quad | \\ 0 \quad 0 \end{array} \right\} = N_1$$

Novamente, $N_1 - M_1 = N_1$ e $M_1 - N_1 = M_1$ e, conseqüentemente, pela definição 5.59 temos de verificar se:

$$s/\langle 1 \rangle = \begin{array}{c} 2 \\ \swarrow \searrow \\ 3 \quad 1 \\ | \quad | \\ 0 \quad 0 \end{array} \begin{array}{c} 2 \\ \swarrow \searrow \\ 3 \quad 0 \\ | \quad | \\ 0 \quad 0 \end{array} >^* t/\langle 1, 1 \rangle = \begin{array}{c} 2 \\ \swarrow \searrow \\ 3 \quad 1 \\ | \quad | \\ 0 \quad 0 \end{array} \begin{array}{c} 2 \\ \swarrow \searrow \\ 3 \quad 3 \\ | \quad | \\ 0 \quad 0 \end{array} = t/\langle 1, 2 \rangle$$

Como $s\langle 1 \rangle = 2 = t\langle 1, 1 \rangle = t\langle 1, 2 \rangle$, então, pela definição 5.61, precisamos verificar se:

$$A) \quad M_2 = \left\{ \begin{array}{c} 3 \\ | \\ 0 \end{array}, \begin{array}{c} 1 \\ \swarrow \searrow \\ 0 \quad 3 \\ \quad | \\ \quad 0 \end{array} \right\} >_{mul}^* \left\{ \begin{array}{c} 3 \\ | \\ 0 \end{array}, 0 \right\} = N_2$$

e

$$B) \quad M_2 = \left\{ \begin{array}{c} 3 \\ | \\ 0 \end{array}, \begin{array}{c} 1 \\ \swarrow \searrow \\ 0 \quad 3 \\ \quad | \\ \quad 0 \end{array} \right\} >_{mul}^* \left\{ \begin{array}{c} 3 \\ | \\ 0 \end{array}, \begin{array}{c} 3 \\ | \\ 0 \end{array} \right\} = N_3$$

Analisando A) temos:

$$N_2 - M_2 = \{0\} \text{ e } M_2 - N_2 = \left\{ \begin{array}{c} 1 \\ \swarrow \searrow \\ 0 \quad 3 \\ \quad | \\ \quad 0 \end{array} \right\} \text{ e neste caso, precisamos verificar se:}$$

$$s/\langle 1, 2 \rangle = \begin{array}{c} 1 \\ \swarrow \searrow \\ 0 \quad 3 \\ \quad | \\ \quad 0 \end{array} >^* 0 = t/\langle 1, 1, 2 \rangle$$

mas $s\langle 1, 2 \rangle = 1 > 0 = t\langle 1, 1, 2 \rangle$ e:

$$\left\{ \begin{array}{c} 1 \\ \swarrow \searrow \\ 0 \quad 3 \\ \quad | \\ \quad 0 \end{array} \right\} >_{mul}^* \{\} \quad (\text{trivialmente}).$$

Analisando B) temos:

$$N_3 - M_2 = \left\{ \begin{array}{c} 3 \\ | \\ 0 \end{array} \right\} \text{ e } M_2 - N_3 = \left\{ \begin{array}{c} 1 \\ \swarrow \searrow \\ 0 \quad 3 \\ \quad | \\ \quad 0 \end{array} \right\} \text{ e neste caso, precisamos verificar se:}$$

$$s/\langle 1 \cdot 2 \rangle = \begin{array}{c} 1 \\ \swarrow \searrow \\ 0 \quad 3 \\ \quad | \\ \quad 0 \end{array} >^* \begin{array}{c} 3 \\ | \\ 0 \end{array} = t/\langle 1 \cdot 2 \cdot 1 \rangle = t/\langle 1 \cdot 2 \cdot 2 \rangle$$

mas, $s \langle 1, 2 \rangle = 1 \not\geq 3 = t \langle 1, 2, 1 \rangle = t \langle 1, 2, 2 \rangle$, logo pela definição 5.61 temos de verificar se:

$$A_1) \quad M_3 = \left\{ 0, \begin{array}{c} 3 \\ | \\ 0 \end{array} \right\} \succ_{mul}^* \left\{ \begin{array}{c} 3 \\ | \\ 0 \end{array} \right\} = N_4$$

ou

$$A_2) \quad M_3 = \left\{ 0, \begin{array}{c} 3 \\ | \\ 0 \end{array} \right\} = \left\{ \begin{array}{c} 3 \\ | \\ 0 \end{array} \right\} = N_4$$

É claro que $M_3 \neq N_4$, logo, precisamos mostrar que vale A_1). Mas, $N_4 - M_3 = \{\}$, logo, trivialmente pela definição 5.59, temos, de fato, $M_3 \succ_{mul}^* N_4$.

Portanto, $s \succ^* t$.

Teorema 5.63. A ordem de caminho recursivo $\succ^*$ é uma ordem de simplificação.

[Der82]

Observe que podemos deixar mais simples a prova apresentada no exemplo 5.62 se usarmos o resultado apresentado pelo Teorema 5.63 pois, toda ordem de simplificação possui a propriedade de subtermo. Sendo assim, observe no exemplo 5.62 que podemos justificar o fato $1(0, 3(0)) \succ^* 3(0)$ simplesmente observando que $\succ^*$ é uma ordem de simplificação e, conseqüentemente, possui a propriedade de subtermo.

Exemplo 5.64. Seja R o seguinte SRT:

$$\begin{aligned} \neg\neg x &\rightarrow x \\ \neg(x \vee y) &\rightarrow (\neg x \wedge \neg y) \\ \neg(x \wedge y) &\rightarrow (\neg x \vee \neg y) \\ x \wedge (y \vee z) &\rightarrow (x \wedge y) \vee (x \wedge z) \\ (y \vee z) \wedge x &\rightarrow (y \wedge x) \vee (z \wedge x) \end{aligned}$$

R é um SRT que reescreve sentenças do cálculo proposicional para suas formas normais disjuntivas.

Queremos provar que R termina. Pode-se mostrar que é impossível provar isto através do uso de alguma interpretação polinomial.

Vamos especificar uma ordem de caminho recursivo $\succ^*$ que já sabemos, pelo teorema 5.63, é uma ordem de simplificação e portanto pelo teorema 5.41 para provarmos que R termina basta provarmos que:

$$\begin{aligned}
(\neg\neg x)\theta &\succ^* x\theta \\
(\neg(x \vee y))\theta &\succ^* (\neg x \wedge \neg y)\theta \\
(\neg(x \wedge y))\theta &\succ^* (\neg x \vee \neg y)\theta \\
(x \wedge (y \vee z))\theta &\succ^* ((x \wedge y) \vee (x \wedge z))\theta \\
((y \vee z) \wedge x)\theta &\succ^* ((y \wedge x) \vee (z \wedge x))\theta
\end{aligned}$$

para qualquer substituição θ .

Consideremos a seguinte ordem total estrita $\succ$ sobre o conjunto de símbolos funcionais $\mathcal{F}_R = \{\neg, \vee, \wedge\}$:

$$\succ = \{(\neg, \wedge), (\wedge, \vee), (\neg, \vee)\}$$

ou seja:

$$\neg \succ \wedge \succ \vee.$$

Observe que $\succ$ é uma ordem parcial estrita sobre $\mathcal{F} \cup \mathcal{V}$

Vamos mostrar que:

Primeiro: $(\neg\neg x)\theta \succ^* x\theta$

Mas, de fato, isto é verdade pela propriedade de subtermo da ordem de simplificação.

Segundo: $(\neg(x \vee y))\theta \succ^* (\neg x \wedge \neg y)\theta$

Ora, $\neg \succ \wedge$ então pela definição 5.61, caso ii), precisamos mostrar que $\{\neg(x\theta \vee y\theta)\} \succ_{mul}^* \{\neg x\theta, \neg y\theta\}$:

Temos $\{\neg(x\theta \vee y\theta)\} \neq \{\neg x\theta, \neg y\theta\}$, logo, pela definição 5.59 precisamos mostrar que $\neg(x\theta \vee y\theta) \succ^* \neg x\theta$ e $\neg(x\theta \vee y\theta) \succ^* \neg y\theta$.

Mas, de fato, $\neg(x\theta \vee y\theta) \succ^* \neg x\theta$, pelo caso i) da definição 5.61, pois $(\neg(x\theta \vee y\theta)) < > = \neg = (\neg x\theta) < >$ e $\{x\theta \vee y\theta\} \succ_{mul}^* \{x\theta\}$ pela definição 5.59, pois:

$\{x\theta \vee y\theta\} \neq \{x\theta\}$ e pela propriedade de subtermo da ordem de simplificação temos $x\theta \vee y\theta \succ^* x\theta$.

Analogamente, pode-se mostrar que $\neg(x\theta \vee y\theta) \succ^* \neg y\theta$.

Terceiro: A prova do fato $(\neg(x \wedge y))\theta \succ^* (\neg x \vee \neg y)\theta$ é análoga à prova anterior.

Quarto: $(x \wedge (y \vee z))\theta \succ^* ((x \wedge y) \vee (x \wedge z))\theta$

Ora, $\wedge \succ \vee$ então pela definição 5.61 caso ii) precisamos provar que $\{x\theta \wedge (y\theta \vee z\theta)\} \succ_{mul}^* \{x\theta \wedge y\theta, x\theta \wedge z\theta\}$:

Temos que $\{x\theta \wedge (y\theta \vee z\theta)\} \neq \{x\theta \wedge y\theta, x\theta \wedge z\theta\}$, logo, pela definição 5.59, precisamos provar que $x\theta \wedge (y\theta \vee z\theta) \succ^* x\theta \wedge y\theta$ e $x\theta \wedge (y\theta \vee z\theta) \succ^* x\theta \wedge z\theta$. Sendo assim, vamos agora provar que:

$$x\theta \wedge (y\theta \vee z\theta) \succ^* x\theta \wedge y\theta:$$

Temos que $(x\theta \wedge (y\theta \vee z\theta))< > = \wedge = (x\theta \wedge y\theta)< >$, logo, pela definição 5.61, caso i), precisamos mostrar que $M = \{x\theta, y\theta \vee z\theta\} \succ_{mul}^* \{x\theta, y\theta\} = N$.

Temos $N - M = \{y\theta\}$ e $M - N = \{y\theta \vee z\theta\}$, logo, pela definição 5.59, temos de provar que $y\theta \vee z\theta \succ^* y\theta$. Mas, de fato, isto é verdade pela propriedade de subtermo da ordem de simplificação.

Analogamente prova-se que: $x\theta \wedge (y\theta \vee z\theta) \succ^* x\theta \wedge z\theta$.

Quinto: $y \vee (z \wedge x)\theta \succ^* ((y \wedge x) \vee (z \wedge x))\theta$

Esta prova é análoga à dada para a regra anterior.

Portanto pelo teorema 5.41, R termina.

[Der82]

Dershowitz [Der87] redefine ordem de caminho recursivo a partir da noção de pré-ordem, do seguinte modo:

Definição 5.65. Seja $\preceq$ uma pré-ordem sobre um conjunto $\mathcal{F} \cup \mathcal{V}$ de símbolos funcionais e variáveis. A ordem de caminho recursivo $\preceq_{rpo}$ sobre o conjunto $\mathcal{T}$ de termos sobre $\mathcal{F} \cup \mathcal{V}$ é definida recursivamente como segue:

$$s = f(s_1, \dots, s_m) \preceq_{rpo} g(t_1, \dots, t_n) = t$$

se

$$s_i \preceq_{rpo} t \text{ para algum } i = 1, \dots, m$$

ou

$$f \succ g \quad e \quad s \succ_{rpo} t_j \text{ para todo } j = 1, \dots, n$$

ou

$$f \approx g \quad e \quad \{s_1, \dots, s_m\} \succ_{rpo_{mul}} \{t_1, \dots, t_n\}$$

onde, $\succ_{rpo_{mul}}$ é a ordem de multi-conjuntos induzida por $\succ_{rpo}$.

Dois termos são equivalentes em relação a $\approx_{rpo}$ se eles forem os mesmos a menos de símbolos equivalentes (em relação a $\simeq$) e da permutação de seus subtermos. Quando dois termos são equivalentes em relação a $\approx_{rpo}$ eles caem completamente no último caso da definição de $\succ_{rpo}$.

Observe que, apesar da definição 5.65 definir $\succ_{rpo}$ como uma *ordem*, na verdade ela é, de fato, uma pré-ordem. Fica-se, neste caso, subentendido que a ordem de caminho recursivo é, de fato, apenas o subconjunto $\succ_{rpo} = \succ_{rpo} - \approx_{rpo}$.

Exemplo 5.66. Suponha que $\times \succ +$ e $a \approx d$. Observe que $\succ$ é uma pré-ordem, logo $+ \approx +$; $\times \approx \times$; $a \approx a$; $b \approx b$ e $d \approx d$. Sendo assim, temos que:

$$s = a \times (b + c) \succ_{rpo} (a \times b) + (c \times d) = t$$

em relação à correspondente ordem de caminho recursivo $\succ_{rpo}$, pela seguinte linha de raciocínio:

$$\begin{aligned} s \succ_{rpo} t & \text{ pois } \times \succ + \quad e \quad s \succ_{rpo} a \times b, c \times d \\ s \succ_{rpo} a \times b & \text{ pois } b + c \succ_{rpo} b \\ b + c \succ_{rpo} b & \text{ pois } b \succ_{rpo} b \\ b \succ_{rpo} b & \text{ pois por trivialidade } \{\} \succ_{rpo_{mul}} \{\}. \\ s \succ_{rpo} c \times d & \text{ pois } a \approx d \quad e \quad b + c \succ_{rpo} c \\ b + c \succ_{rpo} c & \text{ pois } c \succ_{rpo} c. \end{aligned}$$

5.3 Terminação Módulo T

É claro que existem SRT's que não terminam. Como ilustração, considere o seguinte SRT:

$$x \times y \rightarrow y \times x$$

É fácil ver que o sistema acima, constituído por uma única regra, que expressa a propriedade comutativa, não termina. Entretanto, por vezes é possível “dividir” um SRT R que não termina em duas partes disjuntas, digamos R' e S , de modo que $R = R' \cup S$ e busca-se mostrar que R' é um sistema de reescrita noetheriano módulo T , onde T é um conjunto de

possam ser úteis às demonstrações de terminação módulo AC . A interpretação polinomial usada deve, entretanto, preservar seus valores com respeito às leis associativa e comutativa. Isto é, se τ é uma interpretação polinomial monotônica, precisa-se ainda garantir que $\tau(x \times y) = \tau(y \times x)$ e $\tau(x \times (y \times z)) = \tau((x \times y) \times z)$, para todo símbolo $\times \in \mathcal{F}_{AC}$. Sendo assim, se $\tau(l) > \tau(r)$ para cada regra $l \rightarrow r$ de um SRT R , então teríamos que $\tau(t) > \tau(u)$ sempre que $t \stackrel{\tau}{=} \circ \xrightarrow{R} \circ \stackrel{\tau}{=} u$, uma vez que τ preserva seus valores em relação à associatividade e comutatividade.

$f_\tau(x, y) = x \cdot y$ e $f_\tau(x, y) = x + y + 1$ são exemplos de interpretações polinomiais que preservam seus valores em relação às leis associativa e comutativa; $f_\tau(x, y) = x \cdot y + 1$ é um exemplo de uma interpretação polinomial que preserva seus valores apenas em relação à comutatividade, mas não os preserva em relação à associatividade. Note, se $l = (x \times y) \times z \rightarrow x \times (y \times z) = r$ e $\times_\tau(x, y) = x \cdot y + 1$, temos:

$$\tau(l) = \times_\tau(\times_\tau(x, y), z) = \times_\tau(x \cdot y + 1, z) = (x \cdot y + 1) \cdot z + 1 = x \cdot y \cdot z + 1$$

e

$$\tau(r) = \times_\tau(x, \times_\tau(y, z)) = \times_\tau(x, y \cdot z + 1) = x \cdot (y \cdot z + 1) + 1 = x \cdot y \cdot z + x + 1$$

Logo, $\tau(l) \neq \tau(r)$.

Ben-Cherifa e Lescane implementaram, em 1986, um procedimento que prova mecanicamente a terminação de SRT's através do uso de interpretações polinomiais. A decidibilidade do problema de verificação da existência de uma interpretação polinomial τ que preserve seus valores sob as leis associativa e comutativa e que (eventualmente) seja monotônico, tenha a propriedade de subtermo e seja decrescente para cada regra foi mostrada por Lankford em 1979 (Veja Dershowitz [Der87][pg. 102]). Dershowitz observa que apesar da decidibilidade mostrada por Lankford, estes polinômios constituem uma classe muito restrita de ordens de simplificação, isto é, na prática, suponha que R seja um SRT que termina módulo AC , é muito difícil que haja uma interpretação polinomial útil para provar-se a terminação de R módulo AC . Apenas para uma classe muito restrita de SRT's que terminam módulo AC é que existe uma tal interpretação polinomial que prova que o sistema de reescrita termina módulo AC .

Exemplo 5.69. Considere o seguinte SRT R para anéis booleanos:

$$\begin{aligned} x \times 1 &\rightarrow x \\ x \times 0 &\rightarrow 0 \\ x \times x &\rightarrow x \\ x + 0 &\rightarrow x \\ x + x &\rightarrow 0 \\ (x + y) \times z &\rightarrow (x \times z) + (y \times z) \end{aligned}$$

Pode-se usar a seguinte interpretação polinomial para provar-se a terminação de R módulo AC , composta pelas seguintes funções:

$$\begin{aligned}\times_\tau &: \mathbb{N}^2 \mapsto \mathbb{N} \\ +_\tau &: \mathbb{N}^2 \mapsto \mathbb{N} \\ 0_\tau &: \mapsto \mathbb{N} \\ 1_\tau &: \mapsto \mathbb{N} \\ x_\tau &: \mapsto \mathbb{N}\end{aligned}$$

e especificadas, respectivamente, por:

$$\begin{aligned}\times_\tau(x, y) &= x \cdot y \\ +_\tau(x, y) &= x + y + 1 \\ 0_\tau &= 2 \\ 1_\tau &= 2 \\ x_\tau &= 2 \quad \text{para qualquer } x \text{ em } \mathcal{V}\end{aligned}$$

Algumas observações:

- Já vimos que τ preserva seus valores em relação às leis associativa e comutativa.
- τ é monotônica pois $+_\tau(x, y)$ e $\times_\tau(x, y)$ são estritamente positivos, logo se t e s são termos tais que $\tau(t) > \tau(s)$ então:

$$\bullet \tau(+_\tau(u, t)) = +_\tau(\tau(t), \tau(u)) = \tau(t) + \tau(u) + 1 > \tau(s) + \tau(u) + 1 = +_\tau(\tau(s), \tau(u)) = \tau(+_\tau(u, s))$$

e

$$\bullet \tau(\times_\tau(u, t)) = \times_\tau(\tau(t), \tau(u)) = \tau(t) \cdot \tau(u) > \tau(s) \cdot \tau(u) = \times_\tau(\tau(s), \tau(u)) = \tau(\times_\tau(u, s))$$

$$\text{pois } \tau(u) \geq 2, \forall u \in \mathcal{T}$$

Como já citamos, Plaisted [Pla85] mostrou que as interpretações polinomiais monotônicas constituem um caso particular de ordens de terminação. Sendo assim, pelo teorema 5.30, para demonstrar-se que R termina, basta mostrar que para cada regra $l \rightarrow r$ de R temos $\tau(l) > \tau(r)$. É evidente que a regra $l = x \times 1 \rightarrow x = r$ é uma redução, pois a quantidade de $\times$'s em r é zero e a quantidade de $\times$'s em l é um. Analogamente é fácil ver que as demais regras de R , exceto a última, também são reduções. Para a última regra $l = (x + y) \times z \rightarrow (x \times z) + (y \times z) = r$ vamos mostrar que ela é uma redução. Para isto basta observarmos que:

$$\bullet \tau(l) = \tau((x+y) \times z) = \times_{\tau}(\tau(x+y), \tau(z)) = \times_{\tau}(+_{\tau}(\tau(x), \tau(y)), \tau(z)) = +_{\tau}(\tau(x), \tau(y)) \cdot \tau(z) = (\tau(x) + \tau(y) + 1) \cdot \tau(z) = \tau(x) \cdot \tau(z) + \tau(y) \cdot \tau(z) + \tau(z)$$

e

$$\bullet \tau(r) = \tau((x \times z) + (y \times z)) = +_{\tau}(\tau(x \times z), \tau(y \times z)) = +_{\tau}(\times_{\tau}(\tau(x), \tau(z)), \times_{\tau}(\tau(y), \tau(z))) = \times_{\tau}(\tau(x), \tau(z)) + \times_{\tau}(\tau(y), \tau(z)) + 1 = \tau(x) \cdot \tau(z) + \tau(y) \cdot \tau(z) + 1$$

logo, $\tau(l) > \tau(r)$, pois:

$$\tau(l) - \tau(r) = \tau(x) \cdot \tau(z) + \tau(y) \cdot \tau(z) + \tau(z) - (\tau(x) \cdot \tau(z) + \tau(y) \cdot \tau(z) + 1) = \tau(z) - 1 = 1, \text{ pois } \tau(z) = 2.$$

Portanto, R termina.

Mas, queremos mostrar que R termina módulo $AC!$, ou seja, queremos mostrar que não existe seqüência infinita $t_1 \xrightarrow{\overline{AC}} \circ \xrightarrow{\overline{R}} \circ \xrightarrow{\overline{AC}} t_2 \cdots$. Ora, mas se R termina isto significa que não existe seqüência infinita $t_1 \xrightarrow{\overline{R}} t_2 \xrightarrow{\overline{R}} \cdots$. Observe que se $t \xrightarrow{\overline{R}} s$ então $\tau(t) > \tau(s)$ pois $\mathcal{T}$ é monotônica e estável.

Ainda, τ preserva seus valores em relação às leis associativa e comutativa, logo, se $t \xrightarrow{\overline{AC}} s$ então $\tau(t) = \tau(s)$.

Portanto, se $t \xrightarrow{\overline{AC}} \circ \xrightarrow{\overline{R}} \circ \xrightarrow{\overline{AC}} s$ então $\tau(t) > \tau(s)$, e conseqüentemente, não existe seqüência infinita $t_1 \xrightarrow{\overline{AC}} \circ \xrightarrow{\overline{R}} \circ \xrightarrow{\overline{AC}} t_2 \xrightarrow{\overline{AC}} \circ \xrightarrow{\overline{R}} \circ \xrightarrow{\overline{AC}} \cdots$.

Portanto, R termina módulo AC .

[Der87]

Capítulo 6

Completação de Sistemas de Reescrita de Termos

Vimos, no capítulo 5, que a propriedade de terminação de um SRT R garante que não existe R -derivações infinitas, isto é, todo termo t em $\mathcal{T}_R$ possui uma forma normal em relação a $\rightarrow_R$, se R for $\mathfrak{noetheriano}$. Entretanto, a propriedade de terminação de SRT's não garante a unicidade da forma normal de um termo.

Por outro lado, no capítulo 3, caracterizamos SRT's confluente. Num SRT R confluente, se t é um termo tal que t pode ser reescrito como dois outros termos t' e t'' , então existe um termo s tal que t' e t'' podem ser reescritos como s , através de aplicações de regras de reescrita de R .

Para que possamos garantir que todo termo t de um SRT R possua uma forma normal e que esta seja única, é necessário e suficiente que R seja $\mathfrak{noetheriano}$ e confluente e, neste caso, R é dito canônico ou completo.

Em 1942, Newman [New42] mostrou que se um SRT R é $\mathfrak{noetheriano}$ então para que R seja confluente, basta que R seja localmente confluente. Isto é, em SRT's $\mathfrak{noetherianos}$ as propriedades de confluência e confluência local são equivalentes.

Finalmente, com um artigo escrito em 1967 e publicado apenas em 70, Knuth e Bendix [KB70] mostraram que a propriedade de confluência local de um SRT pode ser caracterizada através da convergência de *pares críticos* obtidos pela superposição de pares de regras de reescrita deste SRT.

A seguir, apresentamos as demonstrações do Teorema de Newman (citado por muitos autores como o *Lema do Diamante*) e do Teorema de Knuth e Bendix. Estas demonstrações foram baseadas em Huet [Hue80].

Apresentamos, a seguir, o Princípio de Indução Noetheriana introduzido por Cohn [Coh65] e utilizado por Huet [Hue80] na sua elegante demonstração do Teorema de Newman.

Definição 6.1. (Vizinhança Transitiva) *Seja $\rightarrow_R$ uma relação de reescrita. O conjunto $\Delta^+(t) = \{s : t \xrightarrow{+}_R s\}$ é chamado de vizinhança transitiva de t .*

[Hue80]

Definição 6.2. *Seja P um predicado sobre os termos em $\mathcal{T}_R$. Dizemos que P é $\rightarrow_R$ -completo se e somente se para qualquer t em $\mathcal{T}_R$*

se $P(s)$ para todo s em $\Delta^+(t)$ então $P(t)$.

[Hue80]

Isto é, um predicado é $\rightarrow_R$ -completo se e somente se a validade do predicado para todos os elementos a vizinhança transitiva de t implica na validade do predicado para t .

Princípio de Indução Noetheriana: Se $\rightarrow_R$ é uma relação de reescrita noetheriana, então todo predicado $\rightarrow_R$ -completo é universal, isto é, para todo t $P(t)$.

[Hue80]

Teorema 6.3. (Newman (1942)) *Um SRT R noetheriano é confluyente se e somente se ele for localmente confluyente.*

[Hue80]

Demonstração: ($\Rightarrow$ /somente se)

Hipótese: $\rightarrow_R$ é noetheriana e confluyente.

Tese: $\rightarrow_R$ é localmente confluyente.

Trivial, pois se $t \xrightarrow{+}_R t_1$ e $t \xrightarrow{+}_R t_2$ então $t \xrightarrow{*}_R t_1$ e $t \xrightarrow{*}_R t_2$.

($\Leftarrow$ /se)

Hipótese: $\rightarrow_R$ é noetheriana e localmente confluyente.

Tese: $\rightarrow_R$ é confluyente.

Vamos provar por indução noetheriana que o predicado:

$$P(s) : \forall y, z \quad (s \xrightarrow{*}_R y \text{ e } s \xrightarrow{*}_R z) \Rightarrow y \xrightarrow{R} z$$

é $\rightarrow_R$ -completo. Pela hipótese do teorema, temos que $\exists m, n \in \mathbb{N} : s \xrightarrow{m}_R y$ e $s \xrightarrow{n}_R z$, pois $\rightarrow_R$ é noetheriana. Vamos admitir, por hipótese de indução, que o predicado vale para todo elemento

de $\Delta^+(s)$. Queremos mostrar que P é $\xrightarrow{R}^*$ -completo, ou seja, queremos mostrar que vale $P(s)$, isto é, se $(s \xrightarrow{R}^* y \text{ e } s \xrightarrow{R}^* z)$ então $y \xrightarrow{R}^* z$.

Queremos então mostrar que existe um termo t tal que $y \xrightarrow{R}^* t$ e $z \xrightarrow{R}^* t$.

Base da indução: Se $m = 0$ ou $n = 0$. Ora, se $m = 0$ então $t = z$, e se $n = 0$ então $t = y$.

Agora, sejam y_1 e z_1 tais que $s \xrightarrow{R} y_1 \xrightarrow{R}^* y$ e $s \xrightarrow{R} z_1 \xrightarrow{R}^* z$. Por confluência local, $\exists u : y_1 \xrightarrow{R}^* u$ e $z_1 \xrightarrow{R}^* u$. Como $y_1 \xrightarrow{R}^* y$ e $y_1 \xrightarrow{R}^* u$, então, pela hipótese de indução, $\exists v : y \xrightarrow{R}^* v$ e $u \xrightarrow{R}^* v$, (pois $y_1 \in \Delta^+(s)$). Mas como $z_1 \xrightarrow{R}^* v$ e $z_1 \xrightarrow{R}^* z$, então, pela hipótese de indução, $\exists t : v \xrightarrow{R}^* t$ e $z \xrightarrow{R}^* t$, (pois $u \in \Delta^+(s)$).

Portanto, $y \xrightarrow{R}^* t$ e $z \xrightarrow{R}^* t$, como esquematizado no diagrama 6.1 abaixo:

Figura 6.1: Diamante de Newman.

■

Pelo Teorema de Newman, para verificarmos a propriedade canônica em SRT's noetherianos precisamos mostrar que todo termo t converge localmente, mas há uma quantidade infinita de termos a serem testados e conseqüentemente o Teorema de Newman não se traduz num método efetivo para verificarmos a completeza de tais sistemas de reescrita de termos.

Finalmente, em 1967, Knuth e Bendix, a partir do Teorema de Newman, conseguem um resultado importantíssimo que fornece um método efetivo (o Teorema de Knuth e Bendix) para a verificação da propriedade canônica em SRT's noetherianos.

Eles notaram que se um termo t é tal que $t \xrightarrow{R} t'$ e $t \xrightarrow{R} t''$ então é porque existem regras de reescrita em R , digamos $l_1 \rightarrow r_1$ e $l_2 \rightarrow r_2$ (eventualmente $\langle l_1, r_1 \rangle = \langle l_2, r_2 \rangle$) tais que há subtermos t_1 e t_2 em t que são instâncias de l_1 e l_2 . Suponha, sem perda de generalidades, que $t_1 = l_1\sigma$ e $t_2 = l_2\theta$ e $t' = t[u \leftarrow r_1\sigma]$ e $t'' = t[v \leftarrow r_2\theta]$. Sendo assim, duas situações são possíveis:

Figura 6.2: Confluência local na reescrita de um termo quando os redex's são disjuntos.

Situação (A): $u \perp v$, isto é, t_1 não é subtermo de t_2 e t_2 não é subtermo de t_1 . Neste caso, é fácil ver que existe termo s tal que $t' \xrightarrow{R} s$ e $t'' \xrightarrow{R} s$. Ou seja, a convergência local está assegurada. O diagrama da figura 6.2 ilustra esta situação.

Exemplo 6.4 (Caso (A)). Considere o seguinte SRT $R = \{a \rightarrow_1 c, b \rightarrow_2 d\}$ e considere o termo $f(a, b)$ que queremos reescrever. $f(a, b)$ possui dois subtermos disjuntos que podem ser reescritos por R , são eles: a e b . Podemos reescrever $f(a, b)$ para sua forma normal de uma das duas maneiras abaixo:

$$1. f(a, b) \xrightarrow{R_1} f(c, b) \xrightarrow{R_2} f(c, d).$$

ou

$$2. f(a, b) \xrightarrow{R_2} f(a, d) \xrightarrow{R_1} f(c, d).$$

[Pla94]

Situação (B): $u \preceq v$. Neste caso t_2 é subtermo de t_1 e, conseqüentemente, $\exists w$ tal que $v = u \cdot w$. Aqui, temos duas possibilidades:

Situação (B1): t_2 corresponde a uma posição de variável do lado esquerdo (l_2) da regra de reescrita ($l_2 \rightarrow_2 r_2$) de R que reescreve t para t'' e, novamente, a convergência local está garantida. O diagrama 6.3 ilustra essa situação.

Figura 6.3: A confluência local é garantida quando um redex é ancestral do outro e este outro corresponde a uma posição de variável da regra de reescrita.

Exemplo 6.5 (Caso (B1)). Considere o seguinte SRT $R = \{a \rightarrow_1 b, g(x) \rightarrow_2 f(x, x)\}$ e considere o termo $g(a)$ que queremos reescrever. O termo $g(a)$ possui dois subtermos que podem ser reescritos por R , $g(a)$ e a , onde a é um subtermo de $g(a)$ e a corresponde a uma posição de variável da regra 2 de R que reescreve $g(a)$. Podemos reescrever $g(a)$ para sua forma normal de uma das duas maneiras:

$$1. g(a) \xrightarrow{R}_1 g(b) \xrightarrow{R}_2 f(b, b).$$

ou

$$2. g(a) \xrightarrow{R}_2 f(a, a) \xrightarrow{R}_1 f(b, a) \xrightarrow{R}_1 f(b, b).$$

[Pla94]

Situação (B2): t_2 é subtermo de t_1 mas t_2 não corresponde a uma posição de variável da regra que reescreve t para t'' . Neste caso não há garantia da convergência do par $\langle t', t'' \rangle$.

Exemplo 6.6. Considere o seguinte SRT $R = \{(x+y)+z \rightarrow_1 x+(y+z), x+e \rightarrow_2 x\}$ e considere o termo $t = (x+e)+z$ que queremos reescrever. t possui dois subtermos $t_1 = x+e$ e $t_2 = (x+e)+z$ que podem ser reescritos por R e ainda t_1 é subtermo de t_2 e t_1 não corresponde a

uma posição de variável da regra 1 em R que reescreve t_2 . Podemos reescrever t de uma das duas maneiras:

$$1. t = (x + e) + z \xrightarrow{R}_1 x + (e + z) = t'.$$

ou

$$2. t = (x + e) + z \xrightarrow{R}_2 x + y = t''.$$

t' e t'' são formas normais distintas de t . Portanto R não é localmente confluyente.

[Pla94]

Definição 6.7. A restrição de uma substituição θ ao subconjunto V de $\mathcal{V}$, denotada como θ_V é definida por:

$$x\theta_V = \begin{cases} x\theta & \text{se } x \in V; \\ x & \text{caso contrário.} \end{cases}$$

[Hue80]

Segue da definição 6.7 que $t\theta_V = t\theta$ quando $\text{Var}(t) \subseteq V$.

Definição 6.8. (Par Crítico) Sejam $\langle l_1, r_1 \rangle$ e $\langle l_2, r_2 \rangle$ duas regras de reescrita de um SRT R . Seja $u \in \text{dom } l_1$ tal que $t = l_1/u \notin \mathcal{V}$ (isto é, o subtermo de l_1 localizado na posição u não é uma simples variável) e t e l_2 sejam unificáveis. Seja t' um variante do termo unificador mais geral de t e l_2 tal que $\text{Var}(t') \cap \text{Var}(l_1) = \emptyset$. Sejam θ_1 e θ_2 tais que $t\theta_1 = t'$, $l_2\theta_2 = t'$ e $\theta_{1_{\text{Var}(t)}} = \theta_1$ e $\theta_{2_{\text{Var}(l_2)}} = \theta_2$.

O par crítico determinado pela superposição de $\langle l_2, r_2 \rangle$ sobre $\langle l_1, r_1 \rangle$ em u é o par:

$$\langle l_1\theta_1[u \leftarrow r_2\theta_2], r_1\theta_1 \rangle.$$

[dAG87]

Exemplo 6.9. a) Sejam as regras de reescrita:

$$\langle l_1, r_1 \rangle = \langle f(x, g(y, h(z))), k(x, y, z) \rangle$$

$$\langle l_2, r_2 \rangle = \langle g(a, w), t(w, a) \rangle$$

e a ocorrência $u \in \text{dom } l_1$ tal que $u = 2$. Portanto, $t = l_1/u = g(y, h(z))$. A imagem de t e de l_2 pela unificadora mais geral θ de t e de l_2 é $g(a, h(w))$, onde $y\theta_1 = a$, $z\theta_1 = w$ e $w\theta_2 = h(w)$. Logo, o par crítico determinado pela superposição de $\langle l_2, r_2 \rangle$ sobre $\langle l_1, r_1 \rangle$ em u é $\langle f(x, t(h(w), a)), k(x, a, w) \rangle$.

b) *Sejam as regras de reescrita:*

$$\langle l_1, r_1 \rangle = \langle f(g(h(x, y, z))), h(x, f(y), g(z)) \rangle$$

$$\langle l_2, r_2 \rangle = \langle h(a, b, w), f(w) \rangle$$

e a ocorrência $u \in \text{dom } l_1$ tal que $u = \langle 1, 1 \rangle$. Portanto, $t = l_1/u = h(x, y, z)$. A imagem de t e de l_2 pela unificadora mais geral θ de t e de l_2 é $h(a, b, w)$, onde $x\theta_1 = a, y\theta_1 = b$ e $z\theta_1 = w$. Logo, o par crítico determinado pela superposição de $\langle l_2, r_2 \rangle$ sobre $\langle l_1, r_1 \rangle$ em u é $\langle f(g(f(w))), h(a, f(b), g(w)) \rangle$.

c) *Sejam as regras de reescrita:*

$$\langle l_1, r_1 \rangle = \langle f(g(x, y), g(y, z)), f(x, z) \rangle$$

$$\langle l_2, r_2 \rangle = \langle g(h(w), a), h(w) \rangle$$

e a ocorrência $u \in \text{dom } l_1$ tal que $u = \langle 1 \rangle$. Portanto, $t = l_1/u = g(x, y)$. O par crítico determinado pela superposição de $\langle l_2, r_2 \rangle$ sobre $\langle l_1, r_1 \rangle$ em u é $\langle f(h(w), g(a, z)), f(h(w), z) \rangle$, enquanto que $u = \langle 2 \rangle$ temos o par crítico $\langle f(g(x, h(w)), h(w)), f(x, a) \rangle$.

[dAG87]

Exemplo 6.10. a) *Considere as seguintes regras de reescrita:*

$$\langle l_1, r_1 \rangle = \langle f(x(g(x, a)), h(x)) \rangle$$

$$\langle l_2, r_2 \rangle = \langle g(b, y), k(y) \rangle$$

e a ocorrência $u \in \text{dom } l_1$ tal que $u = \langle 2 \rangle$. Portanto, $t = l_1/u = g(x, a)$. A u.m.g. de l e l_2 é $\theta = \{x \leftarrow b, y \leftarrow a\}$, onde $\theta_1(x) = b$ e $\theta_2(y) = a$. Logo, o par crítico determinado pela superposição de $\langle l_2, r_2 \rangle$ sobre $\langle l_1, r_1 \rangle$ em u é $\langle f(b, k(a)), h(b) \rangle$.

b) *Considere as regras de reescrita:*

$$\langle l_1, r_1 \rangle = \langle f(x, h(y)), k(y, x) \rangle$$

$$\langle l_2, r_2 \rangle = \langle h(g(z, w)), l(z, w) \rangle$$

e a ocorrência $u \in \text{dom } l_1$ tal que $u = \langle 2 \rangle$. Os termos $h(y)$ e $h(g(z, w))$ são unificáveis através da u.m.g. $\theta = \{y \leftarrow g(z, w)\}$. E temos $h(y)\theta = h(g(z, w))\theta = h(g(z, w))$, onde $\theta_1 = \theta$ e $\theta_2 = \emptyset$. Logo, o par crítico determinado pela superposição de $\langle l_2, r_2 \rangle$ sobre $\langle l_1, r_1 \rangle$ em u é $\langle f(x, l(z, w)), k(g(z, w), x) \rangle$.

c) *Considere a regra de reescrita:*

$$\langle l, r \rangle = \langle h(h(x)), k(x) \rangle$$

O termo $h(h(x))$ possui o subtermo próprio $h(x)$.

$h(x')$ (que é uma variante de $h(x)$) unifica-se com $h(h(x))$ através da u.m.g. $\theta = \{x \leftarrow h(x')\}$, onde $\theta_1 = \emptyset$ e $\theta_2 = \theta$. O seguinte par crítico: $\langle h(k(x')), k(h(x')) \rangle$ é gerado.

[Hue80]

Proposição 6.11. *Seja $\rightarrow_R$ uma relação de reescrita sobre os termos em $\mathcal{T}_R$, t um termo de $\mathcal{T}_R$, x uma variável de $\mathcal{V}$, $u_1, u_2, \dots, u_n \in \text{dom } t$ todas as distintas ocorrências de x em t , e θ, σ substituições tais que:*

$$\begin{aligned} x\theta &\rightarrow_R x\sigma \\ y\theta &= y\sigma \quad \forall y \neq x \end{aligned}$$

E definindo $t_0 = t\theta$ e $t_i = t_{i-1}[u_i \leftarrow x\sigma]$, para $1 \leq i \leq n$, temos:

$$t_i \xrightarrow[n-i]{_R} t\sigma, \text{ para } 0 \leq i \leq n.$$

[Hue80]

Demonstração: Como $u_1, \dots, u_n$ são ocorrências da variável x em t , então são todas disjuntas, isto é, $u_i \perp u_j \quad \forall i \neq j$, logo $t_i/t_j = x\theta \quad \forall j > i$. A representação, pela figura 6.4, nos auxilia na compreensão da proposição.

Figura 6.4: As ocorrências de um subtermo são, duas a duas, disjuntas quando o subtermo corresponde a uma variável.

Note que $t_0 \xrightarrow{[u_1]}_R t_1$, pois por hipótese, $x\theta \rightarrow_R t\sigma$ e $\rightarrow_R$ é fechada sob o contexto de aplicação em $\mathcal{T}_R$. Seguindo o mesmo raciocínio, temos: $t_1 \xrightarrow{[u_2]}_R t_2 \xrightarrow{[u_3]}_R \dots \xrightarrow{[u_n]}_R t_n$ ou seja, $t_i \xrightarrow[n-i]{_R} t_n$.

A proposição estará provada se mostrarmos que vale a seguinte igualdade $t_n = t\sigma$, o que faremos por indução no número n de ocorrências de x em t .

A igualdade vale para a base da indução ($n = 0$), pois, neste caso, $t_0 = t\theta = t\sigma$, uma vez que $y\sigma = y\theta, \forall y \neq x$.

Suponha que a igualdade vale para termos com $n - 1$ ocorrências de x , isto é, $t_{n-1} = t\sigma$. Vamos provar que vale para n ocorrências de x . Seja, então, um termo t com n ocorrências de x . Mas,

$t_n \stackrel{\text{def.}}{=} t_{n-1}[u_n \leftarrow x\sigma] \stackrel{\text{hip. indução}}{=} t\sigma[u_n \leftarrow x\sigma] \stackrel{\text{lema 2.49iii)}}{=} t[u_n \leftarrow x]\sigma = t\sigma$. A última igualdade segue do fato que $t/u_n \stackrel{\text{hip.}}{=} x$ ■

Proposição 6.12. *Sejam $\langle l_1, r_1 \rangle, \langle l_2, r_2 \rangle \in R$. Seja $u \in \text{dom } l_1$ tal que $t = l_1/u \notin \mathcal{V}$ e existam θ_1, θ_2 tal que:*

$$t\theta_1 = l_2\theta_2.$$

Então existe um par crítico $\langle t_1, t_2 \rangle$ em R e uma substituição σ tais que:

$$t_1\sigma = l_1\theta_1[u \leftarrow r_2\theta_2]$$

$$t_2\sigma = r_1\theta_1.$$

[Hue80]

Demonstração: Claro que t e l_2 são unificáveis. Seja t' um variante do termo unificador mais geral de t e l_2 tal que $\text{Var}(t') \cap \text{Var}(l_1) = \emptyset$. Sejam ϕ_1 e ϕ_2 , tais que, $t\phi_1 = t', l_2\phi_2 = t'$ e $\phi_1|_{\text{Var}(t)} = \phi_1$ e $\phi_2|_{\text{Var}(l_2)} = \phi_2$, de forma que o mecanismo de superposição nos dá o par crítico $\langle t_1, t_2 \rangle$ onde $t_1 = l_1\phi_1[u \leftarrow r_2\phi_2]$ e $t_2 = r_1\phi_1$.

Sejam as substituições θ e σ , tais que, $l_2\theta_2 = t'$ (ou equivalentemente $t\theta_1 = t'$) e $\sigma = [\theta|_{\text{Var}(t')}] \cup [\theta_1|_{\text{Var}(l_1)}]$ (vale lembrar que $\text{Var}(t') \cap \text{Var}(l_1) = \emptyset$). Por construção, temos $(t\phi_1)\theta = t'\theta = l_2\theta_2 = t\theta_1 = \underbrace{t\phi_1}_{t'}\theta$, onde:

i) $\forall x \in \text{Var}(t), x\theta_1 = (x\phi_1)\theta = (x\phi_1)\sigma$, a última igualdade é válida pois $\text{Var}(x\phi_1) \subseteq \text{Var}(t')$:

ii) $\forall x \in \text{Var}(l_1) - \text{Var}(t), x\phi_1 = x$, pois $\text{Lvar}(\phi_1) \subseteq \text{Var}(t)$; ainda por construção de σ temos $x\sigma = x\theta_1$, logo $x\theta_1 = x\sigma = (x\phi_1)\sigma$.

Logo, por i) e ii), $\forall x \in \text{Var}(l_1), \boxed{x\theta_1 = (x\phi_1)\sigma}^{(*)}$. Analogamente, por construção, $\boxed{l_2\theta_2 = (x\phi_2)\sigma}^{(**)}$.

E, uma vez que $\text{Var}(r_1) \subseteq \text{Var}(l_1)$ e $\text{Var}(r_2) \subseteq \text{Var}(l_2)$, temos: $\boxed{r_1\theta_1 \stackrel{(*)}{=} r_1\phi_1\sigma = t_2\sigma}$

e por fim, $\boxed{l_1\theta_1[u \leftarrow r_2\theta_2] \stackrel{(*)e(**)}{=} (l_1\phi_1)\sigma[u \leftarrow (r_2\phi_2)\sigma] \stackrel{\text{lema 2.49iii}}{=} \underbrace{l_1\phi_1[u \leftarrow r_2\phi_2]\sigma}_{t_1} = t_1\sigma}$. ■

Teorema 6.13. (Knuth e Bendix - 1967) *Dado um SRT R , $\xrightarrow{R}$ é localmente confluyente se e somente se para todo par crítico $\langle t_1, t_2 \rangle$ de R , $t_1 \downarrow_R t_2$.*

[KB70]

Demonstração: ($\Rightarrow$ /somente se)

Hipótese: $\xrightarrow{R}$ é localmente confluyente.

Trivial, pois os dois termos t_1 e t_2 de um par crítico derivam (pela definição de par crítico) de um mesmo termo, que é uma instância do lado esquerdo de alguma regra de reescrita de R , onde por hipótese, $t_1 \xrightarrow{R} t_2$.

($\Leftarrow$ /se)

Hipótese: Para todo par crítico $\langle t_1, t_2 \rangle$ de R , vale $t_1 \xrightarrow{R} t_2$.

Seja t um termo, $\langle l_1, r_1 \rangle$ e $\langle l_2, r_2 \rangle$ regras de R , $u_1, u_2 \in \text{dom } t$, e θ_1, θ_2 substituições tais que:

$$\begin{aligned} t/u_1 &= l_1\theta_1 \quad \text{e} \quad t \xrightarrow{R[u_1]} t[u_1 \leftarrow r_1\theta_1] (= t') \\ t/u_2 &= l_2\theta_2 \quad \text{e} \quad t \xrightarrow{R[u_2]} t[u_2 \leftarrow r_2\theta_2] (= t'') \end{aligned}$$

Segundo os redex's u_1 e u_2 temos duas possibilidades:

a) $u_1 \perp u_2$ (redex's disjuntos (independentes)):

Por persistência, temos $t'/u_2 = l_2\theta_2$ e $t''/u_1 = l_1\theta_1$.

Por comutatividade, temos $\hat{t} = t'[u_2 \leftarrow r_2\theta_2] = t''[u_1 \leftarrow r_1\theta_1]$

e assim, $t' \xrightarrow{R[u_2]} \hat{t}$ e $t'' \xrightarrow{R[u_1]} \hat{t}$, logo $t' \xrightarrow{R} t''$.

b) um redex é prefixo (ancestral) do outro:

Sem perda de generalidades, vamos supor $u_1 \prec u_2$.

Seja $v = u_2/u_1$ (isto é, $u_2 = u_1 \cdot v$).

Mas, $t/u_1 = l_1\theta_1$, logo $(t/u_1)/v = l_1\theta_1/v$, mas $v = u_2/u_1$ e como $u_1 \prec u_2$ temos:

$(t/u_1)/(\underbrace{u_2/u_1}_v) \stackrel{\text{cancela/o}}{\stackrel{\text{lema 2.35i.}}{=}} t/u_2 = l_1\theta_1/v = l_2\theta_2$, a última igualdade se verifica pois $t/u_2 = l_2\theta_2$.

E, $t''/u_1 = t[u_2 \leftarrow r_2\theta_2]/u_1 \stackrel{\text{distributiva}}{\stackrel{\text{lema 2.35ii.}}{=}} \underbrace{t/u_1}_{l_1\theta_1} [\underbrace{u_2/u_1}_v \leftarrow r_2\theta_2] = l_1\theta_1[v \leftarrow r_2\theta_2]$.

Precisamos mostrar que existe $\hat{t}$ tal que $\theta_1(r_1) \xrightarrow{*}_R \hat{t}$ e $t''/u_1 \xrightarrow{*}_R \hat{t}$. Pois assim teremos, pelo fato de $\xrightarrow{R}$ ser fechada sob contexto de aplicação, se $t'/u_1 \xrightarrow{*}_R \hat{t}$ então $t' \xrightarrow{*}_R t'[u_1 \leftarrow \hat{t}]$.

Novamente, por contexto de aplicação de $\xrightarrow{R}$ temos, se $t''/u_1 \xrightarrow{*}_R \hat{t}$ então $t'' \xrightarrow{*}_R t''[u_1 \leftarrow \hat{t}]$.

Mas $t'[u_1 \leftarrow \hat{t}] = t''[u_1 \leftarrow \hat{t}]$, ou seja, se $r_1\theta_1 = t'/u_1 \xrightarrow{*}_R \hat{t}$ e $t''/u_1 \xrightarrow{*}_R \hat{t}$ então $t' \downarrow^R t''$. A figura 6.5 abaixo ilustra a situação deste caso:

Figura 6.5: Diagrama detalhado de confluência local quando um redex é ancestral do outro.

Agora, $l_2\theta_2$ é subtermo de $l_1\theta_1$, então podemos ter apenas uma das duas situações b₁) ou b₂) abaixo. As representações em árvore dos termos que apresentamos nas figuras 6.6 e 6.7, ilustram respectivamente os subcasos b₁) e b₂).

b₁) $v \notin \text{sdom } l_1$:

Neste caso, $v = v_1 \cdot v_2$, onde v_1 é tal que $l_1/v_1 = x \in \mathcal{V}$. Eventualmente $v_2 = \langle \rangle$, e se isto ocorrer então $v = v_1 \cdot \langle \rangle = v_1$ simplesmente. Pelo corolário 2.50 do lema 2.49 ii) temos:

$$l_1\theta_1 / \underbrace{(v_1 \cdot v_2)}_v = x\theta_1/v_2, \text{ mas } l_1\theta_1 / (v_1 \cdot v_2) = l_2\theta_2, \text{ logo } l_2\theta_2 = x\theta_1/v_2.$$

Note que, neste caso, $l_2\theta_2$ é um subtermo de t que não pertence à estrutura de l_1 .

Seja a seguinte substituição: $\theta = \begin{cases} x\theta' = x\theta_1[v_2 \leftarrow r_2\theta_2] \\ y\theta' = y\theta_1 \end{cases} \quad \forall y \neq x$

Seja $\hat{t} = r_1\theta'$. É claro que $x\theta_1 \xrightarrow{*}_R x\theta'$ (aplicando a regra $\langle l_2, r_2 \rangle$, pois $x\theta_1/v_2 = l_2\theta_2$ e $x\theta' = x\theta_1[v_2 \leftarrow r_2\theta_2]$).

Pela proposição 6.11 (fazendo na proposição, $\sigma = \theta'$, $\theta = \theta_1$ e $t = r_1$), teremos:

$$t_0 = r_1\theta_1 \xrightarrow{*}_R r_1\theta' \stackrel{\text{def.}}{=} \hat{t}$$

e novamente pela proposição 6.11 (fazendo na proposição, $\sigma = \theta'$, $\theta = \theta_1$ e $t = l_1$), teremos:

$$t_0 = l_1\theta_1 \xrightarrow{*}_R l_1\theta'; \text{ mas } x\theta' = x\theta_1[v_2 \leftarrow r_2\theta_2], \text{ logo, } t''/u_1 \stackrel{\text{caso b}}{=} l_1\theta_1[v \leftarrow r_2\theta_2] \xrightarrow{*}_R l_1\theta'. \text{ Finalmente,}$$

$l_1\theta' \xrightarrow{*}_R r_1\theta' = \hat{t}$, uma vez que $\rightarrow$ é fechada sob instanciação. Portanto $t' \downarrow^R t''$.

Figura 6.6: $l_2\theta_2$ não pertence à estrutura de l_1 .

b₂) $v \in \text{sdom } l_1$:

Neste caso, $l_1/v \notin \mathcal{V}$, então pelo lema 2.49 ii) temos:

$l_1\theta_1/v = (l_1/v)\theta_1$, mas $l_1\theta_1/v = l_2\theta_2$, logo $(l_1/v)\theta_1 = l_2\theta_2$.

Pela proposição 6.12 existe um par crítico $\langle t_1, t_2 \rangle$ e uma substituição σ tal que

$t_1\sigma = l_1\theta_1[v \leftarrow r_2\theta_2] = t''/u_1$ e $t_2\sigma = r_1\theta_1$. Por hipótese, existe t_3 tal que $t_1 \xrightarrow{*}_R t_3$ e $t_2 \xrightarrow{*}_R t_3$. Mas, $\xrightarrow{*}_R$ é fechada sob instanciação, logo $t_1\sigma \xrightarrow{*}_R t_3\sigma$ e $t_2\sigma \xrightarrow{*}_R t_3\sigma$. Chame $t_3\sigma$ de $\hat{t}$, isto é, $\hat{t} = t_3\sigma$. Note que, $t_1\sigma = t''/u_1$. Portanto, temos $t''/u_1 \xrightarrow{*}_R \hat{t}$ e $r_1\theta_1 = t'/u_1 \xrightarrow{*}_R \hat{t}$.

Portanto $t' \downarrow^R t''$. ■

Corolário 6.14. Um SRT R *nætheriano* é confluente se e somente se para todo par crítico $\langle t_1, t_2 \rangle$ de R , $t_1 \downarrow^R t_2$.

[Hue80]

Demonstração: Conseqüência imediata do Teorema de Newman e do Teorema de Knuth e Bendix. ■

O lema 6.16, a seguir, nos mostra que a propriedade de confluência de um SRT é equivalente a uma outra propriedade, a Church-Rosser, definida a seguir. Este fato, como veremos, é de extrema importância para que possamos usar o Teorema de Knuth e Bendix para calcular, de modo automático, os teoremas de uma teoria equacional.

Figura 6.7: $l_2\theta_2 = (l_1\theta_1)/v$.

Definição 6.15. (Propriedade Church-Rosser) Um SRT R possui a propriedade Church-Rosser se e somente se para quaisquer termos s, t em $\mathcal{T}_R$, se $s \xrightarrow{*}_R t$ então $s \downarrow^R t$.

[DJ90]

Figura 6.8: Propriedade Church-Rosser.

Lema 6.16. Um SRT R possui a propriedade Church-Rosser se e somente se R for confluenta.

[Buc87]

Pelo lema 6.16, para garantirmos que sempre que $s \xrightarrow{*}_R t$ então s e t possuem uma mesma forma normal, basta garantirmos que R seja confluenta e ainda mais, se R for noetheriano, então pelo Teorema de Newman basta apenas que R seja localmente confluenta para que a propriedade Church-Rosser seja garantida.

Em outros termos, num SRT R que seja noetheriano e localmente confluenta, ou seja R é completo, podemos verificar se $s \xrightarrow{*}_R t$ obtendo as formas normais de s e t segundo R e verificando se elas são idênticas, pois neste caso sempre que $s \xrightarrow{*}_R t$ teremos $R(s) = R(t)$.

Dado um conjunto E de equações, estaremos interessados em obter um SRT R completo tal que $s \xrightarrow{*}_R t$ se e somente se $s \equiv_E t$, pois desta forma teremos obtido um método efetivo para a solução do problema da palavra em E , isto é, um algoritmo que, dados quaisquer dois termos s e t em $\mathcal{T}_E$, decida se s e t são ou não congruentes módulo E , ou em outras palavras, decida

se $E \vdash s = t$ ou não.

Se R é um SRT e E um conjunto de equações então, para garantirmos que $\overset{*}{\underset{R}{\rightarrow}} = \overline{\underset{E}{=}}$, é necessário e suficiente que as duas condições abaixo sejam satisfeitas:

- (1) se $s = t \in E$ então $R(s) = R(t)$;
- (2) se $s \xrightarrow{R} t \in R$ então $s \overline{\underset{E}{=}} t$.

A condição (1) garante $\overline{\underset{E}{=}} \subseteq \overset{*}{\underset{R}{\rightarrow}}$ enquanto a condição (2) garante $\overset{*}{\underset{R}{\rightarrow}} \subseteq \overline{\underset{E}{=}}$.

Knuth e Bendix [KB70] propuseram um método efetivo para a obtenção de um SRT capaz de solucionar o problema da palavra para algumas teorias equacionais. O algoritmo de Knuth e Bendix recebe como dados de entrada um conjunto E de equações e encontra de maneira incremental um SRT R completo tal que $\overline{\underset{E}{=}} = \overset{*}{\underset{R}{\rightarrow}}$. O algoritmo vai retirando equações de E e colocando-as em R na forma de regras de reescrita dando uma orientabilidade às equações de tal modo que se $s \rightarrow t \in R$ então $Var(s) \supseteq Var(t)$, esta restrição quanto as variáveis é necessária pois o teorema de Knuth e Bendix é demonstrado assumindo tal restrição para as variáveis das regras de reescrita.

Sejam E_i e R_i respectivamente o conjunto de equações e o SRT obtidos exatamente após a execução da i -ésima iteração do algoritmo. Inicialmente $i := 0$ e o algoritmo gera dois conjuntos, a saber, $E_0 := E$ e $R_0 := \emptyset$. Ao término da execução, se efetuada com sucesso, o algoritmo retorna os conjuntos $E_{k-1} = \emptyset$ e R_k sendo o SRT completo tal que $s \overline{\underset{E}{=}} t$ se e somente se $s \overset{*}{\underset{R}{\rightarrow}} t$, onde k é o número de iterações realizadas pelo algoritmo.

Na $i + 1$ -ésima iteração do algoritmo, uma equação $s = t$ é obtida de E_i ($0 \leq i \leq k - 1$) e tenta-se inserir as formas normalizadas relativas aos termos s e t em R_{i+1} caso estas não sejam idênticas e ainda se a condição de restrição sobre o conjunto das variáveis de s e t for satisfeita.

Sendo assim, podemos ter as seguintes situações:

Caso (A): Suponha, sem perda de generalidades, que $Var(s) \supseteq Var(t)$ e $R_i(s) \neq R_i(t)$. Sendo assim, fazemos $R_{i+1} := R_i \cup \{R_i(s) \rightarrow R_i(t)\}$ e $E_{i+1} := E_i - \{s = t\}$.

Caso (B): Caso ocorra $R_i(s) = R_i(t)$ isto significa que o par $\langle R_i(s), R_i(t) \rangle$ já é um elemento da congruência gerada por e portanto fazemos apenas $E_{i+1} := E_i - \{s = t\}$ e $R_{i+1} := R_i$.

Caso (C): Caso contrário, o algoritmo pára sem sucesso.

Ao final da i -ésima iteração do algoritmo, temos garantido que a congruência gerada por E é a mesma congruência gerada por $R_i \cup E_i$. Conseqüentemente, se o algoritmo termina com sucesso após a k -ésima e última iteração teremos garantido que a congruência gerada por E seja a mesma gerada por R_k , isto é, $\overline{\underset{E}{=}} = \overset{*}{\underset{R_k}{\rightarrow}}$.

Se na i -ésima iteração uma regra de reescrita $l \rightarrow r$ é inserida em R_i ($0 \leq i \leq k-1$) então gera-se todos os pares críticos possíveis em R_i a partir da superposição do lado esquerdo da nova regra inserida com os lados esquerdos das demais regras (inclusive ela própria). Os pares críticos que convergem em relação à R_i são descartados e os demais são inseridos em E_i na forma de equações para futura consulta. Assim ao final da i -ésima iteração do algoritmo, a convergência de todos os pares críticos de R_i é garantida.

Quando o algoritmo termina com sucesso (tendo efetuado k iterações) não há mais regras de reescrita a serem inseridas em R_k e se isto ocorre é porque o conjunto de equações “esvaziou-se” totalmente e todos os pares críticos de R_k convergem e portanto pelo Teorema de Knuth e Bendix a propriedade de confluência local de R_k fica assegurada.

A propriedade noetheriana de cada R_i é verificada e portanto R_k é noetheriano.

Logo, se o algoritmo pára com sucesso, R_k é localmente confluyente e noetheriano e portanto pelo Teorema de Newman R_k é completo.

Conseqüentemente, o algoritmo de completção de Knuth e Bendix é correto (a menos da parada uniforme) no seguinte sentido: se ele pára com sucesso então R_k é de fato completo e satisfaz: $s \xrightarrow{E} t$ se e somente se $s \xrightarrow{R}^* t$.

Observamos que a terminação do algoritmo não é assegurada, devido ao fato de a cardinalidade de E_i não ser necessariamente menor que a cardinalidade de E_j , com $j < i$, ou seja, o “esvaziamento” dos conjuntos das equações ocorre de modo não monotônico, pode, eventualmente, não convergir para um conjunto vazio de equações e, portanto, fazendo com que o algoritmo não termine sua execução.

Ainda, este é um algoritmo é semi-decisão, pois se ele não parar para um dado conjunto E de equações, isto não implica que não exista um SRT R completo tal que $\xrightarrow{R}^* = \xrightarrow{E}$. Uma demonstração detalhada da correção do algoritmo de completção de Knuth e Bendix pode ser encontrada em Huet [Hue81].

A seguir descrevemos o procedimento de completção, de Knuth e Bendix, de um modo mais sistematizado.

Algoritmo 1.3 Algoritmo de completção de Knuth e Bendix.

0: Sejam $i \in \mathbb{N}$, E_i um conjunto de equações e R_i um SRT.

faça $i := 0$; $E_0 := E$; $R_0 := \emptyset$.

1: **se** $E_i = \emptyset$ **então**

faça $R_{i+1} := R_i$; % apenas para retornar na k -ésima iteração o conjunto R_k ao invés do R_{k-1} .
 o procedimento termina com resultado R_{i+1} .

2: Escolha $t = s \in E_i$.

se $R_i(t) = R_i(s)$ **então**

faça $R_{i+1} := R_i$;
 $E_{i+1} := E_i - \{t = s\}$;
 $i := i + 1$;
 volte para 1.
 3: se $Var(R_i(s)) \subseteq Var(R_i(t))$ então
 faça $l := R_i(t)$ e $r := R_i(s)$;
 c.contrário se $Var(R_i(s)) \supsetneq Var(R_i(t))$ então
 faça $l := R_i(s)$ e $r := R_i(t)$;
 c.contrário o procedimento pára sem sucesso.
 4: faça $R_{i+1} := R_i \cup \{l \rightarrow r\}$
 5: se R_{i+1} não for noetheriano então o procedimento termina sem sucesso;
 c.contrário faça $E_{i+1} := (E_i - \{t = s\}) \cup \{u = v : \langle u, v \rangle \text{ é um par crítico de } R_{i+1}\}$;
 $i := i + 1$;
 volte para 1.

.....

Huet e Oppen [HO80] apresentam uma implementação do algoritmo de completção de Knuth e Bendix que devolve SRT's que possuem também a propriedade de terem suas regras na forma "interreduzidas", isto é, se $s \rightarrow t \in R$ então $R(s) = s$ e $R(t) = t$.

Pode-se ainda deixar o algoritmo de completção de Knuth e Bendix mais eficiente racionalizando a geração de pares críticos. Note que, na versão original do algoritmo de completção de Knuth e Bendix, se um par crítico $\langle t_1, t_2 \rangle$ é gerado pelo par de regras $\langle l_1 \rightarrow r_1, l_2 \rightarrow r_2 \rangle$ em R_i , então este mesmo par crítico continuará sendo gerado desnecessariamente nas futuras iterações do algoritmo até que este par crítico seja inserido na forma de uma regra de reescrita no R_{i+j} para algum $j \geq 1$ e $j \leq k - i$. Esta proposta de racionalização na geração dos pares críticos foi proposta originalmente por Huet [Hue81]. Em Dick [Dic91], encontra-se um bom texto introdutório ao estudo do método de completção de SRT's proposto por Knuth e Bendix. A seguir temos um exemplo de operação do algoritmo de completção de Knuth e Bendix.

Exemplo 6.17. Considere o seguinte conjunto E de equações:

$A_1:$	$0 + x = x$
$A_2:$	$-x + x = 0$
$A_3:$	$(x + y) + z = x + (y + z)$

Operação do algoritmo:

<i>Regras derivadas</i>	<i>Derivação (termos críticos)</i>	<i>Origem</i>	<i>Regras aplicadas ao</i>	
			<i>lado esq.</i>	<i>lado dir.</i>
$R_1 : 0 + x \rightarrow x$		A_1		
$R_2 : -x + x \rightarrow 0$		A_2		
$R_3 : (x + y) + z \rightarrow x + (y + z)$		A_3		
$R_4 : -x + (x + y) \rightarrow y$	$(-x + x) + y$	$R_3 R_2$	R_3	R_2, R_1
$R_5 : -0 + x \rightarrow x$	$-0 + (0 + x)$	$R_1 R_4$	R_1	R_4
$R_6 : --x + 0 \rightarrow x$	$--x + (-x + x)$	$R_2 R_4$	R_2	R_4
$R_7 : --x + y \rightarrow x + y$	$--x + (-x + (x + y))$	$R_4 R_4$	R_4	R_4
$R_8 : x + 0 \rightarrow x$	$--x + (-x + x)$	$R_2 R_4$	R_2, R_7	R_4
$R_9 : -0 \rightarrow 0$	$-0 + 0$	$R_8 R_2$	R_8	R_2
$R_{10} : --x \rightarrow x$	$--x + 0$	$R_8 R_7$	R_8	R_7, R_8
$R_{11} : x + -x \rightarrow 0$	$--x + -x$	$R_7 R_2$	R_7	R_2
$R_{12} : x + (-x + y) \rightarrow y$	$--x + (-x + y)$	$R_7 R_4$	R_7	R_4
$R_{13} : x + (y + -(x + y)) \rightarrow 0$	$(x + y) + -(x + y)$	$R_3 R_{11}$	R_3	R_{11}
$R_{14} : x + -(y + x) \rightarrow -y$	$-y + (y + (x + -(y + x)))$	$R_4 R_{13}$	R_4	R_{13}, R_8
$R_{15} : -(x + y) \rightarrow -y + -x$	$-y + (y + -(x + y))$	$R_4 R_{14}$	R_4	R_{14}

Terminou com sucesso, isto é, o algoritmo gerou um sistema de reescrita completo para o conjunto E de equações fornecido.

[Dic91]

Este algoritmo de completção de Knuth e Bendix possui limitações práticas. Por exemplo, não há relação de ordem estrita que dê orientação a uma equação que expresse comutatividade, como em $a + b = b + a$. Muitos têm sido os esforços para obter métodos que consigam completar sistemas de reescrita que possuam operadores comutativos ou associativos-comutativos [Hen91, PS81, BD87, LB77a, LB77c, LB77b].

No capítulo que se segue, apresentamos um algoritmo de completção de sistemas de reescrita de termos, uma extensão do algoritmo de Knuth e Bendix que nos permite solucionar o problema da palavra para uma subclasse de teorias onde há a presença de símbolos funcionais associativo-comutativos, proposto por Peterson e Stickel [PS81].

Veremos, no capítulo 9, que tal algoritmo nos será útil, pois ele garante a existência de um sistema de reescrita completo para os reticulados distributivos, fato este essencial, segundo nossa abordagem, para que possamos resolver o problema da verificação automática da identidade entre operadores morfológicos invariantes por translação e isotônicos.

Capítulo 7

Completação Módulo T de SRT's

Descrevemos, neste capítulo, uma extensão porposta por Peterson e Stickel [PS81], do algoritmo de completção de Knuth e Bendix, apresentado no capítulo 6.

Como já foi discutido, uma limitação do algoritmo clássico de completção é sua incapacidade de obter SRT's completos a partir de conjuntos de equações que possuam alguma equação impossível de ser orientada. Este é o caso, por exemplo, quando o conjunto de equações possui uma equação da forma $f(x, y) = f(y, x)$: tal equação nos diz que o símbolo funcional f é comutativo.

Quando E é um conjunto de equações que não pode ser orientado, é comum, na literatura corrente, mencionar o conjunto $Th(E)$ como sendo uma “teoria equacional não orientável” ou ainda, por abuso de linguagem, “ E é uma teoria equacional não orientável”.

O método de completção proposto por Peterson e Stickel visa abarcar conjuntos de equações E que possam ser divididos em duas partes, uma parte orientável e outra parte não orientável, digamos respectivamente E' e T , isto é, $E = E' \cup T$ de tal modo que:

- i) cada equação em E' possa ser orientada e escrita na forma de uma regra de reescrita de um SRT a partir do qual pode-se tentar gerar um SRT R completo e tal que $\xrightarrow{*}_R = \overline{\quad}_{E'}$;
- ii) T deve ser um conjunto de equações para qual seja conhecido um conjunto finito e completo de unificadoras- T .

Mostraremos, mais adiante, uma prova da aplicabilidade de tal algoritmo para o caso em que T é uma teoria composta por um número finito de símbolos funcionais associativos-comutativos. A implementação de tal algoritmo produziu sistemas de reescrita completos para algumas estruturas algébricas como grupos comutativos e os reticulados distributivos.

Como veremos adiante, este algoritmo aplica-se apenas a conjuntos de equações cujo

subconjunto não orientável induza classes de congruência com cardinalidades finitas. De fato, veremos que é necessário impor mais restrições (cada equação do subconjunto não orientável deve ser linear e cada variável que ocorra em um dos lados de uma equação deve também ocorrer no outro lado da mesma equação) ao conjunto de equações não orientável para que o método possa ser aplicado.

7.1 Método das Classes de Equivalência

Lankford e Ballantyne [LB77a] desenvolveram uma extensão do algoritmo de completção de Kunth e Bendix. Essa extensão passou a suportar a presença de símbolos funcionais comutativos, porém não associativos. Eles ainda generalizaram o método obtido e desenvolveram um algoritmo semi-decidível (pois não garantiam a propriedade noetheriana do sistema obtido) sendo que cada classe de equivalência era composta por termos que podiam ser mostrados iguais através da aplicação de *equações de permutação*.

Entenda-se por uma equação de permutação toda aquela na qual em cada lado da equação há os mesmos símbolos funcionais e cada símbolo funcional ocorre numa mesma quantidade em cada lado da equação. São exemplos de equações de permutação as leis comutativa e associativa. Uma propriedade das classes de equivalência produzidas por equações de permutação é que a classe de equivalência de qualquer termo é finita e há um algoritmo conhecido para gerá-las [LB77c].

As diferenças básicas entre o algoritmo proposto por Lankford e Ballantyne [LB77a] e o algoritmo de Peterson e Stickel [PS81] que iremos estudar adiante são:

- i) O algoritmo proposto por Lankford e Ballantyne é um algoritmo semi-decidível quanto a propriedade noetheriana do sistema de reescrita gerado. Já o algoritmo de Peterson e Stickel é um algoritmo decidível que gera sistemas de reescrita de termos completos;
- ii) Apenas equações de permutação podem ser usadas na definição das classes de equivalências para o algoritmo de Lankford e Ballantyne, enquanto que esta restrição não se faz presente no algoritmo de Peterson e Stickel.

Podemos ainda citar o trabalho de Huet [Hue80] que propôs um algoritmo de completção garantindo a propriedade noetheriana do sistema obtido por seu algoritmo. Para tal, ele impôs a restrição de que as regras de reescrita fossem lineares à esquerda, isto é, não possuísem variáveis duplicadas no lado esquerdo da regra.

O algoritmo proposto por Huet também é limitado no sentido que não pode ser aplicado, por exemplo, na completção de um reticulado distributivo, pois um reticulado distributivo

resulta na presença de reduções que não são lineares à esquerda. Como veremos, o algoritmo de Peterson e Stickel aplica-se a tal exemplo.

Como já mencionado anteriormente (vide capítulo 4), o problema de desenvolver um algoritmo para encontrar conjuntos de unificadoras- T completos é, em geral, uma tarefa difícil. Entretanto, tem sido propostos algoritmos para resolver o problema em alguns casos restritos, como a unificação- AC que vimos no capítulo 4.

Suponha que T seja um conjunto de equações e s e t termos em $\mathcal{T}_T$ a serem unificados. Suponha que exista um algoritmo $\mathcal{A}$ que verifique se s e t são unificáveis- T , e caso sejam, tal algoritmo devolva uma unificadora- T θ de s e t , isto é, θ é uma substituição tal que $s\theta \stackrel{T}{=} t\theta$, onde Φ é uma permutação tal que $Var(s) \cap Var(t\Phi) = \emptyset$ (este é o caso dos algoritmos que geram um conjunto completo e finito de unificadoras- T de s e t em relação a $W \supseteq Var(s) \cup Var(t)$).

O lema 7.1, a seguir, nos mostra que, se existe um tal algoritmo $\mathcal{A}$, então também existirá algoritmos para testar se s é uma instância- T de t .

Lema 7.1. *Seja T um conjunto de equações e s, t termos em $\mathcal{T}_T$. Considere a existência de um algoritmo $\mathcal{A}$ que decida se s e t são ou não unificáveis- T e, caso afirmativo, exiba uma unificadora- T θ de s e t em relação a $W \supseteq Var(s) \cup Var(t)$. Sejam $Var(s) = \{v_1, \dots, v_k\}$ e $c_1, \dots, c_k$ constantes distintas duas a duas tais que não ocorram em s e também não ocorram em t . Considere a substituição $\phi = \{v_1 \leftarrow c_1, \dots, v_k \leftarrow c_k\}$, e seja ϕ^{-1} a “substituição” $\{c_1 \leftarrow v_1, \dots, c_k \leftarrow v_k\}$. Sendo assim, temos:*

s é uma instância- T de t se e somente se $s\phi$ e t são unificáveis- T

[PS81]

Demonstração:

($\Rightarrow$ /somente se)

Hipótese: s é instância- T de t .

Tese: $s\phi$ e t são unificáveis- T .

Se s for uma instância- T de t , então existe uma substituição θ determinada por $Var(t)$ tal que $s \stackrel{T}{=} t\theta$. Pelo lema 2.51 (ii) temos que $s\phi\theta = s\phi$, logo $s\phi\theta\phi = (s\phi\theta)\phi \stackrel{\text{lema 2.51(ii)}}{=} (s\phi)\phi = s\phi$. Mas, por hipótese $s \stackrel{T}{=} t\theta$ e pelo corolário 3.11 temos que $s\phi \stackrel{T}{=} t\theta\phi$. Então temos que $\theta\phi$ é uma unificadora- T de $s\phi$ e t .

($\Leftarrow$ /se)

Hipótese: $s\phi$ e t são unificáveis- T .

Tese: s é instância- T de t .

Se $s\phi$ e t forem unificáveis- T , então existirá uma substituição θ tal que $s\phi\theta \equiv_T t\theta$. Como $Var(s\phi) = \emptyset$, então $s\phi\theta = s\phi$ e isto implica em $s\phi \equiv_T t\theta$, aplicando ϕ^{-1} em ambos os lados, $s\phi\phi^{-1} \equiv_T t\theta\phi^{-1}$, mas $s\phi\phi^{-1} = s$ e $t\theta\phi^{-1} = t\theta'$ com $\theta' = \theta\phi^{-1}$, logo $s \equiv_T t\theta'$ e portanto acabamos de mostrar que s será uma instância- T de t . ■

Similarmente, seja $Var(s) \cup Var(t) = \{u_1, \dots, u_l\}$ e seja $\{d_1, \dots, d_l\}$ um conjunto formado por constantes distintas duas a duas que não ocorrem em s , não ocorrem em t e por fim, não ocorrem em T , e seja $\psi = \{u_1 \leftarrow d_1, \dots, u_l \leftarrow d_l\}$. Então, de modo análogo à demonstração do lema 7.1, pode-se mostrar que $s \equiv_T t$ se e somente se $s\psi$ e $t\psi$ forem unificáveis- T .

Sendo assim, acabamos de ver que se existe um algoritmo capaz de verificar se dois termos, digamos s e t , são unificáveis- T então também existe um algoritmo que determina se s é ou não uma instância- T de t e também existe um algoritmo que determina se s e t são ou não equivalentes módulo T .

Se R é um sistema de reescrita de termos e T um conjunto de equações então, $\mathcal{T}_{RT}$ denota o conjunto dos termos construídos a partir da união dos alfabetos de R e T e de um conjunto $\mathcal{V}$ infinito e enumerável de variáveis que contenha as variáveis presentes nas regras de R e nas equações de T .

Definição 7.2. (Axioma de Reescrita Módulo T) *Sejam R um sistema de reescrita de termos e T um conjunto de equações, então definimos a relação $\xrightarrow{(R,T)}$ sobre $\mathcal{T}_{RT} \times \mathcal{T}_{RT}$ como segue. O par $(s, t) \in \xrightarrow{(R,T)}$ se e somente se existir um termo $u \in [s]_T$, uma regra de reescrita $l \rightarrow r \in R$, uma posição $p \in \text{dom } u$ e uma substituição σ tal que:*

$$u/p = l\sigma \quad e \quad u[p \leftarrow r\sigma] \in [t]_T$$

Em outros termos, $\xrightarrow{(R,T)} = \equiv_T \circ \xrightarrow{R} \circ \equiv_T$.

Os elementos de $\xrightarrow{(R,T)}$ são chamados de axiomas de reescrita de R módulo T .

Nestas circunstâncias escrevemos $s \xrightarrow{(R,T)} t$ ($u, l \rightarrow r, p, \sigma$) ou simplesmente $s \xrightarrow{(R,T)} t$.

[PS81]

Note que, se $T = \emptyset$, então $\xrightarrow{T}^{\text{notas}} \equiv_T = \{ \langle t, t \rangle : t \in \mathcal{T}_T \}$ e, conseqüentemente, $[t]_T = \{t\}$ para todo t em $\mathcal{T}_T$ e, neste caso ($T = \emptyset$), a definição 7.2 torna-se equivalente à definição 3.23 de axioma de reescrita. Em outras palavras “ver” a definição de axioma de reescrita módulo T como sendo uma generalização da definição do axioma de reescrita.

Observe que, pela definição 7.2, se $l_1 \rightarrow r_1 \in R$, então $l_1 \xrightarrow{(R,T)} r_1$. Ainda, pelas definições 3.3 e 7.2, temos que $s \xrightarrow{(R,T)} t \not\equiv_T s \equiv_T t$; e $s \equiv_T t \not\equiv_T s \xrightarrow{(R,T)} t$.

Proposição 7.3. *Se $s \xrightarrow{(R,T)} t(u, \lambda \rightarrow \rho, n, \theta)$, $s' \equiv_T s$, e $t' \equiv_T t$; então $s' \xrightarrow{(R,T)} t'(u, \lambda \rightarrow \rho, n, \theta)$.*

Demonstração: Consequência imediata da definição 7.2. ♦

Definição 7.4. *Seja R um sistema de reescrita de termos e T um conjunto de equações. Sejam A, B em $\mathcal{T}_{RT}/T$, então escrevemos $A \xrightarrow{[R]} B$ se existir termo s em A e termo t em B tais que $s \xrightarrow{(R,T)} t$. Se $s \xrightarrow{(R,T)} t$ ($u, l \rightarrow r, p, \sigma$) escrevemos $A \xrightarrow{[R]} B$ ($u, l \rightarrow r, p, \sigma$).*

[PS81]

A figura abaixo ilustra as definições 7.2 e 7.4.

Figura 7.1: $s \xrightarrow{(R,T)} t \Leftrightarrow u \xrightarrow{R} u[p \leftarrow r\theta]$.

Proposição 7.5. $A \xrightarrow{[R]}^* B$ se e somente se para todo termo s em A e para todo termo t em B , $s \xrightarrow{(R,T)}^* t$.

[PS81]

Demonstração: Consequência imediata da definição 7.4. ♦

Quando escrevemos $\mathcal{T}/T$ estamos assumindo que os termos em $\mathcal{T}$ são construídos a partir de algum alfabeto que contenha o alfabeto de T .

Definição 7.6. *Seja A em $\mathcal{T}/T$ e seja θ uma substituição. Sendo assim, $A\theta = \{t\theta : t \in A\}$.*

[PS81]

Corolário 7.7. *Sejam A em $\mathcal{T}/T$, θ uma substituição e t um termo em A . Então $A\theta = [t\theta]_T$.*

[PS81]

Demonstração: Consequência imediata da definição 7.6 e do corolário 3.11. ■

Daqui até o corolário 7.11 (inclusive) considere R sendo um sistema de reescrita de termos e T um conjunto de equações.

Lema 7.8. (Compatibilidade) Se $u, s, t \in \mathcal{T}_{RT}$, $n \in \text{dom } u$, e $s \xrightarrow{(R,T)} t$, então $[u[n \leftarrow s]]_T \xrightarrow{[R]} [u[n \leftarrow t]]_T$.

[PS81]

Note que, pela proposição 7.5, $s \xrightarrow{(R,T)} t \Leftrightarrow [s]_T \xrightarrow{[R]} [t]_T$ e, portanto, pelo lema 7.8 temos que se $[s]_T \xrightarrow{[R]} [t]_T$ então para qualquer termo u em $\mathcal{T}_{RT}$ e $n \in \text{dom } u$, $[u[n \leftarrow s]]_T \xrightarrow{[R]} [u[n \leftarrow t]]_T$. Observe ainda que, se $s \xrightarrow{(R,T)} t$, então para qualquer termo u em $\mathcal{T}_{RT}$ e $n \in \text{dom } u$, $u[n \leftarrow s] \xrightarrow{(R,T)} u[n \leftarrow t]$.

Corolário 7.9. Suponha que $u, s, t \in \mathcal{T}_{RT}$, $n \in \text{dom } u$. Sendo assim, se $s \xrightarrow{(R,T)}^* u$ então $[u[n \leftarrow s]]_T \xrightarrow{[R]}^* [u[n \leftarrow u]]_T$.

Demonstração: Seja $t \in \mathcal{T}_{RT}$ tal que $s \xrightarrow{(R,T)} t$ e $t \xrightarrow{(R,T)} u$ (observe que basta demonstrarmos para este caso). Sendo assim, temos:

$s \xrightarrow{(R,T)} t \xrightarrow{\text{lema 7.8}} [u[n \leftarrow s]]_T \xrightarrow{[R]} [u[n \leftarrow t]]_T$ e $t \xrightarrow{(R,T)} u \xrightarrow{\text{lema 7.8}} [u[n \leftarrow t]]_T \xrightarrow{[R]} [u[n \leftarrow u]]_T$, segue que, $[u[n \leftarrow s]]_T \xrightarrow{[R]} [u[n \leftarrow t]]_T \xrightarrow{[R]} [u[n \leftarrow u]]_T$, isto é, $[u[n \leftarrow s]]_T \xrightarrow{[R]}^* [u[n \leftarrow u]]_T$. ◆

Lema 7.10. (Estabilidade) Se $s, t \in \mathcal{T}_{RT}$; $A, B \in \mathcal{T}_{RT}/T$; $s \xrightarrow{(R,T)} t$; $A \xrightarrow{[R]} B$ e θ for uma substituição então $s\theta \xrightarrow{(R,T)} t\theta$ e $A\theta \xrightarrow{[R]} B\theta$.

[PS81]

Demonstração: Consequência imediata das definições 7.2 e 7.4 e do corolário 7.7. ■

Os lemas 7.8 e 7.10 nos mostram que a relação binária $\xrightarrow{[R]}$ é compatível e estável sobre as classes de equivalência módulo T . Quando $T = \emptyset$ temos $\xrightarrow{[R]} = \xrightarrow{R}$. Podemos entender $\xrightarrow{[R]}$ como sendo uma relação de reescrita sobre classes de equivalência módulo T e a definição de relação de reescrita sobre termos um caso particular desta, quando $T = \emptyset$.

Corolário 7.11. Se $s, t \in \mathcal{T}_{RT}$; $A, B \in \mathcal{T}_{RT}/T$; $s \xrightarrow{(R,T)}^* t$; $A \xrightarrow{[R]}^* B$ e θ é uma substituição, então $s\theta \xrightarrow{(R,T)}^* t\theta$ e $A\theta \xrightarrow{[R]}^* B\theta$.

Demonstração: Consequência imediata do lema 7.10 e da definição 2.22 de fecho transitivo-reflexivo de uma relação binária.

◆

Pelo Teorema de Newman, num sistema de reescrita de termos noetheriano, as propriedades de confluência e confluência local equivalem-se. A seguir, veremos, pelo teorema 7.28, que esse resultado continua válido quando estendemos a noção de reescrita de termos para classes de equivalência. O Teorema de Newman pode ser visto como um caso particular do teorema 7.28, quando $T = \emptyset$.

Porém, antes disso, vamos apresentar alguns novos conceitos e resultados básicos sobre sistemas de reescrita sobre classes de equivalência que serão utilizados pelo enunciado do teorema supra-citado, bem como no decorrer de sua demonstração.

Definição 7.12. (Extensão/Extensão Protetora) *Seja t um termo. Uma extensão de t é um termo contendo t como um subtermo. Uma extensão protetora de t é uma extensão que contém distintos elementos de $\mathcal{V} - \text{Var}(t)$ em cada nó folha que não é um subtermo de t . Se t^e for uma extensão (respectivamente extensão protetora) de t e $t = t^e/n$, então dizemos que t está contido (respectivamente contido e protegido) em t^e na posição n .*

[PS81]

Exemplo 7.13. *Seja $t = f(a, g(x, y))$ então $h(z, f(a, g(x, y)), w)$ é uma extensão protetora de t , mas $h(x, f(a, g(x, y)), w)$ e $h(z, f(a, g(x, y)), z)$ não são.*

Definição 7.14. *Seja $l \rightarrow r$ uma regra de reescrita. Uma extensão de $l \rightarrow r$ é uma regra de reescrita $l^e \rightarrow r^e$ tal que para algum nó n , l está contido em l^e na posição n e $r^e = l^e[n \leftarrow r]$. Uma extensão protetora de $l \rightarrow r$ é uma extensão $l^e \rightarrow r^e$ tal que l^e contém distintos elementos de $\mathcal{V} - (\text{Var}(l) \cup \text{Var}(r))$ em cada nó folha que não for subtermo de l . Se $l^e \rightarrow r^e$ for uma extensão (respectiva/e extensão protetora) de $l \rightarrow r$ e $l = l^e/n$, então dizemos que $l \rightarrow r$ está contida (respectiva/e contida e protegida) em $l^e \rightarrow r^e$ na posição n .*

[PS81]

Exemplo 7.15. *$g(x, a) \rightarrow g(x, f(y))$ é uma extensão protetora de $a \rightarrow f(y)$, mas $g(y, a) \rightarrow g(y, f(y))$ não é.*

[PS81]

Teorema 7.16. *Se R e R^e são sistemas de reescrita de termos, tais que $R \subseteq R^e$ e todo elemento de R^e for uma extensão de um elemento de R . Então $\overrightarrow{[R_2^e]} = \overrightarrow{[R]}$.*

[PS81]

Demonstração: Precisamos mostrar que $\overrightarrow{[R_2^e]} \subseteq \overrightarrow{[R]}$ e $\overrightarrow{[R]} \subseteq \overrightarrow{[R_2^e]}$. Como, por hipótese, $R \subseteq R^e$, segue trivialmente que $\overrightarrow{[R]} \subseteq \overrightarrow{[R_2^e]}$.

Resta-nos mostrar que $\vec{\rightarrow}_{[R_2^e]} \subseteq \vec{\rightarrow}_{[R]}$, ou seja, vamos mostrar que se $A \vec{\rightarrow}_{[R_2^e]} B$ então $A \vec{\rightarrow}_{[R]} B$.

Mas, $A \vec{\rightarrow}_{[R_2^e]} B \stackrel{\text{def.7.4}}{\iff} (\exists s \in A \text{ e } \exists t \in B : (s \xrightarrow{(R^e, T)} t \stackrel{\text{def.7.2}}{\iff} \exists u \in [s]_T, n \in \text{dom } u \text{ e } l^e \rightarrow r^e \in R^e \text{ e substituição } \theta : u/n = l^e\theta \text{ e } u[n \leftarrow r^e\theta] \in [t]_T))$

De fato, pela hipótese deste teorema, cada elemento $l^e \rightarrow r^e$ em R^e é uma extensão de algum elemento $l \rightarrow r$ em R . Logo, pela definição 7.14, $\exists m \in \text{dom } l^e$ tal que $l = l^e/m$ e $r^e = l^e[m \leftarrow r]$. Como, $l = l^e/m$, temos $l\theta = (l^e/m)\theta = l^e\theta/m$ e portanto $l\theta$ também é subtermo de u , porém localizado na posição $n \cdot m$, isto é, $u/(n \cdot m) = l\theta$.

Por hipótese, $u[n \leftarrow r^e\theta] \in [t]_T$, mas $r^e = l^e[m \leftarrow r]$, logo $u[n \leftarrow r^e\theta] = u[n \leftarrow l^e[m \leftarrow r]\theta] = u[n \leftarrow l^e\theta[m \leftarrow r\theta]] = u[n \leftarrow u/n[m \leftarrow r\theta]] = u[n \cdot m \leftarrow r\theta] \in [t]_T$.

Temos $u/(n \cdot m) = l\theta$ e $u[n \cdot m \leftarrow r\theta] \in [t]_T$, onde $u \in [s]_T$, logo, pela definição 7.2, $s \xrightarrow{(R, T)} t$, mas por hipótese, $s \in A$ e $t \in B$ e, portanto, pela definição 7.4, $A \vec{\rightarrow}_{[R]} B$. ■

Figura 7.2: Diagrama ilustrativo sobre a demonstração do teorema 7.16.

Teorema 7.17. Se $t \rightarrow s(u, l \rightarrow r, n, \theta)$ e $m \preceq n$, então existe uma extensão $l^e \rightarrow r^e$ e uma substituição ϕ tais que $t \rightarrow s(u, l^e \rightarrow r^e, m, \phi)$.

[PS81]

As próximas oito definições foram inspiradas em [DJ90, PS81].

Definição 7.18. (Sistema de Reescrita Confluente Módulo T) Um sistema de reescrita de termos R é dito confluente módulo T se e somente se

$$\forall A, B, C \in \mathcal{T}_{RT}/T, \exists D \in \mathcal{T}_{RT}/T : (C \xrightarrow{[R]}^* A \xrightarrow{[R]}^* B \Rightarrow C \xrightarrow{[R]}^* D \xrightarrow{[R]}^* B).$$

Figura 7.3: Confluência módulo T.

Note que, pela definição 7.4, podemos expressar a definição acima, equivalentemente, do seguinte modo:

Definição 7.19. ((Segunda Forma) Sistema de Reescrita Confluente Módulo T) Um sistema de reescrita de termos R é dito confluente módulo T se e somente se $\forall t, t_1, t_2 \in \mathcal{T}_{RT}/T, \exists t'_1$ e t'_2 ambos em $\mathcal{T}_{RT}/T$ com $t'_1 \equiv_T t'_2$ tal que $t \xrightarrow{(R,T)}^* t_1$ e $t \xrightarrow{(R,T)}^* t_2 \Rightarrow t_1 \xrightarrow{(R,T)}^* t'_1$ e $t_2 \xrightarrow{(R,T)}^* t'_2$

Figura 7.4: Um modo equivalente de expressar a confluência módulo T.

Definição 7.20. ((Terceira Forma) Sistema de Reescrita Noetheriano Módulo T) Um sistema de reescrita de termos R é dito noetheriano módulo T se qualquer seqüência $A = A_0 \xrightarrow{[R]} A_1 \xrightarrow{[R]} \dots \xrightarrow{[R]} A_i \xrightarrow{[R]} \dots$ de reescrita de classes de equivalência módulo T é finita, isto é, $\forall A \in \mathcal{T}_{RT}/T, \exists B \in \mathcal{T}_{RT}/T$ tal que $A \xrightarrow{[R]}^* B$ e $\nexists C \in \mathcal{T}_{RT}/T$ tal que $B \xrightarrow{[R]} C$.

Note que, a primeira forma da definição de SRT noetheriano módulo T é dada pela definição 5.67. Observe que, pela definição 7.4, podemos expressar a definição acima, equivalentemente, das seguinte forma.

Definição 7.21. ((Quarta Forma) Sistema de Reescrita Noetheriano Módulo T) *Um sistema de reescrita de termos R é dito noetheriano módulo T se qualquer seqüência $t = t_1 \xrightarrow{(R,T)} t_2 \xrightarrow{(R,T)} \cdots \xrightarrow{(R,T)} t_i \xrightarrow{(R,T)} \cdots$ de reescrita de termos em $\mathcal{T}_{RT}$ é finita, isto é, $\forall t \in \mathcal{T}_{RT}, \exists t' \in \mathcal{T}_{RT}$ tal que $t \xrightarrow{(R,T)}^* t'$ e $\nexists s \in \mathcal{T}_{RT}$ tal que $t' \xrightarrow{(R,T)} s$.*

Definição 7.22. (Sistema de Reescrita Localmente Confluyente Módulo T) *Dizemos que um sistema de reescrita de termos R é localmente confluyente módulo T quando $\forall t, t_1, t_2 \in \mathcal{T}_{RT}/T, \exists t'_1, t'_2 \in \mathcal{T}_{RT}/T$ com $t'_1 \equiv_T t'_2$ tal que $(t \xrightarrow{(R,T)} t_1 \text{ e } t \xrightarrow{(R,T)} t_2) \Rightarrow (t_1 \xrightarrow{(R,T)}^* t'_1 \text{ e } t_2 \xrightarrow{(R,T)}^* t'_2)$*

Definição 7.23. (Sistema de Reescrita Completo Módulo T) *Um sistema de reescrita de termos é dito completo módulo T se ele for confluyente módulo T e noetheriano módulo T .*

Teorema 7.24. (Versão do Teorema de Newman para Classes de Equivalência) *Um sistema de reescrita de termos R noetheriano módulo T é confluyente módulo T se e somente se R é localmente confluyente módulo T .*

Definição 7.25. (Propriedade Church-Rosser Módulo T) *Um SRT R possui a propriedade Church-Rosser módulo T se e somente se para quaisquer termos s, t em $\mathcal{T}_{RT}$, se $s \xrightarrow{(R,T)}^* t$ então $\exists s', t' \in \mathcal{T}_{RT}$ com $s' \equiv_T t'$ e tais que $s \xrightarrow{(R,T)}^* s'$ e $t \xrightarrow{(R,T)}^* t'$.*

Figura 7.5: Church-Rosser módulo T.

Lema 7.26. *Um SRT R possui a propriedade Church-Rosser módulo T se e somente se R possui a propriedade de confluência módulo T .*

Demonstração: Consequência imediata do lema 6.16 e das definições 7.25 e 7.18. ♦

Observe que $A \xrightarrow{[R]}^* B$ é equivalente a $t \equiv_{R \cup T} s$ para qualquer $t \in A$ e $s \in B$.

Definição 7.27. (Variante de uma Regra de Reescrita/Equação) Uma regra de reescrita $s' \rightarrow t'$, ou equação $s' = t'$, é uma variante, respectivamente, de uma regra de reescrita $s \rightarrow t$, ou equação $s = t$, se $s' = s\Phi$ e $t' = t\Phi$ e ainda $s'\Phi^{-1} = s$ e $t'\Phi^{-1} = t$, onde Φ é uma permutação e Φ^{-1} é sua inversa.

[PS81]

Em outras palavras, pela definição 7.27, uma variante de uma regra de reescrita ou equação é uma outra regra de reescrita ou equação, respectivamente, que difere da primeira apenas pela renomeação de suas variáveis.

Peterson e Stickel [PS81] observam que nada de importante acontece se as regras de reescrita ou equações são trocadas por variantes. Por exemplo, trocando-se uma, duas (quando houver mais de uma), ou todas as regras de reescrita de um sistema de reescrita R por variantes não se altera a relação $\xrightarrow{[R]}$.

Dizemos que duas regras de reescrita $\langle l_1, r_1 \rangle$ e $\langle l_2, r_2 \rangle$ tiveram suas variáveis padronizadas *a priori* quando $\langle l_1, r_1 \rangle$ e $\langle l_2, r_2 \rangle$ são variantes tais que $\text{Var}(\langle l_1, r_1 \rangle) \cap \text{Var}(\langle l_2, r_2 \rangle) = \emptyset$.

Do mesmo modo, se as equações $\{l_1, r_1\}$ e $\{l_2, r_2\}$ tiveram suas variáveis padronizadas *a priori* então $\{l_1, r_1\}$ e $\{l_2, r_2\}$ são variantes tais que $\text{Var}(\{l_1, r_1\}) \cap \text{Var}(\{l_2, r_2\}) = \emptyset$.

Teorema 7.28. Suponha que R seja um sistema de reescrita de termos *noetheriano* módulo T . Suponha também que sempre que $l_1^e \rightarrow r_1^e$ e $l_2^e \rightarrow r_2^e$ forem extensões protetoras de regras de reescrita em R cujas variáveis foram padronizadas *a priori* e σ for uma unificadora- T em um conjunto completo de unificadoras- T de l_1^e e l_2^e sobre $\text{Var}(l_1^e \rightarrow r_1^e) \cup \text{Var}(l_2^e \rightarrow r_2^e)$, haverá $[w]_T \in \mathcal{T}_{RT}/T$ tal que:

$$[r_1^e \sigma]_T \xrightarrow{[R]}^* [w]_T \quad \text{e} \quad [r_2^e \sigma]_T \xrightarrow{[R]}^* [w]_T$$

Então R será um sistema de reescrita de termos completo módulo T .

[PS81]

Demonstração: Pelo teorema 7.24, R será completo módulo T se sempre que $A \in \mathcal{T}_{RT}/T$ for tal que $A \xrightarrow{[R]} A_1$ e $A \xrightarrow{[R]} A_2$, existir $W \in \mathcal{T}_{RT}/T$ tal que $A_1 \xrightarrow{[R]}^* W$ e $A_2 \xrightarrow{[R]}^* W$.

Suponha então que $A, A_1, A_2 \in \mathcal{T}_{RT}/T$ e

$$A \xrightarrow{[R]} A_1(t_1, l_1 \rightarrow r_1, n_1, \theta), \quad A \xrightarrow{[R]} A_2(t_2, l_2 \rightarrow r_2, n_2, \theta),$$

onde $l_1 \rightarrow r_1$ e $l_2 \rightarrow r_2$ tiveram suas variáveis padronizadas à priori. Como as variáveis de $l_1 \rightarrow r_1$ e $l_2 \rightarrow r_2$ são distintas, isto justifica o fato de podermos utilizar a mesma substituição θ em ambas estas regras de reescrita. Pelo teorema 7.17 existem extensões protetoras $l_1^e \rightarrow r_1^e$ e $l_2^e \rightarrow r_2^e$ de $l_1 \rightarrow r_1$ e $l_2 \rightarrow r_2$, respectivamente, e uma substituição ϕ tais que $\text{Var}(\langle l_1^e, r_1^e \rangle) \cap \text{Var}(\langle l_2^e, r_2^e \rangle) = \emptyset$ e

$$A \xrightarrow{[R]} A_1(t_1, l_1^e \rightarrow r_1^e, \langle \rangle, \phi), \quad A \xrightarrow{[R]} A_2(t_2, l_2^e \rightarrow r_2^e, \langle \rangle, \phi).$$

Então observando as definições 7.2 e 7.4 e o fato que $t_1/\langle \rangle = l_1^e \phi$ onde $t_1/\langle \rangle = t_1$ e analogamente para t_2 , $t_2/\langle \rangle = l_2^e \phi$ onde $t_2/\langle \rangle = t_2$ temos que:

$$(i_1) \quad l_1^e \phi \in A, \quad (i_2) \quad r_1^e \phi \in A_1;$$

e

$$(ii_1) \quad l_2^e \phi \in A, \quad (ii_2) \quad r_2^e \phi \in A_2.$$

Note que $l_1^e \phi \in A$ se e somente se $A = [l_1^e \phi]_T$ e, conseqüentemente, ϕ é uma unificadora-T de l_1^e e l_2^e . Seja Γ um conjunto completo de unificadoras-T de l_1^e e l_2^e em relação a V , onde $V = \text{Var}(l_1^e \rightarrow r_1^e) \cup \text{Var}(l_2^e \rightarrow r_2^e)$. Pela definição 4.8 (iii), existe $\sigma \in \Gamma$ e existe uma substituição ψ tal que:

$$iii) \quad \phi_V \xrightarrow{T} (\sigma\psi)_V \text{ (veja a noção de substituições } E\text{-congruentes)}.$$

Por hipótese, existe $[w]_T \in \mathcal{T}_{RT}/T$ tal que

$$[r_1^e \sigma]_T \xrightarrow{[R]}^* [w]_T \quad e \quad [r_2^e \sigma]_T \xrightarrow{[R]}^* [w]_T.$$

Mas,

$$(iv) \quad [r_1^e \sigma]_T \xrightarrow{[R]}^* [w]_T \xrightarrow{\text{lema 7.10}} [r_1^e \sigma \psi]_T \xrightarrow{[R]}^* [w\psi]_T \quad e$$

$$(v) \quad [r_2^e \sigma]_T \xrightarrow{[R]}^* [w]_T \xrightarrow{\text{lema 7.10}} [r_2^e \sigma \psi]_T \xrightarrow{[R]}^* [w\psi]_T$$

Seja $W = [w\psi]_T$, então:

$$A_1 \stackrel{(i)}{=} [r_1^e \phi]_T \stackrel{(iii)}{=} [r_1^e \sigma \psi]_T \xrightarrow{[R]}^* \overset{\text{hip.} + (iv)}{[w\psi]_T} \stackrel{\text{def.}}{=} W,$$

e

$$A_2 \stackrel{(ii)}{=} [r_2^e \phi]_T \stackrel{(iii)}{=} [r_2^e \sigma \psi]_T \xrightarrow{[R]}^* \overset{\text{hip.} + (v)}{[w\psi]_T} \stackrel{\text{def.}}{=} W.$$

Portanto R é um sistema de reescrita de termos completo módulo T . ■

O teorema 7.28 possui uma dificuldade computacional, uma vez que toda regra de reescrita possui uma quantidade infinita de extensões protetoras, isto é, não existe algoritmo que possa verificar a hipótese do teorema utilizando para tal um número finito de passos. O resultado apresentado pelo teorema 7.28, portanto, não possui aplicações práticas. Na próxima seção, apresentaremos algumas restrições ao conjunto T de equações, sob as quais mostraremos a existência de um algoritmo finito para a verificação da completeza módulo T de um sistema de reescrita de termos.

7.2 Compatibilidade de SRT's com Teorias Equacionais

A próxima definição 7.29 estabelece uma restrição ao conjunto T de equações. Tal restrição será utilizada para demonstrarmos o próximo teorema que estende a aplicabilidade do Teorema de Knuth e Bendix para a verificação de completeza módulo T de SRT's. Entretanto, exigimos a seguinte restrição entre o SRT e o conjunto T de equações considerados.

Definição 7.29. (Compatibilidade de R com T) Dizemos que R é compatível com T se sempre que $t \xrightarrow{(R,T)} s$, existir um nó $m \in \text{dom } t$, uma substituição ϕ e uma regra de reescrita $l \rightarrow r$ em R tais que

$$t/m \equiv l\phi \quad e \quad [s]_T \xrightarrow{[R]}^* [t[m \leftarrow r\phi]]_T.$$

[PS81]

A definição acima funciona como uma restrição para os conjuntos R e T considerados apenas quando $T \neq \emptyset$, pois, caso contrário, teremos, obrigatoriamente, a compatibilidade de R com T , independentemente do SRT R escolhido.

Na próxima seção, veremos que, em certas circunstâncias, é possível adicionar a R extensões de regras de reescrita de R , de tal modo que o conjunto de regras de reescrita resultante seja compatível com T .

Exemplo 7.30. Seja $T = \{x + (y + z) = (x + y) + z\}$ e $R = \{(-x) + x \rightarrow 0\}$, então R não é compatível com T . Podemos verificar isto tomando $t = -(-a) + ((-a) + a)$ e $s = 0 + a$, tome $m = \langle 2 \rangle$, $t/m = (-a) + a$, e $\phi = \{x \leftarrow a\}$, logo $t/m \equiv l\phi$ pois $l\phi = ((-x) + x)\phi = (-a) + a$. $[s]_T = [0+a]_T$ e $[t[m \leftarrow l\phi]]_T = [-(-a) + (0+a)]_T$, mas $\langle [0+a]_T, [-(-a) + (0+a)]_T \rangle \notin \xrightarrow{[R]}^*$ pois caso contrário, existiria $s' \in [0+a]_T$ e $t' \in [-(-a) + (0+a)]_T$ tal que $s' \xrightarrow{(R,T)} t' \Leftrightarrow \exists u \in [s']_T, n \in \text{dom } u$ e substituição θ tais que $u/n \equiv l\theta$ e $u[n \leftarrow r\theta] \in [t']_T$. Mas, $[0+a]_T = \{0+a\}$ pois não existe termo v e não existe substituição σ tais que $\langle 0+a, v \rangle \in \xrightarrow{T}$ pois não há subtermo de $0+a$ que seja instância de $x + (y + z)$ ou $(x + y) + z$ (veja a

definição 3.3).

O mesmo pode ser verificado se tomarmos qualquer nó m em $\text{dom } t = \{ \langle \rangle, \langle 1 \rangle, \langle 1, 1 \rangle, \langle 1, 1, 1 \rangle, \langle 2 \rangle, \langle 2, 1 \rangle, \langle 2, 1, 1 \rangle, \langle 2, 2 \rangle \}$.

Entretanto, se adicionarmos a R as seguintes regras de reescrita $y + ((-x) + x) \rightarrow y + 0$ e $((-x) + x) + z \rightarrow 0 + z$, as quais são extensões da regra de reescrita $(-x) + x \rightarrow 0$, teremos $R = \{(-x) + x \rightarrow 0, y + ((-x) + x) \rightarrow y + 0, ((-x) + x) + z \rightarrow 0 + z\}$, um conjunto de regras de reescrita compatível com a teoria equacional T .

[PS81]

Definição 7.31. (Par Crítico Módulo T) Suponha que $l_1 \rightarrow r_2$ e $l_2 \rightarrow r_2$ sejam elementos de R e que suas variáveis foram padronizadas à priori. Seja $k \in \text{sdom } l_1$. Suponha ainda que σ seja uma unificadora- T em um conjunto completo de unificadoras- T de l_1/k e l_2 sobre $\text{Var}(l_1 \rightarrow r_1) \cup \text{Var}(l_2 \rightarrow r_2)$. Então, dizemos que $\langle [l_1[k \leftarrow r_2]\sigma]_T, [r_1\sigma]_T \rangle$ é um par crítico de R módulo T .

[PS81]

Figura 7.6: Par crítico $\langle [u\sigma]_T, [s[k_1 \leftarrow v]\sigma]_T \rangle$ módulo T .

Vamos, a seguir, fazer algumas observações que nos serão úteis para o desenvolvimento da demonstração do próximo teorema 7.32. Note que, pela definição 7.31, temos que, se $\langle A, B \rangle$ for um par crítico de R módulo T , então $A \xrightarrow{[R]}^* B$. Entretanto, a recíproca não é verdadeira, isto é, se $A \xrightarrow{[R]}^* B$ se verifica, isto não implica que $\langle A, B \rangle$ seja um par crítico de R módulo T .

Teorema 7.32. (Generalização do Teorema de Knuth e Bendix) Seja R um sistema de reescrita de termos $\aleph$ -theriano módulo T e compatível com T . Então R será um sistema de reescrita de termos completo módulo T se e somente se para todo par crítico $\langle A, B \rangle$ de R módulo T , existir C em $\mathcal{T}_{RT}/T$ tal que $A \xrightarrow{[R]}^* C$ e $B \xrightarrow{[R]}^* C$.

[PS81]

Demonstração: ($\Rightarrow$ /somente se)

Hipótese: R é completo módulo T .

Tese: Para todo par crítico $\langle A, B \rangle$ de R módulo T , existe $C \in \mathcal{T}_{RT}/T$ tal que $A \xrightarrow{[R]}^* C$ e $B \xrightarrow{[R]}^* C$.

Se R for completo módulo T e $\langle A, B \rangle$ for um par crítico para R e T , então pela definição 7.31 de par crítico temos $A \xleftarrow{[R]}^* B$, e a classe de equivalência procurada C existe pela definição 7.23 de sistema de reescrita de termos completo módulo T .

($\Leftarrow$ /se)

Hipótese: Para todo par crítico $\langle A, B \rangle$ de R módulo T , existe $C \in \mathcal{T}_{RT}/T$ tal que $A \xrightarrow{[R]}^* C$ e $B \xrightarrow{[R]}^* C$.

Tese: R é completo módulo T .

Seja $A \in \mathcal{T}_{RT}/T$ tal que $A \xrightarrow{[R]} A_1$ e $A \xrightarrow{[R]} A_2$. Como por hipótese R é noetheriano módulo T , então pelo teorema 7.24 R será completo módulo T se R for localmente confluyente módulo T , isto é, se existir $W \in \mathcal{T}_{RT}/T$ tal que $A_1 \xrightarrow{[R]}^* W$ e $A_2 \xrightarrow{[R]}^* W$.

Seja t um termo em A . Como R é compatível com T , existem nós m_1 e $m_2 \in \text{dom } t$, uma substituição ϕ , e regras de reescrita $l_1 \rightarrow r_1, l_2 \rightarrow r_2$ em R cujas variáveis foram padronizadas à priori tais que pela definição 7.29:

$$(i) \quad t/m_1 \equiv_T l_1\phi, \quad (ii) \quad t/m_2 \equiv_T l_2\phi,$$

$$(iii) \quad A_1 \xrightarrow{[R]}^* [t[m_1 \leftarrow r_1\phi]]_T, \quad (iv) \quad A_2 \xrightarrow{[R]}^* [t[m_2 \leftarrow r_2\phi]]_T.$$

(Note que podemos tomar ϕ em ambos os casos pois $l_1 \rightarrow r_1$ e $l_2 \rightarrow r_2$ foram padronizadas à priori). O restante da demonstração será dividido em casos, dependendo da relação entre m_1 e m_2 em t .

Caso 1: Suponha que $m_1 \perp m_2$ (um redex é prefixo (ancestral) do outro). Seja

$$W = [t[m_1 \leftarrow r_1\phi][m_2 \leftarrow r_2\phi]]_T.$$

Como $m_1 \perp m_2$ então $t[m_1 \leftarrow r_1\phi]/m_2 = t/m_2 \xrightarrow{(ii)} l_2\phi$ e, conseqüentemente, pelo lema 3.12 $t[m_1 \leftarrow r_1\phi] \equiv_T t[m_1 \leftarrow r_1\phi][m_2 \leftarrow l_2\phi]$ ou equivalentemente $[t[m_1 \leftarrow r_1\phi]]_T = [t[m_1 \leftarrow r_1\phi][m_2 \leftarrow l_2\phi]]_T$, mas por (iii) $A_1 \xrightarrow{[R]}^* [t[m_1 \leftarrow r_1\phi]]_T$ e portanto $A_1 \xrightarrow{[R]}^* [t[m_1 \leftarrow r_1\phi][m_2 \leftarrow l_2\phi]]_T$.

Pelo lema 7.10 $\xrightarrow{(R,T)}$ é estável, logo $l_2\phi \xrightarrow{(R,T)} r_2\phi$, e portanto

$$[t[m_1 \leftarrow r_1\phi][m_2 \leftarrow l_2\phi]]_T \xrightarrow[\text{[R]}]{\text{lema7.8}} [t[m_1 \leftarrow r_1\phi][m_2 \leftarrow r_2\phi]]_T = W$$

Isto é, $\boxed{A_1 \xrightarrow[\text{[R]}]{*} W}$.

Analogamente,

$A_2 \xrightarrow[\text{[R]}]{*} [t[m_2 \leftarrow r_2\phi]]_T \xrightarrow{\text{lema3.12}} [t[m_2 \leftarrow r_2\phi][m_1 \leftarrow l_1\phi]]_T$ e pelo lema 7.10 $l_1\phi \xrightarrow[\text{(R,T)}]{} r_1\phi$, logo pelo lema 7.8 temos que

$$[t[m_2 \leftarrow r_2\phi][m_1 \leftarrow l_1\phi]]_T \xrightarrow[\text{[R]}]{\text{lema7.8}} [t[m_2 \leftarrow r_2\phi][m_1 \leftarrow r_1\phi]]_T = W$$

Isto é, $\boxed{A_2 \xrightarrow[\text{[R]}]{*} W}$.

Note que, se $m_1 \not\preceq m_2$ então ou $m_1 \preceq m_2$ ou $m_2 \preceq m_1$. Vamos demonstrar para apenas um destes casos, pois o outro é simétrico.

Caso 2: Suponha que $m_1 \preceq m_2$ (um redex é prefixo (ancestral) do outro). Então

$$(t/m_1)/(m_2/m_1) \xrightarrow{\text{lema2.35(i)}} t/m_2 \xrightarrow[\text{T}]{(ii)} l_2\phi.$$

Como $t/m_1 \xrightarrow[\text{T}]{(i)} l_1\phi$ e usando a definição 7.2 fazendo na definição $u = t/m_1$ e $n = (m_2/m_1)$ e observando que $l_2 \rightarrow r_2 \in R$ temos que:

$$l_1\phi \xrightarrow[\text{(R,T)}]{} t/m_1[m_2/m_1 \leftarrow r_2\phi].$$

Mas, por hipótese, R é compatível com T . Logo, existe um nó $k \in \text{dom } l_1\phi$, uma substituição ϕ' , e uma variante $l \rightarrow r$ de uma regra de reescrita de R tais que:

$$v) \quad \text{Var}(l \rightarrow r) \cap [\text{Var}(l_1 \rightarrow r_1) \cup \text{Var}(l_2 \rightarrow r_2)] = \emptyset,$$

$$vi) \quad l_1\phi/k \xrightarrow[\text{T}]{\text{defi.7.29}} l\phi',$$

$$vii) \quad [t/m_1[m_2/m_1 \leftarrow r_2\phi]]_T \xrightarrow[\text{[R]}]{*} [l_1\phi[k \leftarrow r\phi']]_T.$$

Podemos assumir, sem perda de generalidades que, $Lvar(\phi) \subseteq \text{Var}(l_1 \rightarrow r_1) \cup \text{Var}(l_2 \rightarrow r_2)$ e $Lvar(\phi') \subseteq \text{Var}(l \rightarrow r)$ e, conseqüentemente, por (v) temos $Lvar(\phi') \cap \text{Var}(l_1) = \emptyset$ e que $Lvar(\phi) \cap \text{Var}(l) = \emptyset$. O restante da demonstração do teorema será dividida em casos, dependendo da localização de k em $\text{dom } l_1\phi$ (veja o lema 2.48).

Caso 2.1($k \notin \text{sdom } l_1$): Suponha que existam nós com posições m e p tais que $k = m \cdot p$, $m \in \text{dom } l_1$, $l_1m \in \mathcal{V}$, e $p \in \text{dom } (l_1m)\phi$. Então pelo lema 2.48 (ii), $l_1\phi/k = (l_1m)\phi/p$. Suponha que $\phi = \{v_1 \leftarrow t_1, \dots, v_q \leftarrow t_q\}$ e defina $N_i = \{n \in \text{dom } l_1 \mid l_1n = v_i\}$. Então

$l_1\phi = l_1[N_1 \leftarrow t_1] \cdots [N_q \leftarrow t_q]$. Para algum $1 \leq j \leq q$, $l_1m = v_j$ e $m \in N_j$. Seja $N'_j = N_j - \{m\}$. Então (veja o lema 2.32(viii)):

$$l_1\phi = l_1[N_1 \leftarrow t_1] \cdots [m \leftarrow t_j][N'_j \leftarrow t_j] \cdots [N_q \leftarrow t_q].$$

Agora, $p \in \text{dom } t_j$ e $l_1\phi/k = t_j/p \xrightarrow[\overline{T}]{(vi)} l\phi'$. Observe que $l \rightarrow r \in R \xrightarrow[\overline{T}]{\text{defi.7.2}} l \xrightarrow[\overline{T}]{(R,T)} r \xrightarrow[\overline{T}]{\text{lema7.10}} l\phi' \xrightarrow[\overline{T}]{\text{lema7.8}} r\phi' \xrightarrow[\overline{T}]{\text{lema7.8}} \boxed{t_j[p \leftarrow l\phi'] \xrightarrow[\overline{T}]{(R,T)} t_j[p \leftarrow r\phi']}^*$, mas $t_j/p \xrightarrow[\overline{T}]{\text{lema3.12}} l\phi' \xrightarrow[\overline{T}]{\text{lema3.12}} \boxed{t_j \xrightarrow[\overline{T}]{\text{lema3.12}} t_j[p \leftarrow l\phi']}^{**}$.

Logo, como valem $*$ e $** \xrightarrow[\overline{T}]{\text{prop.7.3}} t_j \xrightarrow[\overline{T}]{(R,T)} t_j[p \leftarrow r\phi'] = t'_j$.

Sejam

$$\phi'' = \{v_1 \leftarrow t_1, \dots, v_j \leftarrow t'_j, \dots, v_q \leftarrow t_q\},$$

e

$$W = [t[m_1 \leftarrow r_1\phi'']]_T.$$

Por (iii) temos: $A_1 \xrightarrow[\overline{R}]{*} [t[m_1 \leftarrow r_1\phi]]_T$

Como $t_j \xrightarrow[\overline{T}]{(R,T)} t'_j$, fazendo aplicações sucessivas do lema 7.8 uma, para cada v_j existente em r_1 temos que $[t[m_1 \leftarrow r_1\phi]]_T \xrightarrow[\overline{R}]{*} [t[m_1 \leftarrow r_1\phi'']]_T = W$, logo $\boxed{A_1 \xrightarrow[\overline{R}]{*} W}$

Por (iv) temos: $A_2 \xrightarrow[\overline{R}]{*} [t[m_2 \leftarrow r_2\phi]]_T \xrightarrow[\overline{R}]{\text{trivial}} [t[m_1 \leftarrow t/m_1[m_2/m_1 \leftarrow r_2\phi]]]_T$

Observe que por (vii) $([t/m_1[m_2/m_1 \leftarrow r_2\phi]]_T \xrightarrow[\overline{R}]{*} [l_1\phi[k \leftarrow r\phi']]_T) \xrightarrow[\overline{T}]{\text{prop.7.5}} (t/m_1[m_2/m_1 \leftarrow r_2\phi] \xrightarrow[\overline{T}]{*} l_1\phi[k \leftarrow r\phi']) \xrightarrow[\overline{T}]{\text{coro7.9}} ([t[m_1 \leftarrow t/m_1[m_2/m_1 \leftarrow r_2\phi]]]_T \xrightarrow[\overline{R}]{*} [t[m_1 \leftarrow l_1\phi[k \leftarrow r\phi']]_T)$

$$l_1\phi[k \leftarrow r\phi'] = l_1[N_1 \leftarrow t_1][N_2 \leftarrow t_2] \cdots [m \leftarrow t_j][N'_j \leftarrow t_j] \cdots [N_q \leftarrow t_q][k \leftarrow r\phi'] =$$

$$l_1[N_1 \leftarrow t_1][N_2 \leftarrow t_2] \cdots [N'_j \leftarrow t_j] \cdots [N_q \leftarrow t_q][m \leftarrow t_j][k \leftarrow r\phi'] =$$

$$l_1[N_1 \leftarrow t_1][N_2 \leftarrow t_2] \cdots [N'_j \leftarrow t_j] \cdots [N_q \leftarrow t_q][m \leftarrow t_j[p \leftarrow r\phi']],$$

Mas, $t'_j = t_j[p \leftarrow r\phi']$, logo $l_1\phi[k \leftarrow r\phi'] = l_1[N_1 \leftarrow t_1][N_2 \leftarrow t_2] \cdots [N'_j \leftarrow t_j] \cdots [N_q \leftarrow t_q][m \leftarrow t'_j]$ e, conseqüentemente,

$$[t[m_1 \leftarrow l_1\phi[k \leftarrow r\phi']]]_T = [t[m_1 \leftarrow l_1[N_1 \leftarrow t_1][N_2 \leftarrow t_2] \cdots [N'_j \leftarrow t_j] \cdots [N_q \leftarrow t_q][m \leftarrow t'_j]]]_T$$

Novamente, como $t_j \xrightarrow[\overline{T}]{(R,T)} t'_j$, fazendo aplicações sucessivas do lema 7.8 uma para cada v_j existente em l_1 , temos que

$$[t[m_1 \leftarrow l_1[N_1 \leftarrow t_1] \cdots [N'_j \leftarrow t_j] \cdots [N_q \leftarrow t_q][m \leftarrow t'_j]]]_T \xrightarrow[\overline{R}]{*} [t[m_1 \leftarrow l_1\phi'']]_T$$

$l_1 \rightarrow r_1 \in R \xrightarrow[\overline{T}]{\text{defi.7.2}} l_1 \xrightarrow[\overline{T}]{(R,T)} r_1 \xrightarrow[\overline{T}]{\text{lema7.10}} l_1\phi'' \xrightarrow[\overline{T}]{(R,T)} r_1\phi'' \xrightarrow[\overline{T}]{\text{lema7.8}} [t[m_1 \leftarrow l_1\phi'']]_T \xrightarrow[\overline{R}]{*} [t[m_1 \leftarrow r_1\phi'']]_T = W$, logo $\boxed{A_2 \xrightarrow[\overline{R}]{*} W}$

Caso 2.2($k \in \text{sdom } l_1$): pelo lema 2.48(i), $l_1\phi/k = (l_1/k)\phi \xrightarrow[\overline{T}]{(vi)} l\phi'$

Como vale (v), temos $Lvar(\phi) \subseteq Var(l_1 \rightarrow r_1) \cup Var(l_2 \rightarrow r_2)$ e $Var(\phi') \subseteq Var(l \rightarrow r)$ segue que $Lvar(\phi) \cap Lvar(\phi') = \emptyset$ e ainda $(l_1/k)\phi \equiv_T l\phi'$.

Como consequência imediata da definição 2.54 temos que se s e t são termos em $\mathcal{T}_{RT}$, σ' e σ substituições tais que $Lvar(\sigma') \cap Lvar(\sigma) = \emptyset$, $Var(s) \cap Lvar(\sigma) = \emptyset$, e $s\sigma' \equiv_T t\sigma'$ então $s(\sigma' \cup \sigma) \equiv_T t(\sigma' \cup \sigma)$ e, conseqüentemente, s e t são unificáveis- T . Sendo assim $\theta = \phi \cup \phi'$ é uma unificadora- T de l_1/k e l .

Seja Γ um conjunto de unificadoras- T de l_1/k e l em relação a $V = Var(l_1 \rightarrow r_1) \cup Var(l \rightarrow r)$. Então existe $\sigma \in \Gamma$ e uma substituição ψ tais que, pela definição 4.8, $\theta_V \equiv_T (\sigma\psi)_V$.

Temos que, por hipótese, para todo par crítico $\langle A, B \rangle$ de R módulo T , existe $C \in \mathcal{T}_{RT}/T$ tal que $A \xrightarrow{[R]}^* C$ e $B \xrightarrow{[R]}^* C$. Pela definição, 7.31 temos que $\langle [l_1[k \leftarrow r]\sigma]_T, [r_1\sigma]_T \rangle$ é um par crítico de R módulo T , logo, por hipótese, existe $[w] \in \mathcal{T}_{RT}/T$ tal que $[l_1[k \leftarrow r]\sigma]_T \xrightarrow{[R]}^* [w]_T$ e $[r_1\sigma]_T \xrightarrow{[R]}^* [w]_T$.

Seja $W = [t[m_1 \leftarrow w\psi]]_T$ então por (iii) temos $A_1 \xrightarrow{[R]}^* [t[m_1 \leftarrow r_1\phi]]_T$

$\theta_V \equiv_T (\sigma\psi)_V \Rightarrow r_1\theta_V \equiv_T r_1(\sigma\psi)_V$, mas $Var(r_1) \subseteq Var(l_1) \subseteq V$, logo $r_1\theta_V = r_1\theta$ e $r_1(\sigma\psi)_V = r_1(\sigma\psi)$ e, conseqüentemente, $(r_1\theta \equiv_T r_1(\sigma\psi)) \Leftrightarrow (r_1(\phi \cup \phi') \equiv_T r_1(\sigma\psi))$, pois $\theta = \phi \cup \phi'$.

Mas, $Lvar(\phi') \cap Var(l_1) = \emptyset$, logo $Lvar(\phi') \cap Var(r_1) = \emptyset$ uma vez que $Var(r_1) \subseteq Var(l_1)$ e, portanto, $r_1\theta = r_1(\phi \cup \phi') = r_1\phi$ e conseqüentemente, $r_1\phi \equiv_T r_1(\sigma\psi) \xrightarrow{coro.3.13} t[m_1 \leftarrow r_1\phi] \equiv_T t[m_1 \leftarrow r_1(\sigma\psi)] \xrightarrow{def.7.4} [t[m_1 \leftarrow r_1\phi]]_T = [t[m_1 \leftarrow r_1(\sigma\psi)]]_T$

Mas, $[r_1\sigma]_T \xrightarrow{[R]}^* [w]_T \xrightarrow{def.7.4} r_1\sigma \xrightarrow{(R,T)}^* w \xrightarrow{coro.7.11} r_1\sigma\psi \xrightarrow{(R,T)}^* w\psi \xrightarrow{coro.7.9} [t[m_1 \leftarrow r_1\sigma\psi]]_T \xrightarrow{[R]}^* [t[m_1 \leftarrow w\psi]]_T = W$, logo $\boxed{A_1 \xrightarrow{[R]}^* W}$,

Por (iv) temos $A_2 \xrightarrow{[R]}^* [t[m_2 \leftarrow r_2\phi]]_T \xrightarrow{trivial} [t[m_1 \leftarrow t/m_1[m_2/m_1 \leftarrow r_2\phi]]]_T$

Note que por (vii) $[t/m_1[m_2/m_1 \leftarrow r_2\phi]]_T \xrightarrow{[R]}^* [l_1\phi[k \leftarrow r\phi']]_T \xrightarrow{prop.7.5} t/m_1[m_2/m_1 \leftarrow r_2\phi] \xrightarrow{(R,T)}^* l_1\phi[k \leftarrow r\phi'] \xrightarrow{coro.7.9} [t[m_1 \leftarrow t/m_1[m_2/m_1 \leftarrow r_2\phi]]]_T \xrightarrow{[R]}^* [t[m_1 \leftarrow l_1\phi[k \leftarrow r\phi']]]_T$.

$\theta_V \equiv_T (\sigma\psi)_V \Rightarrow r\theta_V \equiv_T r(\sigma\psi)_V$, mas $Var(r) \subseteq Var(l) \subseteq V$, logo $r\theta_V = r\theta$ e $r(\sigma\psi)_V = r(\sigma\psi)$ e, conseqüentemente, $(r\theta \equiv_T r(\sigma\psi)) \Leftrightarrow (r(\phi \cup \phi') \equiv_T r(\sigma\psi))$, pois $\theta = \phi \cup \phi'$.

Mas, $Lvar(\phi) \cap Var(l) = \emptyset$, logo $Lvar(\phi) \cap Var(r) = \emptyset$ uma vez que $Var(r) \subseteq Var(l)$ e portanto $r\theta = r(\phi \cup \phi') = r\phi'$ e, conseqüentemente, $\boxed{r\phi' \equiv_T r(\sigma\psi)}$ ***.

Ainda, $l_1[k \leftarrow r]\sigma\psi = l_1[k \leftarrow r(\sigma\psi)](\sigma\psi) = l_1(\sigma\psi)[k \leftarrow r(\sigma\psi)] \equiv_T l_1\phi[k \leftarrow r(\sigma\psi)] \xrightarrow{coro.3.13} l_1\phi[k \leftarrow r\phi'] \xrightarrow{coro.3.13} [t[m_1 \leftarrow l_1\phi[k \leftarrow r\phi']]]_T = [t[m_1 \leftarrow l_1[k \leftarrow r]\sigma\psi]]_T$

Finalmente, $[l_1[k \leftarrow r]\sigma]_T \xrightarrow{[R]}^* [w]_T \xrightarrow{\text{def.7.4}} l_1[k \leftarrow r]\sigma \xrightarrow{(R,T)}^* w \xrightarrow{\text{coro.7.11}} l_1[k \leftarrow r]\sigma\psi \xrightarrow{(R,T)}^* w\psi \xrightarrow{\text{coro.7.9}} [t[m_1 \leftarrow l_1[k \leftarrow r]\sigma\psi]]_T \xrightarrow{[R]}^* [t[m_1 \leftarrow w\psi]]_T = W$, logo $\boxed{A_2 \xrightarrow{[R]}^* W}$. ■

Definição 7.33. (Par Crítico Especial) *Seja $\langle [l_1[k \leftarrow r_2]\sigma]_T, [r_1\sigma]_T \rangle$ um par crítico de R módulo T . Dizemos que $\langle [l_1[k \leftarrow r_2]\sigma]_T, [r_1\sigma]_T \rangle$ é um par crítico especial de R módulo T caso haja uma regra $l \rightarrow r$ em R contida e protegida em $l_1 \rightarrow r_1$ na posição n e tal que $k \prec n$.* [PS81]

Lema 7.34. *Se para todo par crítico especial $\langle A, B \rangle$ de R módulo T existir C tal que $A \xrightarrow{[R]}^* C$ e $B \xrightarrow{[R]}^* C$ então para todo par crítico $\langle A', B' \rangle$ de R módulo T existe C' tal que $A' \xrightarrow{[R]}^* C'$ e $B' \xrightarrow{[R]}^* C'$.*

[PS81]

Demonstração: Suponha que $PC = \langle [l_1[k \leftarrow r_2]\sigma]_T, [r_1\sigma]_T \rangle$ seja um par crítico de R módulo T que não seja especial. Com certeza, existe alguma regra de reescrita em R tal que esteja contida e protegida em $l_1 \rightarrow r_1$ para alguma posição n (uma vez que $l_1 \rightarrow r_1$ está contida e protegida nela mesma).

Seja $l \rightarrow r$ qualquer regra em R tal que $l \rightarrow r$ está contida e protegida em $l_1 \rightarrow r_1$ na posição n (eventualmente $l = l_1$ e $r = r_1$ e neste caso $n = \langle \rangle$). Como PC não é um par crítico especial isto significa que $k \not\prec n$. Podemos ter duas situações, $(k \perp n)$ ou $(n \preceq k)$:

Situação 1 ($k \perp n$): Ora, $l_1 \rightarrow r_1$ é uma extensão protetora de $l \rightarrow r$ na posição n , logo $r_1 = l_1[n \leftarrow r]$ e, neste caso, não é difícil ver que existe C em $\mathcal{T}_{RT}/T$ tal que $[l_1[k \leftarrow r_2]\sigma]_T \xrightarrow{[R]}^* C$ e $[r_1\sigma]_T \xrightarrow{[R]}^* C$. Ou seja, nesta situação PC converge módulo T . O diagrama 7.7 ilustra este caso.

┐ Na demonstração original, presente no artigo de Peterson e Stickel [PS81], a situação $n \perp k$ é dita ser impossível de ocorrer pois, segundo os autores “se $n \perp k$, então como l está contida e protegida em l_1 então $l_1k \in \mathcal{V}$ que contradiz a definição de par crítico módulo T que se exige $k \in \text{sdom } l_1$ (isto é $l_1k \notin \mathcal{V}$).” Entretanto, pode-se ver que o fato de $l_1 = l^e$ não implica na necessidade de $l_1k = l^ek \in \mathcal{V}$. ┘

Situação 2 ($n \preceq k$): Podemos assumir, sem perda de generalidades, que n é o ancestral mais “próximo” de k tal que l_1/n está contido e protegido em l_1 . Isto é, n é tal que se existir algum n' , $n \prec n' \preceq k$, então não existirá $l' \rightarrow r' \in R$ que seja contida e protegida em $l_1 \rightarrow r_1$ na posição n' (se houvesse um tal n' , poderíamos escolher $l' \rightarrow r'$ ao invés de $l \rightarrow r$). Seja $m = k/n$, então $m \in \text{dom } l$ e $l/m = l_1/k$. Estamos assumindo, por hipótese, que σ é uma unificadora- T , elemento de um conjunto completo de unificadoras- T de l_1/k e l_2 em relação a $\text{Var}(l_1 \rightarrow r_1)$

Figura 7.7: Convergência do par crítico é garantida quando $k \perp n$.

Figura 7.8: Diagrama representativo da situação 2 ($n \preceq k$).

$\cup \text{Var}(l_2 \rightarrow r_2)$. Segue que $\langle [l(m \leftarrow r_2)\sigma], [r\sigma] \rangle$ além de ser um par crítico, fato este trivial uma vez que $l/m = l_1/k$ (observe a definição 7.31 de par crítico módulo T), ele é um par crítico especial (isto segue do fato de n ser o ancestral mais “próximo” de k) e portanto, por hipótese, existe $[c]_T$ tal que $[r\sigma]_T \xrightarrow{[R]}^* [c]_T$ e $[l(m \leftarrow r_2)\sigma]_T \xrightarrow{[R]}^* [c]_T$.

Seja $C = [l_1\sigma[n \leftarrow c]]_T$, então, como $l_1 \rightarrow r_1$ é uma extensão protetora de $l \rightarrow r$ na posição n temos que $r_1 = l_1[n \leftarrow r]$ e, conseqüentemente, $[r_1\sigma]_T = [l_1[n \leftarrow r]\sigma]_T \stackrel{\text{lema 2.49(iii)}}{=} [l_1\sigma[n \leftarrow r\sigma]]_T$, no parágrafo anterior concluímos que $\exists [c]_T$ tal que $[r\sigma]_T \xrightarrow{[R]}^* [c]_T \stackrel{\text{prop. 7.5}}{\iff} r\sigma \xrightarrow{(R,T)}^* c \stackrel{\text{coro. 7.9}}{\implies} [l_1\sigma[n \leftarrow r\sigma]]_T \xrightarrow{[R]}^* [l_1\sigma[n \leftarrow c]]_T = C$.

Por hipótese, l está contido e protegido em l_1 na posição n , isto é, l é subtermo de l_1 em n . E mais, pelas observações já feitas, como $l/m = l_1/k$ temos que $l_1[k \leftarrow r_2] = l_1[n \leftarrow l[m \leftarrow r_2]] \Rightarrow l_1[k \leftarrow r_2]\sigma = l_1[n \leftarrow l[m \leftarrow r_2]]\sigma \Leftrightarrow [l_1[k \leftarrow r_2]\sigma]_T = [l_1[n \leftarrow l[m \leftarrow$

$r_2]]\sigma]_T \stackrel{\text{lema 2.49(iii)}}{=} [l_1\sigma[n \leftarrow l[m \leftarrow r_2]\sigma]]_T$, nos parágrafos anteriores, concluímos que $\exists [c]_T$ tal que $[l[m \leftarrow r_2]\sigma]_T \xrightarrow{[R]}^* [c]_T \stackrel{\text{prop. 7.5}}{\iff} l[m \leftarrow r_2]\sigma \xrightarrow{(R,T)}^* c \stackrel{\text{coro. 7.9}}{\implies} [l_1\sigma[n \leftarrow l[m \leftarrow r_2]\sigma]]_T \xrightarrow{[R]}^* [l_1\sigma[n \leftarrow c]]_T = C$.

Ou seja, o par crítico de R módulo $T < [l_1[k \leftarrow r_2]\sigma]_T, [r_1\sigma]_T >$ converge. ■

O Teorema 7.32 nos permite realizar um teste do tipo do Knuth-Bendix para a verificação da completeza- T de qualquer sistema de reescrita de termos R noetheriano módulo T tal que seja compatível com T . O teste consiste em ir gerando pares críticos $< A, B >$, reduzindo cada um para suas formas normais, e então verifica-se se as formas normais (dos A 's e dos B 's, respectivos) são iguais. Se todo par crítico possuir como formas normais, dois termos iguais módulo T , então R será completo- T .

O lema anterior nos mostra que podemos considerar uma variação do teorema 7.32. Considerando apenas pares críticos especiais, este teorema continua válido, fato este útil porque nos casos de interesses práticos podemos checar a completeza- T apenas testando pares críticos especiais.

Entretanto, o teorema 7.32, mesmo considerando-se sua variante apresentada acima (considerando-se pares críticos especiais ao invés de pares críticos) possui ainda um outro problema para que possamos aplicá-lo em casos práticos. Esse problema, é que devemos saber a priori se R é compatível com T . Ora, precisamos então, encontrar condições suficientes para que possamos afirmar que R seja compatível com T , e é isto que o próximo teorema 7.36, nos apresenta.

Definição 7.35. Dizemos que uma equação $l = r$ possui variáveis idênticas e não repetidas se l e r forem lineares e $\text{Var}(l) = \text{Var}(r)$.

[PS81]

Teorema 7.36. Suponha que T seja um conjunto de equações e R um sistema de reescrita de termos. Suponha que toda equação em T possua variáveis idênticas e não repetidas. Suponha também que sempre que $l = r \in T$ e $l_1 \rightarrow r_1 \in R$ tiveram tido suas variáveis padronizadas a priori e $q \in \text{sdom } l$, existirá $q \neq \langle \rangle$ tal que l/q e l_1 sejam unificáveis- T , disto segue que existe $l_2 \rightarrow r_2 \in R$ e uma substituição γ tais que $l[q \leftarrow l_1] \equiv_T l_2\gamma$ e $[l[q \leftarrow r_1]]_T \xrightarrow{[R]}^* [r_2\gamma]_T$. Então R será compatível com T .

[PS81]

Antes de apresentarmos uma demonstração deste teorema, faremos algumas observações.

O teorema 7.36 não se aplicará para conjuntos T de equações que possuírem equações tais como $x \cdot 0 = 0$, $x \cdot x^{-1} = 1$, ou ainda $x \cdot x = x$. Para mostrarmos que o teorema 7.36 necessita impor a restrição das variáveis idênticas e não repetidas, apresentamos um contra-exemplo.

Exemplo 7.37. *Seja $T = \{x \cdot x = x\}$ e $R = \{b \rightarrow c\}$, onde x é uma variável e b, c são constantes. Este exemplo satisfaz todas as condições impostas pelo teorema 7.36, exceto uma delas, pois a equação em T não é linear. Contudo, R não é compatível com T e podemos verificar isto se considerarmos os termos $s = (a \cdot b) \cdot (a \cdot b)$ e $t = a \cdot c$, onde a é uma constante. Note que, de fato, $s \xrightarrow{(R,T)} t$ pois:*

$$[s]_T = \{(a \cdot b) \cdot (a \cdot b), (a \cdot b)\}$$

$$[t]_T = \{(a \cdot c)\}$$

Tomemos $u \in [s]_T$, $u = (a \cdot b)$, $n = \langle 2 \rangle$ e $\theta = \emptyset$, logo $u/n = b = b\theta$ e $u[n \leftarrow c\theta] = u[\langle 2 \rangle \leftarrow c] = a \cdot c \in [t]_T$ e, portanto, pela definição 7.2 $(a \cdot b) \cdot (a \cdot b) \xrightarrow{(R,T)} (a \cdot c)$ $(a \cdot b, b \rightarrow c, \langle 2 \rangle, \emptyset)$ ou simplesmente $s \xrightarrow{(R,T)} t$.

Logo, pela definição 7.29, se R fosse compatível com T , então deveria existir um nó m em dom s tal que $s/m \equiv b$ e $[t]_T \xrightarrow{[R]}^* [s[m \leftarrow c]]_T$.

Para $m = \langle 1, 2 \rangle$ ou $m = \langle 2, 2 \rangle$ temos $s/m = b$.

Se R fosse compatível com T teríamos, então, para $m = \langle 1, 2 \rangle$ ou $m = \langle 2, 2 \rangle$ que:

$$[t]_T \xrightarrow{[R]}^* [s[m \leftarrow c]]_T$$

Note que, se $m = \langle 1, 2 \rangle$ então $s[m \leftarrow c] = (a \cdot c) \cdot (a \cdot b)$ e, neste caso, temos:

$$[(a \cdot c) \cdot (a \cdot b)]_T = \{(a \cdot c) \cdot (a \cdot b)\}.$$

Note que, se $m = \langle 2, 2 \rangle$ então $s[m \leftarrow c] = (a \cdot b) \cdot (a \cdot c)$ e, neste caso, temos:

$$[(a \cdot b) \cdot (a \cdot c)]_T = \{(a \cdot b) \cdot (a \cdot c)\}$$

Logo, pela definição 7.4 e pelo fato de $[t]_T = \{(a \cdot c)\}$, se R fosse compatível com T teríamos que:

$$a \cdot c \xrightarrow{(R,T)}^* (a \cdot b) \cdot (a \cdot c) \text{ ou}$$

$$a \cdot c \xrightarrow{(R,T)}^* (a \cdot c) \cdot (a \cdot b)$$

Mas, não é difícil ver, pela definição 7.2, que nenhuma das duas situações acima é possível e portanto, de fato, R não é compatível com T .

[PS81]

Entretanto, e felizmente, algumas equações que aparecem em muitas situações de interesse prático tais como as que expressam as leis associativa e comutativa satisfazem a restrição

de possuírem as variáveis idênticas e não repetidas. A outra restrição, entretanto é menos restritiva, no sentido que ela não se impõe sobre toda e qualquer equação, como por exemplo, na comutatividade não há nó q tal que $q \neq \langle \rangle$ e o símbolo que ocorra em q não seja uma variável, isto é, no caso da comutatividade $(x \cdot y = y \cdot x)$ se $q \neq \langle \rangle$ então $q \notin \text{sdom}(x \cdot y)$.

Demonstração do Teorema 7.36: Pela definição 7.29 devemos mostrar que sempre que $t \xrightarrow{(R,T)}$ s , então existe um nó $m \in \text{dom } t$, uma substituição ϕ , e uma regra de reescrita $l_2 \rightarrow r_2 \in R$ tais que

$$t/m \equiv_{\overline{T}} l_2\phi \quad \text{e} \quad [s]_T \xrightarrow{[R]}^* [t[m \leftarrow r_2\phi]]_T.$$

Se t e u são termos e n é um número inteiro não negativo, então $t \xrightarrow{T}^n u$ significa que existe termos $t_0, \dots, t_n$ tais que $t = t_0, u = t_n$ e se $0 \leq i < n$, então $t_i \xrightarrow{T} t_{i+1}$. Vamos definir também,

$$P = \{p \in \mathbb{N} \mid \text{se } t \text{ e } u \text{ são termos tais que } t \xrightarrow{T}^p u, \text{ e } \exists k \in \text{dom } u, l_1 \rightarrow r_1 \in R, \text{ e uma substituição } \theta \text{ tais que } u/k \equiv_{\overline{T}} l_1\theta, \text{ então } \exists m \in \text{dom } t, l_2 \rightarrow r_2 \in R, \text{ e uma substituição } \phi \text{ tais que } t/m \equiv_{\overline{T}} l_2\phi \text{ e } [u[k \leftarrow r_1\theta]]_T \xrightarrow{[R]}^* [t[m \leftarrow r_2\phi]]_T\}$$

Para completarmos a demonstração é suficiente mostrarmos que $P = \mathbb{N}$. Pois se $t \xrightarrow{(R,T)}$ $s(u, l_1 \rightarrow r_1, k, \theta)$, então pela definição 7.2 $u \equiv_{\overline{T}} t, u/k = l_1\theta$, e $s \equiv_{\overline{T}} u[k \leftarrow r_1\theta]$ (note que, $s \equiv_{\overline{T}} u[k \leftarrow r_1\theta] \Leftrightarrow [s]_T = [u[k \leftarrow r_1\theta]]_T$). E ainda, se $P = \mathbb{N}$ então $\exists m \in \text{dom } t, l_2 \rightarrow r_2 \in R$, e existe uma substituição ϕ tais que $t/m \equiv_{\overline{T}} l_2\phi$ e $[u[k \leftarrow r_1\theta]]_T \xrightarrow{[R]}^* [t[m \leftarrow r_2\phi]]_T$: logo, R será compatível com T .

Daqui até o final da demonstração do teorema, t e u serão termos, p será um número em $\mathbb{N}$, $k \in \text{dom } u, l_1 \rightarrow r_1 \in R$, e θ será uma substituição tais que $t \xrightarrow{T}^p u$ e $u/k \equiv_{\overline{T}} l_1\theta$ ¹. Será então, nossa tarefa, encontrar $m \in \text{dom } t, l_2 \rightarrow r_2 \in R$, e uma substituição ϕ tais que $t/m \equiv_{\overline{T}} l_2\phi$ e $[u[k \leftarrow r_1\theta]]_T \xrightarrow{[R]}^* [t[m \leftarrow r_2\phi]]_T$.

Caso 1: Suponha $p = 0$, logo $t \xrightarrow{T}^0 u$, isto é, $t = t_0 = u$. Tome $m = k, l_2 \rightarrow r_2 = l_1 \rightarrow r_1$, e $\phi = \theta$. Então, $t/m = u/k$, pois $t = u$ e $m = k$, e $u/k \stackrel{\text{hip.1}}{\equiv_{\overline{T}}} l_1\theta = l_2\phi$, e $u[k \leftarrow r_1\theta] = t[m \leftarrow r_2\phi]$, logo $[u[k \leftarrow r_1\theta]]_T \xrightarrow{[R]}^* [t[m \leftarrow r_2\phi]]_T$.

Caso 2: Suponha $p = 1$. Então, $u \xrightarrow{T}^1 t(l = r, n, \psi)$ ou equivalentemente $u \xrightarrow{T} t(l = r, n, \psi)$, logo $[l = r \in T]$ ², $[u/n = l\psi]$ ³, e $[t = u[n \leftarrow r\psi]]$ ⁴. O restante da demonstração deste teorema será dividido em casos, dependendo da relação de posicionamento existente entre n e k .

Caso 2.1: Suponha que $n \perp k$. Tome $m = k, l_2 \rightarrow r_2 = l_1 \rightarrow r_1$, e $\phi = \theta$. Então,

$$t/m \stackrel{m=k, \text{hip.4}}{=} u[n \leftarrow r\psi]/k \stackrel{\text{lema2.32(iv)}}{=} u/k \stackrel{\text{hip.1}}{=} l_1\theta \stackrel{\text{hip.caso2.1}}{=} l_2\phi,$$

e

$$t[m \leftarrow r_2\phi] \stackrel{\text{hips.caso 2.1} + \text{hip.4}}{=} u[n \leftarrow r\psi][k \leftarrow r_1\theta]$$

e pela hipótese 2 + corolário 3.11 + corolário 3.13 temos que $u[n \leftarrow r\psi][k \leftarrow r_1\theta] \stackrel{=}{=} u[n \leftarrow l\psi][k \leftarrow r_1\theta] \stackrel{\text{hip.3}}{=} u[k \leftarrow r_1\theta]$. Mas,

$$t[m \leftarrow r_2\phi] \stackrel{=}{=} u[k \leftarrow r_1\theta] \Leftrightarrow [t[m \leftarrow r_2\phi]]_T = [u[k \leftarrow r_1\theta]]_T \Rightarrow [u[k \leftarrow r_1\theta]] \xrightarrow{[R]}^* [t[m \leftarrow r_2\phi]]_T$$

Caso 2.2: Suponha $k \preceq n$. Tome $m = k, l_2 \rightarrow r_2 = l_1 \rightarrow r_1$, e $\phi = \theta$. Então,

$$t/m \stackrel{m=k, \text{hip.4}}{=} u[n \leftarrow r\psi]/k \stackrel{\text{lema2.35(ii)}}{=} (u/k)[n/k \leftarrow r\psi]$$

e pela hipótese 2 + corolário 3.11 + corolário 3.13 temos que $(u/k)[n/k \leftarrow r\psi] \stackrel{=}{=} (u/k)[n/k \leftarrow l\psi]$

$$\stackrel{\text{lema2.35(ii)}}{=} u[n \leftarrow l\psi]/k \stackrel{\text{hip.3}}{=} u/k \stackrel{\text{hip.1}}{=} l_1\theta \stackrel{\text{hip.caso2.2}}{=} l_2\phi$$

e

$$t[m \leftarrow r_2\phi] \stackrel{\text{hip.4}}{=} u[n \leftarrow r\psi][m \leftarrow r_2\phi] \stackrel{\text{hip.m=k}}{=} u[n \leftarrow r\psi][k \leftarrow r_2\phi] \stackrel{\text{lema2.35(iii)}}{=} u[k \leftarrow r_2\phi] \stackrel{\text{hip.caso2.2}}{=} u[k \leftarrow r_1\theta] \text{ e portanto } [t[m \leftarrow r_2\phi]]_T = [u[k \leftarrow r_1\theta]]_T,$$

e, conseqüentemente, $[u[k \leftarrow r_1\theta]]_T \xrightarrow{[R]}^* [t[m \leftarrow r_2\phi]]_T$.

Caso 2.3: Suponha $n \prec k$. Seja $q = k/n$, isto é, $n \cdot q = k$

$$l_1\theta \stackrel{\text{hip.1}}{=} u/k \stackrel{\text{lema2.35(i)}}{=} (u/n)/(k/n) \stackrel{\text{hip.3}}{=} l\psi/(k/n) = l\psi/q \quad ^5.$$

O restante da demonstração deste caso será dividido por casos, dependendo da localização de q em $l\psi$ (veja o lema 2.48).

Caso 2.3.1 ($q \notin \text{sdom } l$): neste caso existem nós m_1 e p_1 tais que $q = m_1 \cdot p_1, m_1 \in \text{dom } l, lm_1 = v \in \mathcal{V}$, e $p_1 \in \text{dom } v\psi$. Então pelo lema 2.48(ii), $l\psi/q = v\psi/p_1$ ⁶.

Como $l = r$ possuem variáveis idênticas e não repetidas então há exatamente um único nó $m_2 \in \text{dom } r$ tal que $rm_2 = v$. Seja $m = n \cdot m_2 \cdot p_1, l_2 \rightarrow r_2 = l_1 \rightarrow r_1$, e $\phi = \theta$ ⁷.

Temos $t \stackrel{\text{hip.4}}{=} u[n \leftarrow r\psi]$. Observe que $n \cdot m_2 \cdot p_1 \in \text{dom } u[n \leftarrow r\psi]$, uma vez que $rm_2 = v, v \in \mathcal{V}$ e $p_1 \in \text{dom } v\psi$. Logo, $n \cdot m_2 \cdot p_1$ pertence ao $\text{dom } t$, pois $t = u[n \leftarrow r\psi]$. Como tomamos $m = n \cdot m_2 \cdot p_1$ temos que $t/m = u[n \leftarrow r\psi]/n \cdot m_2 \cdot p_1 \stackrel{\text{lema2.32(ii)}}{=} r\psi/m_2 \cdot p_1$. Como $m_2 \in \text{dom } r, rm_2 = v$, temos que $r\psi/m_2 = v\psi$, logo $r\psi/m_2 \cdot p_1 = v\psi/p_1 \stackrel{6}{=} l\psi/q \stackrel{5}{=} l_1\theta \stackrel{7}{=} l_2\phi$.

e

$$t[m \leftarrow r_2\phi] \stackrel{4}{=} u[n \leftarrow r\psi][m \leftarrow r_2\phi] \stackrel{7}{=} u[n \leftarrow r\psi][n \cdot m_2 \cdot p_1 \leftarrow r_1\theta] \stackrel{\text{lema2.32(iii)}}{=} u[n \leftarrow r\psi][m_2 \cdot p_1 \leftarrow r_1\theta]$$

como $rm_2 = v$ e $p_1 \in \text{dom } v\psi$, temos que:

$$u[n \leftarrow r\psi[m_2 \cdot p_1 \leftarrow r_1\theta]] = u[n \leftarrow r\psi[m_2 \leftarrow v\psi[p_1 \leftarrow r_1\theta]]]$$

isto é,

$$\boxed{t[m \leftarrow r_2\phi] = u[n \leftarrow r\psi[m_2 \leftarrow v\psi[p_1 \leftarrow r_1\theta]]]}^8.$$

Seja $t' = v\psi[p_1 \leftarrow r_1\theta]$ então $u[n \leftarrow r\psi[m_2 \leftarrow v\psi[p_1 \leftarrow r_1\theta]]] = u[n \leftarrow r\psi[m_2 \leftarrow t']]$. Vamos definir ψ' por $w\psi' = w\psi$ se $w \in \mathcal{V} - \{v\}$ e $v\psi' = t'$ caso contrário. Pela definição dada de ψ' e como cada variável em r ocorre uma única vez e ainda, como $rm_2 = v$ então temos: $r\psi[m_2 \leftarrow t'] = r\psi'$ (veja a figura 7.9).

Figura 7.9: $r\psi[m_2 \leftarrow t'] = r\psi'$.

Portanto, pela identidade 8 temos:

$$\begin{aligned} t[m \leftarrow r_2\phi] &= u[n \leftarrow r\psi'] \\ \text{Como, } l = r \in T \Rightarrow l \equiv_T r &\xrightarrow{\text{coro.3.11}} l\psi' \equiv_T r\psi' \xrightarrow{\text{coro.3.13}} u[n \leftarrow l\psi'] \equiv_T u[n \leftarrow r\psi'] \Leftrightarrow u[n \leftarrow r\psi'] \equiv_T \\ u[n \leftarrow l\psi'], &\text{ pois } \equiv_T \text{ é simétrica.} \end{aligned}$$

Pela definição de ψ' e t' e pelo fato que $lm_1 = v$ e ainda como cada variável em l ocorre uma única vez então temos $l\psi' = l\psi[m_1 \leftarrow t'] = l\psi[m_1 \leftarrow v\psi[p_1 \leftarrow r_1\theta]]$ (veja a figura 7.10):

Figura 7.10: $l\psi' = l\psi[m_1 \leftarrow t']$.

Logo, $u[n \leftarrow l\psi'] = u[n \leftarrow l\psi[m_1 \leftarrow v\psi[p_1 \leftarrow r_1\theta]]]$, como $lm_1 = v$ e $p_1 \in \text{dom } v\psi$ temos:

$$\begin{aligned} u[n \leftarrow l\psi[m_1 \leftarrow v\psi[p_1 \leftarrow r_1\theta]]] &= u[n \leftarrow l\psi[m_1 \cdot p_1 \leftarrow r_1\theta]] \stackrel{\text{lema2.32(iii)}}{=} u[n \leftarrow l\psi][n \cdot m_1 \cdot p_1 \leftarrow \\ r_1\theta] &\stackrel{3}{=} u[n \cdot m_1 \cdot p_1 \leftarrow r_1\theta] \end{aligned}$$

Mas $q = m_1 \cdot p_1$ (hipótese do caso 2.3.1) e $q = k/n$ (hipótese do caso 2.3), logo, $n \cdot m_1 \cdot p_1 =$

$n \cdot q = n \cdot (k/n) \stackrel{\text{lema 2.34(i)}}{=} k$ e, portanto,

$$u[n \cdot m_1 \cdot p_1 \leftarrow r_1 \theta] = u[k \leftarrow r_1 \theta].$$

Acabamos de mostrar que, neste caso 2.3.1, $t/m \stackrel{=}{=} l_2 \phi$ e $t[m \leftarrow r_2 \phi] \stackrel{=}{=} u[k \leftarrow r_1 \theta]$ e, portanto,

$$[u[k \leftarrow r_1 \theta]]_T \xrightarrow{[R]}^* [t[m \leftarrow r_2 \phi]]_T$$

Caso 2.3.2 ($q \in \text{sdom } l$): pelo lema 2.48(i), $l\psi/q = (l/q)\psi$. Então por 5 temos:

$$\boxed{l_1 \theta \stackrel{=}{=} (l/q)\psi.}^9$$

Note que, por hipótese, $l_1 \rightarrow r_1$ e $l = r$ tiveram suas variáveis padronizadas à priori, logo podemos assumir, sem perda de generalidade, que as variáveis $\text{Var}(\psi) \cap \text{Var}(\langle l_1, r_1 \rangle) = \emptyset$ e $\text{Var}(\theta) \cap \text{Var}(\langle l, r \rangle) = \emptyset$. Então a equação 9, acima, nos mostra que l_1 e l/q são unificáveis- T com unificadora- T $\theta \cup \psi$ (observe que, pela definição 2.54, $\theta \cup \psi$ só está definida quando $\text{Lvar}(\theta) \cap \text{Lvar}(\psi) = \emptyset$). Pelas hipóteses do enunciado deste teorema existem $l_2 \rightarrow r_2 \in R$ e uma substituição γ tais que $l[q \leftarrow l_1] \stackrel{=}{=} l_2 \gamma$ e $[l[q \rightarrow r_1]]_T \xrightarrow{[R]}^* [r_2 \gamma]$. (Observe que, pelas

$$\text{hipóteses do caso 2.3, } \left\{ \begin{array}{l} n \prec k \Rightarrow n \neq k \\ e \\ q = k/n \end{array} \right\} \Rightarrow q \neq \langle \rangle.$$

Seja $m = n$ e $\phi = \gamma \circ (\psi \cup \theta) \stackrel{\text{notac.}}{=} \gamma(\psi \cup \theta)$. Então como $m = n$ e por 4, temos:

$$t/m = u[n \leftarrow r\psi]/n = u[n \leftarrow r\psi]/(n \cdot \langle \rangle),$$

pelo lema 2.32(ii), fazendo no lema, $p = \langle \rangle$, temos: $u[n \leftarrow r\psi]/(n \cdot \langle \rangle) = r\psi / \langle \rangle = r\psi$.

Pelo lema 3.10 temos: $r\psi \stackrel{=}{=} l\psi$ pois $(l = r) \in T$.

Pelo parágrafo anterior concluímos que l_1 e l/q são unificáveis- T com unificadora- T $\theta \cup \psi$. Isto é, $\boxed{l/q(\psi \cup \theta) \stackrel{=}{=} l_1(\psi \cup \theta)}$, mas $\text{Var}(\psi) \cap \text{Var}(l_1) = \emptyset$ e $\text{Var}(\theta) \cap \text{Var}(l) = \emptyset$, logo $l_1(\psi \cup \theta) = l_1 \theta$ e $(l/q)(\psi \cup \theta) = (l/q)\psi$.

Portanto, $(l/q(\psi \cup \theta) \stackrel{=}{=} l_1(\psi \cup \theta)) \Rightarrow ((l/q)\psi \stackrel{=}{=} l_1 \theta) \stackrel{\text{lema 2.49(ii)}}{\iff} (l\psi/q \stackrel{=}{=} l_1 \theta) \stackrel{\text{lema 3.12}}{\implies} (l\psi \stackrel{=}{=} l\psi[q \leftarrow l_1 \theta])$, mas $l\psi[q \leftarrow l_1 \theta] = l[q \leftarrow l_1](\psi \cup \theta)$.

Mas, $(l[q \leftarrow l_1] \stackrel{\text{hip.}}{=} l_2 \gamma) \Rightarrow (l[q \leftarrow l_1](\psi \cup \theta) \stackrel{=}{=} (l_2 \gamma)(\psi \cup \theta)) \Leftrightarrow (l[q \leftarrow l_1](\psi \cup \theta) \stackrel{=}{=} l_2(\gamma(\psi \cup \theta)))$, e ainda, $l_2(\gamma(\psi \cup \theta)) = l_2 \phi$, pois por hipótese $\phi = l(\psi \cup \theta)$.

Portanto, acabamos de mostrar que, neste subcaso 2.3.2, $t/m \stackrel{=}{=} l_2 \phi$.

Note que, $u[k \leftarrow r_1 \theta] = u[n \leftarrow u/n[k/n \leftarrow r_1 \theta]]$ (veja a figura 7.11)

logo, $[u[k \leftarrow r_1 \theta]]_T = [u[n \leftarrow u/n[k/n \leftarrow r_1 \theta]]]_T$, mas $q = k/n$ (hipótese do caso 2.3) e $u/n = l\psi$ (hipótese do caso 2), então: $[u[n \leftarrow u/n[k/n \leftarrow r_1 \theta]]]_T = [u[n \leftarrow l\psi[q \leftarrow r_1 \theta]]]_T = [u[n \leftarrow l[q \leftarrow r_1](\psi \cup \theta)]]_T$. Por hipótese do teorema temos:

$$\begin{array}{c} [l[q \leftarrow r_1]]_T \xrightarrow{[R]}^* [r_2 \gamma]_T \\ \xleftrightarrow{\text{prop. 7.5}} l[q \leftarrow r_1] \xrightarrow{[R, T]}^* r_2 \gamma \end{array}$$

Figura 7.11: $u[k \leftarrow r_1 \theta] = u[n \leftarrow u/n[k/n \leftarrow r_1 \theta]]$.

$$\begin{aligned} & \xrightarrow{\text{coro.7.11}} l[q \leftarrow r_1](\psi \cup \theta) \xrightarrow{(R,T)^*} r_2 \gamma(\psi \cup \theta) \\ & \xrightarrow{\text{coro.7.9}} [u[n \leftarrow l[q \leftarrow r_1](\psi \cup \theta)]]_T \xrightarrow{(R)^*} [u[n \leftarrow r_2 \sigma(\psi \cup \theta)]]_T \end{aligned}$$

Temos que $t \stackrel{4}{=} u[n \leftarrow r\psi]$, logo $t[m \leftarrow r_2 \phi] = u[n \leftarrow r\psi][m \leftarrow r_2 \phi]$,
mas, por construção, neste subcaso 2.3.2, $\phi = \gamma(\psi \cup \theta)$, logo:

$$u[n \leftarrow r_2 \gamma(\psi \cup \theta)] = u[n \leftarrow r_2 \phi] \quad (1)$$

também foi adotado, neste subcaso 2.3.2, $m = n$, logo:

$$t[m \leftarrow r_2 \phi] = u[n \leftarrow r\psi][n \leftarrow r_2 \phi] = u[n \leftarrow r_2 \phi] \quad (2)$$

De (1) e (2) temos que:

$$\begin{aligned} u[n \leftarrow r_2 \gamma(\psi \cup \theta)] &= t[m \leftarrow r_2 \phi] \\ &\Rightarrow \\ [u[n \leftarrow r_2 \gamma(\psi \cup \theta)]]_T &= [t[m \leftarrow r_2 \phi]]_T. \end{aligned}$$

Portanto, acabamos de mostrar que, neste subcaso 2.3.2, $[u[k \leftarrow r_1 \theta]]_T \xrightarrow{(R)^*} [t[m \leftarrow r_2 \phi]]_T$.

Caso 3: Suponha que $p > 1$. Vamos demonstrar por indução finita.

Base: $p = 1$.

Hipótese de indução: vale para $q < p$.

Tese: vale para $q = p$

Demonstração da indução: Dem. da base ($p = 1$): a demonstração da base já está feita, pois é justamente a demonstração do Caso 2.

Tomemos $t \xrightarrow{P} u$, logo $\exists t_2 : t \xrightarrow{1} t_2 \xrightarrow{P-1} u$.

Pela hipótese de indução, $p - 1 \in P$, logo existem $m_2 \in \text{dom } t_2, l_3 \rightarrow r_3 \in R$ e uma substituição ϕ_2 tais que:

$$t_2/m_2 \stackrel{=}{=} l_3 \phi_2$$

e

$$[u[k \leftarrow r_1\theta]]_T \xrightarrow{[R]}^* [t_2[m_2 \leftarrow r_3\phi_2]]_T.$$

Como $1 \in P$ então existem $m \in \text{dom } t, l_2 \rightarrow r_2 \in R$ e substituição ϕ tais que:

$$t/m \equiv_{\overline{T}} l_2\phi$$

e

$$[t_2[m_2 \leftarrow r_3\phi_2]]_T \xrightarrow{[R]}^* [t[m \leftarrow r_2\phi]]_T.$$

Logo, pela transitividade da relação $\xrightarrow{[R]}^*$ temos que:

$$[u[k \leftarrow r_1\theta]]_T \xrightarrow{[R]}^* [t[m \leftarrow r_2\phi]]_T, \quad \text{logo vale para } q = p. \quad \blacksquare$$

Na próxima seção, discutiremos a aplicação dos resultados encontrados nesta seção 7.2 para teorias equacionais que consistem das leis associativas e comutativas, definidas a seguir.

7.3 Compatibilidade com T para Teorias Associativas-Comutativas

Definição 7.38. *Seja f um símbolo funcional qualquer e x, y, z variáveis duas a duas distintas. Uma lei associativa é uma equação da forma $f(x, f(y, z)) = f(f(x, y), z)$. Uma lei comutativa é uma equação da forma $f(x, y) = f(y, x)$.*

[PS81]

Definição 7.39. *Uma teoria equacional $Th(T)$ é uma teoria associativa-comutativa ou simplesmente teoria-AC quando todo elemento de T ou é uma lei associativa ou é uma lei comutativa e todo símbolo funcional que ocorre em T é associativo-comutativo. Neste caso, dizemos simplesmente que T é uma teoria-AC.*

[PS81]

Definição 7.40. *Seja T uma teoria-AC, sendo $\mathcal{F}$ seu conjunto de símbolos funcionais. Definimos então:*

$AC(T) = \{f \in \mathcal{F} \text{ tal que pelo menos uma variante de } f(x, f(y, z)) = f(f(x, y), z) \text{ está em } T \text{ e pelo menos uma variante de } f(x, y) = f(y, x) \text{ está em } T\}.$

[PS81]

Definição 7.41. *Seja R um sistema de reescrita de termos e T uma teoria-AC. Definimos então:*

$$R(AC(T)) = \{f(v, l) \rightarrow f(v, r) \mid f \in AC(T), l \rightarrow r \in R, v \in \mathcal{V} - Var(l \rightarrow r) \text{ e } l < > = f\}.$$

e

$$R_T^e = R(AC(T)) \cup R.$$

[PS81]

Teorema 7.42. *Sejam T uma teoria-AC e R um sistema de reescrita de termos. Então R_T^e é compatível com T .*

[PS81]

Demonstração: Vamos mostrar que R_T^e é compatível com T usando o teorema 7.36 e, portanto, precisamos impor que toda equação em T possua variáveis idênticas e não repetidas. Suponha que $l = r \in T, l_1 \rightarrow r_1 \in R_T^e, q \in sdom\ l, q \neq < >, \text{ e } l/q \text{ e } l_1$ sejam unificáveis- T . Note que $l = r$ deve ser uma lei associativa uma vez que não há um tal q para leis comutativas. Portanto, para demonstrarmos o teorema vamos adotar que $l = f(x, f(y, z))$ e $r = f(f(x, y), z)$. Se adotássemos $r = f(x, f(y, z))$ e $l = f(f(x, y), z)$, a demonstração seria similar. Então $l/q = f(y, z)$, mas como l/q é unificável- T com l_1 , l_1 deve ser da forma $f(t_1, t_2)$ para algum termo t_1 e para algum termo t_2 . O restante da demonstração será feita por casos, dependendo da localização da regra de reescrita $l_1 \rightarrow r_1$ em R_T^e .

Caso 1: Suponha $l_1 \rightarrow r_1 \in R$. Como l_1 é da forma $f(t_1, t_2)$ então existe $l_2 \rightarrow r_2 \in R(AC(T))$ da forma $f(v, l_1) \rightarrow f(v, r_1)$.

Seja $\gamma = \{v \leftarrow x\}$, então $l[q \leftarrow l_1] = f(x, l_1) = l_2\gamma$ e $l[q \leftarrow r_1] = f(x, r_1) = r_2\gamma$.

Logo, $l[q \leftarrow l_1] \equiv_T l_2\gamma$ e $[l[q \leftarrow r_1]]_T \xrightarrow{[R]}^* [r_2\gamma]_T$.

Caso 2: Suponha $l_1 \rightarrow r_1 \in R(AC(T))$. Então, $l_1 \rightarrow r_1$ é da forma $f(v, l') \rightarrow f(v, r')$, onde $f \in AC(T), l' \rightarrow r' \in R$, e $v \in \mathcal{V} - Var(l' \rightarrow r')$.

Sejam $l_2 \rightarrow r_2 = l_1 \rightarrow r_1$ (isto é, $l_2 = l_1$ e $r_2 = r_1$) e $\gamma = \{v \leftarrow f(x, v)\}$. Então,

$$l[q \leftarrow l_1] = f(x, f(v, l'))$$

mas $\langle f(x, f(v, z)), f(f(x, v), z) \rangle \in \xrightarrow{T}$, logo $\langle f(x, f(v, l')), f(f(x, v), l') \rangle \in \xrightarrow{T}$ e, conseqüentemente, $\langle f(x, f(v, l')), f(f(x, v), l') \rangle \in \xrightarrow{T}^* \stackrel{\text{notac.}}{=} \overline{\overline{\quad}}$.

$$f(f(x, v), l') = f(v, l')\gamma = l_2\gamma.$$

Ou seja, acabamos de mostrar que $f(x, f(v, l')) \equiv_T l_2\gamma$.

Analogamente temos:

$$l[q \leftarrow r_1] = f(x, f(v, r')) \equiv_T f(f(x, v), r') = f(v, r')\gamma = r_2\gamma.$$

Logo, $[l[q \leftarrow r_1]]_T = [r_2\gamma]_T$ e, conseqüentemente, $[l[q \leftarrow r_1]]_T \xrightarrow{[R]}^* [r_2\gamma]_T$.

Portanto, pelo teorema 7.36, acabamos de mostrar que R_T^e é compatível com T . ■

7.4 Um Algoritmo de Completação Módulo T de SRT's.

Primeiro vamos mostrar que existe um algoritmo que reduz qualquer termo para sua forma normal.

Lema 7.43. *Seja R um sistema de reescrita de termos e T um conjunto de equações. Suponha que R seja compatível com T , $A \in \mathcal{T}_{RT}/T$, e $s \in A$. Então, A será redutível se e somente se existir $m \in \text{dom } s$, uma regra de reescrita $l \rightarrow r \in R$ e uma substituição θ tais que $s/m \equiv l\theta$. Nestas condições, $A \xrightarrow{[R]} [s[m \leftarrow r\theta]]_T$.*

[PS81]

Demonstração: ($\Leftarrow$ /se)

Suponha que existam $m \in \text{dom } s$, uma regra de reescrita $l \rightarrow r \in R$ e uma substituição θ tais que $s/m \equiv l\theta$. Seja $t = s[m \leftarrow l\theta]$. Pelo lema 3.12 $s \equiv s[m \leftarrow l\theta]$, logo $t \equiv s$ e, conseqüentemente, $t \in A$. Ainda mais, $t/m = l\theta$ e $l \rightarrow r \in R$, logo, como consequência imediata da definição 7.2, $l \xrightarrow{(R,T)} r$.

$l \xrightarrow{(R,T)} r \xrightarrow{\text{lema 7.10}} l\theta \xrightarrow{(R,T)} r\theta \xrightarrow{\text{lema 7.8}} [t[m \leftarrow l\theta]]_T \xrightarrow{[R]} [t[m \leftarrow r\theta]]_T \Leftrightarrow [t]_T \xrightarrow{[R]} [t[m \leftarrow r\theta]]_T$
pois $t[m \leftarrow l\theta] = t$ uma vez que $t/m = l\theta$.

Mas $t \in A$, logo $[t]_T = A$, então $A \xrightarrow{[R]} [t[m \leftarrow r\theta]]_T$.

Como $t[m \leftarrow r\theta] = s[m \leftarrow r\theta]$, pois $t = s[m \leftarrow l\theta]$, então $A \xrightarrow{[R]} [s[m \leftarrow r\theta]]_T$. Portanto A é redutível.

($\Rightarrow$ /somente se)

Suponha que A seja redutível. Então existirá $t \in A$ e um termo t' tais que $t \xrightarrow{(R,T)} t'$. Como $s \in A$, então $s \equiv t$ e pela proposição 7.3 $s \xrightarrow{(R,T)} t'$. Como R é compatível com T existe $m \in \text{dom } s$, uma substituição θ e uma regra de reescrita $l \rightarrow r \in R$ tais que $s/m \equiv l\theta$ e ainda $s \xrightarrow{\text{lema 3.12}} s[m \leftarrow l\theta]$ e como $l \xrightarrow{(R,T)} r \xrightarrow{\text{lema 7.10}} l\theta \xrightarrow{(R,T)} r\theta \xrightarrow{\text{lema 7.8}} [s[m \leftarrow l\theta]]_T \xrightarrow{[R]} [s[m \leftarrow r\theta]]_T$ e portanto $A \xrightarrow{[R]} [s[m \leftarrow r\theta]]_T$ pois $[s]_T = A$. ■

Suponha que T seja um conjunto de equações tal que para qualquer termo t , $[t]_T$ seja finito. Suponha ainda que exista um algoritmo para gerá-lo. Suponha também que R seja um sistema de reescrita de termos noetheriano módulo T e compatível com T . Podemos usar o lema 7.43 para criar um algoritmo que reescreva qualquer elemento em $\mathcal{T}_{RT}/T$ para sua forma normal. Para vermos isto, seja $A \in \mathcal{T}_{RT}/T$ e escolha qualquer termo $s \in A$, depois, percorra todos os nós com posições $n \in \text{dom } s$, um por um, e verifique, para cada nó percorrido, se $s/n \equiv l\theta$ para algum l , com $l \rightarrow r \in R$ e uma substituição θ . Se a resposta for afirmativa para algum n , então A será redutível. Caso contrário A não será redutível. Se A for redutível, então

$A \xrightarrow{[R]} [s[n \leftarrow r\theta]]_T = B$, e podemos aplicar o mesmo procedimento para B .

Como, por hipótese, R é noetheriano módulo T , este procedimento deve terminar, obrigatoriamente, após um número finito de passos em um termo irreduzível $C \in \mathcal{T}_{RT}/T$ tal que $A \xrightarrow{[R]}^* C$.

Seja $\mathcal{A}$ um algoritmo como descrito acima que aceita, como entrada, qualquer elemento $A \in \mathcal{T}_{RT}/T$ e retorna, como saída, uma forma normal para A . Se, no teorema 7.32, trocarmos " $A \xrightarrow{[R]}^* C$ e $B \xrightarrow{[R]}^* C$ " por " $\mathcal{A}(A) = \mathcal{A}(B)$ " o resultado do teorema continua válido, e podemos selecionar C como sendo $\mathcal{A}(A)$.

Usando esta versão modificada do teorema 7.32, podemos ser capazes de estender um sistema de reescrita de termos R para um sistema de reescrita de termos completo. Para fazermos isto, precisaremos de um algoritmo de unificação- T finito e completo. Primeiro, estendemos R , se necessário, para torná-lo um conjunto R^e que seja compatível com T . O teorema 7.42 explica como fazer isto, se T for uma teoria- AC .

Para cada par crítico $\langle A, B \rangle$ verificamos se $\mathcal{A}(A) = \mathcal{A}(B)$, isto é, para cada par $l_1 \rightarrow r_1$ e $l_2 \rightarrow r_2$ de regras de reescrita e para cada nó com posição $k \in \text{dom } l_1$ verificamos se l_1/k é unificável- T com l_2 . Se encontrarmos um tal k , então chamamos o algoritmo de unificação- T que irá produzir um conjunto Γ finito e completo de unificadoras- T de l_1/k e l_2 .

Para cada $\sigma \in \Gamma$ verificamos se $\mathcal{A}([r_1\sigma]_T) = \mathcal{A}([l_1[k \leftarrow r_2]\sigma]_T)$. Se $\mathcal{A}([r_1\sigma]_T) = \mathcal{A}([l_1[k \leftarrow r_2]\sigma]_T)$ então continuamos as verificações; caso contrário, isto é, se $\mathcal{A}([r_1\sigma]_T) \neq \mathcal{A}([l_1[k \leftarrow r_2]\sigma]_T)$ teremos as seguintes formas normais $B = \mathcal{A}([r_1\sigma]_T)$ e $C = \mathcal{A}([l_1[k \leftarrow r_2]\sigma]_T)$ tais que $B \neq C$.

Podemos decidir se existem, ou não, termos $b \in B$ e $c \in C$ tais que $b \xrightarrow{(R,T)} c$ ou $c \xrightarrow{(R,T)} b$ possam ser adicionados a R , criando um sistema de reescrita R' que continue sendo noetheriano módulo T . Se isto não for possível, então o algoritmo terminará sua execução sem ter conseguido completar o sistema de reescrita de termos fornecido; caso contrário, criamos R' e uma extensão R'^e compatível com T e tentamos mostrar que R'^e é completo, da mesma maneira que acabamos de descrever.

Observe que se R' for gerado com sucesso, isto é, se $b \xrightarrow{(R,T)} c$ ou $c \xrightarrow{(R,T)} b$, então teremos, obrigatoriamente, que $B \xrightarrow{[R']} C$ ou $C \xrightarrow{[R']} B$. Mas R' terá uma regra de reescrita a mais que R , então mais verificações serão necessárias para tentar-se completar R' .

Este processo de completção que acabamos de descrever pode:

- i) terminar com um erro (no caso do algoritmo ser incapaz de completar o sistema de reescrita de termos fornecido);
- ii) não terminar sua execução gerando infinitas regras;
- iii) terminar com sucesso gerando um sistema de reescrita de termos R^* que seja completo- T .

Mostraremos, a seguir, como podemos transformar cada regra de reescrita num sistema de reescrita R normalizadas em relação às outras regras existentes em R . Note que, teoricamente, não há necessidade de manter-se cada regra de um sistema de reescrita normalizada em relação às demais regras do sistema, mas este procedimento de “enxugamento” do sistema de reescrita torna o algoritmo de completção mais eficiente.

Definição 7.44. Se T e U são conjuntos de equações, então podemos dizer que T e U são equivalentes, e neste caso, escrevemos $T \equiv U$, contanto que, para quaisquer termos s e t , $s \stackrel{T}{=} t$ se e somente se $s \stackrel{U}{=} t$. É fácil ver que $\equiv$ é uma relação de equivalência.

[PS81]

Lema 7.45. Suponha que T seja um conjunto de equações e $t \stackrel{T}{=} s$. Seja $U = T \cup \{t = s\}$. Então $T \equiv U$.

[PS81]

Demonstração: Observando a definição 7.44, precisamos mostrar que, para quaisquer termos l e r , $l \stackrel{T}{=} r$ se e somente se $l \stackrel{U}{=} r$.

($\Rightarrow$ /somente se)

Hipótese: $l \stackrel{T}{=} r$

Tese: $l \stackrel{U}{=} r$

Se $l \stackrel{T}{=} r$, então $l \stackrel{U}{=} r$ pois $U \supset T$.

($\Leftarrow$ /se)

Para demonstrarmos esta implicação é suficiente mostrarmos que $l \stackrel{1}{U} r$ implica $l \stackrel{T}{=} r$.

Hipótese: $l \stackrel{1}{U} r$

Tese: $l \stackrel{T}{=} r$

Suponha que $l \stackrel{1}{U} r (<>, u = v, n, \theta)$. Se $(u = v) \in T$, então $l \stackrel{T}{=} r$; caso contrário $(u = v) = (t = s)$ ou $(u = v) = (s = t)$ e podemos supor por simetria que $(u = v) = (t = s)$. Sendo assim, $l/n = t\theta$ e $r = l[n \leftarrow s\theta]$. Como $t \stackrel{T}{=} s$, temos pelo corolário 3.11 que $t\theta \stackrel{T}{=} s\theta$, e pelo lema 3.12, $l \stackrel{T}{=} r$. ■

Lema 7.46. Suponha que T seja uma teoria equacional qualquer. Seja $U = T \cup \{s = t\}$ e $V = T \cup \{p = q\}$. Se $s \stackrel{U}{=} t$ e $p \stackrel{U}{=} q$, então V e U são equivalentes.

[PS81]

O próximo teorema 7.48 nos mostra que, sob certas circunstâncias, podemos eliminar regras de reescrita do sistema e, conseqüentemente, alguns dos testes exigidos pelo teorema 7.32 não precisam ser realizados, proporcionando assim, uma maior eficiência do algoritmo de completção.

Teorema 7.48. *Suponha que R seja compatível com T e que R seja noetheriano módulo T . Suponha que para algum par de regras distintas entre si $l_1 \rightarrow r_1, l_2 \rightarrow r_2$ em R exista uma substituição θ tal que*

$$l_1 \equiv_{\overline{T}} l_2\theta \quad \text{e} \quad [r_1]_T \xrightarrow{[R]}^* [r_2\theta]_T$$

Além disto, suponha que exista uma regra de reescrita $l \rightarrow r$ em R , contida e protegida em $l_1 \rightarrow r_1$ na posição k , onde $k = \langle n \rangle, n \in \mathbb{Z}_+$. Então, $l_1 \rightarrow r_1$ pode ser removida de R sem que, com isto, a conclusão do teorema 7.32 seja afetada.

[PS81]

7.5 Uma Implementação do Algoritmo de Completção

Peterson e Stickel [PS81] apresentaram uma implementação de seu algoritmo que acabamos de descrever nas seções anteriores e também apresentaram alguns exemplos de aplicações do algoritmo.

Os operadores associativos-comutativos foram expressos em termos de operações com um número variável de argumentos, isto é, $+(+(a, b), c)$ e $+(a, +(b, c))$ foram ambas expressas como $+(a, b, c)$ (ou equivalentemente, em notação infixa, como $(a + b + c)$).

Eles utilizaram o algoritmo de unificação associativa-comutativa desenvolvido por Stickel [Sti81] (veja a seção 4.3) para realizarem as inferências de igualdade entre termos no passo (24) do algoritmo.

Para as simplificações que são realizadas nos passos (6) e (16) do algoritmo foi usado um algoritmo de unificação-AC incompleto mas que é completo para determinar substituições através das quais um termo torna-se instância-AC de um outro termo, isto é, tal algoritmo é suficiente para a realização de matchings-AC e é mais rápido que um algoritmo de unificação-AC completo.

Grande parte do tempo gasto durante execução do algoritmo é consumido no processo de simplificação do passo (6) apesar do alto custo de aplicação do algoritmo de unificação-AC completo no passo (24).

Para determinar se um termo é mais simples que outro (passos (7) e (8)) foi utilizado a medida de complexidade polinomial de Lankford [Lan75a, Lan75b, Lan79] (veja exemplo 5.34). A complexidade de um termo t foi calculada usando a seguinte interpretação polinomial para a função de terminação τ (veja definição 5.32) composta pelas seguintes funções:

$$\begin{aligned}
\cup_\tau &: \mathbb{N}^2 \mapsto \mathbb{N} \\
\cap_\tau &: \mathbb{N}^2 \mapsto \mathbb{N} \\
*_\tau &: \mathbb{N}^2 \mapsto \mathbb{N} \\
+_\tau &: \mathbb{N}^2 \mapsto \mathbb{N} \\
-_\tau &: \mathbb{N} \mapsto \mathbb{N} \\
0_\tau &: \mapsto \mathbb{N} \\
1_\tau &: \mapsto \mathbb{N} \\
x_\tau &: \mapsto \mathbb{N}
\end{aligned}$$

e especificadas, respectivamente, por:

$$\begin{aligned}
\cup_\tau(x, y) &= x_\tau + y_\tau + 5 \\
\cap_\tau(x, y) &= x_\tau \cdot y_\tau \\
*_\tau(x, y) &= x_\tau \cdot y_\tau \\
+_\tau(x, y) &= x_\tau + y_\tau + 5 \\
-_\tau(x) &= 2 \cdot x_\tau + 2 \\
0_\tau &= 2 \\
1_\tau &= 2 \\
x_\tau &= 2 \quad \text{para qualquer } x \text{ em } \mathcal{V}
\end{aligned}$$

Pode-se mostrar que esta interpretação é, de fato, uma função de terminação e, portanto, estabelece uma relação de ordem útil para nossos propósitos de verificação da propriedade de terminação de sistemas de reescrita de termos.

Como consequência imediata das especificações de $+_\tau(x, y)$ e $*_\tau(x, y)$, temos que $\tau(x + y) = \tau(y + x)$, $\tau((x + y) + z) = \tau(x + (y + z))$, $\tau(x * y) = \tau(y * x)$ e $\tau((x * y) * z) = \tau(x * (y * z))$. Analogamente para os símbolos $\cup$ e $\cap$. Isto é, qualquer termo equivalente módulo AC possui a mesma complexidade segundo esta interpretação dada, fato este essencial para que esta interpretação seja útil para garantir-se a terminação- AC (veja seção 5.3) dos sistemas de reescrita obtidos nos próximos três exemplos.

A relação de comparação de complexidade será denotada por $\succ$ e definida como $s \succ t$ se e somente se $\tau(s\theta) > \tau(t\theta)$ para qualquer substituição θ , onde $>$ é a relação usual “maior que” dos inteiros. No programa que segue, $\tau(s\theta) > \tau(t\theta)$ foi verificado apenas quando θ for a substituição identidade, isto é, nula. Entretanto, pode-se verificar que a monotonicidade de $\succ$, especificada acima, também é satisfeita para qualquer substituição θ considerada, isto é, de fato, se $s \succ t$ então $s\theta \succ t\theta$.

Uma *extensão simples* $(l \rightarrow r)^e$ de $l \rightarrow r$ é a regra de reescrita $f(v, l) \rightarrow f(v, r)^*$, onde $l \langle \rangle = f$, f é uma função associativa-comutativa, v é uma variável que não está contida em $l \rightarrow r$, e $f(v, r)^*$ está normalizado em relação a todas as regras em R e $l \rightarrow r$, e $f(v, l) \rightarrow$

$f(v, r)^*$ não é uma instância de $l \rightarrow r$. Não há extensão se alguma das condições acima não for satisfeita. Todas as inferências sobre $l_1 \rightarrow r_1$ e $l_2 \rightarrow r_2$ são realizadas no passo (24), onde as inferências são feitas separadamente para cada par $\langle l_1 \rightarrow r_1, l_2 \rightarrow r_2 \rangle$; $\langle l_1 \rightarrow r_1, (l_2 \rightarrow r_2)^e \rangle$; $\langle (l_1 \rightarrow r_1)^e, l_2 \rightarrow r_2 \rangle$ e $\langle (l_1 \rightarrow r_1)^e, (l_2 \rightarrow r_2)^e \rangle$.

A seguir descrevemos este algoritmo de um modo mais sistematizado.

Algoritmo 1.4 Algoritmo de completção módulo T de Peterson e Stickel.

- 1: Sejam $k, a \in \mathbb{N}$;
 R : o sistema de reescrita de termos corrente;
 E : conjunto de equações a serem simplificadas e possivelmente adicionadas como regras de reescrita;
 T : conjunto de equações para o qual pretende-se obter um SRT completo módulo T .
 $Pairs$: pares de regras de reescrita a partir dos quais serão feitas as inferências.
- 2: % Inicialização
 $R := \emptyset$; $Pairs := \emptyset$. Seja E o conjunto inicial de equações.
- 3: se $E = \emptyset$ então vá para 21.
- 4: % Simplifica uma equação e adiciona-a como regra de reescrita caso não seja identidade.
 Seja $s = t$ uma equação em E com a menor complexidade (segundo $\succ$)
- 5: $E := E - \{s = t\}$.
- 6: se $R(s) \equiv R(t)$ então
 faça vá para 3.
- 7: % Tenta produzir uma regra de reescrita a partir da equação $R(s) = R(t)$.
 se $R(s) \succ R(t)$ então
 faça $l := R(s)$;
 $r := R(t)$;
 vá para 10.
- 8: se $R(t) \succ R(s)$ então
 faça $l := R(t)$;
 $r := R(s)$;
 vá para 10.
- 9: Termina a execução do programa com erro ($R(s)$ e $R(t)$ são incomparáveis segundo a relação de complexidade $\succ$ especificada).
- 10: % Adiciona a regra de reescrita $l \rightarrow r$ em R .
 $R := R \cup \{l \rightarrow r\}$.
- 11: $Pairs := Pairs \cup \{\{l \rightarrow r, l' \rightarrow r'\}, \{(l \rightarrow r)^e, l' \rightarrow r'\}, \{l \rightarrow r, (l' \rightarrow r')^e\}, \{(l \rightarrow r)^e, (l' \rightarrow r')^e\}\}$ para cada $l' \rightarrow r'$ em R .

- 12: % Normalização das regras de reescrita em R com respeito às outras regras em R .
 Seja k o número de elementos de R ;
 $a := 1$.
- 13: se $a > k$ então vá para 3.
- 14: Seja $s \rightarrow t$ a a -ésima regra de reescrita de R .
- 15: Seja $R' := R - \{s \rightarrow t\}$.
- 16: se $R'(s) = s$ e $t = R'(t)$ então
 faça $a := a + 1$;
 vá para 13. % $s \rightarrow t$ já estão em suas formas normais.
- 17: {Remove $s \rightarrow t$ de R .} $Pairs := Pairs - \{\{s \rightarrow t, s' \rightarrow t'\}, \{(s \rightarrow t)^e, s' \rightarrow t'\}, \{(s \rightarrow t), (s' \rightarrow t')^e\}, \{(s \rightarrow t)^e, (s' \rightarrow t')^e\}\}$ para cada $s' \rightarrow t'$ em R , extensão simples $(s \rightarrow t)^e$ de $s \rightarrow t$, e extensão simples $(s' \rightarrow t')^e$ de $s' \rightarrow t'$.
- 18: $R := R - \{s \rightarrow t\}$.
- 19: se $R'(s) \neq R'(t)$ então
 faça $k := k - 1$;
 vá para 13.
- 20: vá para 7.
- 21: se $Pairs = \emptyset$ então termina a execução do programa com R sendo um sistema de reescrita completo- T .
- 22: % Faz novas inferências a partir de um par de regras de reescrita.
 Seja $l_1 \rightarrow r_1, l_2 \rightarrow r_2$ um par de regras em $Pairs$ tal que a soma das complexidades de l_1 e l_2 seja a menor possível dentre todos os pares existentes em $Pairs$.
- 23: $Pairs := Pairs - \{l_1 \rightarrow r_1, l_2 \rightarrow r_2\}$.
- 24: $E := \{r_1\sigma = r_2\sigma \mid \sigma \in \text{conjunto completo de unificadoras-}T \text{ de } l_1 \text{ e } l_2\} \cup \{r_1\sigma \cup (l_1(n \leftarrow r_2))\sigma \mid l_1 \rightarrow r_1 \text{ não seja uma extensão, } n \in \text{dom } l_1, n \neq \langle \rangle, \sigma \in \text{conjunto completo de unificadoras-}T \text{ de } l_1/n \text{ e } l_2\} \cup \{(l_2(n \leftarrow r_1))\sigma = r_2\sigma \mid l_2 \rightarrow r_2 \text{ não seja uma extensão, } n \in \text{dom } l_2, n \neq \langle \rangle, \sigma \in \text{conjunto completo de unificadoras-}T \text{ de } l_1 \text{ e } l_2/n\}$.
- 25: vá para 3.

.....

A seguir, alguns exemplos da utilização deste algoritmo apresentados por Peterson e Stickel [PS81].

Exemplo 7.49. Grupos Comutativos. *O programa referente ao algoritmo apresentado acima recebeu como dados de entrada os seguintes dois axiomas de grupos comutativos separados em duas partes E e T , onde E contém os axiomas de grupos e T contém os axiomas que expressam a propriedade associativa-comutativa do grupo.*

$$E = \{0 + x = x, (-x) + x = 0\};$$

$$T = \{x + (y + z) = (x + y) + z, x + y = y + x\}.$$

O programa formulou regras de reescrita a partir das equações de E dadas e encontrou o seguinte sistema de reescrita completo módulo AC:

- (1) $0 + x \rightarrow x$ pela eq. de entrada $0 + x = x$.
 - (2) $(-x) + x \rightarrow 0$ pela eq. de entrada $(-x) + x = 0$.
 - (3) $-0 \rightarrow 0$ por (1) e (2).
 - (4) $-(-x) \rightarrow x$ por (2)^e e (2)^e.
 - (5) $-(x + y) + x \rightarrow -y$ por (2)^e e (2)^e.
 - (6) $-(x + y) \rightarrow (-x) + (-y)$ por (2)^e e (5).
- (5) removido!.

Obtendo um sistema de reescrita completo para grupos comutativos formado pelas regras de reescrita 1, 2, 3, 4 e 6.

Exemplo 7.50. Anéis Comutativos com Unidade Neste exemplo, foi fornecido como dados de entrada o conjunto completo de regras de reescrita (na forma de equações) para grupos comutativos mais os axiomas da identidade e distributividade para a função associativa-comutativa “ $*$ ” (conjunto E) e também o conjunto T contendo as leis associativas e comutativas dos operadores $*$ e $+$:

$$\begin{aligned}
 E = \{ & 0 + x = x, \\
 & (-x) + x = 0, \\
 & -0 = 0, \\
 & -(-x) = x, \\
 & -(x + y) = (-x) + (-y), \\
 & 1 * x = x \\
 & x * (y + z) = (x * y) + (x * z), \\
 & \} \\
 T = \{ & x + y = y + x, \\
 & x * y = y * x, \\
 & x + (y + z) = (x + y) + z, \\
 & x * (y * z) = (x * y) * z \\
 & \}
 \end{aligned}$$

O algoritmo obteve o seguinte sistema de reescrita completo módulo AC:

- (1) $0 + x \rightarrow x$
- (2) $(-x) + x \rightarrow 0$
- (3) $-0 \rightarrow 0$
- (4) $-(-x) \rightarrow x$
- (5) $-(x + y) \rightarrow (-x) + (-y)$
- (6) $1 * x \rightarrow x$
- (7) $x * (y + z) \rightarrow (x * y) + (x * z)$
- (8) $(x * 0) + (x * y) \rightarrow x * y$ por (1) e (7).
- (9) $(x * (-y)) + (x * y) \rightarrow x * 0$ por (2) e (7).
- (10) $(x * 0) + x \rightarrow x$ por (6) e (8).
- (11) $0 + 0 \rightarrow 0$ por (1) e (10).
- (12) $(-x) + (-(x * 0)) \rightarrow -x$ por (5) e (10).
- (13) $x + (-(x * 0)) \rightarrow x$ por (4) e (12).
- (14) $x * 0 \rightarrow 0$ por (2)^e e (10)^e.
(13), (12), (11), (10), (9) removidas!
- (15) $(x * (-y)) + (x * y) \rightarrow 0$ por (9).
(8) removida!
- (16) $(x * (-1)) + x \rightarrow 0$ por (6) e (15).
- (17) $-x + (-(x * (-1))) \rightarrow 0$ por (5) e (16).
- (18) $x + (-(x * (-1))) \rightarrow 0$ por (4) e (17).
- (19) $x * (-1) \rightarrow -x$ de (2)^e e (16)^e.
(18), (17), (16) removidas!
- (20) $x * (-y) \rightarrow -(x * y)$, por (6) e (19)^e.
(19), (15) removidas!

O sistema formado pelas regras 1, 2, 3, 4, 5, 6, 7, 8, 14 e 20 constitui um sistema de reescrita completo para anéis comutativos com unidade.

Exemplo 7.51. Reticulados Distributivos Foram fornecidos como dados de entrada o conjunto de axiomas dos reticulados distributivos separados em duas partes, E e T , onde T contém a porção das equações não orientáveis da especificação equacional dos reticulados distributivos.

$$\begin{aligned}
 E = \{ & x \cup (x \cap y) = x, \\
 & x \cap (x \cup y) = x, \\
 & x \cap (y \cup z) = (x \cap y) \cup (x \cap z) \\
 & \}
 \end{aligned}$$

$$T = \{x \cup y = y \cup x, \\ x \cap y = y \cap x, \\ x \cup (y \cup z) = (x \cup y) \cup z, \\ x \cap (y \cap z) = (x \cap y) \cap z \\ \}$$

e o algoritmo obteve o seguinte sistema de reescrita completo módulo AC:

- (1) $x \cup (x \cap y) \rightarrow x$
- (2) $x \cap (x \cup y) \rightarrow x$
- (3) $x \cap (y \cup z) \rightarrow (x \cap y) \cup (x \cap z)$
- (2) *removida!*
- (4) $(x \cap x) \cup (x \cap y) \rightarrow x$ por (2)
- (5) $x \cap x \rightarrow x$ por (1) e (4)
- (4) *removida!*
- (6) $x \cup x \rightarrow x$ por (1) e (5)

As regras de reescrita 1, 3, 5 e 6 formam um sistema de reescrita de termos completo para reticulados distributivos. Observe que a outra regra distributiva, $x \cup (y \cap z) = (x \cup y) \cap (x \cup z)$, é obtida a partir das regras acima obtidas. O lado esquerdo desta equação é irreduzível, entretanto pode-se usar (3) repetidamente para reescrever o lado direito para $(x \cap x) \cup (y \cap x) \cup (x \cap z) \cup (y \cap z)$. Então aplicando (5) nesta última expressão produz-se $x \cup (y \cap x) \cup (x \cap z) \cup (y \cap z)$ que, por sua vez, escreve-se à $x \cup (y \cap z)$ após aplicar-se duas mais reescritas, usando (1).

No apêndice B, descrevemos os passos principais, realizados pelo algoritmo de completção descrito nesse capítulo, para a completção dos reticulados distributivos.

Capítulo 8

Morfologia Matemática

A Morfologia Matemática surgiu na década de 60. na École Nationale Supérieure des Mines de Paris, em Fontainebleau. Seus criadores, George Matheron e Jean Serra, estavam interessados em buscar soluções para problemas relacionados à análise de imagens.

Eles propuseram soluções baseadas no conceito de transformações nas formas das imagens realizadas através de dois operadores ou transformações elementares que eles denominaram *dilatação* e *erosão*. Essas transformações são realizadas por *funções estruturantes* que “sondam localmente” a imagem.

Inicialmente, Matheron e Serra trabalharam em problemas envolvendo a classe das imagens binárias. isto é, cada “*pixel*” de uma imagem pode assumir apenas dois valores auto-excluentes, 0 ou 1. Neste contexto, a dilatação e erosão foram estabelecidas, respectivamente, a partir da noção de *adição de Minkowski* e *subtração de Minkowski*.

As imagens binárias planas e espaciais podem ser representadas por subconjuntos de $\mathbb{Z}^2$ e $\mathbb{Z}^3$ respectivamente. Por esta razão, a Morfologia Matemática para imagens binárias é igualmente denominada Morfologia Matemática para conjuntos.

Matheron e Serra encontraram resultados interessantes no estudo da Morfologia Matemática para conjuntos e posteriormente expandiram os conceitos morfológicos para cobrirem também o caso das imagens em níveis de cinza, que chamaram de Morfologia Matemática para funções. As imagens planas e espaciais em níveis de cinza podem ser representadas através de mapeamentos de $\mathbb{Z}^2$ em $\mathbb{N}$ e $\mathbb{Z}^3$ em $\mathbb{N}$ respectivamente.

Considere E um conjunto não vazio, então temos:

Definição 8.1. (Limitante Superior/Inferior) *Seja $\mathcal{X}$ uma subcoleção não vazia de $\mathcal{P}(E)$, isto é, $\mathcal{X} \subseteq \mathcal{P}(E)$ e $\mathcal{X} \neq \emptyset$ e seja A um elemento de $\mathcal{P}(E)$. Dizemos que A é um limitante superior de $\mathcal{X}$ (em $\mathcal{P}(E)$) se e somente se $X \subseteq A$ para todo $X \in \mathcal{X}$.*

Dizemos que A é um limitante inferior de $\mathcal{X}$ (em $\mathcal{P}(E)$) se e somente se $A \subseteq X$ para todo $X \in \mathcal{X}$.

[BB94]

Definição 8.2. (Supremo/Ínfimo) O supremo de $\mathcal{X}$ (em $\mathcal{P}(E)$), denotado $\sup \mathcal{X}$ é, se existir, o menor dos limitantes superiores de $\mathcal{X}$ em $\mathcal{P}(E)$. Em outros termos, para todo Y em $\mathcal{P}(E)$, Y é limitante superior de $\mathcal{X}$ se e somente se $\sup \mathcal{X} \subseteq Y$. O ínfimo de $\mathcal{X}$ (em $\mathcal{P}(E)$), denotado $\inf \mathcal{X}$ é, se existir, o maior dos limitantes inferiores de $\mathcal{X}$ em $\mathcal{P}(E)$. Em outras palavras, para todo Y em $\mathcal{P}(E)$, Y é limitante inferior de $\mathcal{X}$ se e somente se $Y \subseteq \inf \mathcal{X}$.

[BB94]

“A partir da década de oitenta, Matheron e Serra notaram que os resultados obtidos tanto para conjuntos quanto para funções tinham essencialmente um fator em comum: a existência de uma relação de ordem parcial, no caso dos conjuntos, a relação usual de inclusão ($\subseteq$) de conjuntos e no caso das funções, a relação usual de menor ou igual ($\leq$) dos números inteiros estendida à funções.” [BB94, pg. 4]

A observação acima levou à busca de uma possível generalização da Morfologia Matemática, o que acabou ocorrendo. Matheron e Serra mostraram que a estrutura básica $< \mathcal{P}(E)$, $\subseteq$ $>$ da Morfologia Matemática para conjuntos forma um reticulado completo (no caso das imagens planas, $E \subseteq \mathbb{Z}^2$ e no caso das imagens espaciais, $E \subseteq \mathbb{Z}^3$), o mesmo ocorrendo para o caso das funções, cuja estrutura básica é $< \mathcal{F}$, $\leq$ $>$, onde $\mathcal{F}$ é o conjunto de todas as funções definidas de $\mathbb{Z}^2$ em $\mathbb{Z}$, no caso das imagens planas em níveis de cinza, no caso das imagens espaciais em níveis de cinza, temos $\mathcal{F}$ sendo, evidentemente, o conjunto de todas as funções definidas de $\mathbb{Z}^3$ em $\mathbb{Z}$. Com isto, passou-se a “olhar” para a Morfologia Matemática através de uma caracterização mais geral e, conseqüentemente, mais abstrata. A partir deste momento, a Morfologia Matemática passou a ser melhor definida como sendo o estudo de *isomorfismos* entre reticulados completos.

Para as próximas duas definições, considere I sendo um conjunto cujos elementos são chamados de *índices*; $(A_i)_{i \in I}$ sendo uma *família de elementos de $\mathcal{P}(E)$* . Uma família $(A_i)_{i \in I}$ é um mapeamento de I em $\mathcal{P}(E)$.

Definição 8.3. A união da família dos subconjuntos A_i é o subconjunto de E , denotado $\bigcup_{i \in I} A_i$ e dado por:

$$\bigcup_{i \in I} A_i = \{x \in E : \exists i \in I, x \in A_i\}.$$

[BB94]

Definição 8.4. A interseção da família dos subconjuntos A_i é o subconjunto de E , denotado $\bigcap_{i \in I} A_i$ e dado por:

$$\bigcup_{i \in I} A_i = \{x \in E : \forall i \in I, x \in A_i\}.$$

[BB94]

O conjunto parcialmente ordenado $\langle \mathcal{P}(E), \subseteq \rangle$, provido das operações habituais de união e interseção entre conjuntos estendidas às famílias em $\mathcal{P}(E)$, forma um *reticulado completo*. Em outras palavras, para todo conjunto de índices I e toda família $(A_i)_{i \in I}$ de subconjuntos de E valem:

$$\bigcup_{i \in I} A_i = \sup \mathcal{A}_I \quad e \quad \bigcap_{i \in I} A_i = \inf \mathcal{A}_I$$

onde $\mathcal{A}_I$ é a imagem de I através da família $(A_i)_{i \in I}$, isto é,

$$\mathcal{A}_I = \{A \in \mathcal{P}(E) : \exists i \in I, A_i = A\}$$

De modo equivalente, podemos dizer que o conjunto parcialmente ordenado $\langle \mathcal{P}(E), \subseteq \rangle$, provido das operações habituais de união e interseção entre conjuntos estendidas às famílias em $\mathcal{P}(E)$, forma um reticulado completo porque toda subcoleção de $\mathcal{P}(E)$ possui supremo e ínfimo. [BB94, pg.29]

De um modo geral, para cada imagem binária X existe conjunto E tal que X é elemento de $\mathcal{P}(E)$ e a transformação da imagem X em uma imagem Y em $\mathcal{P}(E)$ é uma operação de $\mathcal{P}(E)$ em $\mathcal{P}(E)$.

Os operadores sobre $\mathcal{P}(E)$ são representados, neste texto, por letras gregas, como ψ, α, β . Os operadores sobre $\mathcal{P}(E)$ também são chamados de *operadores sobre conjuntos*.

Definição 8.5. (Dilatação, Erosão, Anti-Dilatação, Anti-Erosão) Um operador ψ sobre $\mathcal{P}(E)$ é

- a) uma dilatação se e somente se para todo $\mathcal{X} \subseteq \mathcal{P}(E)$,
 $\psi(\sup \mathcal{X}) = \sup \psi(\mathcal{X})$;
- b) uma erosão se e somente se para todo $\mathcal{X} \subseteq \mathcal{P}(E)$,
 $\psi(\inf \mathcal{X}) = \inf \psi(\mathcal{X})$;
- c) uma anti-dilatação se e somente se para todo $\mathcal{X} \subseteq \mathcal{P}(E)$,
 $\psi(\sup \mathcal{X}) = \inf \psi(\mathcal{X})$;
- d) uma anti-erosão se e somente se para todo $\mathcal{X} \subseteq \mathcal{P}(E)$.
 $\psi(\inf \mathcal{X}) = \sup \psi(\mathcal{X})$.

[BB94]

O conjunto das dilatações é denotado Δ , o das erosões $\mathbf{E}$, o das anti-dilatações Δ^a e o das anti-erosões $\mathbf{E}^a$. Uma dilatação é denotada, genericamente, por δ , uma erosão por ϵ , uma anti-dilatação por δ^a e uma anti-erosão por ϵ^a . Para um dado conjunto X , os conjuntos $\delta(X)$, $\epsilon(X)$, $\delta^a(X)$ e $\epsilon^a(X)$ chamam-se respectivamente, de dilatação, erosão, anti-dilatação e anti-erosão de X [BB94, pg. 34].

A definição 8.5 que especifica as classes dos operadores elementares da Morfologia Matemática não é útil para implementarmos no computador uma dilatação, erosão, anti-dilatação ou anti-erosão. Precisamos, para isso, caracterizar os operadores elementares de forma construtiva, de modo que possamos implementá-los através de um algoritmo. Abaixo, apresentamos a caracterização da classe das dilatações. Denotamos o conjunto das funções de E em $\mathcal{P}(E)$ por $\mathcal{P}^E$.

Proposição 8.6. (Caracterização das Dilatações) *O mapeamento de Δ em $\mathcal{P}^E$, $\delta \mapsto a_\delta$, onde a_δ é a função dada por $a_\delta(y) = \delta(\{y\})$, com $y \in E$, é uma bijeção. Seu inverso é $a \mapsto \delta_a$, onde δ_a é a dilatação dada por $\delta_a(Y) = \bigcup_{y \in Y} a(y)$, com $Y \in \mathcal{P}(E)$.*

[BB94]

Demonstração: Primeiro vamos mostrar que, de fato, δ_a é uma dilatação. Para todo $a \in \mathcal{P}^E$ e todo $\mathcal{Y} \subseteq \mathcal{P}(E)$,

$$\begin{aligned}
 \delta_a(\sup \mathcal{Y}) &= \bigcup_{y \in \sup \mathcal{Y}} a(y) && \text{(definição de } \delta_a) \\
 &= \bigcup_{y \in \bigcup_{Y \in \mathcal{Y}} Y} a(y) && \text{(propriedade da união)} \\
 &= \left(\bigcup_{y \in Y_1} a(y) \right) \cup \dots \cup \left(\bigcup_{y \in Y_n} a(y) \right), \text{ onde } \mathcal{Y} = \{Y_1, \dots, Y_n\} \\
 &= \bigcup_{Y \in \mathcal{Y}} \left(\bigcup_{y \in Y} a(y) \right) && \text{(idempotência da união)} \\
 &= \bigcup_{Y \in \mathcal{Y}} \bigcup_{y \in Y} a(y) && \text{(associatividade da união)} \\
 &= \bigcup_{Y \in \mathcal{Y}} \delta_a(Y) && \text{(definição de } \delta_a) \\
 &= \sup \delta_a(\mathcal{Y}) && \text{(propriedade da união)}
 \end{aligned}$$

Logo, pela definição 8.5, δ_a é uma dilatação.

Vamos mostrar que o mapeamento $\delta \mapsto a_\delta$ é injetor. Para todo δ em Δ e para todo Y em $\mathcal{P}(E)$,

$$\begin{aligned}
 \delta_{a_\delta}(Y) &= \bigcup_{y \in Y} a_\delta(y) && \text{(definição de } \delta_a) \\
 &= \bigcup_{y \in Y} \delta(\{y\}) && \text{(definição de } a_\delta) \\
 &= \delta\left(\bigcup_{y \in Y} \{y\}\right) && \text{(propriedade da dilatação)} \\
 &= \delta(Y)
 \end{aligned}$$

Em outros termos, para todo δ em Δ , $\boxed{\delta_{a_\delta} = \delta}$. Sendo assim, considere os mapeamentos $\delta \mapsto a_\delta$ e $\delta' \mapsto a_{\delta'}$, com $\delta \neq \delta'$. Suponha, por absurdo, que $a_\delta = a_{\delta'}$, então $\delta' = \delta'_{a_{\delta'}} = \delta'_{a_\delta} \xrightarrow{\text{por } *} \delta = \delta'$, logo, temos uma contradição, pois, por hipótese, $\delta \neq \delta'$. Logo, temos $a_\delta \neq a_{\delta'}$ e portanto o mapeamento $\delta \mapsto a_\delta$ é injetor.

Vamos mostrar que o mapeamento $\delta \mapsto a_\delta$ é sobrejetor. Para todo a em $\mathcal{P}^E$ e todo y em E ,

$$\begin{aligned}
 a_{\delta_a}(y) &= \delta_a(\{y\}) && \text{(definição de } a_\delta) \\
 &= \bigcup_{\nu \in \{y\}} a(\nu) && \text{(definição de } \delta_a) \\
 &= a(y).
 \end{aligned}$$

Em outras palavras, para todo a em $\mathcal{P}^E$ e todo y em E , $a_{\delta_a} = a$. Sendo assim, considere os mapeamentos $\delta \mapsto a_\delta$ e $\delta' \mapsto a_{\delta'}$, com $\delta \neq \delta'$. Suponha por absurdo que $a_\delta = a_{\delta'}$, então como, consequência da definição de a_δ , temos $\delta = \delta'$, isto é, uma contradição, pois, por hipótese, $\delta \neq \delta'$. Logo, $a_\delta \neq a_{\delta'}$ e portanto, o mapeamento $\delta \mapsto a_\delta$ é sobrejetor.

Portanto, acabamos de mostrar que o mapeamento $\delta \mapsto a_\delta$ é injetor e sobrejetor, logo é uma bijeção. ■

A proposição 8.6 mostra que existe uma correspondência de um para um entre Δ e $\mathcal{P}^E$. As funções a com valores nas partes de E caracterizam sem ambigüidades as dilatações. A figura 8.1, obtida em [BB94], ilustra este resultado. A função a_δ é chamada de *função estruturante* da dilatação δ .

De modo análogo, podemos caracterizar as erosões, anti-dilatações e anti-erosões.

Figura 8.1: Bijeção entre as dilatações e as funções estruturantes.

Estruturando um conjunto E como um grupo abeliano, pode-se definir a noção de translação de qualquer subconjunto X de E por $u \in E$. Por abuso de linguagem, chamamos u de *vetor*, apesar dele não ser elemento de um espaço vetorial. Daqui até o final deste capítulo considere E sendo um grupo abeliano, com a operação binária denotada por $+$, unária denotada por $-$ e o elemento neutro (ou origem) denotado por $\mathbf{o}$.

Notação: Se $a, b \in E$ então $a - b \stackrel{\text{notação}}{=} a + (-b)$.

Definição 8.7. (Translação por um Vetor) *Seja X um subconjunto de um grupo abeliano E . O translado de X por um vetor u de E é o subconjunto denotado $X + u$ e dado por:*

$$X + u = \{x \in E : x - u \in X\}.$$

O operador τ_u é um mapeamento em $\mathcal{P}(E)$ dado por $\tau_u(X) = X + u$ e é chamado de operador de translação.

[BB94]

Proposição 8.8. (Propriedades Básicas do Translado) *Para todo X, Y em $\mathcal{P}(E)$ e para todo u, v em E :*

- (i) $X + \mathbf{o} = X$;
- (ii) $(X + u) + v = X + (u + v)$;
- (iii) $X \subseteq Y$ se e somente se $(X + u) \subseteq (Y + u)$.

[BB94]

Como consequência da proposição 8.8, temos o seguinte corolário:

Corolário 8.9. *Para qualquer X_1, X_2 em $\mathcal{P}(E)$ e qualquer u em E :*

- (i) $(X_1 \cup X_2) + u = (X_1 + u) \cup (X_2 + u);$
(ii) $(X_1 \cap X_2) + u = (X_1 + u) \cap (X_2 + u).$

[BB94]

Definição 8.10. (Invariância por Translação) *Seja E um grupo abeliano. Um operador ψ sobre $\mathcal{P}(E)$ é invariante por translação se e somente se para todo $u \in E$,*

$$\psi(X + u) = \psi(X) + u.$$

[BB94]

Definição 8.11. (Transposto de um Subconjunto) *Seja E um grupo abeliano. O transposto (em relação à origem) de um subconjunto X de E , denotado X^t , é dado por:*

$$X^t = \{y \in E : y = -x, x \in X\}.$$

[BB94]

Proposição 8.12. (Propriedade Básica do Transposto) *Sejam $X, Y \in \mathcal{P}(E)$, então:*

$$(X \cup Y)^t = X^t \cup Y^t$$

[BB94]

Demonstração:

$$\begin{aligned} X^t \cup Y^t &= \{z_1 \in E : -z_1 \in X\} \cup \{z_2 \in E : -z_2 \in Y\} && \text{(definição de transposto)} \\ &= \{z \in E : -z \in X \text{ ou } -z \in Y\} && \text{(propriedade de } \cup) \\ &= \{z \in E : -z \in (X \cup Y)\} && \text{(definição de } \cup) \\ &= (X \cup Y)^t && \text{(definição de transposto)} \end{aligned}$$

◆

Definição 8.13. (Adição de Minkowski) *Seja E um grupo abeliano. Sejam A e B dois subconjuntos de E . A soma de Minkowski de A e B é o subconjunto de E , denotado $A \oplus B$ e dado por:*

$$A \oplus B = \{x \in E : \exists a \in A \text{ e } \exists b \in B, x = a + b\}$$

A adição de Minkowski, denotada $\oplus$, é o mapeamento dado por:

$$(A, B) \mapsto A \oplus B$$

[BB94]

Proposição 8.14. (Propriedades Básicas da Soma de Minkowski) *Para todo A, B e C em $\mathcal{P}(E)$ valem:*

Figura 8.2: Soma de Minkowski de dois subconjuntos.

- (i) $A \oplus B = \bigcup_{b \in B} A + b$ (definição equivalente);
- (ii) $A \oplus B = B \oplus A$ (comutatividade);
- (iii) $(A \oplus B) \oplus C = A \oplus (B \oplus C)$ (associatividade);
- (iv) $A \oplus (B \cup C) = (A \oplus B) \cup (A \oplus C)$ (distributividade);
- (v) $A \oplus B = \{x \in E : (B^t + x) \cap A \neq \emptyset\}$ (definição equivalente).

[BB94]

Demonstração: Propriedade (i): Para todo A, B em $\mathcal{P}(E)$ e para todo x em E ,

$$\begin{aligned}
 x \in A \oplus B &\Leftrightarrow \exists a \in A \text{ e } \exists b \in B : x = a + b && \text{(definição de } \oplus) \\
 &\Leftrightarrow \exists b \in B, (\exists a \in A, x = a + b) && \text{(equivalência lógica)} \\
 &\Leftrightarrow \exists b \in B, (\exists a \in A, a = x - b) && \text{(propriedade de } +) \\
 &\Leftrightarrow \exists b \in B, x - b \in A && \text{(equivalência lógica)} \\
 &\Leftrightarrow \exists b \in B, x \in A + b && \text{(definição de translado)} \\
 &\Leftrightarrow x \in \bigcup_{b \in B} A + b && \text{(definição de união)}
 \end{aligned}$$

□

Propriedade (ii):

$$\begin{aligned}
 A \oplus B &= \{x \in E : \exists a \in A \text{ e } \exists b \in B : x = a + b\} && \text{(definição de } \oplus) \\
 &= \{x \in E : \exists b \in B \text{ e } \exists a \in A : x = b + a\} && \text{(comutatividade de } +) \\
 &= B \oplus A && \text{(definição de } \oplus)
 \end{aligned}$$

◇

Propriedade (iii):

$$\begin{aligned}
A \oplus (B \oplus C) &= \{a + d, d \in B \oplus C\} && \text{(definição de soma de Minkowski)} \\
&= \{a + (b + c) : a \in A, b \in B, c \in C\} && \text{(definição de soma de Minkowski)} \\
&= \{(a + b) + c : a \in A, b \in B, c \in C\} && \text{(associatividade de grupos)} \\
&= \{e + c : e \in A \oplus B, c \in C\} && \text{(definição de soma de Minkowski)} \\
&= (A \oplus B) \oplus C && \text{(definição de soma de Minkowski)}
\end{aligned}$$

◇

Propriedade (iv):

$$\begin{aligned}
A \oplus (B \cup C) &= \bigcup_{d \in (B \cup C)} A + d && \text{(definição equivalente de } \oplus \text{)} \\
&= \left(\bigcup_{b \in B} A + b \right) \cup \left(\bigcup_{c \in C} A + c \right) && \text{(associativ. e idempotência da união)} \\
&= (A \oplus B) \cup (A \oplus C) && \text{(definição equivalente)}
\end{aligned}$$

◆

Definição 8.15. (Subtração de Minkowski) *Seja E um grupo abeliano. Sejam A e B dois subconjuntos de E . A diferença de Minkowski de A e B é o subconjunto de E , denotado $A \ominus B$ e dado por:*

$$A \ominus B = \{ x \in E : \exists a \in A \text{ e } \exists b \in B, x = a - b \}$$

A subtração de Minkowski, denotada $\ominus$, é o mapeamento dado por:

$$(A, B) \mapsto A \ominus B.$$

[BB94]

Figura 8.3: Diferença de Minkowski de dois subconjuntos.

Proposição 8.16. (Propriedades Básicas da Diferença de Minkowski) *Para todo A, B e C em $\mathcal{P}(E)$ vale:*

$$(i) \quad A \ominus B = \bigcap_{b \in B} A - b \quad \text{(definição equivalente);}$$

$$(ii) A \ominus (B \cup C) = (A \ominus B) \cap (A \ominus C);$$

$$(iii) A \ominus B = \{y \in E : (B + y) \subseteq A\} \quad (\text{definição equivalente}).$$

[BB94]

Demonstração: Vamos mostrar que vale a propriedade (i).

$$\begin{aligned} x \in A \ominus B &\Leftrightarrow \forall b \in B, (\exists a \in A, y = a - b) && (\text{definição de subtração de Minkowski}) \\ &\Leftrightarrow \forall b \in B, (\exists a \in A, a = y + b) && (\text{propriedade de } +) \\ &\Leftrightarrow \forall b \in B, y + b \in A && (\text{equivalência lógica}) \\ &\Leftrightarrow \forall b \in B, y \in A - b && (\text{definição de translado}) \\ &\Leftrightarrow \bigcap_{b \in B} A - b && (\text{definição de interseção}) \end{aligned}$$

□

Vamos mostrar que vale (ii).

$$\begin{aligned} A \ominus (B \cup C) &= \bigcap_{x \in (B \cup C)} A - x && (\text{definição equivalente de } \ominus) \\ &= \bigcap \{A + x : x \in (B \cup C)^t\} && (\text{definição de transposto}) \\ &= \bigcap \{A + x : x \in (B^t \cup C^t)\} && (\text{propriedade do transposto}) \\ &= \bigcap \{A + x : x \in B^t\} \cup \bigcap \{A + x : x \in C^t\} && (\text{definição de } \cup) \\ &= \left(\bigcap_{x \in B} A - x \right) \cup \left(\bigcap_{x \in C} A - x \right) && (\text{definição equivalente de } \ominus) \\ &= (A \ominus B) \cup (A \ominus C) \end{aligned}$$

◆

Definição 8.17. (Função Invariante por Translação) *Seja E um grupo abeliano. Uma função b de E em $\mathcal{P}(E)$ é dita invariante por translação se e somente se, as propriedades equivalentes abaixo são verificadas:*

$$a) \forall x, y \in E, b(y + u) = b(y) + u;$$

$$b) \exists B \in \mathcal{P}(E), \forall y \in E, b(y) = B + y.$$

[BB94]

Proposição 8.18. (propriedades das dilatações invariantes por translação) *Seja E um grupo abeliano. Seja δ uma dilatação sobre $\mathcal{P}(E)$ e seja b sua função estruturante, então as três propriedades abaixo são equivalentes:*

$$(i) b \text{ é invariante por translação};$$

$$(ii) \delta(Y) = Y \oplus B, \text{ sendo } Y \in \mathcal{P}(E) \text{ e } B = \delta(\{\mathbf{o}\}), \text{ onde 'o' é o elemento neutro do grupo}$$

Figura 8.4: Três exemplos de soma de Minkowski.

abeliano E ;

(iii) δ é invariante por translação.

[BB94]

Demonstração: Vamos mostrar que (i) implica (ii). Para todo Y em $\mathcal{P}(E)$ e para todo x em E temos,

$$x \in \delta(Y) \Leftrightarrow x \in \bigcup_{y \in Y} b(y) \quad (\text{caracterização das dilatações})$$

$$\Leftrightarrow x \in \bigcup_{y \in Y} (B + y) \quad (\text{definição de função estruturante e hipótese (i)})$$

$$\Leftrightarrow x \in B \oplus Y \quad (\text{definição de } \oplus)$$

Isto é, pela comutatividade de $\oplus$, para todo $Y \in \mathcal{P}(E)$, $\delta(Y) = Y \oplus B$. Como consequência,

$$\begin{aligned} \delta(\{\mathbf{o}\}) &= \{\mathbf{o}\} \oplus B & (Y = \{\mathbf{o}\}) \\ &= B & (\text{propriedade de } \oplus) \end{aligned}$$

Vamos mostrar que (ii) implica (iii). Para todo x em E e para todo Y em $\mathcal{P}(E)$,

$$\begin{aligned}
\delta(Y + u) &= (Y + u) \oplus B && \text{(hipótese (ii))} \\
&= (Y \oplus B) + u && \text{(propriedade de } \oplus) \\
&= \delta(Y) + u && \text{(hipótese (ii))}
\end{aligned}$$

Isto é, δ é invariante por translação.

Vamos mostrar que (iii) implica (i). Para todo y em E temos,

$$\begin{aligned}
b(y + u) &= \delta(\{y + u\}) && \text{(definição de função estruturante de } \delta) \\
&= \delta(\{x \in E : x = y + u\}) \\
&= \delta(\{x \in E : x - u = y\}) && \text{(propriedade de +)} \\
&= \delta(\{x \in E : x - u \in \{y\}\}) \\
&= \delta(\{y\} + u) && \text{(definição de translado)} \\
&= \delta(\{y\}) + u && \text{(hipótese (iii))} \\
&= b(y) + u && \text{(definição de função estruturante de } \delta)
\end{aligned}$$

Isto é, b é invariante por translação. ■

Podemos, a partir da proposição 8.18 caracterizar as dilatações invariantes por translação. Os conjuntos das dilatações, erosões, anti-dilatações e anti-erosões invariantes por translação são denotados, respectivamente, Δ' , $\mathbf{E}'$, $\Delta^{a'}$, $\mathbf{E}^{a'}$.

Proposição 8.19. (Caracterização das Dilatações Invariantes por Translação) *Seja Δ' o conjunto das dilatações invariantes por translação. O mapeamento de Δ' em $\mathcal{P}^E$, $\delta \mapsto B_\delta$, onde B_δ é o conjunto dado por $B_\delta = \delta(\{\mathbf{o}\})$ é uma bijeção. Seu inverso é $B \mapsto \delta_B$, onde δ_B é a dilatação invariante por translação dada por $\delta_B(Y) = Y \oplus B$ ($Y \in \mathcal{P}(E)$).*

[BB94]

Demonstração: Vamos primeiro verificar que δ_B é, de fato, uma dilatação invariante por translação. Seja $B \in \mathcal{P}(E)$ e seja b um mapeamento de E em $\mathcal{P}(E)$ tal que $b(y) = B + y$, com $Y \in E$ e seja, δ_b a dilatação pela função estruturante b . Para todo Y e B em $\mathcal{P}(E)$,

$$\begin{aligned}
\delta_b(Y) &= Y \oplus B && \text{(proposição 8.18((i) implica (ii)))} \\
&= \delta_B(Y)
\end{aligned}$$

Isto é, pela proposição 8.6, δ_B é uma dilatação e pela proposição 8.18 ((ii) implica (iii)) δ_B é invariante por translação.

Vamos mostrar que $\delta \mapsto B_\delta$ é uma bijeção. Em primeiro lugar, para todo δ em Δ' e para todo Y em $\mathcal{P}(E)$,

$$\begin{aligned}
\delta_{B_\delta}(Y) &= Y \oplus B_\delta && \text{(definição de } \delta_B) \\
&= Y \oplus \delta(\{\mathbf{o}\}) && \text{(definição } B_\delta) \\
&= \delta(Y) && \text{(proposição 8.18((iii) implica (ii))}
\end{aligned}$$

Em outras palavras, para todo $\delta \in \Delta'$, $\delta_{B_\delta} = \delta$. Isto demonstra que o mapeamento $\delta \mapsto B_\delta$ é injetor. Em segundo lugar, para todo $B \in \mathcal{P}(E)$ e para todo $x \in E$ temos,

$$\begin{aligned}
x \in B_{\delta_B} &\Leftrightarrow x \in \delta_B(\{\mathbf{o}\}) && \text{(definição de } B_\delta) \\
&\Leftrightarrow x \in \{\mathbf{o}\} \oplus B && \text{(definição de } \delta_B) \\
&\Leftrightarrow x \in B && \text{(propriedade de } \oplus)
\end{aligned}$$

Em outros termos, para todo $B \in \mathcal{P}(E)$, $B_{\delta_B} = B$. Isto demonstra que o mapeamento $\delta \mapsto B_\delta$ é sobrejetor e conseqüentemente é uma bijeção. ■

Figura 8.5: Bijeção entre as dilatações invariantes por translação e os conjuntos.

De modo análogo, pode-se caracterizar as erosões, anti-erosões e anti-dilatações invariantes por translação.

Dado um conjunto X , os conjuntos $\epsilon_B(X)$, $\delta_B^a(X)$ e $\epsilon_B^a(X)$ são chamados respectivamente *erosão*, *anti-dilatação* e *anti-erosão* de X pelo elemento estruturante B e são dados respectivamente por:

$$\begin{aligned}
\epsilon_B(X) &= X \ominus B; \\
\delta_B^a(X) &= (X \oplus B^c)^c; \\
\epsilon_B^a(X) &= (X \ominus B)^c.
\end{aligned}$$

onde, c é a operação usual de complemento (negação) de conjuntos.

Figura 8.6: Dilatação de um conjunto por um elemento estruturante.

Figura 8.7: Dois modos de construir o dilatado.

A figura 8.7, obtida em [BB94], apresenta dois modos de construir o dilatado de um conjunto. O primeiro (a) baseia-se na definição equivalente de soma de Minkowski apresentada pela proposição 8.14(i): neste caso, o dilatado é obtido “pintando” o elemento estruturante sempre que o centro deste estiver contido na imagem a ser dilatada. O segundo modo considera a definição equivalente de soma de Minkowski dada pela proposição 8.14(v): neste caso, obtemos o dilatado posicionando o elemento estruturante sobre a imagem a ser dilatada e sempre que ele “tocar” a figura a ser dilatada, “pintamos” o centro do elemento estruturante.

Definição 8.20. (União e Interseção entre Operadores) *Sejam ψ_1 e ψ_2 dois operadores sobre $\mathcal{P}(E)$. A união dos operadores ψ_1 e ψ_2 é o operador sobre $\mathcal{P}(E)$, denotado $\psi_1 \vee \psi_2$ e dado por:*

$$(\psi_1 \vee \psi_2)(X) = \psi_1(X) \cup \psi_2(X) \quad (X \in \mathcal{P}(E))$$

A interseção dos operadores ψ_1 e ψ_2 é o operador sobre $\mathcal{P}(E)$, denotado $\psi_1 \wedge \psi_2$ e dado por:

$$(\psi_1 \wedge \psi_2)(X) = \psi_1(X) \cap \psi_2(X) \quad (X \in \mathcal{P}(E))$$

[BB94]

Proposição 8.21. (União e Interseção versus Composição) *Para quaisquer operadores ϕ , ψ_1, ψ_2 sobre $\mathcal{P}(E)$*

$$\begin{aligned}
((\psi_1 \vee \psi_2)\phi(X) &= \psi_1(\phi(X)) \vee \psi_2(\phi(X)) \\
((\psi_1 \wedge \psi_2)\phi(X) &= \psi_1(\phi(X)) \wedge \psi_2(\phi(X))
\end{aligned}$$

para qualquer $X \in \mathcal{P}(E)$.

[BB94]

Figura 8.8: Erosão de um conjunto por um elemento estruturante.

A figura 8.9, obtida em [BB94], mostra um modo de construir o erodido de um conjunto, baseado na definição equivalente de diferença de Minkowski, dada pela proposição 8.16(iii): deste modo, o erodido é obtido “pintando” o centro do elemento estruturante sempre que o elemento estruturante estiver contido na imagem a ser erodida.

Figura 8.9: Modo de construir o erodido.

Proposição 8.22. (Propriedades Básicas da Dilatação e Erosão por um Elemento Estruturante) Sejam δ_B e ϵ_B a dilatação e erosão, respectivamente, pelo elemento estruturante B , isto é,

$$\delta_B(Y) = Y \oplus B \quad e \quad \epsilon_B(Y) = Y \ominus B$$

Então valem as seguintes propriedades para todo B_1 e B_2 em $\mathcal{P}(E)$:

- (i) $\delta_{B_1} \circ \delta_{B_2} = \delta_{B_1 \oplus B_2}$;
- (ii) $\delta_{B_1} \vee \delta_{B_2} = \delta_{B_1 \cup B_2}$;
- (iii) $\epsilon_{B_1} \circ \epsilon_{B_2} = \epsilon_{B_1 \oplus B_2}$;

$$(iv) \epsilon_{B_1} \wedge \epsilon_{B_2} = \epsilon_{B_1 \cup B_2}.$$

[BB94]

Demonstração: Vamos mostrar que vale (i). Queremos mostrar que para todo X em $\mathcal{P}(E)$, $\delta_{B_1} \delta_{B_2}(X) = \delta_{B_1 \oplus B_2}(X)$,

$$\begin{aligned} \delta_{B_1} \circ \delta_{B_2}(X) &= \delta_{B_1}(X \oplus B_2) && \text{(caracterização das dilatações invariantes por translação)} \\ &= (X \oplus B_2) \oplus B_1 && \text{(caracterização das dilatações invariantes por translação)} \\ &= X \oplus (B_2 \oplus B_1) && \text{(associatividade de } \oplus \text{)} \\ &= X \oplus (B_1 \oplus B_2) && \text{(comutatividade de } \oplus \text{)} \\ &= \delta_{B_1 \oplus B_2}(X) && \text{(caracterização das dilatações invariantes por translação)} \end{aligned}$$

◇

Vamos mostrar que vale (ii). Queremos mostrar que para todo X em $\mathcal{P}(E)$, $(\delta_{B_1} \vee \delta_{B_2})(X) = \delta_{B_1 \cup B_2}(X)$,

$$\begin{aligned} \delta_{B_1 \cup B_2}(X) &= X \oplus (B_1 \cup B_2) && \text{(caracterização das dilatações invariantes por translação)} \\ &= (X \oplus B_1) \cup (X \oplus B_2) && \text{(distributividade de } \oplus \text{)} \\ &= \delta_{B_1}(X) \cup \delta_{B_2}(X) && \text{(caracterização das dilatações invariantes por translação)} \\ &= (\delta_{B_1} \vee \delta_{B_2})(X) && \text{(definição de união de operadores)} \end{aligned}$$

◇

A demonstração de (iii) é análoga a demonstração de (i).

Vamos mostrar que vale (iv). Queremos mostrar que para todo X em $\mathcal{P}(E)$, $(\epsilon_{B_1} \wedge \epsilon_{B_2})(X) = \epsilon_{B_1 \cup B_2}(X)$,

$$\begin{aligned} \epsilon_{B_1 \cup B_2}(X) &= X \ominus (B_1 \cup B_2) && \text{(caracterização das erosões invariantes por translação)} \\ &= (X \ominus B_1) \cap (X \ominus B_2) && \text{(propriedade de } \ominus \text{)} \\ &= \epsilon_{B_1}(X) \cap \epsilon_{B_2}(X) && \text{(caracterização das erosões invariantes por translação)} \\ &= (\epsilon_{B_1} \wedge \epsilon_{B_2})(X) && \text{(definição de interseção de operadores)} \end{aligned}$$

◆

Capítulo 9

Verificação de Identidade e Simplificação de Operadores Morfológicos

Suponha que um problema qualquer relativo ao tratamento de imagens seja proposto a dois especialistas em Morfologia Matemática. Seja ψ_1 e ψ_2 dois operadores morfológicos que solucionam o problema proposto, cada um deles tendo sido encontrado por um dos especialistas consultados. Isto sugere tentarmos responder corretamente as seguintes questões:

a) ψ_1 e ψ_2 são, de fato, equivalentes ? Em outros termos, $\psi_1(X) = \psi_2(X)$, $\forall X \in \mathcal{P}(E)$? Existe um algoritmo capaz de verificar, após uma quantidade finita de passos, a equivalência entre operadores morfológicos?

b) Se ψ_1 e ψ_2 são equivalentes, existe um algoritmo capaz de encontrar, após uma quantidade finita de passos, um operador ψ , equivalente a ψ_1 e ψ_2 , tal que ψ seja “mais eficiente” (em relação a alguma noção útil de eficiência adotada, por exemplo, a quantidade de supremos e ínfimos realizados pelo operador) que ambos, ψ_1 e ψ_2 ?

Dizemos que um operador ψ' é “ótimo” se e somente se não existir operador ψ'' , equivalente a ψ' , tal que ψ'' seja “mais eficiente” que ψ' . Nesses termos,

c) Dado um operador ψ qualquer, existe um algoritmo capaz de encontrar (após uma quantidade finita de passos) um operador ψ' , equivalente a ψ , tal que ψ' seja ótimo?

É sabido que existe um algoritmo (implementado) que verifica a equivalência entre operadores morfológicos invariantes por translação e localmente definidos em relação à uma janela W . A verificação da equivalência entre tais operadores, neste caso, é dada pelo cálculo das bases canônicas dos respectivos operadores morfológicos (veja Barrera e Salas [BS96]).

Aqui, a morfologia matemática relativa à classe dos operadores morfológicos invariantes por translação e isotônicos é abordada através da especificação de teorias equacionais, a partir das quais exibimos um método efetivo para a verificação de equivalência entre operadores morfológicos invariantes por translação e isotônicos.

Como vimos, no capítulo 3, qualquer conjunto T de equações induz uma teoria equacional, denotada $Th(T)$, sendo que os elementos de $Th(T)$ são pares de termos. Os teoremas “induzidos” por T , ou, simplesmente, os teoremas de T , são exatamente os elementos de $Th(T)$.

Sabemos que o problema de verificar se um par de termos é, ou não, um teorema (problema da palavra) de uma teoria equacional é indecidível (no caso geral), entretanto, para casos particulares de conjuntos de equações, é possível descrever algoritmos que decidam se um par de termos é, ou não, teorema da teoria dada. Uma abordagem possível para a busca de tais algoritmos baseia-se na tentativa de obter sistemas de reescrita de termos completos (noetherianos e confluentes) e que induzam as mesmas classes de congruência da teoria inicial.

Por um lado, podemos dizer que, um sistema de reescrita de termos é um sistema formal, pois suas regras de reescrita são constituídas por termos e estes são expressões bem-formadas (segundo as regras de formação das frases da linguagem) construídas a partir de um conjunto finito de símbolos funcionais (alfabeto) e um conjunto enumerável de variáveis.

Por outro lado, a morfologia matemática pode ser expressa através de uma linguagem formal [BB94, BB92], a Linguagem Morfológica, constituída de uma gramática e uma semântica, na qual seus operadores são expressos em termos da composição, união e interseção dos 4 operadores elementares (dilatação, erosão, anti-dilatação e anti-erosão) e a Linguagem Morfológica é completa [BB94, BB90], ou seja, qualquer operação entre conjuntos pode ser representada como uma frase desta linguagem. Isso nos motivou a tentarmos descrever um sistema de reescrita para o qual a morfologia matemática constituísse um modelo, de tal modo que pudéssemos resolver o problema proposto, o da verificação automática da identidade entre operadores morfológicos.

Banon e Barrera explicam: “Podemos garantir que a Linguagem Morfológica é completa, pois qualquer operador entre conjuntos pode ser representado por formas canônicas, similares àquelas apresentadas para a decomposição de operadores invariantes por translação, e essas formas são frases válidas da Linguagem Morfológica” [BB94, pg.153].

Neste trabalho, abordamos apenas a classe dos operadores morfológicos invariantes por translação e isotônicos, e neste caso, vimos, no capítulo 8, que as funções estruturantes de tais operadores são chamadas de elementos estruturantes e são representados por símbolos como

$A, B, C, D, A_1, B_1, C_1, D_1, \dots$

As tabelas 9.1 e 9.2 especificam ,respectivamente, a sintaxe e a semântica da morfologia matemática dos operadores invariantes por translação e isotônicos e foram baseadas em [BB94, pg. 152]. A notação dada pela tabela 9.1 possui a forma Backus-Naur (veja a referência [Pagan81] em [BB94]).

$\langle \text{operador} \rangle$	$::=$	$\langle \text{operador elementar} \rangle \mid \langle \text{limitante} \rangle \mid \langle \text{composição} \rangle$
$\langle \text{limitante} \rangle$	$::=$	$\langle \text{argumento} \rangle \langle \text{operação de reticulado} \rangle \langle \text{argumento} \rangle$
$\langle \text{argumento} \rangle$	$::=$	$\langle \text{termo} \rangle \mid \langle \text{composição} \rangle$
$\langle \text{termo} \rangle$	$::=$	$\langle \text{operador elementar} \rangle \mid (\langle \text{limitante} \rangle)$
$\langle \text{operação de composição} \rangle$	$::=$	$\circ$
$\langle \text{composição} \rangle$	$::=$	$\langle \text{termo} \rangle \langle \text{operação de composição} \rangle \langle \text{termo} \rangle \mid \langle \text{composição} \rangle \langle \text{termo} \rangle$
$\langle \text{operador elementar} \rangle$	$::=$	$\langle \text{operador morfológico} \rangle \downarrow \{ \langle \text{elemento estruturante} \rangle \}$
$\langle \text{elemento estruturante} \rangle$	$::=$	$\langle \text{letra} \rangle \mid \langle \text{letra} \rangle \langle \text{número} \rangle$
$\langle \text{número} \rangle$	$::=$	$\langle \text{dígito} \rangle \mid \langle \text{número} \rangle \langle \text{dígito} \rangle$
$\langle \text{operação de reticulado} \rangle$	$::=$	$\vee \mid \wedge$
$\langle \text{operador morfológico} \rangle$	$::=$	$\epsilon \mid \delta$
$\langle \text{letra} \rangle$	$::=$	$A \mid B \mid C \mid D$
$\langle \text{dígito} \rangle$	$::=$	$0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

Tabela 9.1: Sintaxe da Linguagem Morfológica restrita aos operadores i.t. e isotônicos.

Sendo assim, chamamos de operador morfológico qualquer expressão bem-formada segundo a sintaxe apresentada pela tabela 9.1.

Exemplo 9.1. Segundo a sintaxe dada pela tabela 9.1, as expressões $(\epsilon_A \wedge \delta_{B_1}) \vee (\delta_A \circ \delta_{A_2})$ e $(\delta_{A_1} \wedge \epsilon_{C_2}) \circ \epsilon_{A_2}$ são termos da Linguagem Morfológica.

$\mathcal{I}[A] \equiv$	$X \in \mathcal{P}(E)$	
$\mathcal{I}[\delta_A] \equiv$	$\psi : \mathcal{P}(E) \mapsto \mathcal{P}(E), \psi(X) = X \oplus A$	$(X \in \mathcal{P}(E)), \text{onde } \oplus \text{ é a soma de Minkovisk.}$
$\mathcal{I}[\epsilon_A] \equiv$	$\psi : \mathcal{P}(E) \mapsto \mathcal{P}(E), \psi(X) = X \ominus A$	$(X \in \mathcal{P}(E)), \text{onde } \ominus \text{ é a diferença de Minkovisk.}$
$\mathcal{I}[(\psi)] \equiv$	$\mathcal{I}[\psi]$	
$\mathcal{I}[\psi_1 \vee \psi_2] \equiv$	$\psi : \mathcal{P}(E) \mapsto \mathcal{P}(E), \psi(X) = \mathcal{I}[\psi_1](X) \cup \mathcal{I}[\psi_2](X)$	$(X \in \mathcal{P}(E)), \cup \text{ é a união usual de conjuntos.}$
$\mathcal{I}[\psi_1 \wedge \psi_2] \equiv$	$\psi : \mathcal{P}(E) \mapsto \mathcal{P}(E), \psi(X) = \mathcal{I}[\psi_1](X) \cap \mathcal{I}[\psi_2](X)$	$(X \in \mathcal{P}(E)), \cap \text{ é a interseção usual de conjuntos.}$
$\mathcal{I}[\psi_1 \circ \psi_2] \equiv$	$\psi : \mathcal{P}(E) \mapsto \mathcal{P}(E), \psi(X) = \mathcal{I}[\psi_1](\mathcal{I}[\psi_2](X))$	$(X \in \mathcal{P}(E))$

Tabela 9.2: Semântica da Linguagem Morfológica restrita aos operadores i.t. e isotônicos.

Por exemplo, a frase $\delta_A \circ (\epsilon_A \circ \delta_A)$ tem exatamente o mesmo significado da frase δ_A segundo a semântica da morfologia matemática. Isto é, para qualquer X em $\mathcal{P}(E)$ temos $\mathcal{I}[\delta_A \circ (\epsilon_A \circ \delta_A)](X) = \mathcal{I}[\delta_A](X)$, pois:

$$\begin{aligned} \mathcal{I}[\delta_A \circ (\epsilon_A \circ \delta_A)](X) &= \mathcal{I}[\delta_A](\mathcal{I}[(\epsilon_A \circ \delta_A)](X)) = \mathcal{I}[\delta_A](\mathcal{I}[\epsilon_A \circ \delta_A](X)) = \mathcal{I}[\delta_A](\mathcal{I}[\epsilon_A](\mathcal{I}[\delta_A](X))) = \\ &= \mathcal{I}[\delta_A](\mathcal{I}[\epsilon_A](X \oplus A)) = \mathcal{I}[\delta_A]((X \oplus A) \ominus A) = ((X \oplus A) \ominus A) \oplus A. \end{aligned}$$

e

$$\mathcal{I}[\delta_A](X) = X \oplus A.$$

A identidade entre $((X \oplus A) \ominus A) \oplus A$ e $X \oplus A$ segue como consequência das propriedades de soma e subtração de Minkowski e, portanto, de fato, $\mathcal{I}[\delta_A \circ (\epsilon_A \circ \delta_A)](X) = \mathcal{I}[\delta_A](X)$, $\forall X \in \mathcal{P}(E)$ e neste caso, diremos que $\delta_A \circ (\epsilon_A \circ \delta_A)$ e δ_A são operadores morfológicos *equivalentes*.

Pela proposição 8.19, as dilatações e erosões invariantes por translação são caracterizadas por elementos estruturantes, que são subconjuntos de E . Daqui por diante, as imagens binárias consideradas são planas, isto é, $E \subseteq \mathbb{Z}^2$.

Os *elementos estruturantes primitivos* são subconjuntos do quadrado 3×3 centrado na origem (*quadrado elementar*) de $E \subseteq \mathbb{Z}^2$.

São alguns exemplos de elementos estruturantes primitivos: $\begin{bmatrix} 111 \\ 111 \\ 111 \end{bmatrix}$, $\begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix}$, $\begin{bmatrix} 111 \\ 000 \\ 111 \end{bmatrix}$, $\begin{bmatrix} 011 \\ 011 \\ 011 \end{bmatrix}$.

Suponha que B seja um subconjunto (finito) qualquer de $\mathbb{Z}^2$. É fácil ver que B pode ser representado pela união de translados de elementos estruturantes primitivos, isto é, $B = \bigcup_{(1 \leq i \leq n)} B_i + u_i$, onde $B_i \subseteq \mathbb{Z}^2$ e $u_i = (x_i, y_i)$, com $x_i, y_i \in \mathbb{Z}$, para algum n em $\mathbb{Z}_+$.

A figura 9.1 ilustra este fato e foi baseada num exemplo obtido em Banon e Barrera [BB94, pg.154].

Figura 9.1: Decomposição em termos de subconjuntos do quadrado elementar.

Podemos também decompor qualquer subconjunto não vazio (finito) de $\mathbb{Z}^2$ como uniões de translações do subconjunto $\{(0,0)\}$ que denotamos simplesmente por $\{\mathbf{o}\}$. Isto é, se B é um subconjunto qualquer de $\mathbb{Z}^2$, então $B = \bigcup_{i(1 \leq i \leq n)} \{\mathbf{o}\} + u_i$, onde $u_i = (x_i, y_i)$, com $x_i, y_i \in \mathbb{Z}$, para algum n em $\mathbb{Z}_+$. A figura 9.2 ilustra este fato.

$$B = \{(\{\mathbf{o}\} + (1,1)) \cup (\{\mathbf{o}\} + (1,2)) \cup (\{\mathbf{o}\} + (0,3)) \cup (\{\mathbf{o}\} + (2,2)) \cup (\{\mathbf{o}\} + (3,1)) \cup (\{\mathbf{o}\} + (2,3))\}$$

Figura 9.2: Decomposição de um subconjunto em termos de uniões de translações da origem.

Os elementos estruturantes primitivos $\begin{bmatrix} 000 \\ 001 \\ 000 \end{bmatrix}$, $\begin{bmatrix} 000 \\ 100 \\ 000 \end{bmatrix}$, $\begin{bmatrix} 010 \\ 000 \\ 000 \end{bmatrix}$ e $\begin{bmatrix} 000 \\ 000 \\ 010 \end{bmatrix}$ são denotados, respectivamente, por L (leste), O (oeste), N (norte), S (sul).

Note que, o elemento estruturante primitivo L é o conjunto unitário cujo elemento é o par ordenado (elemento de $\mathbb{Z}^2$) $(0,1)$, isto é, $L = \{(0,1)\}$. Analogamente, $O = \{(0,-1)\}$, $N = \{(-1,0)\}$ e $S = \{(1,0)\}$. Como a composição de operadores é associativa, então não há necessidade do uso do símbolo de parênteses na construção de uma frase formada por composições de operadores.

Exemplo 9.2. Ao invés de escrevermos $\psi_1 \circ (\psi_2 \circ \psi_3) \circ (\psi_4 \circ \psi_5)$ podemos escrever simplesmente $\psi_1 \circ \psi_2 \circ \psi_3 \circ \psi_4 \circ \psi_5$, uma vez que a forma pela qual os ψ 's se associam é irrelevante para o resultado final no cálculo do operador morfológico. Em outras palavras, o resultado produzido pelo operador $\psi_1 \circ (\psi_2 \circ \psi_3) \circ (\psi_4 \circ \psi_5)$ é idêntico ao resultado produzido por qualquer outro operador que difira deste apenas na ordem de posicionamento dos parênteses. Ou seja, $\psi_1 \circ (\psi_2 \circ \psi_3) \circ (\psi_4 \circ \psi_5) = (((\psi_1 \circ \psi_2) \circ \psi_3) \circ \psi_4) \circ \psi_5 = (\psi_1 \circ \psi_2) \circ (\psi_3 \circ \psi_4) \circ \psi_5 = \dots$

Definição 9.3. ("Singleton") Um "singleton" é um elemento estruturante de cardinalidade unitária.

Exemplo 9.4. O elemento estruturante $A = \{(3,2)\}$ é um singleton, pois $|A| = 1$. O elemento estruturante $B = [101]$ não é um singleton, pois $|B| = 2$.

Observe que, como consequência da proposição 8.22 (i) e da definição 8.13 (soma de

Minkowski), qualquer dilatação cujo elemento estruturante é um singleton pode ser realizada através de composições de dilatações pelos elementos estruturantes L, O, N ou S .

Exemplo 9.5. $\delta_{\{(2,3)\}} = \delta_S \circ (\delta_S \circ (\delta_L \circ (\delta_L \circ \delta_L))) \stackrel{\text{notac.}}{=} \delta_S \circ \delta_S \circ \delta_L \circ \delta_L \circ \delta_L$.

Do mesmo modo, como consequência da proposição 8.22 (iii) e da definição 8.15 (subtração de Minkowski) qualquer erosão cujo elemento estruturante é um singleton pode ser realizada através de composições de erosões pelos elementos estruturantes L, O, N ou S .

Exemplo 9.6. $\epsilon_{\{(-1,2)\}} = \epsilon_N \circ \epsilon_L \circ \epsilon_L$.

É fácil ver que, quando um operador morfológico é formado por composições de dilatações e/ou erosões pelos elementos estruturantes L, O, N ou S , tem-se que a operação de composição “ $\circ$ ”, neste caso, possui a propriedade de ser associativa-comutativa e, portanto, não importa a ordem com a qual é aplicada a composição das dilatações e/ou erosões.

Exemplo 9.7. $\delta_{\{(2,3)\}} = \delta_S \circ \delta_S \circ \delta_L \circ \delta_L \circ \delta_L = \delta_L \circ \delta_S \circ \delta_S \circ \delta_L \circ \delta_L = \delta_L \circ \delta_L \circ \delta_S \circ \delta_S \circ \delta_L$;
 $\delta_{(1,0)} \circ \epsilon_{(2,1)} = \delta_S \circ ((\epsilon_S \circ \epsilon_S) \circ \epsilon_L) = \delta_S \circ \epsilon_S \circ \epsilon_S \circ \epsilon_L = \epsilon_L \circ \epsilon_S \circ \epsilon_S \circ \delta_S = \delta_S \circ \epsilon_L \circ \epsilon_S \circ \epsilon_S = \delta_S \circ \epsilon_S \circ \epsilon_L \circ \epsilon_S$.

Se B é um subconjunto de $\mathbb{Z}^2$ e u um vetor, isto é, u um elemento de $\mathbb{Z}^2$, então $\{u\}$ é um subconjunto de $\mathbb{Z}^2$ e, portanto, pela proposição 8.14 (i) (definição equivalente de soma de Minkowski), $B \oplus \{u\} = \bigcup_{v \in \{u\}} B + v = B + u$. Em particular, se $B = \{\mathbf{o}\}$ então $\{\mathbf{o}\} + u = \{\mathbf{o}\} \oplus \{u\}$.

Denotamos por ι o operador identidade sobre $\mathcal{P}(E)$, isto é, $\iota(X) = X$, $\forall X \in \mathcal{P}(E)$.

Note que o símbolo funcional $\vee$ é binário. Como sabemos, a operação de união entre operadores morfológicos, representada pelo símbolo $\vee$, é associativa e, sendo assim, podemos, por simplicidade de notação, escrever as frases formadas por uniões de operadores morfológicos sem o auxílio dos símbolos de parênteses. Por exemplo, podemos escrever simplesmente $\psi_1 \vee \psi_2 \vee \psi_3 \vee \psi_4 \vee \psi_5$ ao invés de $((\psi_1 \vee \psi_2) \vee (\psi_3 \vee \psi_4)) \vee \psi_5$. O mesmo raciocínio aplica-se ao símbolo funcional $\wedge$ (interseção entre operadores morfológicos). Entretanto, em certas ocasiões, estaremos interessados em deixar explícito a colocação dos parênteses na frase morfológica formada por uniões, erosões ou composições de operadores morfológicos.

Seja δ_A uma dilatação invariante por translação. Logo, pelas observações acima e pela proposição 8.22 (i) e (ii), temos:

$$\begin{aligned}
\delta_A &= \delta_{(\{o\}+u_1) \cup \dots \cup (\{o\}+u_n)} \\
&= \delta_{\{o\}+u_1} \vee \dots \vee \delta_{\{o\}+u_n} \\
&= \delta_{\{o\} \oplus \{u_1\}} \vee \dots \vee \delta_{\{o\} \oplus \{u_n\}} \\
&= (\delta_{\{o\}} \circ \delta_{\{u_1\}}) \vee \dots \vee (\delta_{\{o\}} \circ \delta_{\{u_n\}}) \\
&= (\iota \circ \delta_{\{u_1\}}) \vee \dots \vee (\iota \circ \delta_{\{u_n\}}) \\
&= \delta_{\{u_1\}} \vee \dots \vee \delta_{\{u_n\}} \\
&= (\delta_{1_1} \circ \dots \circ \delta_{1_{p_1}}) \vee \dots \vee (\delta_{n_1} \circ \dots \circ \delta_{n_{p_n}})
\end{aligned}$$

para algum $n \in \mathbb{Z}_+$ e algum $u_1, \dots, u_n \in \mathbb{Z}^2$,

onde, $\delta_{\{u_k\}} = (\delta_{k_1} \circ \delta_{k_2} \circ \dots \circ \delta_{k_{p_k-1}} \circ \delta_{k_{p_k}})$, $1 \leq k \leq n$, e para cada k , $p_k \in \mathbb{Z}_+$ e $\delta_{k_j} \in \{\delta_L, \delta_O, \delta_N, \delta_S, \iota\}$, com $1 \leq j \leq p_k$.

Conseqüentemente, cada dilatação invariante por translação pode ser expressa através de uma frase que contém unicamente símbolos do conjunto $\{\delta_L, \delta_O, \delta_N, \delta_S, \iota, \circ, \vee\}$. mais os símbolos auxiliares de parênteses “(” e “)”.

Exemplo 9.8. Considere a dilatação δ_B , onde $B = \begin{bmatrix} 0001 \\ 0110 \\ 0010 \\ 0010 \\ 0100 \end{bmatrix}$.

$$\begin{aligned}
\delta_B &= \delta_{(\{o\}+(1,1)) \cup (\{o\}+(1,2)) \cup (\{o\}+(0,3)) \cup (\{o\}+(2,2)) \cup (\{o\}+(3,1))} \\
&= \delta_{\{o\}+(1,1)} \vee \delta_{\{o\}+(1,2)} \vee \delta_{\{o\}+(0,3)} \vee \delta_{\{o\}+(2,2)} \vee \delta_{\{o\}+(3,1)} \\
&= \delta_{\{o\} \oplus \{(1,1)\}} \vee \delta_{\{o\} \oplus \{(1,2)\}} \vee \delta_{\{o\} \oplus \{(0,3)\}} \vee \delta_{\{o\} \oplus \{(2,2)\}} \vee \delta_{\{o\} \oplus \{(3,1)\}} \\
&= (\delta_{\{o\}} \circ \delta_{\{(1,1)\}}) \vee (\delta_{\{o\}} \circ \delta_{\{(1,2)\}}) \vee (\delta_{\{o\}} \circ \delta_{\{(0,3)\}}) \vee (\delta_{\{o\}} \circ \delta_{\{(2,2)\}}) \vee (\delta_{\{o\}} \circ \delta_{\{(3,1)\}}) \\
&= (\iota \circ \delta_{\{(1,1)\}}) \vee (\iota \circ \delta_{\{(1,2)\}}) \vee (\iota \circ \delta_{\{(0,3)\}}) \vee (\iota \circ \delta_{\{(2,2)\}}) \vee (\iota \circ \delta_{\{(3,1)\}}) \\
&= \delta_{\{(1,1)\}} \vee \delta_{\{(1,2)\}} \vee \delta_{\{(0,3)\}} \vee \delta_{\{(2,2)\}} \vee \delta_{\{(3,1)\}} \\
&= (\delta_S \circ \delta_L) \vee (\delta_S \circ \delta_L \circ \delta_L) \vee (\delta_L \circ \delta_L \circ \delta_L) \vee (\delta_S \circ \delta_S \circ \delta_L \circ \delta_L) \vee (\delta_S \circ \delta_S \circ \delta_S \circ \delta_L)
\end{aligned}$$

Sendo assim, podemos representar δ_B através do seguinte diagrama de árvore.

De modo análogo, se ϵ_A é uma erosão invariante por translação então, pelas observações acima e pela proposição 8.22 (iii) e (iv) temos:

$$\begin{aligned}
\epsilon_A &= \epsilon(\{o\}+u_1) \cup \dots \cup (\{o\}+u_n) \\
&= \epsilon_{\{o\}+u_1} \wedge \dots \wedge \epsilon_{\{o\}+u_n} \\
&= \epsilon_{\{o\} \oplus \{u_1\}} \wedge \dots \wedge \epsilon_{\{o\} \oplus \{u_n\}} \\
&= (\epsilon_{\{o\}} \circ \epsilon_{\{u_1\}}) \wedge \dots \wedge (\epsilon_{\{o\}} \circ \epsilon_{\{u_n\}}) \\
&= (\iota \circ \epsilon_{\{u_1\}}) \wedge \dots \wedge (\iota \circ \epsilon_{\{u_n\}}) \\
&= \epsilon_{\{u_1\}} \wedge \dots \wedge \epsilon_{\{u_n\}} \\
&= (\epsilon_{1_1} \circ \dots \circ \epsilon_{1_{p_1}}) \wedge \dots \wedge (\epsilon_{n_1} \circ \dots \circ \epsilon_{n_{p_n}})
\end{aligned}$$

para algum $n \in \mathbb{Z}_+$ e algum $u_1, \dots, u_n \in \mathbb{Z}^2$, onde, $\epsilon_{\{u_k\}} = (\epsilon_{k_1} \circ \epsilon_{k_2} \circ \dots \circ \epsilon_{k_{p_k-1}} \circ \epsilon_{k_{p_k}})$, $1 \leq k \leq n$, e para cada $k, p_k \in \mathbb{Z}_+$ e $\epsilon_{k_j} \in \{\epsilon_L, \epsilon_O, \epsilon_N, \epsilon_S, \iota\}$, com $(1 \leq j \leq p_k)$.

Entretanto, note que $\delta_L(X) = X \oplus L = X \oplus \{(0, 1)\} = X + (0, 1) = X - (0, -1) = X \ominus \{(0, -1)\} = X \ominus O = \epsilon_O(X)$, $\forall X \in \mathcal{P}(\mathbb{Z}^2)$. Isto é, $\delta_L = \epsilon_O$. De modo análogo, verifica-se que $\delta_O = \epsilon_L$, $\delta_N = \epsilon_S$ e $\delta_S = \epsilon_N$. Conseqüentemente, cada erosão invariante por translação pode ser expressa através de uma frase que contém unicamente símbolos do conjunto $\{\delta_L, \delta_O, \delta_N, \delta_S, \iota, \circ, \wedge\}$ mais os símbolos auxiliares de parênteses “(” e “)”. Portanto, se ϵ_A é um erosão invariante por translação temos:

$$\begin{aligned}
\epsilon_A &= \\
&\vdots \\
&= (\epsilon_{1_1} \circ \dots \circ \epsilon_{1_{p_1}}) \wedge \dots \wedge (\epsilon_{n_1} \circ \dots \circ \epsilon_{n_{p_n}}) \\
&= (\delta_{1_1} \circ \dots \circ \delta_{1_{p_1}}) \wedge \dots \wedge (\delta_{n_1} \circ \dots \circ \delta_{n_{p_n}})
\end{aligned}$$

onde, para cada $1 \leq k \leq n$ e para cada $1 \leq j \leq p_k$, $\delta_{k_j} = \epsilon_{k_j}$.

Exemplo 9.9. Considere a erosão ϵ_B , onde $B = \begin{bmatrix} 0001 \\ 0110 \\ 0010 \\ 0100 \end{bmatrix}$.

$$\begin{aligned}
\epsilon_B &= \epsilon(\{o\}+(0,0)) \cup (\{o\}+(1,-1)) \cup (\{o\}+(-1,-1)) \cup (\{o\}+(-1,0)) \cup (\{o\}+(-2,1)) \\
&= \epsilon_{\{o\}+(0,0)} \wedge \epsilon_{\{o\}+(1,-1)} \wedge \epsilon_{\{o\}+(-1,-1)} \wedge \epsilon_{\{o\}+(-1,0)} \wedge \epsilon_{\{o\}+(-2,1)} \\
&= \epsilon_{\{o\} \oplus \{(0,0)\}} \wedge \epsilon_{\{o\} \oplus \{(1,-1)\}} \wedge \epsilon_{\{o\} \oplus \{(-1,-1)\}} \wedge \epsilon_{\{o\} \oplus \{(-1,0)\}} \wedge \epsilon_{\{o\} \oplus \{(-2,1)\}} \\
&= (\epsilon_{\{o\}} \circ \epsilon_{\{(0,0)\}}) \wedge (\epsilon_{\{o\}} \circ \epsilon_{\{(1,-1)\}}) \wedge (\epsilon_{\{o\}} \circ \epsilon_{\{(-1,-1)\}}) \wedge (\epsilon_{\{o\}} \circ \epsilon_{\{(-1,0)\}}) \wedge (\epsilon_{\{o\}} \circ \epsilon_{\{(-2,1)\}}) \\
&= (\iota \circ \epsilon_{\{(0,0)\}}) \wedge (\iota \circ \epsilon_{\{(1,-1)\}}) \wedge (\iota \circ \epsilon_{\{(-1,-1)\}}) \wedge (\iota \circ \epsilon_{\{(-1,0)\}}) \wedge (\iota \circ \epsilon_{\{(-2,1)\}}) \\
&= \epsilon_{\{(0,0)\}} \wedge \epsilon_{\{(1,-1)\}} \wedge \epsilon_{\{(-1,-1)\}} \wedge \epsilon_{\{(-1,0)\}} \wedge \epsilon_{\{(-2,1)\}} \\
&= \iota \wedge (\epsilon_S \circ \epsilon_O) \wedge (\epsilon_N \circ \epsilon_O) \wedge \epsilon_N \wedge (\epsilon_N \circ \epsilon_N \circ \epsilon_L) \\
&= \iota \wedge (\delta_N \circ \delta_L) \wedge (\delta_S \circ \delta_L) \wedge \delta_S \wedge (\delta_S \circ \delta_S \circ \delta_O)
\end{aligned}$$

Sendo assim, podemos representar ϵ_B através do seguinte diagrama de árvore.

Ainda nos primeiros anos de pesquisa de Matheron e Serra sobre as propriedades das dilatações e erosões, eles mostraram que qualquer operador invariante por translação e isotônico pode ser decomposto em termos de uniões e interseções de composições de dilatações ou erosões invariantes por translação.

Exemplo 9.10. $\psi = (\delta_A \circ \epsilon_A) \vee \delta_B$, com $A, B \in \mathcal{P}(E)$ é um operador invariante por translação e isotônico. A figura 9.3 exibe este operador como uma estrutura de árvore.

Figura 9.3: Estrutura de árvore do operador $(\delta_A \circ \epsilon_A) \vee \delta_B$.

Portanto, pelo que acabamos de ver, qualquer operador invariante por translação e isotônico para imagens binárias pode ser expresso através de uma frase que contém unicamente símbolos do conjunto $\{\delta_L, \delta_O, \delta_N, \delta_S, \iota, \vee, \wedge, \circ\}$ mais os símbolos auxiliares “(” e “)”.

Definição 9.11. (A Forma Decomposta de um Operador Morfológico) Dizemos que ψ é um operador morfológico decomposto se ψ está escrito em termos dos símbolos do conjunto $\{\delta_L, \delta_O, \delta_N, \delta_S, \iota, \vee, \wedge, \circ\}$ mais os símbolos auxiliares “(” e “)”.

Note que todo operador morfológico decomposto é um operador invariante por translação e isotônico.

Exemplo 9.12. $(\delta_L \vee \delta_O \vee \iota) \circ \delta_N$ é um operador morfológico decomposto.

$\delta_{[011]}$ não é um operador morfológico decomposto.

$\delta_{\begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix}}$ não é um operador morfológico decomposto.

$(\delta_{\begin{bmatrix} 111 \\ 111 \\ 111 \end{bmatrix}} \circ \delta_L) \wedge \epsilon_N$ não é um operador morfológico decomposto.

Exemplo 9.13. O operador morfológico $\delta_A \circ \epsilon_A$, onde $A = \begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix}$, pode ser decomposto como segue:

$$\begin{aligned} \delta \begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix} \circ \epsilon \begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix} &= \\ &= (\delta_{\{(0,0)\}} \vee \delta_{\{(1,0)\}} \vee \delta_{\{(0,-1)\}} \vee \delta_{\{(0,1)\}} \vee \delta_{\{(-1,0)\}}) \circ \\ &\quad (\epsilon_{\{(0,0)\}} \wedge \epsilon_{\{(1,0)\}} \wedge \epsilon_{\{(0,-1)\}} \wedge \epsilon_{\{(0,1)\}} \wedge \epsilon_{\{(-1,0)\}}) \\ &= (\iota \vee \delta_S \vee \delta_O \vee \delta_L \vee \delta_N) \circ (\iota \wedge \epsilon_S \wedge \epsilon_O \wedge \epsilon_L \wedge \epsilon_N) \end{aligned}$$

Exemplo 9.14. O operador morfológico invariante por translação e isotônico $\left(\delta \begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix} \circ \epsilon \begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix} \right) \vee$

$\delta \begin{bmatrix} 001 \\ 001 \\ 111 \end{bmatrix}$ pode ser decomposto como segue:

$$\begin{aligned} & \left(\delta \begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix} \circ \epsilon \begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix} \right) \vee \delta \begin{bmatrix} 001 \\ 001 \\ 111 \end{bmatrix} = \\ &= ((\iota \vee \delta_S \vee \delta_O \vee \delta_L \vee \delta_N) \circ (\iota \wedge \epsilon_S \wedge \epsilon_O \wedge \epsilon_L \wedge \epsilon_N)) \vee \delta \begin{bmatrix} 001 \\ 001 \\ 111 \end{bmatrix} \\ &= ((\iota \vee \delta_S \vee \delta_O \vee \delta_L \vee \delta_N) \circ (\iota \wedge \epsilon_S \wedge \epsilon_O \wedge \epsilon_L \wedge \epsilon_N)) \vee (\delta_{\{(1,-1)\}} \vee \delta_{\{(1,0)\}} \vee \delta_{\{(1,1)\}} \vee \delta_{\{(0,1)\}} \vee \delta_{\{(-1,1)\}}) \\ &= ((\iota \vee \delta_S \vee \delta_O \vee \delta_L \vee \delta_N) \circ (\iota \wedge \epsilon_S \wedge \epsilon_O \wedge \epsilon_L \wedge \epsilon_N)) \vee ((\delta_S \circ \delta_O) \vee \delta_S \vee (\delta_S \circ \delta_L) \vee \delta_L \vee (\delta_N \circ \delta_L)) \\ &= ((\iota \vee \delta_S \vee \delta_O \vee \delta_L \vee \delta_N) \circ (\iota \wedge \delta_N \wedge \delta_L \wedge \delta_O \wedge \delta_S)) \vee ((\delta_S \circ \delta_O) \vee \delta_S \vee (\delta_S \circ \delta_L) \vee \delta_L \vee (\delta_N \circ \delta_L)) \end{aligned}$$

O diagrama a seguir expressa o operador morfológico (em sua forma decomposta) $\left(\delta \begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix} \circ \epsilon \begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix} \right) \vee \delta \begin{bmatrix} 001 \\ 001 \\ 111 \end{bmatrix}$ como um diagrama de árvore.

É fácil ver que processo de decompor um operador morfológico pode ser realizado por

um algoritmo trivial. Seja $\mathcal{A}$ um tal algoritmo.

Exemplo 9.15.
$$\mathcal{A}\left(\left(\delta\begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix} \circ \epsilon\begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix}\right) \vee \delta\begin{bmatrix} 001 \\ 001 \\ 111 \end{bmatrix}\right) = \left(\mathcal{A}\left(\delta\begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix}\right) \circ \mathcal{A}\left(\epsilon\begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix}\right)\right) \vee \mathcal{A}\left(\delta\begin{bmatrix} 001 \\ 001 \\ 111 \end{bmatrix}\right) =$$

$$= ((\iota \vee \delta_S \vee \delta_O \vee \delta_L \vee \delta_N) \circ (\iota \wedge \delta_S \wedge \delta_O \wedge \delta_L \wedge \delta_N)) \vee ((\delta_S \circ \delta_O) \vee \delta_S \vee (\delta_S \circ \delta_L) \vee \delta_L \vee (\delta_N \circ \delta_L)).$$

Consideremos a *complexidade* de um operador morfológico invariante por translação e isotônico como sendo a quantidade de uniões e interseções que são necessárias para o cálculo do operador, considerando para tal, que as implementações da dilatação e erosão numa máquina morfológica sejam baseadas, respectivamente, pelas definições equivalentes de soma (proposição 8.14(i)) e subtração (proposição 8.16(i)) de Minkowski.

Denotemos $\|\psi\|$ a complexidade de um operador morfológico ψ invariante por translação e isotônico.

Exemplo 9.16. A complexidade do operador morfológico $\delta\begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix}$ é $\|\delta\begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix}\| = 4;$

a complexidade do operador morfológico $\epsilon\begin{bmatrix} 111 \\ 111 \\ 111 \end{bmatrix}$ é $\|\epsilon\begin{bmatrix} 111 \\ 111 \\ 111 \end{bmatrix}\| = 8;$

$$\left\| \left(\delta\begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix} \vee \epsilon\begin{bmatrix} 111 \\ 111 \\ 111 \end{bmatrix} \right) \circ \delta\begin{bmatrix} 0001 \\ 0110 \\ 0010 \\ 0100 \end{bmatrix} \right\| = 4 + 1 + 8 + 4 = 17.$$

É fácil ver que qualquer operador morfológico possui exatamente a mesma complexidade de que sua respectiva forma decomposta, isto é, se ψ é um operador morfológico, então $\|\psi\| = \|\mathcal{A}(\psi)\|$, é isto o que mostra o lema 9.17 a seguir.

Lema 9.17. *Seja ψ um operador morfológico invariante por translação e isotônico. A complexidade de ψ é exatamente a mesma complexidade da forma decomposta de ψ , isto é, $\|\psi\| = \|\mathcal{A}(\psi)\|$.*

Demonstração: Como consequência imediata da noção dada no parágrafo anterior sobre complexidade de um operador morfológico invariante por translação e isotônico, temos que a complexidade de um operador ψ é a soma das complexidades das dilatações e erosões que ocorrem em ψ mais a quantidade de símbolos de união ($\vee$) e interseção ($\wedge$) que ocorrem em ψ . Sabemos que a decomposição de um operador ψ consiste em decompor as dilatações e erosões que ocorrem em ψ . Entretanto, é fácil ver que, $\|\delta_A\| = \|\mathcal{A}(\delta_A)\|$ e $\|\epsilon_A\| = \|\mathcal{A}(\epsilon_A)\|$ e, portanto,

concluimos que $\|\psi\| = \|\mathcal{A}(\psi)\|$.

◆

Observe que $\|\delta_L\| = \|\delta_O\| = \|\delta_N\| = \|\delta_S\| = \|\iota\| = 0$ e, conseqüentemente, para calcular a complexidade de um operador decomposto basta contar a quantidade de uniões ($\vee$) e interseções ($\wedge$) que ocorrem no termo que expressa o operador.

Exemplo 9.18.

$$\|(\iota \vee \delta_S \vee \delta_O \vee \delta_L \vee \delta_N) \circ (\iota \wedge \epsilon_N \wedge \epsilon_L \wedge \epsilon_O \wedge \epsilon_S)\| = 8;$$

$$\|((\iota \vee \delta_S \vee \delta_O \vee \delta_L \vee \delta_N) \circ (\iota \wedge \delta_S \wedge \delta_O \wedge \delta_L \wedge \delta_N)) \vee (\delta_S \circ \delta_O \vee \delta_S \vee \delta_S \circ \delta_L \vee \delta_L \vee \delta_N \circ \delta_L)\| = 13.$$

Como já mencionamos no início deste capítulo, a frase $\delta_A \circ (\epsilon_A \circ \delta_A)$ tem exatamente o mesmo significado da frase δ_A segundo a semântica da morfologia matemática, isto é, para qualquer X em $\mathcal{P}(E)$ temos $\mathcal{I}[\delta_A \circ (\epsilon_A \circ \delta_A)](X) = \mathcal{I}[\delta_A](X)$ ou simplesmente $\delta_A \circ (\epsilon_A \circ \delta_A) = \delta_A$.

Considere o elemento estruturante $A = [101]$. Neste caso $\mathcal{A}(\delta_A \circ (\epsilon_A \circ \delta_A)) = (\delta_L \vee \delta_O) \circ ((\delta_O \wedge \delta_L) \circ (\delta_L \vee \delta_O))$ e $\mathcal{A}(\delta_A) = \delta_L \vee \delta_O$.

Para o cálculo de $\delta_A \circ (\epsilon_A \circ \delta_A)$ são realizadas duas operações de uniões e uma de interseção, enquanto que para o cálculo de δ_A é realizada apenas uma união. Entretanto, estes são operadores morfológicos equivalentes pois, como já vimos, a identidade $\delta_A \circ (\epsilon_A \circ \delta_A) = \delta_A$ é satisfeita pela semântica da morfologia matemática. Isto sugere pensarmos que haja operações “desnecessárias” realizadas por $\delta_A \circ (\epsilon_A \circ \delta_A)$.

Definição 9.19. (Operador Morfológico Seqüencial) Dizemos que ψ é um operador morfológico seqüencial se e somente se

$$\psi = \begin{cases} f & \text{onde } f \in \{\delta_L, \delta_O, \delta_S, \delta_N, \iota\}; \\ \text{ou} \\ (\psi_1 \circ \psi_2) & \text{onde } \psi_1 \text{ e } \psi_2 \text{ são operadores morfológicos seqüenciais.} \end{cases}$$

Exemplo 9.20. $(\delta_N \circ \delta_O)$ é um operador morfológico seqüencial;

$((\iota \circ (\delta_L \circ \delta_L)) \circ \delta_O)$ é um operador morfológico seqüencial.

Definição 9.21. (Uniões ou Interseções de Operadores Morfológicos Seqüenciais) Dizemos que ψ é um operador morfológico formado por uniões ou interseções de operadores morfológicos seqüenciais se e somente se

$$\psi = \begin{cases} \psi' & \text{onde } \psi' \text{ é um operador morfológico seqüencial;} \\ \text{ou} \\ (\psi_1 \vee \psi_2) & \text{onde } \psi_1 \text{ e } \psi_2 \text{ são operadores morfológicos formados por uniões ou} \\ & \text{interseções de operadores morfológicos seqüenciais.} \\ \text{ou} \\ (\psi_1 \wedge \psi_2) & \text{onde } \psi_1 \text{ e } \psi_2 \text{ são operadores morfológicos formados por uniões ou} \\ & \text{interseções de operadores morfológicos seqüenciais.} \end{cases}$$

Podemos reduzir a quantidade de “operações desnecessárias” efetuadas por um operador morfológico invariante por translação e isotônico decompondo-o e reescrevendo a decomposição obtida na forma de uniões ou interseções de operadores morfológicos seqüenciais, como mostra o exemplo 9.22 a seguir.

Exemplo 9.22. $\delta_L \circ (\delta_O \vee \delta_O) = \underbrace{(\delta_L \circ \delta_O)}_{\iota} \vee \underbrace{(\delta_L \circ \delta_O)}_{\iota} = \underbrace{\iota \vee \iota}_{\iota}$

Veremos que há um sistema de reescrita R_1 noetheriano que nos permite reescrever qualquer operador morfológico decomposto num operador morfológico formado por uniões ou interseções de operadores morfológicos seqüenciais, irreduzível em relação a R_1 . Este sistema de reescrita R_1 é obtido a partir de um conjunto de identidades da morfologia matemática que exibimos a seguir.

Sejam ψ_1, ψ_2, ψ_3 operadores morfológicos invariantes por translação e isotônicos.

A composição de operadores morfológicos é associativa, isto é,

$$(\psi_1 \circ \psi_2) \circ \psi_3 = \psi_1 \circ (\psi_2 \circ \psi_3) \quad (I_1)$$

Pela proposição 8.21,

$$(\psi_1 \vee \psi_2) \circ \psi_3 = (\psi_1 \circ \psi_3) \vee (\psi_2 \circ \psi_3) \quad (I_2)$$

$$(\psi_1 \wedge \psi_2) \circ \psi_3 = (\psi_1 \circ \psi_3) \wedge (\psi_2 \circ \psi_3) \quad (I_3)$$

Pelo corolário 8.9,

$$(\psi_1(X) \cup \psi_2(X)) + u = (\psi_1(X) + u) \cup (\psi_2(X) + u)$$

$$(\psi_1(X) \cap \psi_2(X)) + u = (\psi_1(X) + u) \cap (\psi_2(X) + u)$$

ou, equivalentemente,

$$\tau_u \circ (\psi_1 \vee \psi_2) = (\tau_u \circ \psi_1) \vee (\tau_u \circ \psi_2)$$

$$\tau_u \circ (\psi_1 \wedge \psi_2) = (\tau_u \circ \psi_1) \wedge (\tau_u \circ \psi_2)$$

e, portanto, por particularização do corolário 8.9,

$$\begin{aligned}
\tau_{(0,1)} \circ (\psi_1 \vee \psi_2) &= (\tau_{(0,1)} \circ \psi_1) \vee (\tau_{(0,1)} \circ \psi_2) \\
\tau_{(0,-1)} \circ (\psi_1 \vee \psi_2) &= (\tau_{(0,-1)} \circ \psi_1) \vee (\tau_{(0,-1)} \circ \psi_2) \\
\tau_{(-1,0)} \circ (\psi_1 \vee \psi_2) &= (\tau_{(-1,0)} \circ \psi_1) \vee (\tau_{(-1,0)} \circ \psi_2) \\
\tau_{(1,0)} \circ (\psi_1 \vee \psi_2) &= (\tau_{(1,0)} \circ \psi_1) \vee (\tau_{(1,0)} \circ \psi_2) \\
\tau_{(0,0)} \circ (\psi_1 \vee \psi_2) &= (\tau_{(0,0)} \circ \psi_1) \vee (\tau_{(0,0)} \circ \psi_2) \\
\tau_{(0,1)} \circ (\psi_1 \wedge \psi_2) &= (\tau_{(0,1)} \circ \psi_1) \wedge (\tau_{(0,1)} \circ \psi_2) \\
\tau_{(0,-1)} \circ (\psi_1 \wedge \psi_2) &= (\tau_{(0,-1)} \circ \psi_1) \wedge (\tau_{(0,-1)} \circ \psi_2) \\
\tau_{(-1,0)} \circ (\psi_1 \wedge \psi_2) &= (\tau_{(-1,0)} \circ \psi_1) \wedge (\tau_{(-1,0)} \circ \psi_2) \\
\tau_{(1,0)} \circ (\psi_1 \wedge \psi_2) &= (\tau_{(1,0)} \circ \psi_1) \wedge (\tau_{(1,0)} \circ \psi_2) \\
\tau_{(0,0)} \circ (\psi_1 \wedge \psi_2) &= (\tau_{(0,0)} \circ \psi_1) \wedge (\tau_{(0,0)} \circ \psi_2)
\end{aligned}$$

Mas, como consequências imediatas da caracterização das dilatações invariantes por translação (proposição 8.19) e definição de translação por um vetor (definição 8.7), temos $\delta_L = \tau_{(0,1)}$, $\delta_O = \tau_{(0,-1)}$, $\delta_N = \tau_{(-1,0)}$, $\delta_S = \tau_{(1,0)}$ e $\iota = \tau_{(0,0)}$ e, portanto, podemos representar as identidades acima, respectivamente, de modo equivalente, como segue:

$$\delta_L \circ (\psi_1 \vee \psi_2) = (\delta_L \circ \psi_1) \vee (\delta_L \circ \psi_2) \quad (I_4)$$

$$\delta_O \circ (\psi_1 \vee \psi_2) = (\delta_O \circ \psi_1) \vee (\delta_O \circ \psi_2) \quad (I_5)$$

$$\delta_N \circ (\psi_1 \vee \psi_2) = (\delta_N \circ \psi_1) \vee (\delta_N \circ \psi_2) \quad (I_6)$$

$$\delta_S \circ (\psi_1 \vee \psi_2) = (\delta_S \circ \psi_1) \vee (\delta_S \circ \psi_2) \quad (I_7)$$

$$\iota \circ (\psi_1 \vee \psi_2) = (\iota \circ \psi_1) \vee (\iota \circ \psi_2) \quad (I_8)$$

$$\delta_L \circ (\psi_1 \wedge \psi_2) = (\delta_L \circ \psi_1) \wedge (\delta_L \circ \psi_2) \quad (I_9)$$

$$\delta_O \circ (\psi_1 \wedge \psi_2) = (\delta_O \circ \psi_1) \wedge (\delta_O \circ \psi_2) \quad (I_{10})$$

$$\delta_N \circ (\psi_1 \wedge \psi_2) = (\delta_N \circ \psi_1) \wedge (\delta_N \circ \psi_2) \quad (I_{11})$$

$$\delta_S \circ (\psi_1 \wedge \psi_2) = (\delta_S \circ \psi_1) \wedge (\delta_S \circ \psi_2) \quad (I_{12})$$

$$\iota \circ (\psi_1 \wedge \psi_2) = (\iota \circ \psi_1) \wedge (\iota \circ \psi_2) \quad (I_{13})$$

Sejam l_j e r_j os lados direito e esquerdo, respectivamente, da identidade I_j , com $1 \leq j \leq 13$.

Consideremos o seguinte sistema de reescrita $R_1 = \bigcup_j \{l_j \rightarrow r_j\}$, com $1 \leq j \leq 13$.

Trivialmente, a morfologia matemática é um modelo para R_1 , uma vez que as regras de reescrita de R_1 foram obtidas dando orientações ao conjunto $\{I_1, \dots, I_{13}\}$ de identidades da morfologia matemática e, portanto, como consequência do Teorema de Birkhoff 3.20 (a parte deste teorema correspondente à correção), garante-se que qualquer R_1 -prova expressa uma verdade para a morfologia matemática.

O que queremos mostrar a seguir é que, de fato, qualquer operador morfológico decom-

posto pode ser reescrito, através de uma quantidade finita de aplicações de regras de reescrita de R_1 (isto é, uma R_1 -prova), a um termo irreduzível formado por uniões e interseções de operadores morfológicos seqüenciais.

Notação: Daqui por diante, por simplicidade na escrita, os símbolos $\delta_L, \delta_O, \delta_N$ e δ_S serão representados, respectivamente, pelas letras d (translação da imagem à direita), e (translação da imagem à esquerda), c ($\dots$ cima), b ($\dots$ baixo). Também, daqui por diante, ao invés de denotarmos genericamente um operador morfológico invariante por translação e isotônico através do uso de uma letra grega minúscula como ψ , ou sua composição com índices inteiros estritamente positivos, como $\psi_1, \psi_2, \dots$, passamos a representar genericamente um tal operador usando as letras minúsculas do final do alfabeto arábico, como x, y, z e suas composições com índices inteiros estritamente positivos, como $x_1, x_2, \dots$.

Sendo assim, R_1 fica determinado como segue:

$$R_1 = \{ \begin{array}{ll} r_1 : & (x \circ y) \circ z \rightarrow x \circ (y \circ z), \\ r_2 : & (x \vee y) \circ z \rightarrow (x \circ z) \vee (y \circ z), \\ r_3 : & (x \wedge y) \circ z \rightarrow (x \circ z) \wedge (y \circ z), \\ r_4 : & d \circ (x \vee y) \rightarrow (d \circ x) \vee (d \circ y), \\ r_5 : & e \circ (x \vee y) \rightarrow (e \circ x) \vee (e \circ y), \\ r_6 : & c \circ (x \vee y) \rightarrow (c \circ x) \vee (c \circ y), \\ r_7 : & b \circ (x \vee y) \rightarrow (b \circ x) \vee (b \circ y), \\ r_8 : & \iota \circ (x \vee y) \rightarrow (\iota \circ x) \vee (\iota \circ y), \\ r_9 : & d \circ (x \wedge y) \rightarrow (d \circ x) \wedge (d \circ y), \\ r_{10} : & e \circ (x \wedge y) \rightarrow (e \circ x) \wedge (e \circ y), \\ r_{11} : & c \circ (x \wedge y) \rightarrow (c \circ x) \wedge (c \circ y), \\ r_{12} : & b \circ (x \wedge y) \rightarrow (b \circ x) \wedge (b \circ y), \\ r_{13} : & \iota \circ (x \wedge y) \rightarrow (\iota \circ x) \wedge (\iota \circ y) \end{array} \}$$

O lema 9.23, a seguir, nos mostra que R_1 não admite nenhuma seqüência de reescrita que seja infinita.

Lema 9.23. *O sistema de reescrita R_1 , especificado acima, é noetheriano.*

Demonstração: Vamos mostrar que, de fato, R_1 é noetheriano; para isto especificamos uma função de terminação (veja definição 5.32) baseada em uma interpretação polinomial.

O alfabeto Σ dos termos de R_1 é dado por $\Sigma = \{\circ, \vee, \wedge, d, e, c, b, \iota\}$, onde $\text{Arid}(\circ) = \text{Arid}(\vee) = \text{Arid}(\wedge) = 2$ e $\text{Arid}(d) = \text{Arid}(e) = \text{Arid}(c) = \text{Arid}(b) = \text{Arid}(\iota) = 0$.

Considere a seguinte interpretação polinomial para a função de terminação τ , composta pelos seguintes mapeamentos:

$$\begin{aligned}
\circ_\tau &: \mathbb{N}^2 \mapsto \mathbb{N} \\
\vee_\tau &: \mathbb{N}^2 \mapsto \mathbb{N} \\
\wedge_\tau &: \mathbb{N}^2 \mapsto \mathbb{N} \\
d &: \mapsto \mathbb{N} \\
e &: \mapsto \mathbb{N} \\
c &: \mapsto \mathbb{N} \\
b &: \mapsto \mathbb{N} \\
\iota &: \mapsto \mathbb{N} \\
x &: \mapsto \mathbb{N}, \quad \forall x \in \mathcal{V}
\end{aligned}$$

e especificados, respectivamente, por:

$$\begin{aligned}
\circ_\tau(x, y) &= x_\tau \cdot y_\tau \\
\vee_\tau(x, y) &= 2 \cdot x_\tau + y_\tau + 1 \\
\wedge_\tau(x, y) &= 2 \cdot x_\tau + y_\tau + 1 \\
d_\tau &= 2 \\
e_\tau &= 2 \\
c_\tau &= 2 \\
b_\tau &= 2 \\
\iota_\tau &= 2 \\
x_\tau &= 2, \quad \forall x \in \mathcal{V}
\end{aligned}$$

Observe que se $\tau(t) > \tau(s)$ então $\tau(f(\cdots t \cdots)) > \tau(f(\cdots s \cdots)), \forall t, s \in \mathcal{T}_{R_1}$ (monotonicidade), pois:

- i) o valor de um termo t segundo a interpretação τ é calculado indutivamente a partir do produto $(\cdot)$ dos valores de seus subtermos (caso $t / < > = \circ$) ou é calculado a partir da soma $(+)$ ponderada positiva do produto dos valores de seus subtermos (caso $t / < > \in \{\vee, \wedge\}$);
- ii) $\tau(d) = d_\tau = 2; \tau(e) = e_\tau = 2; \tau(c) = c_\tau = 2; \tau(b) = b_\tau = 2; \tau(\iota) = \iota_\tau = 2;$ e $\tau(x) = x_\tau = 2, \forall x \in \mathcal{V}$, logo, para qualquer termo t em $\mathcal{T}_{R_1}$, $\tau(t)$ é estritamente positivo ($\tau(t) \geq 2$).

Ainda, a relação “maior que” $(>)$ usual dos naturais é noetheriana, logo τ é, de fato, uma função de terminação.

Considere a ordem parcial $\succ$ noetheriana sobre $\mathbb{N}$ especificada como segue:

$$\tau(t) \succ \tau(s) \text{ se e somente se } \begin{cases} \tau(t) > \tau(s) \\ \text{ou} \\ \tau(t) = \tau(s) \text{ e } \tau(t / \langle 1 \rangle) > \tau(s / \langle 1 \rangle) \end{cases}$$

Precisamos mostrar que $\succ$ assim especificada é, de fato, noetheriana.

Note que se $\tau(t_1) \succ \tau(t_2)$ e $\tau(t_1) = \tau(t_2)$ então $\tau(t_1 / \langle 1 \rangle) > \tau(t_2 / \langle 1 \rangle)$. Então não há seqüência infinita de termos $t_1, t_2, \dots$ tais que $\tau(t_1) \succ \tau(t_2) \succ \dots$ com $\tau(t_1) = \tau(t_2) = \dots$ pois, caso contrário, haveria uma seqüência infinita $\tau(t_1 / \langle 1 \rangle) > \tau(t_2 / \langle 1 \rangle) \dots$ o que é um absurdo pois vimos que $\tau(t) \geq 2$ para todo termo t em $\mathcal{T}_{R_1}$ e a relação “maior que” ($>$) usual dos naturais é noetheriana. Logo, acabamos de mostrar que $\succ$ é noetheriana.

Portanto, pelo teorema 5.33, para que R_1 seja noetheriano basta que $\tau(l\sigma) \succ \tau(r\sigma)$ para cada regra $l \rightarrow r$ em R_1 e qualquer substituição σ para as variáveis da regra $l \rightarrow r$. É isto que vamos mostrar a seguir:

- Para a regra r_1 de R_1 e para qualquer substituição σ temos:

$$\begin{aligned} \tau(((x \circ y) \circ z)\sigma) &= \tau((x\sigma \circ y\sigma) \circ z\sigma) \\ &\stackrel{\text{notaç. pré-fixa}}{=} \tau(\circ(\circ(x\sigma, y\sigma), z\sigma)) \\ &= \circ_\tau(\tau(\circ(x\sigma, y\sigma)), \tau(z\sigma)) \\ &= \tau(\circ(x\sigma, y\sigma)) \cdot \tau(z\sigma) \\ &= \circ_\tau(\tau(x\sigma), \tau(y\sigma)) \cdot \tau(z\sigma) \\ &= (\tau(x\sigma) \cdot \tau(y\sigma)) \cdot \tau(z\sigma) \\ &= \tau(x\sigma) \cdot \tau(y\sigma) \cdot \tau(z\sigma). \end{aligned}$$

$$\begin{aligned} \tau((x \circ (y \circ z))\sigma) &= \tau(x\sigma \circ (y\sigma \circ z\sigma)) \\ &\stackrel{\text{notaç. pré-fixa}}{=} \tau(\circ(x\sigma, \circ(y\sigma, z\sigma))) \\ &= \circ_\tau(\tau(x\sigma), \tau(\circ(y\sigma, z\sigma))) \\ &= \tau(x\sigma) \cdot \tau(\circ(y\sigma, z\sigma)) \\ &= \tau(x\sigma) \cdot \circ_\tau(\tau(y\sigma), \tau(z\sigma)) \\ &= \tau(x\sigma) \cdot (\tau(y\sigma) \cdot \tau(z\sigma)) \\ &= \tau(x\sigma) \cdot \tau(y\sigma) \cdot \tau(z\sigma). \end{aligned}$$

Isto é, $\tau(((x \circ y) \circ z)\sigma) = \tau((x \circ (y \circ z))\sigma)$, precisamos então verificar para a regra r_1 se $\tau(((x \circ y) \circ z)\sigma / \langle 1 \rangle) > \tau((x \circ (y \circ z))\sigma / \langle 1 \rangle)$. ou seja, queremos mostrar que

$\tau((x \circ y)\sigma) > \tau(x\sigma)$. Mas, $\tau((x \circ y)\sigma) = \tau(x\sigma \circ y\sigma) = \tau(x\sigma) \cdot \tau(y\sigma) > \tau(x\sigma)$ pois $\tau(t) \geq 2$ para qualquer termo t em $\mathcal{T}_{R_1}$. Logo, acabamos de mostrar que para a regra r_1 e para qualquer substituição σ ,

$$\tau(((x \circ y) \circ z)\sigma) \succ \tau((x \circ (y \circ z))\sigma)$$

• Para a regra r_2 de R_1 e para qualquer substituição σ temos:

$$\begin{aligned} \tau(((x \vee y) \circ z)\sigma) &= \tau((x\sigma \vee y\sigma) \circ z\sigma) \\ &\stackrel{\text{notaç. pré-fixa}}{=} \tau(\circ(\vee(x\sigma, y\sigma), z\sigma)) \\ &= \circ_\tau(\tau(\vee(x\sigma, y\sigma)), \tau(z\sigma)) \\ &= \tau(\vee(x\sigma, y\sigma)) \cdot \tau(z\sigma) \\ &= \vee_\tau(\tau(x\sigma), \tau(y\sigma)) \cdot \tau(z\sigma) \\ &= (2 \cdot \tau(x\sigma) + \tau(y\sigma) + 1) \cdot \tau(z\sigma) \\ &= 2 \cdot \tau(x\sigma) \cdot \tau(z\sigma) + \tau(y\sigma) \cdot \tau(z\sigma) + \tau(z\sigma). \end{aligned}$$

$$\begin{aligned} \tau(((x \circ z) \vee (y \circ z))\sigma) &= \tau((x\sigma \circ z\sigma) \vee (y\sigma \circ z\sigma)) \\ &\stackrel{\text{notaç. pré-fixa}}{=} \tau(\vee(\circ(x\sigma, z\sigma), \circ(y\sigma, z\sigma))) \\ &= \vee_\tau(\tau(\circ(x\sigma, z\sigma)), \tau(\circ(y\sigma, z\sigma))) \\ &= 2 \cdot \tau(\circ(x\sigma, z\sigma)) + \tau(\circ(y\sigma, z\sigma)) + 1 \\ &= 2 \cdot (\circ_\tau(\tau(x\sigma), \tau(z\sigma))) + \circ_\tau(\tau(y\sigma), \tau(z\sigma)) + 1 \\ &= 2 \cdot (\tau(x\sigma) \cdot \tau(z\sigma)) + \tau(y\sigma) \cdot \tau(z\sigma) + 1 \\ &= 2 \cdot \tau(x\sigma) \cdot \tau(z\sigma) + \tau(y\sigma) \cdot \tau(z\sigma) + 1. \end{aligned}$$

Logo, $\tau(((x \vee y) \circ z)\sigma) - \tau(((x \circ z) \vee (y \circ z))\sigma) = 2 \cdot \tau(x\sigma) \cdot \tau(z\sigma) + \tau(y\sigma) \cdot \tau(z\sigma) + \tau(z\sigma) - 2 \cdot \tau(x\sigma) \cdot \tau(z\sigma) - \tau(y\sigma) \cdot \tau(z\sigma) - 1 = \tau(z\sigma) - 1 > 0$ pois $\tau(z\sigma) \geq 2$. Isto é, $\tau(((x \vee y) \circ z)\sigma) > \tau(((x \circ z) \vee (y \circ z))\sigma)$ e portanto acabamos de mostrar que, para a regra r_2 e para qualquer substituição σ ,

$$\tau(((x \vee y) \circ z)\sigma) \succ \tau(((x \circ z) \vee (y \circ z))\sigma)$$

• Para a regra r_3 de R_1 , a análise é similar a que acabamos de fazer para a regra r_2 .

• Para a regra r_4 de R_1 e para qualquer substituição σ temos:

$$\begin{aligned}
\tau((d \circ (x \vee y))\sigma) &= \tau(d\sigma \circ (x\sigma \vee y\sigma)) \\
&\stackrel{\text{notaç. pré-fixa}}{=} \tau(\circ(d\sigma, \vee(x\sigma, y\sigma))) \\
&= \circ_\tau(\tau(d\sigma), \tau(\vee(x\sigma, y\sigma))) \\
&= \tau(d\sigma) \cdot \tau(\vee(x\sigma, y\sigma)) \\
&= \tau(d\sigma) \cdot \vee_\tau(\tau(x\sigma), \tau(y\sigma)) \\
&= \tau(d\sigma) \cdot (2 \cdot \tau(x\sigma) + \tau(y\sigma) + 1) \\
&= 2 \cdot \tau(d\sigma) \cdot \tau(x\sigma) + \tau(d\sigma) \cdot \tau(y\sigma) + \tau(d\sigma) \\
&= 4 \cdot \tau(x\sigma) + 2 \cdot \tau(y\sigma) + 2.
\end{aligned}$$

$$\begin{aligned}
\tau(((d \circ x) \vee (d \circ y))\sigma) &= \tau((d\sigma \circ x\sigma) \vee (d\sigma \circ y\sigma)) \\
&\stackrel{\text{notaç. pré-fixa}}{=} \tau(\vee(\circ(d\sigma, x\sigma), \circ(d\sigma, y\sigma))) \\
&= \vee_\tau(\tau(\circ(d\sigma, x\sigma)), \tau(\circ(d\sigma, y\sigma))) \\
&= 2 \cdot \tau(\circ(d\sigma, x\sigma)) + \tau(\circ(d\sigma, y\sigma)) + 1 \\
&= 2 \cdot \circ_\tau(\tau(d\sigma), \tau(x\sigma)) + \circ_\tau(\tau(d\sigma), \tau(y\sigma)) + 1 \\
&= 2 \cdot \tau(d\sigma) \cdot \tau(x\sigma) + \tau(d\sigma) \cdot \tau(y\sigma) + 1 \\
&= 4 \cdot \tau(x\sigma) + 2 \cdot \tau(y\sigma) + 1.
\end{aligned}$$

Logo $\tau((d \circ (x \vee y))\sigma) - \tau(((d \circ x) \vee (d \circ y))\sigma) = 1$. Sendo assim acabamos de mostrar que para a regra r_4 e para qualquer substituição σ ,

$$\tau((d \circ (x \vee y))\sigma) \succ \tau(((d \circ x) \vee (d \circ y))\sigma)$$

- Para as demais regras, $r_5, \dots, r_{13}$, a análise é similar a que acabamos de fazer para r_4 .

Portanto, pelo teorema 5.33, R_1 é noetheriano.

◆

O lema 9.24, a seguir, nos mostra que, se ψ é um operador morfológico decomposto, então $R_1(\psi)$ é um operador morfológico formado por uniões e interseções de operadores morfológicos seqüenciais.

Lema 9.24. *Seja ψ um operador morfológico decomposto. Então, $R_1(\psi)$ é um operador morfológico formado por uniões e interseções de operadores morfológicos seqüenciais.*

Demonstração: Seja $\psi' = R_1(\psi)$. Queremos mostrar que ψ' é formado por uniões ou interseções de operadores seqüenciais. Suponha, por absurdo, que ψ' não seja formado por uniões ou

interseções de operadores sequenciais. Logo, existe algum subtermo de ψ' , digamos ψ'_1 , tal que ψ'_1 possui uma das seguintes formas: $(\psi'_2 \vee \psi'_3) \circ \psi'_4$ ou $(\psi'_2 \wedge \psi'_3) \circ \psi'_4$ ou $\psi'_2 \circ (\psi'_3 \vee \psi'_4)$ ou $\psi'_2 \circ (\psi'_3 \wedge \psi'_4)$.

Caso 1: $\psi'_1 = (\psi'_2 \vee \psi'_3) \circ \psi'_4$.

Neste caso ψ'_1 pode ser reescrito através da regra r_2 de R_1 e, conseqüentemente, ψ' não seria irreduzível pois ψ'_1 é subtermo de ψ' , o que é um absurdo pois, pelo lema 9.23, ψ' é irreduzível.

Caso 2: $\psi'_1 = (\psi'_2 \wedge \psi'_3) \circ \psi'_4$.

Neste caso, ψ'_1 pode ser reescrito através da regra r_3 de R_1 e teríamos que ψ' seria redutível, e pelo lema 9.23, isto seria um absurdo.

Caso 3: $\psi'_1 = \psi'_2 \circ (\psi'_3 \vee \psi'_4)$.

Por hipótese, ψ é um operador decomposto, conseqüentemente ψ' é um operador decomposto e, portanto, podemos ter apenas um dos seguintes subcasos:

Caso 3.1: ψ'_2 é um operador sequencial.

$\psi'_2 = (\gamma_1 \circ \dots \circ \gamma_n)$, onde $\gamma_i \in \{\delta_L, \delta_O, \delta_N, \delta_S, \iota\}$, com $1 \leq i \leq n$. Neste caso, ψ'_1 pode ser reescrito através da regra r_1 de R_1 que expressa a associatividade da composição de operadores morfológicos:

$$((\gamma_1 \circ \dots \circ \gamma_n) \circ (\psi'_3 \vee \psi'_4)) \xrightarrow{r_1} (\gamma_1 \circ \dots \circ (\gamma_n \circ (\psi'_3 \vee \psi'_4)))$$

e teríamos que ψ' seria redutível, mas, pelo lema 9.23, isto é um absurdo.

Caso 3.2: $\psi'_2 = (\psi'_5 \vee \psi'_6)$

Seja $\psi'_7 = \psi'_5 \vee \psi'_6$. Logo, $\psi'_1 = (\psi'_5 \vee \psi'_6) \circ \psi'_4$ e este é justamente o Caso 1 já analisado.

Caso 3.3: $\psi'_2 = (\psi'_5 \wedge \psi'_6)$.

Seja $\psi'_7 = \psi'_5 \wedge \psi'_6$. Logo, $\psi'_1 = (\psi'_5 \wedge \psi'_6) \circ \psi'_4$ e este é justamente o Caso 2 já analisado.

Portanto, acabamos de mostrar que, de fato, ψ' é formado por uniões ou interseções de operadores morfológicos sequenciais.

Exemplo 9.25. Seja $\psi = \delta \begin{bmatrix} 010 \\ 000 \\ 010 \end{bmatrix} \circ \epsilon \begin{bmatrix} 000 \\ 010 \\ 100 \end{bmatrix}$, logo $\mathcal{A}(\psi) = ((\delta_N \circ \delta_N) \vee \iota) \circ (\iota \wedge (\delta_N \circ \delta_L))$.

A seguir apresentamos um operador morfológico equivalente a ψ formado apenas por uniões ou interseções de operadores morfológicos seqüenciais. Como vimos, tal operador é obtido calculando-se a forma normal (irredutível), segundo o sistema de reescrita R_1 , da forma decomposta de ψ . Em outras palavras, precisamos calcular $R_1(\mathcal{A}(\psi))$.

Apresentamos a seguir uma possível seqüência de reescrita do termo $\mathcal{A}(\psi)$ para uma sua respectiva forma normal.

$$\begin{aligned}
 ((\delta_N \circ \delta_N) \vee \iota) \circ (\iota \wedge (\delta_N \circ \delta_L)) &\stackrel{\text{notação}}{=} ((c \circ c) \vee \iota) \circ (\iota \wedge (c \circ d)) \\
 &\xrightarrow[\tau_2]{[p_1]} ((c \circ c) \circ (\iota \wedge (c \circ d))) \vee (\iota \circ (\iota \wedge (c \circ d))) \\
 &\hspace{15em} (p_1 = \langle \rangle) \\
 &\xrightarrow[\tau_1]{[p_2]} (c \circ (c \circ (\iota \wedge (c \circ d)))) \vee (\iota \circ (\iota \wedge (c \circ d))) \\
 &\hspace{15em} (p_2 = \langle 1 \rangle) \\
 &\xrightarrow[\tau_{11}]{[p_3]} (c \circ ((c \circ \iota) \wedge (c \circ (c \circ d)))) \vee (\iota \circ (\iota \wedge (c \circ d))) \\
 &\hspace{15em} (p_3 = \langle 1, 2 \rangle) \\
 &\xrightarrow[\tau_{13}]{[p_4]} (c \circ ((c \circ \iota) \wedge (c \circ (c \circ d)))) \vee ((\iota \circ \iota) \wedge (\iota \circ (c \circ d))) \\
 &\hspace{15em} (p_4 = \langle 2 \rangle) \\
 &\xrightarrow[\tau_{11}]{[p_5]} ((c \circ (c \circ \iota)) \wedge (c \circ (c \circ (c \circ d)))) \vee ((\iota \circ \iota) \wedge (\iota \circ (c \circ d))) \\
 &\hspace{15em} (p_5 = \langle 1 \rangle)
 \end{aligned}$$

Note que o último termo obtido pela seqüência apresentada é irredutível, isto é,

$$\begin{aligned}
 R_1(\mathcal{A}(\psi)) &= ((c \circ (c \circ \iota)) \wedge (c \circ (c \circ (c \circ d)))) \vee ((\iota \circ \iota) \wedge (\iota \circ (c \circ d))) \\
 &\stackrel{\text{notação}}{=} ((\delta_N \circ (\delta_N \circ \iota)) \wedge (\delta_N \circ (\delta_N \circ (\delta_N \circ \delta_L)))) \vee ((\iota \circ \iota) \wedge (\iota \circ (\delta_N \circ \delta_L))) \\
 &\stackrel{\text{notação}}{=} ((\delta_N \circ \delta_N \circ \iota) \wedge (\delta_N \circ \delta_N \circ \delta_N \circ \delta_L)) \vee ((\iota \circ \iota) \wedge (\iota \circ \delta_N \circ \delta_L)).
 \end{aligned}$$

O diagrama 9.4 a seguir mostra a representação na forma de árvore do termo irredutível obtido.

Exibimos, a seguir, um sistema de reescrita R_2 que “simplifica” operadores morfológicos seqüenciais. Mostramos também que os termos irredutíveis obtidos por R_2 são ótimos em relação à quantidade de translações que o operador realiza, isto é, se ψ e ψ' são operadores seqüenciais, tal que $\psi = \psi'$, então a quantidade de translações que $R_2(\psi)$ realiza em qualquer imagem é menor ou igual à quantidade de translações que ψ' realiza que qualquer imagem.

Tal sistema de reescrita R_2 é obtido a partir de um conjunto de identidades da morfologia matemática que exibimos a seguir, e que são conseqüências imediatas da proposição 8.22, da definição 8.13 (adição de Minkowski) e do operador ι (operador identidade sobre $\mathcal{P}(E)$).

Figura 9.4: Diagrama de árvore do termo irreduzível obtido pelo exemplo 9.25.

$$\delta_S \circ \delta_N = \delta_{S \oplus N} = \delta_{\{o\}} = \iota \quad (I_{14})$$

$$\delta_L \circ \delta_O = \delta_{L \oplus O} = \delta_{\{o\}} = \iota \quad (I_{15})$$

$$\iota \circ \delta_S = \delta_S \quad (I_{16})$$

$$\iota \circ \delta_N = \delta_N \quad (I_{17})$$

$$\iota \circ \delta_L = \delta_L \quad (I_{18})$$

$$\iota \circ \delta_O = \delta_O \quad (I_{19})$$

Note que,

$$\begin{aligned} \delta_S \circ \delta_N &= \delta_{S \oplus N} && \text{(proposição 8.22(i))} \\ &= \delta_{N \oplus S} && \text{(comutatividade de } \oplus) \\ &= \delta_N \circ \delta_S && \text{(proposição 8.22(i))} \end{aligned}$$

Isto é, $\delta_S \circ \delta_N = \delta_N \circ \delta_S$. Analogamente $\delta_L \circ \delta_O = \delta_O \circ \delta_L$, $\iota \circ \delta_S = \delta_S \circ \iota$, $\iota \circ \delta_N = \delta_N \circ \iota$, $\iota \circ \delta_L = \delta_L \circ \iota$, $\iota \circ \delta_O = \delta_O \circ \iota$.

Acabamos de mostrar que $\psi_1 \circ \psi_2 = \psi_2 \circ \psi_1$, para quaisquer ψ_1, ψ_2 em $\{\delta_S, \delta_N, \delta_L, \delta_O, \iota\}$.

A composição de operadores é associativa (proposição 2.9).

Como consequência imediata das observações feitas nos dois parágrafos anteriores, conclui-se que $\psi_1 \circ \psi_2 = \psi_2 \circ \psi_1$, para quaisquer ψ_1, ψ_2 operadores morfológicos seqüenciais.

Exemplo 9.26. *Sejam ψ_1, ψ_2, ψ_3 operadores morfológicos seqüenciais.*

Sendo assim, $\psi_1 \circ \psi_2 \circ \psi_3 = \psi_1 \circ \psi_3 \circ \psi_2 = \psi_2 \circ \psi_1 \circ \psi_3 = \psi_2 \circ \psi_3 \circ \psi_1 = \psi_3 \circ \psi_1 \circ \psi_2 = \psi_3 \circ \psi_2 \circ \psi_1$.

Sejam l_j e r_j os lados direito e esquerdo, respectivamente, da identidade I_j , com $14 \leq j \leq 19$.

Consideremos o seguinte sistema de reescrita $R_2 = \bigcup_j \{l_j \rightarrow r_j\}$, com $1 \leq j \leq 13$.

Trivialmente a morfologia matemática é um modelo para R_2 , uma vez que as regras de reescrita de R_2 foram obtidas dando orientações ao conjunto $\{I_{14}, \dots, I_{19}\}$ de identidades da morfologia matemática.

O sistema de reescrita de termos R_2 é especificado como segue, onde $\circ$ é um operador- AC .

$$R_2 = \{ \begin{array}{ll} r_{14} : & b \circ c \rightarrow \iota, \\ r_{15} : & d \circ e \rightarrow \iota, \\ r_{16} : & \iota \circ b \rightarrow b, \\ r_{17} : & \iota \circ c \rightarrow c, \\ r_{18} : & \iota \circ d \rightarrow d, \\ r_{19} : & \iota \circ e \rightarrow e \end{array} \}$$

Lema 9.27. *O sistema de reescrita R_2 , especificado acima, é completo módulo AC .*

Demonstração: Seja T um conjunto de equações. Para que possamos verificar se um dado sistema de reescrita é completo módulo T , segundo o método de completção descrito no capítulo 7, é necessário que o sistema de reescrita considerado seja compatível com T .

No nosso caso, temos,

$$T = \{ \begin{array}{ll} x \circ y & = y \circ x, \\ x \circ (y \circ z) & = (x \circ y) \circ z \end{array} \}$$

Isto é, T é uma teoria- AC e portanto, pelo teorema 7.42, $R_{2_T}^e$ é compatível com T .

Mas $R_{2_T}^e$ é, por definição, uma extensão de R e, conseqüentemente, pelo teorema 7.16,

$$\overrightarrow{[R_{2_T}^e]} = \overrightarrow{[R_2]}.$$

Logo, se mostrarmos que $R_{2_T}^e$ é completo módulo T teremos mostrado também que R_2 é completo módulo T .

$$R_{2_T}^e \stackrel{\text{def.}}{=} \{ \begin{array}{ll} r_{14} : & b \circ c \rightarrow \iota, \\ r_{15} : & d \circ e \rightarrow \iota, \\ r_{16} : & \iota \circ b \rightarrow b, \\ r_{17} : & \iota \circ c \rightarrow c, \\ r_{18} : & \iota \circ d \rightarrow d, \\ r_{19} : & \iota \circ e \rightarrow e, \\ r_{14}^e : & x \circ (b \circ c) \rightarrow x \circ \iota, \\ r_{15}^e : & x \circ (d \circ e) \rightarrow x \circ \iota, \\ r_{16}^e : & x \circ (\iota \circ b) \rightarrow x \circ b, \\ r_{17}^e : & x \circ (\iota \circ c) \rightarrow x \circ c, \\ r_{18}^e : & x \circ (\iota \circ d) \rightarrow x \circ d, \\ r_{19}^e : & x \circ (\iota \circ e) \rightarrow x \circ e \end{array} \}$$

É fácil ver que R_{2T}^e é noetheriano módulo T . Para isto, basta observar que, para cada regra $l \rightarrow r$ em R_{2T}^e , o termo r do lado direito da regra possui uma quantidade estritamente menor de símbolos funcionais.

Como R_{2T}^e é noetheriano módulo T e compatível com T então, pelo teorema 7.32, basta mostrarmos que todo par crítico de R_{2T}^e módulo T é convergente. Sendo assim, é suficiente mostrarmos que, para qualquer par $l_1 \rightarrow r_1, l_2 \rightarrow r_2$ de regras de R_{2T}^e , todos os pares críticos de R_{2T}^e módulo T , obtidos pelas superposições dos pares de termos $\langle l_1, l_2 \rangle$ convergem ou, equivalentemente, basta mostrarmos que para qualquer par $l_1 \rightarrow r_1, l_2 \rightarrow r_2$ de regras de R_2 , todos os pares críticos de R_2 módulo T obtidos pelas superposições dos pares de termos $\langle l_1, l_2 \rangle, \langle l_1, l_2^e \rangle, \langle l_1^e, l_2 \rangle, \langle l_1^e, l_2^e \rangle$ convergem, onde l_1^e e l_2^e são, respectivamente, as extensões simples de l_1 e l_2 .

Caso 1: Superposições $\langle l_1, l_2 \rangle$.

É fácil ver que R_2 não admite superposições entre os lados esquerdos de suas regras. Como as regras em R_2 são constituídas por termos fechados (básicos), então a superposição entre os lados esquerdos de cada par de regras de R_2 consiste um caso particular de superposição, neste caso, se t_1 e t_2 são termos básicos a serem superpostos, então basta verificarmos se existe subtermo de t_1 que seja igual módulo T a t_2 e se existe subtermo de t_2 que seja igual módulo T a t_1 .

Para as regras $l_1 \rightarrow r_1, l_2 \rightarrow r_2$, isto é, $b \circ c \rightarrow \iota$ e $d \circ e \rightarrow \iota$ não há como superpor $l_1 = b \circ c$ sobre algum subtermo de $l_2 = d \circ e$, pois como ambos, l_1 e l_2 , são termos básicos, então seria necessário que $l_1/n \equiv_T l_2$, com $n \in \{\langle \rangle, \langle 1 \rangle, \langle 2 \rangle\}$, o que é impossível. Analogamente, a superposição de l_1 sobre l_2 é impossível. O mesmo argumento pode ser feito para qualquer par de regras de R_2 e portanto, para quaisquer par de regras $l_1 \rightarrow r_1, l_2 \rightarrow r_2$ de R_2 , não há como superpor o par de termos $\langle l_1, l_2 \rangle$.

Caso 2: Superposições $\langle l_1, l_2^e \rangle$.

Sejam l_1 o lado esquerdo da regra de reescrita r_{14} e l_2 o lado esquerdo da regra de reescrita r_{15} . Sendo assim, temos $l_1 = b \circ c$ e $l_2 = d \circ e$. A extensão simples de l_2 é $l_2^e = x \circ (d \circ e)$, onde x é uma variável. É fácil ver que não há como superpor (de modo não trivial) o par de termos $\langle b \circ c, x \circ (d \circ e) \rangle$. Lembre-se que as superposições triviais são aquelas que ocorrem numa posição correspondente a uma variável. Neste nosso caso, se tomarmos a substituição $\sigma = \{x \leftarrow (b \circ c)\}$ teremos $l_1\sigma \equiv_T l_2^e/\langle 1 \rangle \sigma$, onde $l_1\sigma = l_1$ pois l_1 é termo básico e $l_2^e/\langle 1 \rangle \sigma = (b \circ c)$.

Como esta superposição é trivial, pois ocorre num subtermo de l_2^e que corresponde a

uma variável, a convergência do par crítico gerado a partir desta superposição fica garantida. O diagrama 9.5 ilustra esta situação.

Figura 9.5: Convergência fica garantida quando o subtermo onde ocorre a superposição corresponde a uma posição de uma variável.

De modo análogo, podemos mostrar que, para qualquer par de regras $l_1 \rightarrow r_1, l_2 \rightarrow r_2$ de R_2 , as superposições obtidas pelo par $\langle l_1, l_2^e \rangle$ não geram pares críticos divergentes.

Caso 3: Superposições $\langle l_1^e, l_2 \rangle$.

A não possibilidade de superposição dos termos l_1^e e l_2 para quaisquer par de regras $l_1 \rightarrow r_1, l_2 \rightarrow r_2$ de R_2 , decorre como consequência imediata da análise feita no Caso 2.

Caso 4: Superposições $\langle l_1^e, l_2^e \rangle$.

A única maneira de superpor l_1^e sobre l_2^e é através de uma superposição trivial, localizada num subtermo de l_2^e que corresponde a uma posição de variável e portanto, como já vimos, a convergência módulo T do par crítico gerado fica garantida.

De modo análogo, a única forma de superpor l_2^e sobre l_1^e é através de uma superposição trivial, isto é, localizada numa posição de variável de l_1^e . Podemos facilmente verificar que, para quaisquer pares de regras de R_2 , as superposições possíveis são apenas as triviais.

Como $R_{2_T}^e$ é tal que não admite geração de pares críticos módulo T divergentes, então,

pelo teorema 7.32, R_2^e é completo módulo T e portanto, pelo teorema 7.16, R_2 é completo módulo T . ◆

Logo, qualquer operador morfológico ψ da forma $\psi_1 \circ \dots \circ \psi_n$, onde ψ_i ($1 \leq i \leq n$) $\in \{\delta_L, \delta_O, \delta_N, \delta_S, \iota\}$ possui uma forma normal $R_2(\psi)$ que é única módulo T . É fácil ver que $\psi' = R_2(\psi)$ é da forma $\psi'_1 \circ \dots \circ \psi'_m$, com $m \leq n$ e ψ_i ($1 \leq i \leq m$) $\in \{\delta_L, \delta_O, \delta_N, \delta_S, \iota\}$ e ainda para quaisquer i e j , $i \neq j$, $1 \leq i, j \leq m$ temos: se $\psi'_i = \iota$ então $\psi'_j \neq \iota$; se $\psi'_i = \delta_L$ então $\psi'_j \neq \delta_O$; se $\psi'_i = \delta_N$ então $\psi'_j \neq \delta_S$.

É fácil ver que se $\psi = \psi_1 \circ \dots \circ \psi_n$, onde ψ_i ($1 \leq i \leq n$) $\in \{\delta_L, \delta_O, \delta_N, \delta_S, \iota\}$, então ψ é tal que realiza apenas uma translação na imagem fornecida ao operador ψ , isto é, $\psi(X) = X + u$, onde $u = (x, y)$, com $x = x_2 - x_1$ e $y = y_2 - y_1$, onde x_1, x_2, y_1, y_2 são, respectivamente, a quantidade de símbolos $\delta_N, \delta_S, \delta_O, \delta_L$ que ocorrem em ψ .

Podemos entender que a translação de uma imagem é efetuada através de um “caminho” que leva a imagem de uma posição à outra.

O que o sistema R_2 faz é, justamente, encontrar o caminho mais curto (4 conexo) que translada a imagem de uma posição à outra e, conseqüentemente, R_2 é um sistema de reescrita útil para a verificação de identidade entre operadores morfológicos seqüenciais.

De fato, se ψ e ψ' são operadores morfológicos seqüenciais então $\psi = \psi'$ se e somente se $R_2(\psi) \equiv_T R_2(\psi')$.

Exemplo 9.28. Considere $\psi = \delta_L \circ \delta_L \circ \delta_N \circ \delta_L \circ \delta_S \circ \delta_S \circ \delta_L \circ \delta_N \circ \delta_N$.
 $R_2(\psi) = \delta_L \circ \delta_L \circ \delta_L \circ \delta_L \circ \delta_N$.

Como conseqüência da definição 8.20 de união e interseção entre operadores sobre

$\mathcal{P}(E)$, a tripla $A = \langle \mathcal{P}(E)^{\mathcal{P}(E)}; \vee, \wedge \rangle$ forma um reticulado distributivo (por herança de reticulados distributivos dos subconjuntos) e, portanto, pela definição de reticulado distributivo, A satisfaz as seguintes identidades:

$$\begin{array}{lll}
 x \vee x & = & x \quad \forall x \in \mathcal{P}(E)^{\mathcal{P}(E)} \quad (\text{idempotência}) \\
 x \wedge x & = & x \quad \forall x \in \mathcal{P}(E)^{\mathcal{P}(E)} \quad (\text{idempotência}) \\
 x \vee y & = & y \vee x \quad \forall x, y \in \mathcal{P}(E)^{\mathcal{P}(E)} \quad (\text{comutatividade}) \\
 x \wedge y & = & y \wedge x \quad \forall x, y \in \mathcal{P}(E)^{\mathcal{P}(E)} \quad (\text{comutatividade}) \\
 (x \vee y) \vee z & = & x \vee (y \vee z) \quad \forall x, y, z \in \mathcal{P}(E)^{\mathcal{P}(E)} \quad (\text{associatividade}) \\
 (x \wedge y) \wedge z & = & x \wedge (y \wedge z) \quad \forall x, y, z \in \mathcal{P}(E)^{\mathcal{P}(E)} \quad (\text{associatividade}) \\
 x \vee (x \wedge y) & = & x \quad \forall x, y \in \mathcal{P}(E)^{\mathcal{P}(E)} \quad (\text{absorção}) \\
 x \wedge (x \vee y) & = & x \quad \forall x, y \in \mathcal{P}(E)^{\mathcal{P}(E)} \quad (\text{absorção}) \\
 x \vee (y \wedge z) & = & (x \vee y) \wedge (x \vee z) \quad \forall x, y, z \in \mathcal{P}(E)^{\mathcal{P}(E)} \quad (\text{distributividade}) \\
 x \wedge (y \vee z) & = & (x \wedge y) \vee (x \wedge z) \quad \forall x, y, z \in \mathcal{P}(E)^{\mathcal{P}(E)} \quad (\text{distributividade})
 \end{array}$$

Trivialmente, o reticulado distributivo $A = \langle \mathcal{P}(E)^{\mathcal{P}(E)}; \vee, \wedge \rangle$ é um modelo para o conjunto E_3 de equações abaixo:

$$\begin{aligned}
 E_3 = \{ \quad & e_{20} : \quad x \vee x = x, \\
 & e_{21} : \quad x \wedge x = x, \\
 & e_{22} : \quad x \vee y = y \vee x, \\
 & e_{23} : \quad x \wedge y = y \wedge x, \\
 & e_{24} : \quad (x \vee y) \vee z = x \vee (y \vee z), \\
 & e_{25} : \quad (x \wedge y) \wedge z = x \wedge (y \wedge z), \\
 & e_{26} : \quad x \vee (x \wedge y) = x, \\
 & e_{27} : \quad x \wedge (x \vee y) = x, \\
 & e_{28} : \quad x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z), \\
 & e_{29} : \quad x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z) \quad \}
 \end{aligned}$$

E, portanto, pelo teorema de completeza de Birkhoff [Bir35], se quisermos verificar que $\psi = \psi'$, para qualquer par (ψ, ψ') de operadores morfológicos invariantes por translação basta mostrarmos que $E_3 \vdash \psi = \psi'$.

O problema da verificação de identidade entre operadores morfológicos através desta abordagem reside na necessidade em saber se as partes que compõem os operadores, que se quer verificar a identidade, são iguais ou não. Por exemplo, sejam $\psi = (\psi_1 \vee \psi_2) \wedge (\psi_3 \wedge \psi_4)$ e $\psi' = (\psi'_1 \wedge \psi'_3) \vee (\psi'_2 \vee \psi'_4) \wedge \psi'_5$. Podemos deduzir que $E_3 \vdash \psi = \psi'$ ou $E_3 \not\vdash \psi = \psi'$ se já sabemos previamente se $\psi_i = \psi'_j$, para cada par (i, j) de índices, com $1 \leq i \leq 4$ e $1 \leq j \leq 5$.

Entretanto, como já vimos, cada operador morfológico invariante por translação pode ser expresso como um operador morfológico decomposto e, pelo lema 9.24, cada operador morfológico decomposto pode ser expresso através de uniões e interseções de operadores morfológicos

seqüenciais.

Suponha que ψ e ψ' , dados no parágrafo acima, são tais que cada ψ_i e ψ'_j (em ψ e ψ' , respectivamente) são operadores seqüenciais. Neste caso, podemos decidir se para cada par (ψ, ψ') , $\psi_i = \psi'_j$. Para isto, basta calcularmos as formas normais de ψ e ψ' em relação ao sistema de reescrita R_2 que, como já vimos, é um sistema útil para a verificação de identidade entre operadores morfológicos seqüenciais, pois $R_2(\psi_i) \equiv R_2(\psi'_j)$ se e somente se $\psi_i = \psi'_j$.

Exemplo 9.29. Considere os seguintes operadores morfológicos invariantes por translação e isotônicos, $\psi = \delta_{\begin{bmatrix} 10 \\ 01 \end{bmatrix}} \circ (\delta_{[10]} \vee \epsilon_{[01]})$ e $\psi' = \delta_{[10]} \circ \delta_{\begin{bmatrix} 10 \\ 01 \end{bmatrix}}$.

Queremos verificar através de um algoritmo se (ou não) $\psi = \psi'$.

Primeiro: vamos representar ψ e ψ' em suas respectivas formas decompostas, são elas:

$$\mathcal{A}(\psi) = (\delta_N \vee \delta_L) \circ (\delta_O \vee \delta_O) \quad \text{e} \quad \mathcal{A}(\psi') = \delta_O \circ (\delta_N \vee \delta_L)$$

Segundo: a partir de $\mathcal{A}(\psi)$ e $\mathcal{A}(\psi')$ obtemos as representações de ψ e ψ' formadas por uniões e interseções de operadores morfológicos seqüenciais, que são dadas, respectivamente por $R_1(\mathcal{A}(\psi))$ e $R_1(\mathcal{A}(\psi'))$.

$$R_1(\mathcal{A}(\psi)) = (\delta_N \circ \delta_O) \vee (\delta_N \circ \delta_O) \vee (\delta_L \circ \delta_O) \vee (\delta_L \circ \delta_O) \quad \text{e} \quad R_1(\mathcal{A}(\psi')) = (\delta_O \circ \delta_N) \vee (\delta_O \circ \delta_L)$$

Terceiro: $R_1(\mathcal{A}(\psi))$ e $R_1(\mathcal{A}(\psi'))$ são operadores morfológicos formados por uniões ou interseções de operadores morfológicos seqüenciais. Através do sistema de reescrita R_2 , obtemos as formas normais dos operadores morfológicos seqüenciais que compõem $R_1(\mathcal{A}(\psi))$ e $R_1(\mathcal{A}(\psi'))$.

$$\begin{aligned} R_2(R_1(\mathcal{A}(\psi))) &= R_2((\delta_N \circ \delta_O) \vee (\delta_N \circ \delta_O) \vee (\delta_L \circ \delta_O) \vee (\delta_L \circ \delta_O)) \\ &= (R_2(\delta_N \circ \delta_O)) \vee (R_2(\delta_N \circ \delta_O)) \vee (R_2(\delta_L \circ \delta_O)) \vee (R_2(\delta_L \circ \delta_O)) \\ &= (\delta_N \circ \delta_O) \vee (\delta_N \circ \delta_O) \vee \iota \vee \iota \end{aligned}$$

$$\begin{aligned} R_2(R_1(\mathcal{A}(\psi'))) &= R_2((\delta_O \circ \delta_N) \vee (\delta_O \circ \delta_L)) \\ &= (R_2(\delta_O \circ \delta_N)) \vee (R_2(\delta_O \circ \delta_L)) \\ &= (\delta_O \circ \delta_N) \vee \iota \end{aligned}$$

Quarto: $\psi = \mathcal{A}(\psi) = R_1(\mathcal{A}(\psi)) = R_2(R_1(\mathcal{A}(\psi)))$ e $\psi = \mathcal{A}(\psi') = R_1(\mathcal{A}(\psi')) = R_2(R_1(\mathcal{A}(\psi')))$. Isto é, ψ e $R_2(R_1(\mathcal{A}(\psi)))$ são termos que possuem o mesmo significado segundo a semântica da morfologia matemática. Analogamente, para ψ' e $R_2(R_1(\mathcal{A}(\psi')))$.

Portanto, a identidade entre ψ e ψ' fica verificada se decidirmos sobre a validade da identidade $(\delta_N \circ \delta_O) \vee (\delta_N \circ \delta_O) \vee \iota \vee \iota = (\delta_O \circ \delta_N) \vee \iota$.

Seja $\psi_1 = \delta_N \circ \delta_O$, $\psi_2 = \iota$, $\psi'_1 = \delta_O \circ \delta_N$ e $\psi'_2 = \iota$. Sendo assim, queremos verificar a validade

da identidade $\psi_1 \vee \psi_1 \vee \psi_2 \vee \psi_2 = \psi'_1 \vee \psi'_2$.

Como, $\psi_1, \psi_2, \psi'_1, \psi'_2$ são operadores morfológicos seqüenciais ótimos, torna-se trivial verificarmos se $\psi_i = \psi'_i$, para cada $1 \leq i, j \leq 2$.

De fato, se $\gamma = \gamma_1 \circ \dots \circ \gamma_n \stackrel{\text{notação pré-fixa}}{=} \circ(\gamma_1, \dots, \gamma_n)$ e $\gamma' = \gamma'_1 \circ \dots \circ \gamma'_m \stackrel{\text{notação pré-fixa}}{=} \circ(\gamma'_1, \dots, \gamma'_m)$, onde $\gamma_{i(1 \leq i \leq n)}, \gamma'_{j(1 \leq j \leq m)} \in \{\delta_N, \delta_S, \delta_L, \delta_O, \iota\}$, sendo γ e γ' termos irredutíveis, obtidos através de R_2 então, γ e γ' são ótimos e temos que, $\gamma = \gamma'$ se e somente se γ diferir de γ' apenas pela permutação de seus subtermos.

Neste exemplo temos $\psi_1 = \psi'_1$, pois diferem apenas de uma permutação de seus subtermos, e trivialmente temos $\psi_2 = \psi'_2$, e conseqüentemente,

$$\psi = \psi' \text{ se e somente se } \psi_1 \vee \psi_1 \vee \psi_2 \vee \psi_2 = \psi_1 \vee \psi_2, \quad \text{com } \psi_1 \neq \psi_2.$$

Mas, ψ_1 e ψ_2 são elementos de $\mathcal{P}(E)^{\mathcal{P}(E)}$ e como vimos $A = \langle \mathcal{P}(E)^{\mathcal{P}(E)}, \vee, \wedge \rangle$ forma um reticulado distributivo e portanto, $\psi_1 \vee \psi_1 \vee \psi_2 \vee \psi_2 = \psi_1 \vee \psi_2$ se e somente se $x \vee x \vee y \vee y = x \vee y, \forall x, y \in \mathcal{V}$. Conseqüentemente, pelo Teorema de Birkhoff [Bir35], verificar se $x \vee x \vee y \vee y = x \vee y, \forall x, y \in \mathcal{V}$ é conseqüência semântica de A (que é um modelo de E_3), é equivalente a verificar se $E_3 \vdash x \vee x \vee y \vee y = x \vee y$.

Pelo o que acabamos de ver, para conseguirmos resolver, de modo automático (através de um algoritmo), o problema da verificação da identidade entre operadores morfológicos invariantes por translação e isotônicos, resta-nos encontrar um método de calcularmos os teoremas de E_3 através de um algoritmo. Em nossa abordagem, isto significa encontrar um sistema de reescrita de termos, digamos R_3 , tal que $\xrightarrow{*}_{R_3} = \equiv_{E_3}$ e R_3 seja confluyente e noetheriano, isto é, R_3 deve ser canônico (completo).

Note que, não há como dar uma “orientabilidade” para as equações do conjunto $E_3 = \{e_{20}, \dots, e_{29}\}$, uma vez que não existe relação de ordem capaz de orientar equações que expressem a lei comutativa, como é o caso das equações e_{22} e e_{23} . Se conseguíssemos obter um sistema de reescrita R_3 completo, possuindo as mesmas classes de congruência induzidas por E_3 , isto é, $\xrightarrow{*}_{R_3} = \equiv_{E_3}$, teríamos, então, conseguido exibir um processo automático e finito para a verificação da igualdade entre os termos de $\mathcal{T}_{E_3}$. Mas, tal sistema R_3 completo (no sentido clássico, isto é, módulo $\emptyset$) não existe devido ao fato da não orientabilidade das equações e_{22} e e_{23} em E_3 .

Porém, como vimos no capítulo 7, podemos dividir o conjunto E_3 em duas partes disjuntas com o intuito de separarmos a porção de E_3 que é impossível de ser orientável (seja T_3 esta parte), daquela porção de E_3 que pode ser orientada (seja E'_3 esta parte), com $E_3 = E'_3 \cup T_3$. Sendo assim, temos $E'_3 = \{e_{20}, e_{21}, e_{24}, e_{25}, e_{26}, e_{27}, e_{28}, e_{29}\}$ e $T_3 = \{e_{22}, e_{23}\}$.

Como as classes de congruência induzidas por T_3 são finitas e construtivas (há um algoritmo, que termina, para gerar seus elementos), basta encontrarmos um sistema de reescrita R_3 que induza as mesmas classes de congruência que E'_3 e que seja completo módulo T_3 , ou em

outros termos, queremos encontrar um sistema de reescrita completo módulo T_3 .

Entretanto, sabemos que há um sistema de reescrita completo módulo T_3 , ou equivalentemente, completo módulo AC (veja exemplo 7.51), obtido pelo método de completção apresentado no capítulo 7, e conseqüentemente temos um método efetivo para a verificação de identidade entre operadores morfológicos invariantes por translação e isotônicos.

Abaixo apresentamos o sistema de reescrita R_3 completo módulo T_3 para a teoria equacional E_3 que expressa as identidades da estrutura algébrica de um reticulado distributivo.

$$R_3 = \left\{ \begin{array}{l} x \vee (x \wedge y) \rightarrow x, \\ x \wedge (y \vee z) \rightarrow (x \wedge y) \vee (x \wedge z), \\ x \wedge x \rightarrow x, \\ x \vee x \rightarrow x \end{array} \right\}$$

Daqui por diante, o termo canônico $R_3(R_2(R_1(\mathcal{A}(\psi))))$ é denotado por $\psi\downarrow$.

Um operador morfológico ψ é uma frase (termo) da Linguagem Morfológica e portanto, daqui por diante, por abuso de linguagem, quando dizemos que ψ é um operador canônico, estamos querendo dizer que o termo que expressa o operador morfológico ψ é canônico.

Exemplo 9.30. *Vamos verificar a identidade entre os operadores morfológicos ψ e ψ' dados no exemplo 9.29. Pelo exemplo 9.29 temos:*

$$R_2(R_1(\mathcal{A}(\psi))) = (\delta_N \circ \delta_O) \vee (\delta_N \circ \delta_O) \vee \iota \vee \iota;$$

$$R_2(R_1(\mathcal{A}(\psi')))) = (\delta_O \circ \delta_N) \vee \iota.$$

Logo, resta-nos mostrar que a identidade $(\delta_N \circ \delta_O) \vee (\delta_N \circ \delta_O) \vee \iota \vee \iota = (\delta_O \circ \delta_N) \vee \iota$ é satisfeita segundo a semântica da morfologia matemática.

Mas, $\delta_N \circ \delta_O$ e $\delta_O \circ \delta_N$ são operadores morfológicos equivalentes, isto é, $\delta_N \circ \delta_O(X) = \delta_O \circ \delta_N(X) \forall X \in \mathcal{P}(E)$, e neste caso, como já vimos, escrevemos simplesmente $\delta_N \circ \delta_O = \delta_O \circ \delta_N$. A identidade existente entre $\delta_N \circ \delta_O$ e $\delta_O \circ \delta_N$ fica garantida pelo cálculo das suas formas canônicas em relação ao sistema de reescrita R_2 que é completo módulo T e, portanto, temos $\delta_N \circ \delta_O = \delta_O \circ \delta_N$ se e somente se $R_2(\delta_N \circ \delta_O) \equiv_T R_2(\delta_O \circ \delta_N)$. Neste exemplo temos, trivialmente, a validade da identidade $\delta_O \circ \delta_N \equiv_T \delta_N \circ \delta_O$ e portanto, $R_2(\delta_N \circ \delta_O) \equiv_T R_2(\delta_O \circ \delta_N)$.

Portanto, mostrar que a identidade $(\delta_N \circ \delta_O) \vee (\delta_N \circ \delta_O) \vee \iota \vee \iota = (\delta_O \circ \delta_N) \vee \iota$ é satisfeita pela semântica da morfologia matemática é equivalente a mostrar que:

$$\psi_1 \vee \psi_1 \vee \psi_2 \vee \psi_2 \stackrel{?}{=} \psi_1 \vee \psi_2$$

onde $\psi_1, \psi_2 \in \mathcal{P}(E)^{\mathcal{P}(E)}$, sendo $\psi_1 = \delta_N \circ \delta_O$ e $\psi_2 = \iota$.

Como vimos, $\langle \mathcal{P}(E)^{\mathcal{P}(E)}, \vee, \wedge \rangle$ forma um reticulado distributivo e, neste caso,

$$\psi_1 \vee \psi_1 \vee \psi_2 \vee \psi_2 = \psi_1 \vee \psi_2, \text{ onde } \psi_1, \psi_2 \in \mathcal{P}(E)^{\mathcal{P}(E)}$$

se e somente se

$$x \vee x \vee y \vee y = x \vee y, \quad \forall x, y \in \mathcal{V}.$$

Seja σ a “substituição” dada por $\sigma = \{\psi_1 \leftarrow x, \psi_2 \leftarrow y\}$ e seja ψ^{-1} sua inversa, isto é, $\psi^{-1} = \{x \leftarrow \psi_1, y \leftarrow \psi_2\}$. Sendo assim, $(\psi_1 \vee \psi_1 \vee \psi_2 \vee \psi_2)\sigma = x \vee x \vee y \vee y$ e $(x \vee x \vee y \vee y)\sigma^{-1} = \psi_1 \vee \psi_1 \vee \psi_2 \vee \psi_2$, analogamente, $(\psi_1 \vee \psi_2)\sigma = x \vee y$ e $(x \vee y)\sigma^{-1} = \psi_1 \vee \psi_2$.

$\langle \mathcal{P}(E)^{\mathcal{P}(E)}, \vee, \wedge \rangle$ é um modelo para E_3 e portanto, pelo Teorema da completeza de Birkhoff [Bir35], a identidade acima é satisfeita para o reticulado distributivo $\langle \mathcal{P}(E)^{\mathcal{P}(E)}, \vee, \wedge \rangle$ se e somente se $E_3 \vdash x \vee x \vee y \vee y = x \vee y$.

Entretanto, queremos garantir que todos os teoremas de E_3 possam ser deduzidos através de um algoritmo (um método efetivo), após uma quantidade finita de passos. Como vimos, segundo nossa abordagem, este método efetivo para o cálculo dos teoremas de E_3 equivale a existência de um sistema de reescrita completo que induza as mesmas classes de congruência de E_3 . Como vimos, R_3 é o sistema de reescrita procurado, e portanto o problema da verificação da identidade entre dois termos, digamos t_1 e t_2 , num reticulado distributivo, pode ser resolvido calculando os termos canônicos $R_3(t_1)$ e $R_3(t_2)$ e comparando-os se são iguais módulo T_3 .

$$R_3((x \vee x) \vee y \vee y) \equiv_{T_3} R_3(x \vee y)$$

Mas,

$$R_3((x \vee x) \vee y \vee y) = x \vee y = R_3(x \vee y)$$

E portanto, acabamos de mostrar que $E_3 \vdash x \vee x \vee y \vee y = x \vee y$ ou equivalentemente (pelo Teorema de completeza de Birkhoff), a identidade $x \vee x \vee y \vee y = x \vee y$ é satisfeita pelo reticulado distributivo $\langle \mathcal{P}(E)^{\mathcal{P}(E)}, \vee, \wedge \rangle$, isto é, $\gamma \vee \gamma \vee \gamma' \vee \gamma' = \gamma \vee \gamma', \forall \gamma, \gamma' \in \mathcal{P}(E)^{\mathcal{P}(E)}$.

Em particular, tomando $\gamma = \psi_1$ e $\gamma' = \psi_2$ temos que $\psi_1 \vee \psi_1 \vee \psi_2 \vee \psi_2 = \psi_1 \vee \psi_2$.

Mas, $\psi_1 = \delta_N \circ \delta_O$ e $\psi_2 = \iota$, logo acabamos de mostrar que, de fato,

$$(\delta_N \circ \delta_O) \vee (\delta_N \circ \delta_O) \vee \iota \vee \iota = (\delta_O \circ \delta_N) \vee \iota.$$

Mas,

$$(\delta_N \circ \delta_O) \vee (\delta_N \circ \delta_O) \vee \iota \vee \iota = R_2(R_1(\mathcal{A}(\delta_{\begin{bmatrix} 10 \\ 01 \end{bmatrix}} \circ (\delta_{[10]} \vee \epsilon_{[01]})))) \text{ e } (\delta_O \circ \delta_N) \vee \iota = R_2(R_1(\mathcal{A}(\delta_{[10]} \circ \delta_{\begin{bmatrix} 10 \\ 01 \end{bmatrix}}))))$$

e portanto,

$$\delta_{\begin{bmatrix} 10 \\ 01 \end{bmatrix}} \circ (\delta_{[10]} \vee \epsilon_{[01]}) = \delta_{[10]} \circ \delta_{\begin{bmatrix} 10 \\ 01 \end{bmatrix}}.$$

Daqui por diante, escrevemos $\psi \downarrow$ ao invés de $R_3(R_2(R_1(\mathcal{A}(\psi)))\sigma)\sigma^{-1}$, isto é, $\psi \downarrow$ denota o operador canônico equivalente a ψ .

Neste exemplo, a forma canônica dos termos (operadores morfológicos) ψ e ψ' é dada pelo termo (operador morfológico):

$$\psi \downarrow = (\delta_N \circ \delta_O) \vee I$$

A foto de tela a seguir ilustra, na linguagem de programação visual Cantata do software de processamento de imagens Khoros, a execução dos operadores ψ e ψ' para uma instância. Este software permite ao usuário criar sua própria biblioteca de operadores, como por exemplo, a dilatação e erosão. Uma vez o usuário tendo implementado seus operadores de biblioteca, ele pode criar operadores mais complexos através do modo de programação visual, o Cantata. Desse modo, cada operador novo é criado compondo operadores elementares. Na foto de tela abaixo, por exemplo, cada caixinha representa um operador. A caixinha denominada “3x3 Str.El.” contém a especificação de um elemento estruturante 3x3. A “Dilation” representa o operador da dilatação para subconjuntos, a “Erosion” representa o operador da erosão para subconjuntos. “XOR” recebe como entrada duas imagens binárias e devolve como saída a imagem resultante da operação lógica ou-exclusivo, deste modo, se são fornecidas duas imagens iguais ao operador “XOR” teremos como saída a imagem vazia. “Display Image” mostra ao usuário a imagem que recebe como entrada. Em todas as 4 fotos de tela que tiramos neste capítulo, cada uma delas contém exatamente 3 display images. O primeiro display image, mais acima, representa a imagem de entrada, o segundo, um pouco mais abaixo, representa a imagem de saída obtida por um dos operadores morfológicos dados no respectivo exemplo, e o terceiro display image mostra o resultado da operação lógica ou-exclusivo, realizada com a imagem de saída relativa ao operador dado para o exemplo com a imagem de saída relativa ao operador canônico obtido em cada exemplo. Utilizamos a operação “XOR” para mostrarmos que, para a instância dada, o operador canônico obtido realiza exatamente a mesma tarefa do operador dado. Informações mais detalhadas sobre o software Khoros pode ser obtida no endereço eletrônico “<http://ftp.khoros.com/khoros/whatis.html>”.

Em geral, como veremos nos próximos exemplos, as formas canônicas obtidas para os operadores morfológicos possuem uma grande quantidade de símbolos funcionais. O sistema de reescrita R_4 , a seguir, tem o objetivo de “reconstruir” os operadores canônicos obtidos por R_3 , e assim, obter uma representação para o mesmo operador morfológico com uma quantidade menor de símbolos funcionais.

$$R_4 = \{ (x \wedge y) \vee (x \wedge z) \rightarrow x \wedge (y \vee z) \}$$

É fácil ver que R_4 é noetheriano pois a quantidade de símbolos funcionais do lado esquerdo de cada regra é estritamente maior que a quantidade de símbolos funcionais do lado direito da respectiva regra. Além disso, a quantidade de vezes que uma variável aparece no lado esquerdo de cada regra é maior ou igual a quantidade de vezes que esta mesma variável aparece no lado direito da respectiva regra.

Seja ψ é um operador morfológico, a *complexidade paralela* de ψ é definida como sendo a quantidade de “passos” que uma máquina paralela “ideal” (uma máquina que possui processadores suficientes para o paralelismo maciço de uma tarefa) necessita realizar para o cálculo de ψ e é denotada por $\|\psi\|^p$.

Se um operador morfológico ψ possuir a menor complexidade paralela dentre todos os operadores morfológicos equivalentes a ψ , então dizemos que ψ é um operador *ótimo*^p.

O cálculo da complexidade paralela é um tanto quanto intuitiva, partindo-se da noção de complexidade seqüencial dada neste texto. Sendo assim, não nos preocupamos em defini-la, mas vamos apresentá-la intuitivamente através do exemplo que segue.

Exemplo 9.31. Seja $\psi = (\delta_A \wedge \delta_B) \vee (\epsilon_C \vee \delta_D)$.

A complexidade de ψ é $\|\psi\| = |A| - 1 + |B| - 1 + |C| - 1 + |D| - 1 + 3 = |A| + |B| + |C| + |D| - 1$.

A complexidade paralela de ψ é $\|\psi\|^p = \max\{|A|, |B|, |C|, |D|\} - 1 + 2 = \max\{|A|, |B|, |C|, |D|\} + 1$.

Apesar de não demonstrarmos, conjecturamos que, de fato, um operador canônico reconstruído por R_4 constitui um operador morfológico ótimo^p. Sendo assim, o operador canônico (forma irreduzível em relação a R_3) obtido pelo exemplo 9.30 constituiria um operador ótimo^p pois, neste caso, $R_4((\delta_O \circ \delta_O) \vee \iota) = (\delta_O \circ \delta_O) \vee \iota$.

Por fim, assim como vimos a existência de um algoritmo trivial $\mathcal{A}$ que, quando aplicado a um operador morfológico invariante por translação isotônico, devolve um operador equivalente ao primeiro, porém, utilizando-se apenas os símbolos $\{\delta_O, \delta_L, \delta_N, \delta_S, \iota\}$ mais os símbolos auxiliares de parênteses. É fácil ver que existe um algoritmo trivial, vamos chamá-lo de $\mathcal{A}^{-1}$, que equivale ao processo inverso efetuado por $\mathcal{A}$. Assim, como a complexidade de um operador morfológico não se altera pela aplicação do algoritmo $\mathcal{A}$, o mesmo ocorre em relação ao algoritmo

$\mathcal{A}^{-1}$.

No exemplo 9.30 temos $\mathcal{A}^{-1}((\delta_O \circ \delta_O) \vee \iota) = \delta_{[101]}$. Note que, $\|\mathcal{A}^{-1}((\delta_O \circ \delta_O) \vee \iota)\| = \|\delta_{[101]}\| = 1$.

Exemplo 9.32. Considere os operadores morfológicos $\psi = \delta_A \circ (\epsilon_A \circ \delta_A)$ e $\psi' = \delta_A$, onde $A = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$.

Vamos mostrar a identidade existente entre ψ e ψ' , isto é, queremos mostrar que $\psi = \psi'$.

Vamos calcular a forma canônica de ψ .

Primeiro: através do algoritmo $\mathcal{A}$ representamos ψ e ψ' usando apenas os símbolos do conjunto $\{\vee, \wedge, \circ, \delta_O, \delta_L, \delta_N, \delta_S, \iota\}$ mais os símbolos auxiliares “(” e “)”. Em outros termos, vamos representá-los em suas respectivas formas decompostas, são elas:

$$\mathcal{A}(\psi) = (\delta_N \vee \delta_S) \circ ((\delta_S \wedge \delta_N) \circ (\delta_N \vee \delta_S)) \text{ e } \mathcal{A}(\psi') = \delta_N \vee \delta_S.$$

Segundo: a partir de $\mathcal{A}(\psi)$ e $\mathcal{A}(\psi')$ obtemos as representações de ψ e ψ' formadas por uniões ou interseções de operadores morfológicos sequenciais, que são dadas, respectivamente por $R_1(\mathcal{A}(\psi))$ e $R_1(\mathcal{A}(\psi'))$.

Vamos calcular $R_1(\mathcal{A}(\psi))$:

$$\begin{aligned} & (\delta_N \vee \delta_S) \circ ((\delta_S \wedge \delta_N) \circ (\delta_N \vee \delta_S)) \stackrel{\text{notação}}{=} (c \vee b) \circ ((b \wedge c) \circ (c \vee b)) \\ & \xrightarrow[r_2]{[p_1]} ((c \circ ((b \wedge c) \circ (c \vee b))) \vee (b \circ ((b \wedge c) \circ (c \vee b)))) \quad p_1 = \langle \rangle \\ & \xrightarrow[r_3]{[p_2]} ((c \circ ((b \circ (c \vee b)) \wedge (c \circ (c \vee b)))) \vee (b \circ ((b \wedge c) \circ (c \vee b)))) \quad p_2 = \langle 1, 2 \rangle \\ & \xrightarrow[r_3]{[p_3]} ((c \circ ((b \circ (c \vee b)) \wedge (c \circ (c \vee b)))) \vee (b \circ ((b \circ (c \vee b)) \wedge (c \circ (c \vee b)))) \quad p_3 = \langle 2, 2 \rangle \\ & \xrightarrow[r_7]{[p_4]} \circ \xrightarrow[r_7]{[p_5]} ((c \circ (((b \circ c) \vee (b \circ b)) \wedge (c \circ (c \vee b)))) \vee (b \circ (((b \circ c) \vee (b \circ b)) \wedge (c \circ (c \vee b)))) \quad p_4 = \langle 1, 2, 1 \rangle, p_5 = \langle 2, 2, 1 \rangle \\ & \xrightarrow[r_6]{[p_6]} \circ \xrightarrow[r_6]{[p_7]} \\ & ((c \circ (((b \circ c) \vee (b \circ b)) \wedge ((c \circ c) \vee (c \circ b)))) \vee (b \circ (((b \circ c) \vee (b \circ b)) \wedge ((c \circ c) \vee (c \circ b)))) \quad p_6 = \langle 1, 2, 2 \rangle, p_7 = \langle 2, 2, 2 \rangle \\ & \xrightarrow[r_{11}]{[p_8]} \circ \xrightarrow[r_{11}]{[p_9]} \\ & (((c \circ ((b \circ c) \vee (b \circ b))) \wedge (c \circ ((c \circ c) \vee (c \circ b)))) \vee ((b \circ ((b \circ c) \vee (b \circ b))) \wedge (b \circ ((c \circ c) \vee (c \circ b))))) \quad p_8 = \langle 1 \rangle, p_9 = \langle 2 \rangle \\ & \xrightarrow[r_6]{[p_{10}]} \circ \xrightarrow[r_6]{[p_{11}]} \circ \xrightarrow[r_7]{[p_{12}]} \circ \xrightarrow[r_7]{[p_{13}]} \end{aligned}$$

$$(((c \circ (b \circ c)) \vee (c \circ (b \circ b))) \wedge ((c \circ (c \circ c)) \vee (c \circ (c \circ b)))) \vee (((b \circ (b \circ c)) \vee (b \circ (b \circ b))) \wedge ((b \circ (c \circ c)) \vee (b \circ (c \circ b)))) \\ p_{10} = \langle 1, 1 \rangle, p_{11} = \langle 1, 2 \rangle, p_{12} = \langle 2, 1 \rangle, p_{13} = \langle 2, 2 \rangle$$

Observe que o termo obtido acima é irreduzível segundo R_1 .

Vamos calcular $R_1(\mathcal{A}(\psi'))$:

ψ' já é irreduzível em relação a $R_1(\mathcal{A}(\psi'))$.

Terceiro: $R_1(\mathcal{A}(\psi))$ e $R_1(\mathcal{A}(\psi'))$ são operadores morfológicos formados por uniões ou interseções de operadores morfológicos sequenciais. Através do sistema de reescrita R_2 obtemos as formas normais dos operadores morfológicos sequenciais que compõem $R_1(\mathcal{A}(\psi))$ e $R_1(\mathcal{A}(\psi'))$.

$$R_2(R_1(\mathcal{A}(\psi))) = ((c \vee b) \wedge ((c \circ c \circ c) \vee c)) \vee ((b \vee (b \circ b \circ b)) \wedge (c \vee b)) \\ \stackrel{\text{notação}}{=} ((\delta_N \vee \delta_S) \wedge ((\delta_N \circ \delta_N \circ \delta_N) \vee \delta_N)) \vee ((\delta_S \vee (\delta_S \circ \delta_S \circ \delta_S)) \wedge (\delta_N \vee \delta_S))$$

$$R_2(R_1(\mathcal{A}(\psi'))) = R_2(c \vee b) = c \vee b \stackrel{\text{notação}}{=} \delta_N \vee \delta_S.$$

Quarto: $R_2(R_1(\mathcal{A}(\psi))) = ((\psi_1 \vee \psi_2) \wedge (\psi_3 \vee \psi_1)) \vee ((\psi_2 \vee (\psi_4 \wedge (\psi_1 \vee \psi_2)))$, onde $\psi_1 = \delta_N, \psi_2 = \delta_S, \psi_3 = \delta_N \circ (\delta_N \circ \delta_N), \psi_4 = \delta_S \circ (\delta_S \circ \delta_S)$.

$$R_2(R_1(\mathcal{A}(\psi'))) = \psi'_1 \vee \psi'_2, \text{ onde } \psi'_1 = \delta_N \text{ e } \psi'_2 = \delta_S.$$

Mas, $\psi_1 = \psi'_1$ e $\psi_2 = \psi'_2$ e portanto queremos verificar a validade da seguinte identidade:

$$((\psi_1 \vee \psi_2) \wedge (\psi_3 \vee \psi_1)) \vee ((\psi_2 \vee (\psi_4 \wedge (\psi_1 \vee \psi_2))) \stackrel{?}{=} \psi_1 \vee \psi_2, \text{ onde } \psi_1, \psi_2, \psi_3, \psi_4 \in \mathcal{P}(E)^{\mathcal{P}(E)}.$$

Como vimos, $\langle \mathcal{P}(E)^{\mathcal{P}(E)}, \vee, \wedge \rangle$ forma um reticulado distributivo e, neste caso,

$$((\psi_1 \vee \psi_2) \wedge (\psi_3 \vee \psi_1)) \vee ((\psi_2 \vee (\psi_4 \wedge (\psi_1 \vee \psi_2))) = \psi_1 \vee \psi_2, \text{ onde } \psi_1, \psi_2, \psi_3, \psi_4 \in \mathcal{P}(E)^{\mathcal{P}(E)} \\ \text{se e somente se}$$

$$((z \vee x) \wedge ((y \vee z)) \vee ((x \vee w) \wedge (z \vee x)) = (z \vee x), \quad \forall x, y, z, w \in \mathcal{V}.$$

Daqui por diante escrevemos $\stackrel{AC}{=}$ no lugar de $\stackrel{T_3}{=}$, uma vez que T_3 é uma teoria AC.

Vamos calcular a forma canônica do termo $((z \vee x) \wedge ((y \vee z)) \vee ((x \vee w) \wedge (z \vee x)))$ segundo R_3 :

$$((z \vee x) \wedge ((y \vee z)) \vee ((x \vee w) \wedge (z \vee x))) \xrightarrow{r_2} \circ \xrightarrow{r_2} \\ (((z \vee x) \wedge y) \vee ((z \vee x) \wedge z)) \vee (((x \vee w) \wedge z) \vee ((x \vee w) \wedge x)) \stackrel{AC}{=} \circ \stackrel{AC}{=} \\ (((z \vee x) \wedge y) \vee (z \wedge (z \vee x))) \vee (((x \vee w) \wedge z) \vee (x \wedge (x \vee w))) \xrightarrow{r_2} \circ \xrightarrow{r_2} \\ (((z \vee x) \wedge y) \vee ((z \wedge z) \vee (z \wedge x))) \vee (((x \vee w) \wedge z) \vee ((x \wedge x) \vee (x \wedge w))) \xrightarrow{r_3} \circ \xrightarrow{r_3} \\ (((z \vee x) \wedge y) \vee (z \vee (z \wedge x))) \vee (((x \vee w) \wedge z) \vee (x \vee (x \wedge w))) \xrightarrow{r_1} \circ \xrightarrow{r_1} \\ (((z \vee x) \wedge y) \vee z) \vee (((x \vee w) \wedge z) \vee x) \stackrel{AC}{=} \circ \stackrel{AC}{=}$$

$$\begin{aligned}
& (z \vee ((z \vee x) \wedge y)) \vee (x \vee ((x \vee w) \wedge z)) \xrightarrow{\overline{\overline{AC}}} \circ \xrightarrow{\overline{\overline{AC}}} \\
& (z \vee (y \wedge (z \vee x))) \vee (x \vee (z \wedge (x \vee w))) \xrightarrow{\overline{\overline{r_2}}} \circ \xrightarrow{\overline{\overline{r_2}}} \\
& (z \vee ((y \wedge z) \vee (y \wedge x))) \vee (x \vee ((z \wedge x) \vee (z \wedge w))) \xrightarrow{\overline{\overline{AC}}} \circ \xrightarrow{\overline{\overline{AC}}} \\
& ((z \vee (y \wedge z)) \vee (y \wedge x)) \vee ((x \vee (z \wedge x)) \vee (z \wedge w)) \xrightarrow{\overline{\overline{AC}}} \circ \xrightarrow{\overline{\overline{AC}}} \\
& ((z \vee (z \wedge y)) \vee (y \wedge x)) \vee ((x \vee (x \wedge z)) \vee (z \wedge w)) \xrightarrow{\overline{\overline{r_1}}} \circ \xrightarrow{\overline{\overline{r_1}}} \\
& (z \vee (y \wedge x)) \vee (x \vee (z \wedge w)) \xrightarrow{\overline{\overline{AC}}} \\
& ((z \vee (y \wedge x)) \vee x) \vee (z \wedge w) \xrightarrow{\overline{\overline{AC}}} \\
& (z \vee ((y \wedge x) \vee x)) \vee (z \wedge w) \xrightarrow{\overline{\overline{AC}}} \circ \xrightarrow{\overline{\overline{AC}}} \\
& (z \vee (x \vee (x \wedge y))) \vee (z \wedge w) \xrightarrow{\overline{\overline{r_1}}} \\
& (z \vee x) \vee (z \wedge w) \xrightarrow{\overline{\overline{AC}}} \\
& (x \vee z) \vee (z \wedge w) \xrightarrow{\overline{\overline{AC}}} \\
& x \vee (z \vee (z \wedge w)) \xrightarrow{\overline{\overline{r_1}}} \\
& x \vee z.
\end{aligned}$$

Acabamos de mostrar que $R_3(R_2(R_1(\mathcal{A}(\psi)))) = x \vee z = R_3(R_2(R_1(\mathcal{A}(\psi'))))$ e portanto $\psi = \psi'$.

O operador canônico, obtido por R_3 , equivalente a ψ e ψ' é $\psi \downarrow = \delta_S \vee \delta_N$.

Note que, neste exemplo, a reconstrução de $\psi \downarrow$ por R_4 coincide com $\psi \downarrow$, mas como veremos nos próximos exemplos, isto nem sempre ocorre.

Se pudermos comprovar a conjectura feita nos parágrafos anteriores teremos que $\delta_S \vee \delta_N$ constituir-se um operador ótimo^p, isto é, não haveria operador morfológico γ pertencente à classe de equivalência de ψ tal que $\|\gamma\|^p < \|\psi \downarrow\|^p = \|\delta_S \vee \delta_N\|^p = 1$.

Por fim, $\mathcal{A}^{-1}(\delta_S \vee \delta_N) = \delta \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = \delta_A$

A foto de tela a seguir ilustra que a verificação de equivalência entre ψ e ψ' é feita através do cálculo do operador canônico $\psi \downarrow$ que é equivalente a ψ e ψ' .

A foto de tela a seguir ilustra a execução dos operadores ψ e ψ' para uma instância.

Exemplo 9.33. Vamos simplificar o seguinte operador morfológico $\psi = \delta_A \vee \delta_B$, onde $A = \begin{bmatrix} 010 \\ 111 \\ 000 \end{bmatrix}$

e $B = \begin{bmatrix} 010 \\ 111 \\ 000 \end{bmatrix}$.

A complexidade de ψ é $\|\psi\| = 4 + 1 + 3 = 8$.

A complexidade paralela de ψ é $\|\psi\|^p = \max\{4, 3\} + 1 = 5$.

$$\mathcal{A}(\psi) = (\delta_N \vee \delta_S \vee \delta_O \vee \delta_L \vee \iota) \vee (\delta_N \vee \delta_L \vee \delta_O \vee \iota).$$

Neste exemplo é fácil ver que $R_2(R_1(\mathcal{A}(\psi))) = R_1(\mathcal{A}(\psi)) = \mathcal{A}(\psi)$.

Facilmente pode-se observar que o operador canônico $\psi \downarrow$ é tal que $\psi \downarrow = R_3(\mathcal{A}(\psi)) = \delta_N \vee \delta_S \vee \delta_L \vee \delta_O \vee \iota$.

A complexidade do operador canônico $\psi \downarrow$ obtido é dada por $\|\psi \downarrow\| = \|\delta_N \vee \delta_S \vee \delta_L \vee \delta_O \vee \iota\| = 4$.

A complexidade paralela de $\psi \downarrow$ é $\|\psi \downarrow\|^p = \|\delta_N \vee \delta_S \vee \delta_L \vee \delta_O \vee \iota\|^p = 4$.

Como ocorreu no exemplo anterior, a reconstrução de $\psi \downarrow$ por R_4 coincide com $\psi \downarrow$, isto é, $R_4(\psi \downarrow) = \psi \downarrow$.

Por fim, $\mathcal{A}^{-1}(\psi \downarrow) = \delta \begin{bmatrix} 010 \\ 111 \\ 010 \end{bmatrix} = \delta_A$.

Por vezes, as seqüências de reescritas necessárias para a obtenção dos termos irreduzíveis são muito extensas para serem detalhadas, como foi feito no exemplo 9.32. Por este motivo, vamos utilizar o software CiME, desenvolvido por Evelyne Contejan e Claude Marché em 1996, para calcularmos as formas irreduzíveis necessárias. O software CiME é útil não só para o processo de reescrita de um termo, mas também para o processo de completção de sis-

temas de reescrita. Ao leitor mais interessado, este sistema está disponível via FTP no endereço eletrônico "<http://www.lri.fr/~demon/cime.html>".

Exemplo 9.34. Vamos verificar a equivalência entre os operadores morfológicos ψ e ψ' dados por $\psi = (\delta_C \wedge \epsilon_A) \vee (\epsilon_A \wedge \iota)$ e $\psi' = ((\epsilon_A \wedge \epsilon_B) \wedge \delta_D) \vee ((\epsilon_B \wedge \epsilon_A) \wedge \iota)$, onde os elementos estruturantes são $A = \begin{bmatrix} 100 \\ 111 \\ 100 \end{bmatrix}$, $B = \begin{bmatrix} 11 \\ 10 \end{bmatrix}$, $C = \begin{bmatrix} 01 \\ 00 \\ 10 \end{bmatrix}$ e $D = \begin{bmatrix} 00 \\ 00 \\ 10 \end{bmatrix}$.

Note que as complexidades de ψ e ψ' são, respectivamente, $\|\psi\| = 12$ e $\|\psi'\| = 17$.

As complexidades paralelas de ψ e ψ' são, respectivamente, $\|\psi\|^p = 6$ e $\|\psi'\|^p = 7$.

Primeiro: vamos calcular o termo canônico $\psi\downarrow$.

$$\mathcal{A}(\psi) = ((\iota \vee ((\delta_S \circ \delta_S) \circ \delta_O)) \wedge (\delta_S \wedge \iota \wedge \delta_N \wedge \delta_O \wedge (\delta_O \circ \delta_O))) \vee ((\delta_S \wedge \iota \wedge \delta_N \wedge \delta_O \wedge (\delta_O \circ \delta_O)) \wedge \iota).$$

$$R_2(R_1(\mathcal{A}(\psi))) = R_1(\mathcal{A}(\psi)) = ((\iota \vee (\delta_S \circ (\delta_S \circ \delta_O))) \wedge (\delta_S \wedge \iota \wedge \delta_N \wedge \delta_O \wedge (\delta_O \circ \delta_O))) \vee ((\delta_S \wedge \iota \wedge \delta_N \wedge \delta_O \wedge (\delta_O \circ \delta_O)) \wedge \iota).$$

$$\psi\downarrow = R_3(R_2(R_1(\mathcal{A}(\psi)))) = \iota \wedge \delta_S \wedge \delta_N \wedge (\delta_O \circ \delta_O) \wedge \delta_O.$$

$$R_4(\psi\downarrow) = \psi\downarrow.$$

A complexidade de $\psi\downarrow$ é $\|\psi\downarrow\| = 4$.

A complexidade paralela de $\psi\downarrow$ é $\|\psi\downarrow\|^p = 4$.

$$\text{Por fim, } \mathcal{A}^{-1}(\psi\downarrow) = \epsilon \begin{bmatrix} 100 \\ 111 \\ 100 \end{bmatrix} = \epsilon_A.$$

Segundo: vamos calcular o termo canônico $\psi'\downarrow$.

$$\mathcal{A}(\psi') = (\delta_S \wedge \iota \wedge \delta_N \wedge \delta_O \wedge (\delta_O \circ \delta_O) \wedge \iota \wedge \delta_N \wedge \delta_O \wedge ((\delta_S \circ \delta_S) \circ \delta_O)) \vee (\iota \wedge \delta_N \wedge \delta_O \wedge \delta_S \wedge \iota \wedge \delta_N \wedge \delta_O \wedge (\delta_O \circ \delta_O) \wedge \iota).$$

$$R_2(R_1(\mathcal{A}(\psi'))) = R_1(\mathcal{A}(\psi')) = (\delta_S \wedge \iota \wedge \delta_N \wedge \delta_O \wedge (\delta_O \circ \delta_O) \wedge \iota \wedge \delta_N \wedge \delta_O \wedge (\delta_S \circ \delta_S \circ \delta_O)) \vee (\iota \wedge \delta_N \wedge \delta_O \wedge \delta_S \wedge \iota \wedge \delta_N \wedge \delta_O \wedge (\delta_O \circ \delta_O) \wedge \iota).$$

$$\psi'\downarrow = R_3(R_2(R_1(\mathcal{A}(\psi')))) = \iota \wedge \delta_S \wedge \delta_N \wedge (\delta_O \circ \delta_O) \wedge \delta_O.$$

Pelo o que acabamos de ver, as formas canônicas de ψ e ψ' coincidem (são iguais módulo

AC), isto é, $\psi \downarrow_{AC} \psi' \downarrow$, e portanto os operadores morfológicos ψ e ψ' são equivalentes, isto é, $\psi(X) = \psi'(X) \forall X \in \mathcal{P}(E)$, ou em outros termos, $\psi = \psi'$.

Note que as complexidade e complexidade paralela do operador morfológico canônico $\psi \downarrow$ obtido são menores, respectivamente, que as complexidade e complexidade paralela de ψ e ψ' .

$$\|\psi \downarrow\| = 4 < \|\psi\| = 12 < \|\psi'\| = 17.$$

$$\|\psi \downarrow\|^p = 4 < \|\psi\|^p = 6 < \|\psi'\|^p = 7.$$

E pela conjectura dada, $\psi \downarrow$ constituiria-se, de fato, o operador morfológico ótimo^p, isto é, o de menor complexidade paralela dentre todos os operadores na classe de equivalência de ψ .

A foto de tela a seguir ilustra que a verificação de equivalência entre ψ e ψ' é feita através do cálculo do operador canônico $\psi \downarrow$ que é equivalente a ψ e ψ' .

$$(\delta_C \wedge \epsilon_A) \vee (\epsilon_A \wedge \iota) = ((\epsilon_A \wedge \epsilon_B) \wedge \delta_D) \vee ((\epsilon_B \wedge \epsilon_A) \wedge \iota)$$

A foto de tela a seguir ilustra, na linguagem de programação visual Cantata, a execução dos operadores ψ e $A^{-1}(\psi \downarrow)$ para uma instância.

O próximo exemplo nos mostra que nem sempre o operador canônico possui complexidade inferior (estrita ou não) ao operador inicialmente fornecido para o cálculo da simplificação do mesmo.

Exemplo 9.35. Vamos “tentar” simplificar o seguinte operador morfológico $\psi = \delta_A \wedge ((\delta_B \circ \epsilon_A) \vee (\delta_C \circ \epsilon_C))$, onde $A = \begin{bmatrix} 000 \\ 100 \end{bmatrix}$, $B = \begin{bmatrix} 00001 \\ 00000 \\ 00000 \\ 00000 \\ 10000 \end{bmatrix}$, $C = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$.

A complexidade de ψ é $\|\psi\| = 3$.

A complexidade paralela de ψ é $\|\psi\|^p = 2$.

$$\mathcal{A}(\psi) = ((\delta_S \circ \delta_O) \circ \delta_O) \wedge (((\delta_N \circ \delta_N \vee (((\delta_S \circ \delta_S) \circ \delta_O) \circ \delta_O) \circ \delta_O) \circ ((\delta_L \circ \delta_L) \circ \delta_N)) \vee (\delta_S \circ \delta_N)).$$

$$R_1(\mathcal{A}(\psi)) = (\delta_S \circ (\delta_O \circ \delta_O)) \wedge ((\delta_S \circ (\delta_S \circ (\delta_O \circ (\delta_O \circ (\delta_O \circ (\delta_L \circ (\delta_L \circ \delta_N))))))) \vee (\delta_N \circ (\delta_N \circ (\delta_L \circ (\delta_L \circ \delta_N)))) \vee (\delta_S \circ \delta_N)).$$

$$R_2(R_1(\mathcal{A}(\psi))) = (\delta_S \circ \delta_O \circ \delta_O) \wedge (((\delta_O \circ \delta_O \circ \delta_S) \vee (\delta_N \circ \delta_N \circ \delta_L \circ \delta_L \circ \delta_N)) \vee \iota).$$

$$\psi \downarrow = R_3(R_2(R_1(\mathcal{A}(\psi)))) = (\delta_S \circ \delta_O \circ \delta_O).$$

A complexidade do operador canônico obtido é $\|\psi \downarrow\| = \|\psi \downarrow\|^p = 0$.

$$R_4(\psi \downarrow) = \psi \downarrow.$$

$$\text{Por fim, } \mathcal{A}^{-1}(R_4(\psi \downarrow)) = \delta_A.$$

O próximo exemplo nos mostra que nem sempre um operador canônico reconstruído constitui-se num operador ótimo, e, portanto, o método descrito neste trabalho não pode ser utilizado para o cálculo de operadores morfológicos ótimos.

Exemplo 9.36. Considere o operador morfológico $\psi = \epsilon \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \circ \delta_{[101]}$.

Vamos calcular e comparar as complexidades de ψ , $\psi \downarrow$ e $R_4(\psi \downarrow)$.

$$\mathcal{A}(\psi) = (\delta_S \wedge \delta_N) \circ (\delta_O \vee \delta_L).$$

$$R_2(R_1(\psi)) = R_1(\psi) = ((\delta_N \circ \delta_O) \vee (\delta_N \circ \delta_L)) \wedge ((\delta_S \circ \delta_O) \vee (\delta_S \circ \delta_L)).$$

$$\psi \downarrow = R_3(R_2(R_1(\psi))) = ((\delta_O \circ \delta_S) \wedge (\delta_O \circ \delta_N)) \vee ((\delta_O \circ \delta_S) \wedge (\delta_L \circ \delta_N)) \vee ((\delta_L \circ \delta_S) \wedge (\delta_O \circ \delta_N)) \vee ((\delta_L \circ \delta_S) \wedge (\delta_L \circ \delta_N)).$$

$$R_4(\psi\downarrow) = ((\delta_N \circ \delta_L) \vee (\delta_N \circ \delta_O)) \wedge ((\delta_S \circ \delta_O) \vee (\delta_S \circ \delta_L)).$$

Note que a complexidade de ψ é menor que as complexidades do operador canônico $\psi\downarrow$ e do operador reconstruído por R_4 .

$$\|\psi\| = 2 < 7 = \|\psi\downarrow\| \quad \text{e} \quad \|\psi\| = 2 < 3 = \|R_4(\psi\downarrow)\|.$$

Quanto as complexidades paralelas, temos:

$$\|\psi\|^p = 2 < 3 = \|\psi\downarrow\|^p \quad \text{enquanto que} \quad \|\psi\|^p = 2 = \|R_4(\psi\downarrow)\|^p.$$

O próximo exemplo nos dá uma noção de como pode ser “difícil” computacionalmente obter o termo canônico de um operador morfológico.

Exemplo 9.37. Vamos simplificar o operador $\psi = \delta_A \circ (\epsilon_A \circ \delta_A)$, onde $A = \begin{bmatrix} 101 \\ 000 \\ 100 \end{bmatrix}$.

A complexidade do operador ψ é dada por $\|\psi\| = 6$.

A complexidade paralela do operador ψ é dada por $\|\psi\|^p = 6$.

$$\begin{aligned} \mathcal{A}(\psi) &= ((\delta_N \circ \delta_L) \vee (\delta_N \circ \delta_O) \vee (\delta_S \circ \delta_O)) \circ ((\delta_S \circ \delta_O) \wedge (\delta_S \circ \delta_L) \wedge (\delta_N \circ \delta_L)) \circ ((\delta_N \circ \delta_L) \vee (\delta_N \circ \delta_O) \vee (\delta_S \circ \delta_O)) \\ &\stackrel{\text{notação}}{=} ((c \circ d) \vee (c \circ e) \vee (b \circ e)) \circ ((b \circ e) \wedge (b \circ d) \wedge (c \circ d)) \circ ((c \circ d) \vee (c \circ e) \vee (b \circ e)). \end{aligned}$$

$$\begin{aligned} R_1(\mathcal{A}(\psi)) &= (((c \circ (e \circ (c \circ (d \circ (c \circ e)))))) \vee (c \circ (e \circ (c \circ (d \circ (c \circ d)))))) \vee (c \circ (e \circ (c \circ (d \circ (b \circ e)))))) \wedge (((c \circ (e \circ (b \circ (e \circ (c \circ e)))))) \vee (c \circ (e \circ (b \circ (e \circ (c \circ d)))))) \vee (c \circ (e \circ (b \circ (e \circ (b \circ e)))))) \wedge (((c \circ (e \circ (b \circ (d \circ (c \circ e)))))) \vee (c \circ (e \circ (b \circ (d \circ (c \circ d)))))) \vee (c \circ (e \circ (b \circ (d \circ (b \circ e)))))) \vee (((c \circ (d \circ (c \circ (d \circ (c \circ e)))))) \vee (c \circ (d \circ (c \circ (d \circ (c \circ d)))))) \vee (c \circ (d \circ (c \circ (d \circ (b \circ e)))))) \wedge (((c \circ (d \circ (b \circ (e \circ (c \circ e)))))) \vee (c \circ (d \circ (b \circ (e \circ (c \circ d)))))) \vee (c \circ (d \circ (b \circ (e \circ (b \circ e)))))) \wedge (((c \circ (d \circ (b \circ (d \circ (c \circ e)))))) \vee (c \circ (d \circ (b \circ (d \circ (c \circ d)))))) \vee (c \circ (d \circ (b \circ (d \circ (b \circ e)))))) \vee (((b \circ (e \circ (c \circ (d \circ (c \circ e)))))) \vee (b \circ (e \circ (c \circ (d \circ (c \circ d)))))) \vee (b \circ (e \circ (c \circ (d \circ (b \circ e)))))) \wedge (((b \circ (e \circ (b \circ (e \circ (c \circ e)))))) \vee (b \circ (e \circ (b \circ (e \circ (c \circ d)))))) \vee (b \circ (e \circ (b \circ (e \circ (b \circ e)))))) \wedge (((b \circ (e \circ (b \circ (d \circ (c \circ e)))))) \vee (b \circ (e \circ (b \circ (d \circ (c \circ d)))))) \vee (b \circ (e \circ (b \circ (d \circ (b \circ e)))))). \end{aligned}$$

$$\begin{aligned} R_2(R_1(\mathcal{A}(\psi))) &= ((((((c \circ c \circ e \circ c) \vee (c \circ c \circ d \circ c) \vee (e \circ c)) \wedge (((e \circ e \circ e \circ c) \vee (e \circ c)) \vee (e \circ e \circ e \circ b))) \wedge (((e \circ c) \vee (d \circ c)) \vee (e \circ b))) \vee ((((((c \circ c \circ d \circ c) \vee (c \circ d \circ c \circ d \circ c \circ d)) \vee (d \circ c)) \wedge (((e \circ c) \vee (d \circ c)) \vee (e \circ b))) \wedge (((d \circ c) \vee (d \circ d \circ d \circ c)) \vee (d \circ b))) \vee ((((((e \circ c) \vee (d \circ c)) \vee (e \circ b)) \wedge (((e \circ e \circ e \circ b) \vee (e \circ b)) \vee (b \circ e \circ b \circ e \circ b \circ e))) \wedge (((e \circ b) \vee (d \circ b)) \vee (b \circ b \circ e \circ b))))). \end{aligned}$$

$$R_4(\psi\downarrow) = R_4(R_3(R_2(R_1(\mathcal{A}(\psi)))) = R_3(R_2(R_1(\mathcal{A}(\psi)))) = (e \circ b) \vee (e \circ c) \vee (d \circ c) \\ \stackrel{\text{notação}}{=} (\delta_O \circ \delta_S) \vee (\delta_O \circ \delta_N) \vee (\delta_L \circ \delta_N).$$

Por fim, $\mathcal{A}^{-1}(\psi\downarrow) = \delta_A$.

A complexidade do operador canônico $\psi\downarrow$ obtido é $\|\psi\downarrow\| = 2$.

A complexidade paralela do operador canônico $\psi\downarrow$ obtido é $\|\psi\downarrow\|^p = 2$.

Exemplo 9.38. Vamos verificar a equivalência entre os operadores morfológicos $\psi = \delta_A \circ (\epsilon_A \circ \delta_A)$ e $\psi' = (((\epsilon_N \circ \delta_A) \wedge \delta_{[101]}) \circ \delta_N) \vee \delta_{\begin{bmatrix} 101 \\ 000 \\ 100 \end{bmatrix}}$, onde $A = \begin{bmatrix} 101 \\ 000 \\ 100 \end{bmatrix}$.

As complexidades de ψ e ψ' são respectivamente:

$$\|\psi\| = 6 \text{ e } \|\psi'\| = 6.$$

As complexidades paralelas de ψ e ψ' são respectivamente:

$$\|\psi\|^p = 6 \text{ e } \|\psi'\|^p = 4.$$

Pelo exemplo 9.37, a forma canônica de ψ é $\psi\downarrow = \delta_A$.

Resta-nos calcular $\psi'\downarrow$ e verificar se, de fato, $\psi\downarrow = \psi'\downarrow$.

$$\mathcal{A}(\psi') = (((\delta_S \circ ((\delta_N \circ \delta_O) \vee (\delta_N \circ \delta_L) \vee (\delta_S \circ \delta_O))) \wedge (\delta_O \vee \delta_L)) \circ \delta_N) \vee (\delta_S \circ \delta_O) \\ \stackrel{\text{notação}}{=} (((b \circ ((c \circ e) \vee (c \circ d) \vee (b \circ e))) \wedge (e \vee d)) \circ c) \vee (b \circ e) \\ R_1(\mathcal{A}(\psi')) = (((e \circ c) \vee (d \circ c)) \wedge ((b \circ (c \circ (e \circ c))) \vee (b \circ (c \circ (d \circ c))) \vee (b \circ (b \circ (e \circ c))))) \vee (b \circ e).$$

$$R_2(R_1(\mathcal{A}(\psi'))) = (((e \circ c) \vee (d \circ c)) \wedge (((e \circ c) \vee (d \circ c)) \vee (e \circ b))) \vee (b \circ e).$$

$$R_4(R_3(R_2(R_1(\mathcal{A}(\psi'))))) = R_3(R_2(R_1(\mathcal{A}(\psi')))) = (d \circ c) \vee (b \circ e) \vee (e \circ c) \stackrel{\text{notação}}{=} (\delta_L \circ \delta_N) \vee (\delta_S \circ \delta_O) \vee (\delta_O \circ \delta_N).$$

As complexidades de $\psi\downarrow$ são dadas por:

$$\|\psi\downarrow\| = \|\psi\downarrow\|^p = 2.$$

$$\text{Por fim, } \mathcal{A}^{-1}(R_4(\psi\downarrow)) = \delta_{\begin{bmatrix} 101 \\ 000 \\ 100 \end{bmatrix}} = \delta_A.$$

A foto de tela a seguir ilustra, na linguagem de programação visual Cantata, a execução dos operadores ψ' e $\mathcal{A}^{-1}(\psi'\downarrow)$ para uma instância.

A tabela a seguir apresenta as complexidades dos operadores dos exemplos dados acima.

	$\ \psi\ $	$\ \psi'\ $	$\ \psi\ ^P$	$\ \psi'\ ^P$	$\ \psi\downarrow\ $	$\ \psi\downarrow\ ^P$	$\ R_4(\psi\downarrow)\ $	$\ R_4(\psi\downarrow)\ ^P$
exemplo 9.30	2	1	2	1	1	1	1	1
exemplo 9.32	3	1	3	1	1	1	1	1
exemplo 9.33	8	-	5	-	4	4	4	4
exemplo 9.34	12	17	6	7	4	4	4	4
exemplo 9.35	3	-	2	-	0	0	0	0
exemplo 9.36	2	-	2	-	7	3	3	2
exemplo 9.37	6	-	6	-	2	2	2	2
exemplo 9.38	6	6	6	4	2	2	2	2

Tabela 9.3: Complexidades de alguns operadores.

O cálculo dos termos canônicos obtidos pelo sistema Cime foram realizados sob a plataforma *SunOS 5.5.1 Generic sun4u sparc SUNW, Ultra-Enterprise*. A seguir, apresentamos uma tabela com os tempos de máquina, bem como a quantidade de matchings realizados pelo Cime para estes cálculos.

	tempo(seg.) para o cálculo de $R_4(\psi\downarrow)$	tempo(seg.) para o cálculo de $R_4(\psi'\downarrow)$	chamadas p/ matchings (ψ)	chamadas p/ matchings (ψ')	reescritas (ψ)	reescritas (ψ')
exemplo 9.34	0,65	0,64	114	169	5	9
exemplo 9.35	0,54	-	963	-	22	-
exemplo 9.36	0,50	-	183	-	9	-
exemplo 9.37	482,07 (8,03 minutos)	-	94868	-	315	-
exemplo 9.38	-	0,51	94868	940	315	25

Tabela 9.4: Estatísticas fornecidas pelo sistema Cime para o cálculo de alguns operadores morfológicos canônicos reconstruídos.

Capítulo 10

Considerações finais

Neste trabalho, foi apresentado uma introdução sobre Sistemas de Reescrita de Termos no contexto de teorias equacionais e uma aplicação para a Morfologia Matemática.

Citamos, no capítulo 5, que qualquer máquina de Turing pode ser vista com um sistema de reescrita. Conseqüentemente, o problema da verificação da propriedade de terminação de sistemas de reescrita é indecidível, para o caso geral, uma vez que é sabido que o problema da parada uniforme para as máquinas de Turing é indecidível. Com isto, conclui-se que o problema de completção de sistemas de reescrita é indecidível para o caso geral.

Como consequência desta forte relação existente entre o conceito de máquinas de Turing e sistemas de reescrita de termos, tem-se que a computabilidade de um problema qualquer está diretamente ligada a existência de um sistema de reescrita completo para o problema.

Vimos que para completarmos sistemas de reescrita precisamos resolver o sub-problema de unificação de termos. Métodos efetivos para a unificação de termos são bem conhecidos e triviais quando os termos envolvidos são ordinários. Um primeiro algoritmo a solucionar o problema de unificação de termos foi proposto, como vimos no capítulo 4, por Herbrand [Her30]. Foi a partir do resultado de Herbrand que Robinson [Rob65] desenvolveu o seu conhecido método de Resolução. Entretanto, o problema da unificação de termos de primeira ordem, para o caso geral, é indecidível, por exemplo, o problema de unificação módulo ACD é indecidível [Kir90].

Apesar dos problemas da terminação e unificação serem indecidíveis, é possível verificar, para casos particulares, se um sistema de reescrita possui a propriedade de terminação, ou ainda, se há métodos efetivos que solucionam o problema de unificação para o caso particular considerado.

Sabemos, por Birkhoff [Bir35], que as noções de consequência sintática e consequência semântica de uma mesma teoria equacional de primeira ordem coincidem. Isso nos permite verificar a validade de identidades nos modelos de uma teoria equacional observando, unicamente,

os teoremas da teoria dada.

No contexto das teorias equacionais, a existência de um método efetivo para o cálculo dos teoremas coincide com a existência de um sistema de reescrita completo para a teoria dada.

Nesta dissertação estudamos a problemática de completção de sistemas de reescrita para o caso particular de teorias que possuem operadores associativos-comutativos. Um exemplo de teoria associativa-comutativa é aquela formada por equações que expressam as identidades dos reticulados distributivos, e como vimos, há um sistema de reescrita completo módulo AC para os reticulados distributivos.

É sabido, da Morfologia Matemática, que a tripla formada pelo conjunto dos operadores sobre $\mathcal{P}(E)$ mais as operações de união e interseção entre operadores sobre $\mathcal{P}(E)$ forma um reticulado distributivo. Tal “característica” da Morfologia Matemática dos operadores sobre subconjuntos nos permitiu descrever um método efetivo tanto para a verificação da equivalência de operadores morfológicos invariantes por translação e isotônicos como para a simplificação destes operadores. A verificação da equivalência de operadores morfológicos, segundo o método descrito neste trabalho, resulta no cálculo do que chamamos de “operadores canônicos”. O problema de encontrar um método efetivo para a verificação de equivalência de operadores morfológicos para imagens binárias, invariantes por translação e localmente definidos por uma janela W é tratado por Barrera e Salas [BS96] através do cálculo das “bases canônicas” destes operadores.

A classe dos operadores invariantes por translação e isotônicos constituem um caso particular de operadores invariantes por translação e localmente definidos por uma janela W e portanto, o método das bases canônicas pode ser usado para a verificação de equivalência entre operadores invariantes por translação e isotônicos. Entretanto, o método descrito nesta dissertação não pode ser usado para a verificação da identidade entre operadores invariantes por translação não isotônicos.

Um aspecto positivo para a abordagem dada em nossa dissertação é que, os operadores canônicos obtidos estão escritos na Linguagem Morfológica, o que não ocorre com o método das bases canônicas que são formadas por coleções intervalos fechados maximais.

Apesar de existir um algoritmo trivial, que reescreva as bases canônicas em frases da Linguagem Morfológica, não há resultados que garantam que esta reconstrução resulte num operador eficiente, seja numa implementação paralela ou sequencial. Por outro lado, os operadores canônicos mostraram-se, na grande maioria dos casos, mais eficientes que os operadores originalmente fornecidos e, portanto, o método descrito aqui mostrou-se útil para a simplificação de operadores morfológicos.

Entretanto, mostramos através de um contra-exemplo, que os operadores canônicos não contituem operadores ótimos quando implementados numa máquina sequencial.

Por fim, apesar de não demonstrarmos, conjecturamos que os operadores canônicos reconstruídos são, de fato, operadores ótimos quando implementados numa máquina paralela.

10.1 Perspectivas Futuras

Um resultado inédito que se pode buscar a partir deste trabalho é mostrar que, realmente, os operadores canônicos reconstruídos são ótimos se implementados numa máquina paralela.

Pode-se ainda tentar estender os resultados apresentados aqui para a classe de operadores sobre subconjuntos ou operadores sobre funções que sejam invariantes por translação não necessariamente isotônicos e localmente definidos em relação a uma janela W .

Apêndice A

O Estado da Arte em SRT's

Apresentamos, a seguir, um resumo em ordem cronológica dos principais trabalhos desenvolvidos sobre completção de sistemas de reescrita de termos, bem como alguns resultados conhecidos sobre a decidibilidade e indecidibilidade para o problema da palavra.

- 1951 T. Evans [Eva51]: Apresenta o primeiro algoritmo para a verificação automática de meta-teoremas da lógica de primeira ordem, da forma termo igual a termo. Isto é, Evans foi o primeiro a descrever um algoritmo para a solução do problema da palavra para teorias de primeira ordem. Lembre-se que, dado dois termos, digamos s e t , e um conjunto E de equações, o problema da palavra consiste em decidir se (ou não) $E \vdash s = t$. O algoritmo proposto por Evans consegue deduzir a validade de equações para uma classe muito restrita de teorias. Uma das limitações deste método é a incapacidade de lidar com a propriedade associativa.

- 1967 Donald E. Knuth e Peter B. Bendix [KB70]: O trabalho dos autores, realizado em 1967 e publicado apenas em 1970, consiste em um algoritmo para a solução do problema da palavra. Ao abordarem este problema, os autores introduzem o conceito de completção de sistemas de reescrita e mostram um algoritmo de completção cujo paradigma central é a verificação da convergência de pares críticos (o conceito de par crítico foi introduzido por estes autores).

Nesta abordagem, a possibilidade de verificação da identidade, através de um algoritmo, entre dois termos quaisquer de uma teoria de primeira ordem (um conjunto E de equações), materializa-se na existência de um sistema de reescrita R que satisfaça três condições:

- i)* o sistema de reescrita R deve induzir as mesmas classes de congruência que E ;
- ii)* R não deve admitir seqüência de reescrita infinita (R deve ter a propriedade noetheriana);

iii) todo par crítico de R deve ser convergente (R deve ter a propriedade de confluência).

O principal avanço deste método em relação ao proposto por Evans é a sua capacidade de lidar com a lei associativa. Entretanto, como observa Peterson e Stickel [PS81], o tratamento da associatividade dado por Knuth e Bendix não é totalmente geral (ao leitor mais interessado no assunto veja o contra-exemplo dado em [PS81, pg.236]).

Uma limitação deste método é a incapacidade de tratar símbolos funcionais não ordinários, como a comutatividade, pois neste caso não há relação de ordem que garanta que cada sequência de reescrita gerada pelo sistema de reescrita seja finita.

- 1977: D. S. Lankford e A. M. Ballantyne [LB77c]: Os autores foram os primeiros a apresentar uma alternativa para a completção de sistemas de reescrita que possuam operadores de natureza tipicamente não noetheriana, como é o caso dos operadores comutativos.

Os autores tiveram a idéia de dividir o conjunto E de equações em duas partes disjuntas, digamos E' e T , isto é, $E = E' \cup T$, onde T possui as equações de E que não são possíveis de ser orientadas como, por exemplo, uma equação que expresse a propriedade comutativa de um operador. Assim, o problema passa a ser o de encontrar um sistema de reescrita completo módulo T .

Uma equação $s = t$ é dita *permutativa* se s e t diferem apenas de uma permutação de seus símbolos funcionais ou variáveis. São exemplos de equações permutativas, as leis comutativa ($x + y = y + x$) e associativa ($x + (y + z) = (x + y) + z$). Quando um conjunto T é formado apenas por equações permutativas tem-se que o tamanho das classes de congruência de termos, induzidas por T , é finita, e há um algoritmo trivial para gerá-las. Lankford e Ballantyne conseguiram apresentar um algoritmo de completção módulo T que pode ser aplicado apenas quando T é formado por equações permutativas.

- 1980 G. Huet [Hue80]: Aborda o problema de completção módulo teoria equacional impondo a restrição que as regras de reescrita devam ser lineares à esquerda. Note que esta restrição impossibilita, por exemplo, a aplicação do método de Huet para a tentativa de completção da teoria dos reticulados distributivos e para a teoria dos grupos comutativos.

- 1981 G. E. Peterson e M. E. Stickel [PS81]: Generalizam os resultados obtidos por Lankford e Ballantyne [LB77c]. Os principais avanços obtidos foram:

- i) O método proposto por Lankford e Ballantyne não garante que o algoritmo de completção termine sua execução, mesmo quando um sistema de reescrita completo é obtido. O método de completção de Peterson e Stickel garante que o algoritmo de completção termina

sempre que um sistema de reescrita completo é obtido.

- ii) O método de completção de Lankford e Ballantyne permite que apenas equações permutativas possam compor a parte não orientável do conjunto de equações considerado, enquanto que o método proposto por Peterson e Stickel não há essa restrição.

Apesar de Peterson e Stickel não limitarem a aplicabilidade de seu método para classes de congruência geradas apenas por equações permutativas, outras restrições aparecem no lugar desta, restrições estas que se materializam através da noção de ‘compatibilidade com uma teoria equacional’. Se T é o conjunto de equações para o qual se quer obter um sistema de reescrita completo módulo T então, para que o método de completção destes autores possa ser aplicado, as duas condições devem ser satisfeitas:

- i) T deve ser linear;
 - ii) O sistema de reescrita do qual se quer verificar a propriedade de completeza deve ser compatível com T .
- 1986 J. Jouannaud e H. Kirchner [JK86]: Através da noção de coerência módulo T os autores conseguem desenvolver uma abordagem mais geral para o problema de completção módulo T que as abordagens dadas até então. Entretanto, este método de completção módulo T ainda restringe-se a conjuntos T de equações que induzem classes de congruência de cardinalidades finitas.
 - 1987 Leo Bachmair e N. Dershowitz [BD87]: Encontraram um método de completção módulo T que generaliza os métodos de completção anteriores. A única restrição imposta ao conjunto T é que haja um algoritmo finito e completo de unificação- T . Pela primeira vez, consegue-se um método de completção módulo um conjunto de equações que induza classes de congruência de cardinalidade infinita. Em particular, este método pode ser usado para tentar completar teorias equacionais que contenham a lei da identidade, $f(x, e) = x$, ou equipotência, $f(f(x)) = x$.
 - 1989 T. B. Baird, G. E. Peterson e R. W. Wilkerson [BPW89]: Apresentam um método de completção módulo teorias associativas-comutativas com identidade ($x + 0 = x$), costumeiramente chamadas por teorias- $AC1$, ou ainda, teorias- ACU . Apesar de as classes de congruência de uma teoria- ACU serem infinitas, o método de completção de Bachmair e Dershowitz não se aplica neste caso, pois a propriedade noetheriana módulo ACU não se verifica neste caso (condição esta exigida para aplicabilidade do método de Bachmair e Dershowitz). Os autores apresentam o seguinte exemplo que ilustra este fato: considere $R = \{-(x + y) \rightarrow (-x) + (-y)\}$ e $T =$

$\{x + y = y + x, x + (y + z) = (x + y) + z, x + 0 = x\}$, logo, T é uma teoria- ACU e a seguinte seqüência de reescrita é infinita:

$$-0 \stackrel{\overline{ACU}}{=} -(0 + 0) \xrightarrow{R} (-0) + (-0) \stackrel{\overline{ACU}}{=} -(0 + 0) + (-0) \xrightarrow{R} (-0) + (-0) + (-0) \dots$$

Os autores mostram ser impossível tentar completar um sistema de reescrita módulo ACU sem que restrições sejam impostas sobre as substituições que são aplicadas aos termos a serem reescritos. Métodos como este, que impõem restrições sobre quando uma regra de reescrita pode ou não ser utilizada para reescrever um termo são, costumeiramente chamados nos textos em inglês por “*rewriting via constraints*”.

- 1991 C. Kirchner, H. Kirchner e M. Rusinowitch [KKR91]: É apresentada uma abordagem bastante geral sobre rewriting via constraints.

- 1993-1996 C. Marché [Mar93, Mar96a]: Em sua tese de doutorado [Mar93], publicada em 1993, C. Marché apresenta uma nova técnica de reescrita que batizou de “*normalized rewriting*” como alternativa e generalização à abordagem de reescrita via constraints. O autor observa que, apesar dos métodos de completção baseados em rewriting via constraints serem úteis para a abordagem da completção módulo ACU , estes métodos não conseguem lidar com outras teorias como as associativas-comutativas com idempotência ($x + x = x$), costumeiramente abreviadas por ACI , ou ainda, deixam de ser úteis para grupos comutativos, costumeiramente abreviados por AG .

Seja R um sistema de reescrita, $l \rightarrow r$ uma regra de reescrita em R e S uma teoria- ACI , então a seguinte seqüência infinita é possível:

$$l \stackrel{\overline{ACI}}{=} l + l \xrightarrow{R} l + r \stackrel{\overline{ACI}}{=} l + l + r \dots$$

Se S for uma teoria- AG então:

$$0 \stackrel{\overline{AG}}{=} l + (-l) \xrightarrow{R} r + (-l) \stackrel{\overline{AG}}{=} l + (-l) + r + (-l) \dots$$

Dado um conjunto E de equações para o qual se quer obter um sistema de reescrita completo, a idéia principal do trabalho de C. Marché consiste em dividir E em duas partes disjuntas, digamos E' e S , $E = E' \cup S$, de modo que os teoremas de E' possam ser calculados através de um sistema de reescrita completo módulo AC . A idéia principal deste método é: sempre que um termo possa ser reescrito deve-se, primeiro, obter sua forma canônica em relação a E' . O autor mostra que esta nova técnica de reescrita generaliza o trabalho de reescrita via constraints de Baird, Peterson e Wilkerson [BPW89], bem como permite a reescrita de termos módulo idempotência, teoria dos grupos, entre outras. Em 1996 é publicado em forma de artigo [Mar96a] seus resultados obtidos no doutorado.

- 1996 C. Marché [Mar96b]: Mostra que o algoritmo de completção de sistemas de reescrita de Knuth e Bendix e o algoritmo de Buchberger [Buc65] (1965) que calcula a *base de Gröbner*

de ideais de polinômios constituem casos particulares de seu algoritmo de completção, “*S-normalized completion*”, baseado em normalized rewriting.

A indecidibilidade do problema da palavra para teorias fechadas (do inglês: “*ground theories*”) módulo associatividade foi demonstrada independentemente por Post [Pos47] e Markov [Mar47] em 1947, a demonstração da indecidibilidade no caso da teoria dos grupos é de Novikov [Nov55] em 1955. Mais recentemente, em 1992, Marché [Mar92] mostrou que o problema da palavra para teorias fechadas módulo ACD é indecidível. Em 1990 a indecidibilidade do problema de unificação módulo ACD é demonstrada por C. Kirchner [Kir90]. A decidibilidade do problema da palavra para teorias fechadas módulo AC é dada por Marché [Mar91] em 1991.

Como consequência do método “normalized rewriting”, desenvolvido por Marché em sua tese de doutorado [Mar93] em 1993, o autor mostra a propriedade de decidibilidade do problema da palavra para teorias fechadas módulo ACU , ACI , $ACUI$, AG , anéis comutativos e anéis booleanos. O diagrama abaixo foi obtido em [Mar96a].

Figura A.1: Decidibilidade do problema da palavra de teorias fechadas módulo E , para alguns E 's

Apêndice B

Passos da Completação do SRT para a Teoria dos Reticulados Distributivos

A seguir vamos comentar sucintamente a completção do sistema de reescrita para os reticulados distributivos, apresentada no exemplo 7.51. O algoritmo 1.4 de completção módulo T recebeu como dados de entrada os seguintes conjuntos E e T de equações:

$$E = \{e_1 : x \cup (x \cap y) = x, \\ e_2 : x \cap (x \cup y) = x, \\ e_3 : x \cap (y \cup z) = (x \cap y) \cup (x \cap z) \\ \}$$

$$T = \{t_1 : x \cup y = y \cup x, \\ t_2 : x \cap y = y \cap x, \\ t_3 : x \cup (y \cup z) = (x \cup y) \cup z, \\ t_4 : x \cap (y \cap z) = (x \cap y) \cap z \\ \}$$

Observe que as equações $x \cup x = x$ e $x \cap x = x$, que expressam leis de idempotência, bem como a equação $x \cup (y \cap z) = (x \cup y) \cap (x \cup z)$, são teoremas da teoria equacional induzida por $E \cup T$. Para convencer-se disto note que $x \stackrel{e_2}{=} x \cap (x \cup y) \stackrel{e_3}{=} (x \cap x) \cup (x \cap y)$. Portanto,

$$\boxed{(x \cap x) \cup (x \cap y) = x}^1.$$

$$x \stackrel{=}{\underset{t_1}{(x \cap x) \cup (x \cap (x \cap y))}} \stackrel{=}{\underset{t_4}{(x \cap x) \cup ((x \cap x) \cap y)}} \stackrel{=}{\underset{e_1}{x \cap x}}.$$

Portanto, acabamos de mostrar que $E \cup T \vdash x = x \cap x$ ².

Analogamente, mostra-se que $E \cup T \vdash x = x \cup x$ ³.

Vamos mostrar agora que, de fato, $x \cup (y \cap z) = (x \cup y) \cap (x \cup z)$ é teorema de $E \cup T$.

$$\begin{aligned} (x \cup y) \cap (x \cup z) &\stackrel{=}{\underset{e_3}{((x \cup y) \cap x) \cup ((x \cup y) \cap z)}} \stackrel{=}{\underset{t_2}{\circ}} \stackrel{=}{\underset{t_2}{(x \cap (x \cup y)) \cup (z \cap (x \cup y))}} \stackrel{=}{\underset{e_2}{((x \cap x) \cup (x \cap y)) \cup (z \cap (x \cup y))}} \\ &\stackrel{=}{\underset{2}{(x \cup (x \cap y)) \cup (z \cap (x \cup y))}} \stackrel{=}{\underset{e_1}{x \cup (z \cap (x \cup y))}} \stackrel{=}{\underset{e_3}{x \cup ((z \cap x) \cup (z \cap y))}} \stackrel{=}{\underset{t_3}{(x \cup (z \cap x)) \cup (z \cap y)}} \\ &\stackrel{=}{\underset{t_2}{(x \cup (x \cap z)) \cup (z \cap y)}} \stackrel{=}{\underset{e_1}{x \cup (z \cap y)}} \stackrel{=}{\underset{t_2}{x \cup (y \cap z)}}. \end{aligned}$$

Portanto, acabamos de mostrar que $E \cup T \vdash x \cup (y \cap z) = (x \cup y) \cap (x \cup z)$.

A seguir, vamos comentar os passos principais que ocorrem durante o processo de completção módulo T do conjunto E de equações. Daqui por diante escrevemos $t \downarrow$ para denotarmos que t é um termo irredutível em relação ao sistema de reescrita R .

- A remoção, do sistema de reescrita R , da regra (2) $x \cap (x \cup y) \rightarrow x$ e, a inserção em R , da regra (4) $(x \cap x) \cup (x \cap y) \rightarrow x$:

Como consequência da inserção em R da regra (3), a saber $x \cap (y \cup z) \rightarrow (x \cap y) \cup (x \cap z)$, tem-se que a regra (2), já existente em R , pode ser reescrita por (3), pois $x \cap (x \cup y) \xrightarrow{(3)} (x \cap x) \cup (x \cap y)$ e consequentemente (2) é removida de R pois ela não está na forma interreduzida. No lugar de (2) insere-se a regra (4) dada por $(x \cap x) \cup (x \cap y) \rightarrow x$ pois:

Sendo as complexidades dos termos irredutíveis, apresentados pelo diagrama acima, dadas por $\|x\| = 2 > 13 = 2 \cdot 2 + 2 \cdot 2 + 5 = \|(x \cap x) \cup (x \cap y)\|$; isto justifica o fato da regra (4) ter sido “orientada” do modo visto acima.

- Inserção da regra (5) por (1) e (4):

Neste ponto temos,

$$R = \{ \begin{array}{l} (1) x \cup (x \cap y) \rightarrow x, \\ (3) x \cap (y \cup z) \rightarrow (x \cap y) \cup (x \cap z), \\ (4) (x \cap x) \cup (x \cap y) \rightarrow x \end{array} \}$$

As extensões simples das regras (1) e (4) são dadas respectivamente por:

$$\begin{aligned} (x \cup (x \cap y) \rightarrow x)^e &\stackrel{\text{def.}}{=} z \cup (x \cup (x \cap y)) \rightarrow (z \cup x)^*, \text{ mas } (z \cup x)^* = (z \cup x), \text{ logo} \\ (x \cup (x \cap y) \rightarrow x)^e &= z \cup (x \cup (x \cap y)) \rightarrow z \cup x, \\ ((x \cap x) \cup (x \cap y) \rightarrow x)^e &\stackrel{\text{def.}}{=} z \cup ((x \cap x) \cup (x \cap y)) \rightarrow (z \cup x)^*, \text{ mas } (z \cup x)^* = (z \cup x), \text{ logo} \\ ((x \cap x) \cup (x \cap y) \rightarrow x)^e &= z \cup ((x \cap x) \cup (x \cap y)) \rightarrow z \cup x. \end{aligned}$$

Precisamos considerar as possibilidades de superposições de termos para cada um dos pares de regras do conjunto abaixo:

$$\{ \begin{array}{ll} < x \cup (x \cap y) \rightarrow x, (x \cap x) \cup (x \cap y) \rightarrow x >, \\ < x \cup (x \cap y) \rightarrow x, z \cup ((x \cap x) \cup (x \cap y)) \rightarrow z \cup x >, \\ < z \cup (x \cup (x \cap y)) \rightarrow z \cup x, (x \cap x) \cup (x \cap y) \rightarrow x >, \\ < z \cup (x \cup (x \cap y)) \rightarrow z \cup x, z \cup ((x \cap x) \cup (x \cap y)) \rightarrow z \cup x > \end{array} \}.$$

Vamos fazer a análise das superposições possíveis dos lados esquerdos das regras do primeiro par de regras de reescrita do conjunto acima, isto é, o par $< x \cup (x \cap y) \rightarrow x, (x \cap x) \cup (x \cap y) \rightarrow x >$.

Vamos então tentar superpor o termo $s = x \cup (x \cap y)$ sobre o termo $t = (x \cap x) \cup (x \cap y)$, isto é, queremos verificar se existe alguma posição p em $sdom\ s$ tal que s/p e t são unificáveis-AC, em outros termos, queremos verificar se existe alguma substituição θ e alguma posição p em $sdom\ s$ tal que $s/p\theta \stackrel{AC}{=} t\Phi\theta$, onde Φ é uma permutação.

É fácil ver que isto não é possível, pois os únicos subtermos de s que não correspondem a alguma variável (pois $s/p \notin Var(s)$) são $s/<> = s$ e $s/<2> = x \cap y$.

Acabamos de ver que não é possível superpor s sobre t , entretanto pode-se superpor t sobre s , isto é, existe substituição θ e posição p em $sdom\ t$ tal que $t/p\theta \stackrel{AC}{=} s\Phi\theta$, onde Φ é uma permutação. Para isto, tomemos $\Phi = \{x \leftarrow z, y \leftarrow w\}$, $p = <>$ e $\theta = \{z \leftarrow z_1 \cap z_1, w \leftarrow z_3, x \leftarrow z_1, y \leftarrow z_1 \cap z_3\}$.

Sendo assim, $z_1 \cap z_1$ e z_1 são irreduzíveis em relação ao sistema de reescrita R formado pelas regras (1), (3) e (4). $\|z_1 \cap z_1\| = 2 \cdot 2 = 4 > 2 = \|z_1\|$ e portanto a regra $z_1 \cap z_1 \rightarrow z_1$ (a menos de uma permutação de suas variáveis) é inserida no sistema de reescrita que está sendo completado, e portanto passamos a ter $R = R \cup \{x \cap x \rightarrow x\}$.

$$s\Phi\theta = (z_1 \cap z_1) \cup ((z_1 \cap z_1) \cap z_3) \stackrel{AC}{=} (z_1 \cap z_1) \cup (z_1 \cap (z_1 \cap z_3)) = t\theta$$

Pode-se verificar que, os demais pares de regras de reescrita dados no conjunto acima não resultam em superposições que geram pares críticos divergentes, e portanto não geram novas regras de reescrita a serem inseridas em R .

- A remoção da regra (4):

Neste ponto temos,

$$R = \{ \begin{array}{l} (1) \ x \cup (x \cap y) \rightarrow x, \\ (3) \ x \cap (y \cup z) \rightarrow (x \cap y) \cup (x \cap z), \\ (4) \ (x \cap x) \cup (x \cap y) \rightarrow x, \\ (5) \ x \cap x \rightarrow x \end{array} \}$$

Por decorrência da inserção da regra (5) no sistema de reescrita R , a regra (4) deixa de possuir a forma interreduzida, e por isto é eliminada. Neste caso não há regra de reescrita para substituir a regra (4), pois a forma interreduzida de (4) não gera par crítico divergente como mostra o próximo diagrama:

- Inserção da regra (6):

A regra (6), a saber, $x \cup x \rightarrow x$, é gerada pela superposição dos lados esquerdos das regras (1) e (5).

Note que $s = x \cup (x \cap y)$ pode ser superposto sobre $t = x \cap x$, isto é, existe substituição θ e posição p em $sdom\ s$ tais que $(x \cup (x \cap y))/p\theta \stackrel{AC}{=} (x \cap x)\Phi\theta$, onde Φ é uma permutação.

É fácil ver que a única posição p em $sdom\ s$ é $p = \langle 2 \rangle$ e portanto $s/p = x \cap y$.

Tomemos $\Phi = \{x \leftarrow x_1, y \leftarrow x_2\}$ e $\Phi' = \{x \leftarrow y_1\}$ e vamos tentar unificar $s/p\Phi = x_1 \cap x_2$ com $y_1 \cap y_1 = t\Phi'$.

A título de ilustração, a análise desta unificação será feita com maiores detalhes. Precisamos, primeiro, calcular a base de soluções não negativas da equação diofantina homogênea linear $\boxed{x_1 + x_2 = 2 \cdot y_1}$ (*).

Neste caso temos: $m = 2, n = 1, a_1 = a_2 = 1, b_1 = 2, \max a = 1, \max b = 2, d_{11} = d_{21} = 2$ e $e_{11} = e_{21} = 1$.

◇ $k = 1$

$$x = \begin{array}{cc} & 2 & 1 \\ \boxed{0} & \boxed{1} \\ & \uparrow & \\ & k & \end{array}$$

$$E_1^{k=1} = \emptyset.$$

$$\max y_1^{k=1}(x_1) = 1.$$

$$E_1^{m=2} = \emptyset.$$

$$\max y_1^{m=2}(x_1, x_2) = 1.$$

$$(a) \ x_1 = 1 < 2 = \max b \quad \text{ok!}$$

$$(b) \ \sum_{i=1}^k a_i \cdot x_i = 1 < \sum_{j=1}^n b_j \cdot \max y_j^k(x_1, \dots, x_k) = \sum_{j=1}^n b_j \cdot \max y_j^{k=1}(x_1) = 2 \quad \text{ok!}$$

◇ $l = 1$

$$y = \begin{array}{c} 1 \\ \boxed{1} \\ \uparrow \\ l \end{array}$$

$$(c) \ y_1 = 1 < 2 = \max y_1^m(x_1, \dots, x_m) = \max y_1^2(x_1, x_2) \quad \text{ok!}$$

$$(d) \ \sum_{j=1}^l b_j \cdot y_j = b_1 \cdot y_1 = 1 = \sum_{i=1}^m a_i \cdot x_i = a_1 \cdot x_1 + a_2 \cdot x_2 \quad \text{ok!}$$

mas (x, y) não é solução de $(*)$!

$$y = \begin{array}{c} 1 \\ \boxed{2} \\ \uparrow \\ l \end{array}$$

(c) $y_1 = 2 > \max y_1^m(x_1, \dots, x_m) = \max y_1^2(x_1, x_2) = 1$ (portanto não satisfaz o teorema 4.18)!

$$x = \begin{array}{cc} 2 & 1 \\ \boxed{0} & \boxed{2} \\ & \uparrow \\ & k \end{array}$$

$$E_1^{k=1} = e_{11} = \{1\}.$$

$$\max y_1^{k=1}(x_1) = 1.$$

$$E_1^{m=2} = \{1\}.$$

$$\max y_1^{m=2}(x_1, x_2) = 1.$$

$$(a) x_1 = 2 = \max b \quad \text{ok!}$$

$$(b) \sum_{i=1}^k a_i \cdot x_i = a_1 \cdot x_1 = 1 \cdot 2 = 2 = \sum_{j=1}^n b_j \cdot \max y_j^k(x_1, \dots, x_k) = \sum_{j=1}^n b_j \cdot \max y_j^{k=1}(x_1) \quad \text{ok!}$$

◇ $l = 1$

$$y = \begin{array}{c} 1 \\ \boxed{1} \\ \uparrow \\ l \end{array}$$

$$(c) y_1 = 1 = \max y_1^m(x_1, \dots, x_m) = \max y_1^2(x_1, x_2) \quad \text{ok!}$$

$$(d) \sum_{j=1}^l b_j \cdot y_j = b_1 \cdot y_1 = 2 \cdot 1 = 2 = \sum_{i=1}^m a_i \cdot x_i = a_1 \cdot x_1 + a_2 \cdot x_2 = 1 \cdot 2 + 1 \cdot 0 \quad \text{ok!}$$

(x, y) é solução de $(*)$ e ainda $(x, y) = S_{11}$ pois $x_{11} = d_{11} = 2, y_1 = e_{11} = 1$ e $x_2 = 0$.

Base:

x_2	x_1	y_1	
0	2	1	$\leftarrow S_{11}$

⋮

continuando a simulação do algoritmo de Stickel [Sti81], visto no capítulo 4, obtemos a seguinte Base das soluções não negativas da equação diofantina homogênea linear (*):

Número da solução	x_1	x_2	y_1		$x_1 + x_2$	$2 \cdot y_1$	
1	0	2	1	$\leftarrow S_{11}$	2	2	z_1
2	1	1	1		2	2	z_2
3	2	0	1	$\leftarrow S_{21}$	2	2	z_3

Para a base das soluções não negativas da equação (*), apresentada pela tabela acima, há um total de $2^3 = 8$ formas diferentes de compor as 3 soluções parciais apresentadas, de modo que cada solução da base apareça no máximo uma única vez em cada composição, são elas: $\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}$.

Dentre estas 8 soluções há 5 delas, tais que, todas as variáveis (x_1, x_2, y_1) possuem pelo menos uma atribuição de algum termo em seu valor, são elas: $\{2\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}$.

Abaixo apresentamos as 5 unificadoras-AC mais gerais de $s/p\Phi = x_1 \cap x_2$ com $t\Phi' = y_1 \cap y_1$ em relação a $W = \{x_1, x_2, y_1\}$.

$$\begin{aligned}
 \sigma_1 &= \{ \quad x_1 \leftarrow z_2 \quad \quad \quad , \quad x_2 \leftarrow z_2 \quad \quad \quad , \quad y_1 \leftarrow z_2 \}. \\
 \sigma_2 &= \{ \quad x_1 \leftarrow \cap(z_1 z_1 z_2) \quad , \quad x_2 \leftarrow z_2 \quad \quad \quad , \quad y_1 \leftarrow \cap(z_1 z_2) \}. \\
 \sigma_3 &= \{ \quad x_1 \leftarrow \cap(z_1 z_1) \quad \quad \quad , \quad x_2 \leftarrow \cap(z_3 z_3) \quad \quad \quad , \quad y_1 \leftarrow \cap(z_1 z_3) \}. \\
 \sigma_4 &= \{ \quad x_1 \leftarrow z_2 \quad \quad \quad , \quad x_2 \leftarrow \cap(z_2 z_3 z_3) \quad , \quad y_1 \leftarrow \cap(z_2 z_3) \}. \\
 \sigma_5 &= \{ \quad x_1 \leftarrow \cap(z_1 z_1 z_2) \quad , \quad x_2 \leftarrow \cap(z_2 z_3 z_3) \quad , \quad y_1 \leftarrow \cap(z_1 z_2 z_3) \}.
 \end{aligned}$$

e portanto,

$$\begin{aligned}
 s/p\Phi\sigma_1 &= z_2 \cap z_2 = t\Phi'\sigma_1. \\
 s/p\Phi\sigma_2 &= (z_1 \cap z_1 \cap z_2) \cap z_2 \stackrel{AC}{=} (z_1 \cap z_2) \cap (z_1 \cap z_2) = t\Phi'\sigma_2. \\
 s/p\Phi\sigma_3 &= (z_1 \cap z_1) \cap (z_3 \cap z_3) \stackrel{AC}{=} (z_1 \cap z_3) \cap (z_1 \cap z_3) = t\Phi'\sigma_3. \\
 s/p\Phi\sigma_4 &= z_2 \cap (z_2 \cap z_3 \cap z_3) \stackrel{AC}{=} (z_2 \cap z_3) \cap (z_2 \cap z_3) = t\Phi'\sigma_4. \\
 s/p\Phi\sigma_5 &= (z_1 \cap z_1 \cap z_2) \cap (z_2 \cap z_3 \cap z_3) \stackrel{AC}{=} (z_1 \cap z_2 \cap z_3) \cap (z_1 \cap z_2 \cap z_3) = t\Phi'\sigma_5.
 \end{aligned}$$

Os pares críticos resultantes da unificação dos lados esquerdos das regras (1) e (5), isto é,

respectivamente s/p e t , são adicionados como equações em E (passo 24 do algoritmo 1.4) para que sejam verificadas se estas novas equações geram ou não novos pares críticos.

$$E = \left\{ \begin{array}{ll} z_2 & = z_2 \cup z_2, \\ (z_1 \cap z_1 \cap z_2) & = (z_1 \cap z_1 \cap z_2) \cup (z_1 \cap z_2), \\ (z_1 \cap z_1) & = (z_1 \cap z_1) \cup (z_1 \cap z_3), \\ z_2 & = z_2 \cup (z_2 \cap z_3), \\ (z_1 \cap z_1 \cap z_2) & = (z_1 \cap z_1 \cap z_2) \cup (z_1 \cap z_2 \cap z_3) \end{array} \right\}$$

A equação $z_2 = z_2 \cup z_2$ é retirada do conjunto E de equações, $E = E - \{z_2 = z_2 \cup z_2\}$.

$R(z_2) = z_2$ e $R(z_2 \cup z_2) = z_2 \cup z_2$.

$\| z_2 \cup z_2 \| = 2 + 2 + 5 = 9 > 2 = \| z_2 \|$.

Portanto a regra $z_2 \cup z_2 \rightarrow z_2$ (a menos de uma permutação em suas variáveis) é adicionada ao conjunto R de regras de reescritas.

E temos,

$$R = \left\{ \begin{array}{l} (1) \ x \cup (x \cap y) \rightarrow x, \\ (3) \ x \cap (y \cup z) \rightarrow (x \cap y) \cup (x \cap z), \\ (4) \ (x \cap x) \cup (x \cap y) \rightarrow x, \\ (5) \ x \cap x \rightarrow x, \\ (6) \ x \cup x \rightarrow x \end{array} \right\}$$

A equação $z_2 = z_2 \cup (z_2 \cap z_3)$ é retirada do conjunto E , $E = E - \{z_2 = z_2 \cup (z_2 \cap z_3)\}$, entretanto ela não é inserida como uma nova regra de reescrita em R , pois as formas normais dos termos que a compõe não geram par crítico, como mostra o diagrama a seguir:

A equação $(z_1 \cap z_1) = (z_1 \cap z_1) \cup (z_1 \cap z_3)$ é retirada do conjunto E , $E = E - \{(z_1 \cap z_1) = (z_1 \cap z_1) \cup (z_1 \cap z_3)\}$, entretanto ela não é inserida como uma nova regra de reescrita em R , pois as formas normais dos termos que a compõe não geram par crítico, como mostra o diagrama a seguir:

$$(z_1 \cap z_1) = (z_1 \cap z_1) \cup (z_1 \cap z_3)$$

A equação $(z_1 \cap z_1 \cap z_2) = (z_1 \cap z_1 \cap z_2) \cup (z_1 \cap z_2)$ é retirada de E , $E = E - \{(z_1 \cap z_1 \cap z_2) = (z_1 \cap z_1 \cap z_2) \cup (z_1 \cap z_2)\}$, mas esta equação não é inserida na forma de uma regra de reescrita em R , pois as formas normais dos termos que a compõe não geram par crítico, como mostra o diagrama a seguir:

$$(z_1 \cap z_1 \cap z_2) = (z_1 \cap z_1 \cap z_2) \cup (z_1 \cap z_2)$$

O algoritmo 1.4 de completção módulo- AC continua sua execução enquanto houver elementos em *Pairs*; entretanto neste nosso caso, nenhuma outra regra de reescrita é adicionada ao sistema de reescrita R .

Apêndice C

Algumas Estatísticas

Por fim, apresentamos algumas estatísticas básicas sobre os exemplos 9.34, 9.35, 9.36, 9.37 e 9.38. Para o cálculo de cada termo canônico, dos exemplos citados acima, apresentamos a quantidade de tentativas de matching-AC, realizadas pelo sistema CiME e o tempo total de máquina necessário para o cálculo dos termos canônicos. Como a interface do sistema de dedução usado (CiME system) para os cálculos das formas normais não admite caracteres especiais, foi necessário uma renomeação de alguns símbolos que compõem os termos. O símbolo ι foi substituído pelo símbolo ‘i’ e, analogamente, $\vee$ foi substituído por ‘+’, $\wedge$ por ‘*’ e $\circ$ por ‘o’.

Como o sistema CiME foi utilizado, nesta dissertação, apenas para reescrever termos para suas respectivas formas canônicas e não para completar sistemas de reescrita de termos, então só nos interessa as estatísticas fornecidas pelo sistema referentes a quantidade de chamadas do algoritmo de matching-AC bem como as informações sobre o tempo de máquina necessário para a obtenção dos termos canônicos. Sendo assim, as informações sobre quantidade de chamadas ao algoritmo de unificação-AC, a quantidade de pares críticos produzidos e a quantidade de regras de reescrita geradas serão omitidos aqui, uma vez que, em nosso caso, estes valores são todos iguais a zero, pois só seriam diferente disso caso o sistema CiME estivesse sendo usado para a completção de sistemas de reescrita, e não é nesse sentido que o estamos utilizando.

C.1 Exemplo 9.34

- Estatísticas fornecidas pelo sistema CiME sobre o exemplo 9.34:

- Estatísticas para o operador $\psi = (\delta_C \wedge \epsilon_A) \vee (\epsilon_A \wedge \iota)$:

(estatísticas sobre o cálculo de $R_1(\psi)$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $((i + ((b \circ b) \circ e)) * b * i * c * e * (e \circ e)) + (b * i * c * e * (e \circ e) * i)$

is

$((i + (b \circ (b \circ e))) * b * i * c * e * (e \circ e)) + (b * i * c * e * (e \circ e) * i)$

(by 1 reductions)

Number of calls to AC matching : 55

Number of successful calls : 1 (1%)

times	user	system	total
Total time	0.060	0.280	0.340
Matching times	0.000	0.010	0.010

(estatísticas sobre o cálculo de $R_2(R_1(\psi))$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $(((((i + (b \circ b \circ e)) * b) * i) * c) * e * (e \circ e)) + (((((b * i) * c) * e) * (e \circ e)) * i)$ is

$(((((i + (b \circ b \circ e)) * b) * i) * c) * e * (e \circ e)) + (((((b * i) * c) * e) * (e \circ e)) * i)$

(by 0 reductions)

Number of calls to AC matching : 36

Number of successful calls : 0 (0%)

times	user	system	total
Total time	0.080	0.110	0.190
Matching times	0.010	0.000	0.010

(estatísticas sobre o cálculo de $\psi \downarrow \stackrel{\text{def.}}{=} R_3(R_2(R_1(\psi)))$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $((A+D)*B*A*C*E*F)+(B*A*C*E*F*A)$ is
 $F*E*C*B*A$

(by 4 reductions)

Number of calls to AC matching : 23

Number of successful calls : 4 (17%)

times	user	system	total
Total time	0.040	0.040	0.080
Matching times	0.020	0.000	0.020

(estadísticas sobre o cálculo de $R_4(\psi\downarrow)$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $F*E*C*B*A$ is

$F*E*C*B*A$

(by 0 reductions)

Number of calls to AC matching : 0

Number of successful calls : 0

times	user	system	total
Total time	0.010	0.030	0.040
Matching times	0.000	0.000	0.000

o Estatísticas para o operador $\psi' = ((\epsilon_A \wedge \epsilon_B) \wedge \delta_D) \vee ((\epsilon_B \wedge \epsilon_A) \wedge \iota)$:

(estadísticas sobre o cálculo de $R_1(\psi')$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $(b*i*c*e*(e \circ e)*i*c*e*((b \circ b) \circ e))+(i*c*e*b*i*c*e*(e \circ e)*i)$ is
 $(b*i*c*e*(e \circ e)*i*c*e*(b \circ (b \circ e)))+(i*c*e*b*i*c*e*(e \circ e)*i)$

(by 1 reductions)

Number of calls to AC matching : 68
 Number of successful calls : 1 (1%)

times	user	system	total
Total time	0.080	0.160	0.240
Matching times	0.010	0.030	0.040

(estatísticas sobre o cálculo de $R_2(R_1(\psi'))$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $((((((((b*i)*c)*e)*(e\ o\ e))*i)*c)*e)*(b\ o\ b\ o\ e))+(((((((i*c)*e)*b)*i)*c)*e)*(e\ o\ e))*i)$ is

$((((((((b*i)*c)*e)*(e\ o\ e))*i)*c)*e)*(b\ o\ b\ o\ e))+(((((((i*c)*e)*b)*i)*c)*e)*(e\ o\ e))*i)$

(by 0 reductions)

Number of calls to AC matching : 36

Number of successful calls : 0 (0%)

times	user	system	total
Total time	0.060	0.170	0.230
Matching times	0.000	0.010	0.010

(estatísticas sobre o cálculo de $\psi' \downarrow^{\text{def.}} R_3(R_2(R_1(\psi')))$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $(B*A*C*E*F*A*C*E*D)+(A*C*E*B*A*C*E*F*A)$ is $F*E*C*B*A$

(by 8 reductions)

Number of calls to AC matching : 65

Number of successful calls : 8 (12%)

times	user	system	total
Total time	0.030	0.070	0.100
Matching times	0.020	0.020	0.040

(estatísticas sobre o cálculo de $R_4(\psi' \downarrow)$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $F^*E^*C^*B^*A$ is

$F^*E^*C^*B^*A$

(by 0 reductions)

Number of calls to AC matching : 0

Number of successful calls : 0

times	user	system	total
Total time	0.030	0.040	0.070
Matching times	0.000	0.000	0.000

C.2 Exemplo 9.35

- Estatísticas fornecidas pelo sistema CiME sobre o exemplo 9.35:

- Estatísticas para o operador $\psi = \delta_A \wedge ((\delta_B \circ \epsilon_A) \vee (\delta_C \circ \epsilon_C))$:

(estatísticas sobre o cálculo de $R_1(\psi)$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $((b \circ e) \circ e)^*(((c \circ c) + (((((b \circ b) \circ e) \circ e) \circ e) \circ e)) \circ ((d \circ d) \circ c)) + (b \circ c))$ is

$(b \circ (e \circ e))^*((b \circ (b \circ (e \circ (e \circ (e \circ (e \circ (d \circ (d \circ c)))))))) + (c \circ (c \circ (d \circ (d \circ c)))) + (b \circ c))$

(by 10 reductions)

Number of calls to AC matching : 769

Number of successful calls : 10 (1%)

times	user	system	total
Total time	0.180	0.110	0.290
Matching times	0.130	0.030	0.160

(estatísticas sobre o cálculo de $R_2(R_1(\psi))$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $(b o e o e)*(((b o b o e o e o e o d o d o c)+(c o c o d o d o c))+(b o c))$ is

$(b o e o e)*(((e o e o b)+(c o c o d o d o c))+i)$

(by 7 reductions)

Number of calls to AC matching : 173

Number of successful calls : 7 (4%)

times	user	system	total
Total time	0.040	0.090	0.130
Matching times	0.040	0.030	0.070

(estatísticas sobre o cálculo de $\psi \downarrow \stackrel{\text{def.}}{=} R_3(R_2(R_1(\psi)))$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $A*(A+B+C)$ is

A

(by 5 reductions)

Number of calls to AC matching : 21

Number of successful calls : 5 (23%)

times	user	system	total
Total time	0.010	0.060	0.070
Matching times	0.000	0.000	0.000

(estatísticas sobre o cálculo de $R_4(\psi \downarrow)$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of A is

A

(by 0 reductions)

Number of calls to AC matching : 0

Number of successful calls : 0

times	user	system	total
Total time	0.010	0.040	0.050
Matching times	0.000	0.000	0.000

C.3 Exemplo 9.36

- Estatísticas fornecidas pelo sistema CiME sobre o exemplo 9.36:

- Estatísticas para o operador $\psi = \epsilon \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \circ \delta_{[101]}$:

(estatísticas sobre o cálculo de $R_1(\psi)$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $(b*c) \circ (e+d)$ is $((c \circ e) + (c \circ d)) * ((b \circ e) + (b \circ d))$

(by 3 reductions)

Number of calls to AC matching : 93

Number of successful calls : 3 (3%)

times	user	system	total
Total time	0.030	0.130	0.160
Matching times	0.000	0.020	0.020

(estatísticas sobre o cálculo de $R_2(R_1(\psi))$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $((c \circ e) + (c \circ d)) * ((b \circ e) + (b \circ d))$ is $((c \circ e) + (c \circ d)) * ((b \circ e) + (b \circ d))$

(by 0 reductions)

Number of calls to AC matching : 48

Number of successful calls : 0 (0%)

times	user	system	total
Total time	0.050	0.090	0.140
Matching times	0.010	0.000	0.010

(estatísticas sobre o cálculo de $\psi \downarrow \stackrel{\text{def.}}{=} R_3(R_2(R_1(\psi)))$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $(A+B)*(C+D)$ is

$(C*A)+(C*B)+(D*A)+(D*B)$

(by 3 reductions)

Number of calls to AC matching : 33

Number of successful calls : 3 (9%)

times	user	system	total
Total time	0.030	0.030	0.060
Matching times	0.020	0.000	0.020

(estatísticas sobre o cálculo de $R_4(\psi \downarrow)$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $(C*A)+(C*B)+(D*A)+(D*B)$ is

$(B+A)*(C+D)$

(by 3 reductions)

Number of calls to AC matching : 9

Number of successful calls : 3 (33%)

times	user	system	total
Total time	0.030	0.110	0.140
Matching times	0.010	0.000	0.010

C.4 Exemplo 9.37

- Estatísticas fornecidas pelo sistema CiME sobre o exemplo 9.37:

◦ Estatísticas para o operador $\psi = \delta_A \circ (\epsilon_A \circ \delta_A)$:

(estatísticas sobre o cálculo de $R_1(\psi)$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $((((c \circ e) + (c \circ d) + (b \circ e)) \circ ((b \circ d) * (b \circ e) * (c \circ d))) \circ ((c \circ e) + (c \circ d) + (b \circ e)))$ is

$((((c \circ (e \circ (c \circ (d \circ (c \circ e)))) + (c \circ (e \circ (c \circ (d \circ (c \circ d)))) + (c \circ (e \circ (c \circ (d \circ (b \circ e)))))) * ((c \circ (e \circ (b \circ (e \circ (c \circ e)))) + (c \circ (e \circ (b \circ (e \circ (c \circ d)))) + (c \circ (e \circ (b \circ (e \circ (b \circ e)))))) * ((c \circ (e \circ (b \circ (d \circ (c \circ e)))) + (c \circ (e \circ (b \circ (d \circ (c \circ d)))) + (c \circ (e \circ (b \circ (d \circ (b \circ e)))))) + (((c \circ (d \circ (c \circ (d \circ (c \circ e)))) + (c \circ (d \circ (c \circ (d \circ (c \circ d)))) + (c \circ (d \circ (c \circ (d \circ (b \circ e)))))) * ((c \circ (d \circ (b \circ (e \circ (c \circ e)))) + (c \circ (d \circ (b \circ (e \circ (c \circ d)))) + (c \circ (d \circ (b \circ (e \circ (b \circ e)))))) * ((c \circ (d \circ (b \circ (d \circ (c \circ e)))) + (c \circ (d \circ (b \circ (d \circ (c \circ d)))) + (c \circ (d \circ (b \circ (d \circ (b \circ e)))))) + (((b \circ (e \circ (c \circ (d \circ (c \circ e)))) + (b \circ (e \circ (c \circ (d \circ (c \circ d)))) + (b \circ (e \circ (c \circ (d \circ (b \circ e)))))) * ((b \circ (e \circ (b \circ (e \circ (c \circ e)))) + (b \circ (e \circ (b \circ (e \circ (c \circ d)))) + (b \circ (e \circ (b \circ (e \circ (b \circ e)))))) * ((b \circ (e \circ (b \circ (d \circ (c \circ e)))) + (b \circ (e \circ (b \circ (d \circ (c \circ d)))) + (b \circ (e \circ (b \circ (d \circ (b \circ e))))))$

(by 105 reductions)

Number of calls to AC matching : 79665

Number of successful calls : 105 (0%)

times	user	system	total
Total time	15.700	2.810	18.510
Matching times	12.600	1.650	14.250

(estatísticas sobre o cálculo de $R_2(R_1(\psi))$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of

$(((((c \circ e \circ c \circ d \circ c \circ e) + (c \circ e \circ c \circ d \circ c \circ d)) + (c \circ e \circ c \circ d \circ b \circ e)) * (((c \circ e \circ b \circ e \circ c \circ c$

$e)+(c o e o b o e o c o d)+(c o e o b o e o b o e)) * (((c o e o b o d o c o e)+(c o e o b o d o c o d)+(c o e o b o d o b o e)))+((((c o d o c o d o c o e)+(c o d o c o d o c o d)+(c o d o c o d o b o e)) * (((c o d o b o e o c o e)+(c o d o b o e o c o d)+(c o d o b o e o b o e)) * (((c o d o b o d o c o e)+(c o d o b o d o c o d)+(c o d o b o d o b o e)))+((((b o e o c o d o c o e)+(b o e o c o d o c o d)+(b o e o c o d o b o e)) * (((b o e o b o e o c o e)+(b o e o b o e o c o d)+(b o e o b o e o b o e)) * (((b o e o b o d o c o e)+(b o e o b o d o c o d)+(b o e o b o d o b o e)) is$
 $(((((c o c o e o c)+(c o c o d o c)+(e o c)) * (((e o e o e o c)+(e o c)+(e o e o e o b)) * (((e o c)+(d o c)+(e o b)))+((((c o c o d o c)+(c o d o c o d o c o d)+(d o c)) * (((e o c)+(d o c)+(e o b)) * (((d o c)+(d o d o d o c)+(d o b)))+((((e o c)+(d o c)+(e o b)) * (((e o e o e o b)+(e o b)+(b o e o b o e o b o e)) * (((e o b)+(d o b)+(b o b o e o b)))$

(by 84 reductions)

Number of calls to AC matching : 14072

Number of successful calls : 84 (0%)

times	user	system	total
Total time	3.710	0.880	4.590
Matching times	2.280	0.510	2.790

(estatísticas sobre o cálculo de $\psi \downarrow \stackrel{\text{def.}}{=} R_3(R_2(R_1(\psi)))$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of

$((A+B+I)*(C+I+F)*(I+K+J))+((B+E+K)*(I+K+J)*(K+D+L))+((I+K+J)*(F+J+H)*(J+L+G))$
 is $J+I+K$

(by 126 reductions)

Number of calls to AC matching : 1129

Number of successful calls : 126 (11%)

times	user	system	total
Total time	458.800	0.130	458.930
	(7mn 38s)	(0mn 00s)	(7mn 38s)
Matching times	458.710	0.050	458.760
	(7mn 38s)	(0mn 00s)	(7mn 38s)

(estatísticas sobre o cálculo de $R_4(\psi \downarrow)$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of J+I+K is

J+I+K

(by 0 reductions)

Number of calls to AC matching : 2

Number of successful calls : 0 (0%)

times	user	system	total
Total time	0.010	0.030	0.040
Matching times	0.000	0.000	0.000

C.5 Exemplo 9.38

- Estatísticas fornecidas pelo sistema CiME sobre o exemplo 9.38:

◦ Estatísticas para o operador $\psi' = (((\epsilon_N \circ \delta_A) \wedge \delta_{[101]}) \circ \delta_N) \vee \delta_{\begin{bmatrix} 000 \\ 100 \end{bmatrix}}$, onde $A = \begin{bmatrix} 101 \\ 000 \\ 100 \end{bmatrix}$:

(estatísticas sobre o cálculo de $R_1(\psi')$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $((((b \circ ((c \circ e) + (c \circ d) + (b \circ e))) * (e + d)) \circ c) + (b \circ e))$ is

$((((e \circ c) + (d \circ c)) * ((b \circ (c \circ (e \circ c))) + (b \circ (c \circ (d \circ c))) + (b \circ (b \circ (e \circ c))))) + (b \circ e)$

(by 10 reductions)

Number of calls to AC matching : 577

Number of successful calls : 10 (1%)

times	user	system	total
Total time	0.120	0.110	0.230
Matching times	0.070	0.020	0.090

(estatísticas sobre o cálculo de $R_2(R_1(\psi'))$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $((e \circ c) + (d \circ c)) * (((b \circ c \circ e \circ c) + (b \circ c \circ d \circ c)) + (b \circ b \circ e \circ c)) + (b \circ e)$ is

$((e \circ c) + (d \circ c)) * (((e \circ c) + (d \circ c)) + (e \circ b)) + (b \circ e)$

(by 6 reductions)

Number of calls to AC matching : 316

Number of successful calls : 6 (1%)

times	user	system	total
Total time	0.110	0.040	0.150
Matching times	0.040	0.010	0.050

(estatísticas sobre o cálculo de $\psi' \downarrow \stackrel{\text{def.}}{=} R_3(R_2(R_1(\psi')))$)

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $((A+B)*(A+B+C))+C$ is

$B+C+A$

(by 9 reductions)

Number of calls to AC matching : 45

Number of successful calls : 9 (20%)

times	user	system	total
Total time	0.050	0.010	0.060
Matching times	0.020	0.000	0.020

(estatísticas sobre o cálculo de $(R_4(\psi' \downarrow))$

Welcome to CiME v1.13 - SunOS-5.4 - CSL opt - 09/25/96 19:36:38

Solving problems:

The normal form of $B+C+A$ is

$B+C+A$

(by 0 reductions)

Number of calls to AC matching : 2

Number of successful calls : 0 (0%)

times	user	system	total
Total time	0.040	0.030	0.070
Matching times	0.000	0.000	0.000

Referências Bibliográficas

- [BB90] Gerald J. F. Banon and Junior Barrera. Set mapping decompositions by mathematical morphology. In R. M. Haralick, editor, *Mathematical Morphology: theory and Hardware*. 1990.
- [BB92] J. Barrera and J. G. F. Banon. Expressiveness of the morphological language. In *Image Algebra and Morphological Image Processing III*, pages 264–275, San Diego, California (1769), 1992. SPIE.
- [BB94] Gerald J. F. Banon and Junior Barrera. *Bases da Morfologia Matemática Para a Análise de Imagens Binárias*. IX Escola de Computação, Recife, 24 a 31 de julho de 1994.
- [BD87] Leo Bachmair and Nachum Dershowitz. Completion for rewriting modulo a congruence. In *Lectures Notes in Computer Science - Rewriting Techniques and Applications*, vol 256, pages 192–203. Springer-Verlag, 1987.
- [Bir35] Garrett Birkhoff. On the structure of abstracts algebras. In *Proceedings of the Cambridge Philosophical Society*, pages 31:433–452, October 1935.
- [BKN85] D. Benanav, D. Kapur, and P. Narendran. Complexity of matching problems. In *Proceedings of First International Conference on Rewriting and Applications, Dijon, France*, maio de 1985.
- [BPW89] Timothy B. Baird, Gerald E. Peterson, and Ralph W. Wilkerson. Complete sets of reductions modulo associativity, commutativity and identity. In *Lectures Notes in Computer Science*, n. 355 - *Rewriting Techniques and Applications*. RTA-89, 1989.
- [BS92] F. Baader and K. U. Schulz. Unification in the union of disjoint equational theories: Combining decision procedures. In *Lecture Notes in Artificial Intelligence - Proceedings of the 11th International Conference on Automated Deduction*, volume 607, pages 50–65. Springer-Verlag, Saratoga Springs, NY, USA 1992.

- [BS94] F. Baader and J. H. Siekmann. Unification theory. In C. Hogger, D. Gabbay and J. A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming (Vol. 2)*. Oxford University Press, 1994.
- [BS96] Junior Barrera and Guilherme Pablo Salas. Set operations on closed intervals and their applications to the automatic programming of morphological machines. *Journal of Electronic Imaging*, 5(3):335–352, July 1996.
- [Buc65] Bruno Buchberger. *An Algorithm for Finding a Basis for the Residue Class Ring of a Zero-Dimensional Ideal*. PhD thesis, University of Innsbruck, Austria (escrito em alemão), 1965.
- [Buc87] Bruno Buchberger. History and basic features of the critical-pair/completion procedure. *Journal of Symbolic Computation*, 3:3–38, 1987.
- [Coh65] P. M. Cohn. *Universal Algebra*. Harper and Row, 1965.
- [CP67] A. H. Clifford and G. B. Preston. *The Algebraic Theory of Semigroups*. American Mathematical Society, 1967.
- [dA91] Jorge de Almeida. *Semigrupos Finitos e Algebra Universal*. Instituto de Matemática e Estatística da USP, São Paulo, 1991.
- [dAG87] Leucio de Azevedo Guerra. Sistemas de reescrita de termos: Aplicações a tipos abstratos de dados. Master's thesis, Depto. Informática do Centro de Ciências Exatas e da Natureza, Univ. de Pernambuco, Brazil, Dezembro 1987.
- [Der82] Nachum Dershowitz. Ordering of term-rewriting systems. *Theoretical Computer Science*, 17:279–301, 1982.
- [Der87] Nachum Dershowitz. Termination of rewriting. *Journal of Symbolic Computation*, 3:69–116, 1987.
- [Dic91] A. J. J. Dick. An introduction to knuth-bendix completion. *The computer Journal*, 34(1):2–15, 1991.
- [DJ90] N. Dershowitz and J. Jouannaud. Rewrite systems. In van Leeuwen, editor, *Handbook of theoretical Computer Science (ed. 3)*, pages 243–320. Elsevier Science Publishers B.V, 1990.
- [dO95] Anjolina Grisi de Oliveira. Transformações entre provas para a dedução natural rotulada via sistemas de reescrita de termos. Master's thesis, Depto. Informática do Centro de Ciências Exatas e da Natureza, Univ. de Pernambuco, Brazil, 1995.

- [Eva51] T. Evans. On multiplicative systems defined by generators and relations. i. normal form theorems. In *Proc. Cambridge Phil. Soc.*, volume 47, pages 637–649, 1951.
- [Fag84] François Fages. Associative-commutative unification. In *Lectures Notes in Computer Science*, n. 170 - 7th International Conference on Automated Deduction, 1984.
- [FH83] François FAGES and Gérard HUET. Complete sets of unifiers and matchers in equational theories. In *Lecture Notes in Computer Science n.159 - CAAP, L'Aquila*, (ed. by G. Ausiello, M. Protasi), pages 205–220, 1983.
- [Gor67] S. Gorn. Explicit definitions and linguistic dominoes. In *Systems and Computer Science*, pages 77–115. University of Toronto Press, Toronto, 1967.
- [Grä79] George Grätzer. *Universal Algebra (2.ed)*. Springer-Verlag, New York, 1979.
- [Har92] David Harel. *Algorithmics: The Spirit of Computing (2.ed)*. Addison-Wesley, The Weizmann Institute of Science, Rehovot, Israel, 1992.
- [Hen91] Martin Joachim Henz. Term rewriting in associative commutative theories with identities. Master's thesis, State University of New York at Stony Brook, December 1991.
- [Her30] J. Herbrand. *Recherches sur la théorie de la démonstration*. PhD thesis, University of Paris, 1930.
- [HL78] G. Huet and D. S. Lankford. On the uniform halting problem for term rewriting systems. Technical Report Report 283, INRIA, Le Chesnay - France, 1978.
- [HO80] G. Huet and Derek C. Oppen. Equations and rewriting rules - a survey. In R. book, editor, *Formal Language Theory: Perspectives and Open Problems*, pages 349–405. Academic Press, 1980.
- [Hue78] G. Huet. An algorithm to generate the basis of solutions to homogeneous linear diophantine equations. *Information Processing Letters*, 7, N. 3:144–147, 1978.
- [Hue80] G. Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *Journal of Association for Computing Machinery*, 27(4):797–821, October 1980.
- [Hue81] G. Huet. A complete proof of correctness of the knuth-bendix completion algorithm. *Journal of Computer and System Sciences*, 23:11–21, 1981.
- [Ius87] Alfredo Iusem. *P = NP ou as Sutilezas da Complexidade Computacional*. Inst. de Mat. Pura e Aplicada - CNPq, Estrada Dona Castorina 110, Jardim Botânico 22.460 R.Janeiro, RJ, 1987.

- [JK86] J. P. Jouannaud and H. Kirchner. Completion of a set of rules modulo a set of equations. *SIAM Journal on Computing*, 15(4):1155–1194, novembro 1986.
- [KB70] D. E. Knuth and P. B. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, 1970.
- [Kir90] Claude Kirchner. *Unification*. Academic Press, 1990.
- [KKR91] C. Kirchner, H. Kirchner, and M. Rusinowitch. Deduction with symbolic constraints. *Revue Française d'Intelligence Artificielle*, 4, special issue on automated deduction:9–52, 1991.
- [Klo87] Jan Willen Klop. Term rewriting systems: A tutorial. *Bulletin of the EATCS*, 32:143–183, 1987.
- [Lan75a] D. S. Lankford. Canonical algebraic simplification in computational logic. Technical report, Mathematics Dep., University of Texas, Austin, Texas, May 1975.
- [Lan75b] D. S. Lankford. Canonical inference. Technical report, Mathematics Dep., University of Texas, Austin, Texas, December 1975.
- [Lan79] D. S. Lankford. On proving term rewriting systems are noetherian. Technical report, Mathematics Dep., Louisiana Tech University, Ruston, La., May 1979.
- [LB77a] D. S. Lankford and A. M. Ballantyne. Decision procedures for simple equational theories with a commutative axiom: Complete sets of commutative reductions. Technical report, Mathematics Dep., University of Texas, Austin, Texas, March 1977.
- [LB77b] D. S. Lankford and A. M. Ballantyne. Decision procedures for simple equational theories with commutative-associative axioms: Complete sets of commutative-associative reductions. Technical report, Mathematics Dep., University of Texas, Austin, Texas, Aug 1977.
- [LB77c] D. S. Lankford and A. M. Ballantyne. Decision procedures for simple equational theories with permutative axioms: Complete sets of permutative reductions. Technical report, Mathematics Dep., University of Texas, Austin, Texas, April 1977.
- [Mar47] A. A. Markov. On the impossibility of certain algorithms in the theory of associative systems. *C.R.Acad. Sci. URSS*, 55:533–586, 1947.
- [Mar91] Claude Marché. On ground ac-completion. In *LNCS n.488 - 4th Rewriting Techniques and Applications.*, Como, Italy, April 1991. Springer-Verlag.

- [Mar92] Claude Marché. The word problem of acd-ground theories is undecidable. *International Journal of Foundations of Computer Science*, 3(1):81–92, 1992.
- [Mar93] Claude Marché. *Réécriture modulo une théorie présentée par un système convergent et décidabilité des problèmes du mot dans certaines classes de théories équationnelles*. PhD thesis, Université Paris-Sud, Orsay, France, 1993.
- [Mar96a] Claude Marché. Normalized rewriting: an alternative to rewriting modulo a set of equations. *Journal of Symbolic Computation*, 21:253–288, 1996.
- [Mar96b] Claude Marché. Normalized rewriting: an unified view of knuth-bendix completion and gröbner bases computation. Technical report, Lab. de Recherche en Informatique - Université Paris-Sud. et CNRS URA 410, Orsay, France, 1996.
- [Men66] Elliott Mendelson. *Introduction to Mathematical Logic (3.ed)*. Van Nostrand Reinhold Company, New York, 1966.
- [Mon71] L. H. Jacy Monteiro. *Elementos de álgebra*. Um livro da coleção/Elementos de Matemática. IMPA Instituto de Matemática Pura e Aplicada, Rio de Janeiro, 1971.
- [New42] M. H. A. Newman. On theories with a combinatorial definition of ‘equivalence’. *Annals of Mathematics*, 43,2:223–243, 1942.
- [Nov55] P. S. Novikov. On the algorithms unsolvability of the word problem in group theory. *Trudy Mat. Inst. Steklov (escrito em russo)*, 44:1–143, 1955.
- [Pla85] D. A. Plaisted. The undecidability of self embedding for term rewriting systems. *Information Processing Letters*, 20:61–64, 1985.
- [Pla94] David A. Plaisted. Equational reasoning and term rewriting systems. In C.Hogger D. Gabbay and J.A Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming (Vol. 1)*, pages 273–364. Oxford University Press, 1994.
- [Plo72] G. D. Plotkin. *Building-in equational theories*. Edinburgh University Press, Edinburgh, Scotland, 1972.
- [Pos47] E. L. Post. Recursive unsolvability of a problem of thue. *Journal of Symbolic Logic*, 13:1–11, 1947.
- [PS81] Gerald E. Peterson and Mark E. Stickel. Complete sets of reductions for some equational theories. *Journal of the Association for Computing Machinery*, 28(2):233–264, April 1981.

- [Rob65] J. A. Robinson. A machine-oriented logic based on the resolution principle. *JACM*, 12:32–41, 1965.
- [Ros73] B. K. Rosen. Tree-manipulating systems and church-rosser theorems. *ACM*, 12(1):160–187, January 1973.
- [RSSU79] P. Raulefs, J. Siekmann, P. Szabó, and E. Unvericht. A short survey on the state of the art in matching and unification problems. *ACM-SIGSAM Bulletin*, 13, N. 2:14–20, 1979.
- [Sti81] Mark E. Stickel. An unification algorithm for associative-commutative functions. *Journal of the Association for Computing Machinery*, 28, N. 3:423–434, 1981.
- [Sti86] Mark E. Stickel. An introduction to automated deduction. In *Fundamentals of Artificial Intelligence - An Automated Course, Lecture Notes in Computer Science 232*, pages 75–132. Springer-Verlag, 1986.
- [Wec92] Wolfgang Wechler. *Universal Algebra for Computer Scientists*. Springer-Verlag, 1992.

Índice Remissivo

$\langle \rangle$, 12	$\emptyset$, 11
$>_{KB}$, 107	ϵ , 190
R/n , 19	ϵ^a , 190
$Th(E)$, 30	ϵ_A , 199
$Th(R)$, 42	$\overline{\overline{E}}$, 30
X^* , 13	$\in$, 12
$\mathbb{N}$, 12	$\in_{mul}$, 117
$\mathbb{Z}_+$, 12	ι , 208
Δ , 190	CSU_E , 51
$[s]_E$, 31	μCSU_E , 51
$\mathcal{P}^E$, 190	$\nrightarrow$, 13
$\mathcal{T}$, 23	$\ominus$, 195
$\cap$, 12	$\oplus$, 193
$\cdot^c$, 199	$\perp$, 19
$\circ$, 14	$\xrightarrow{n}$, 14
$\cup$, 12	$\subseteq$, 12
δ , 190	$\subsetneq$, 12
δ^a , 190	$\succ^*$, 118
$\delta_L, \delta_O, \delta_N, \delta_S$, 208	$\succ_{mul}$, 118
δ_A , 198	τ_u , 192
$\xrightarrow{R}$, 41	$\supseteq$, 93
$\xrightarrow{(R,T)}$, 150	$\vee$, 200
$\xrightarrow{*}$, 17	$\wedge$, 200
$\xrightarrow{*}_R$, 42	f_τ , 103
$\xrightarrow{\equiv}$, 17	$inf \mathcal{X}$, 188
$\xrightarrow{+}$, 17	$n(x, t)$, 107
$\xleftrightarrow{E}$, 30	n/m , 20
$\xleftrightarrow{*}$, 17	$sdom\ t$, 23
$\xleftrightarrow{*}_E$, 30	$sup \mathcal{X}$, 188
$\xleftrightarrow{*}_R$, 43	t^e , 153

- $\vdash$, 53
 $(l \rightarrow r)^e$, 181
 (s, t) , 12
 $\langle s, t \rangle$, 12
 $\langle x_1, \dots, x_k \rangle$, 12
 $AC(T)$, 174
 $A\theta$, 151
 $Arid(f)$, 22
 $Const(t)$, 24
 F^n , 14
 $Lvar(\theta)$, 24
 $R(AC(T))$, 175
 $R(t)$, 44
 $R[n \leftarrow S]$, 19
 R_T^e , 175
 $Rvar(\theta)$, 24
 $Var(\langle s, t \rangle)$, 24
 $Var(\theta)$, 24
 $Var(t)$, 24
 X^t , 193
 Δ^+ , 130
 Σ , 22
 Σ_A – álgebra, 33
 base, 33
 finita, infinita, 34
 suporte, 34
 universo de discurso, 34
 $\vec{\rightarrow}_{[R]}$, 151
 $\rightarrow^{-1}$, 13
 $\models$, 40
 $\|\psi\|^p$, 235
 $\|t\|^v$, 37
 $\|w\|$, 12
 $\psi\downarrow$, 233
 $\rightarrow$, 13
 $\downarrow_R$, 43
 $\frac{V}{E}$, 49
 $\frac{V}{E}$, 49
 $\frac{V}{E}$, 49
 $\frac{[p]}{R}$, 42
 $\vdash$, 40
 $\{s, t\}$, 12, 29
 $\{\mathbf{o}\}$, 207
 $f(\dots t \dots)$, 24
 $f(t_1 t_2 \dots t_{n-1} t_n)$, 76
 $l^e \rightarrow r^e$, 153
 t^* , 44
 $\mathcal{F}$, 22
 $\mathcal{F}_*$, 23
 $\mathcal{I}_A$, 33
 $\mathcal{P}(X)$, 12
 $\mathcal{T}(\mathcal{F} \cup \mathcal{V})$, 22
 $\mathcal{T}/E$, 31
 $\mathcal{T}_E$, 29
 $\mathcal{T}_R$, 41
 $\mathcal{T}_{RT}$, 150
 $\mathcal{V}$, 22
 ínfimo, 188
 Δ^a , 190
 $\mathbf{E}^a$, 190
 $\mathbf{E}$, 190
 $\mathbf{U}_E(s, t)$, 50
 $\mathbf{o}$, 192
 alfabeto, 22
 álgebra de Boole, 36
 anel booleano, 36
 anel comutativo, 35
 anti-dilatação, 189
 anti-erosão, 189
 árvore, 19
 atribuição, 37

- axioma de reescrita, 41
 - módulo T , 150
- axioma equacional, 30
- “bags”, 118
- base de Gröbner, 254
- Church-Rosser, 141, 156
- CiME system, 239
- classe de equivalência, 16
- confluência local, 44
- congruência, 24
- conjunto das partes, 12
- conjunto independente, 19
- conjunto quociente, 16
- contexto de aplicação, 22
- corpo finito, 37
- derivação auto-embutida, 96
- dilatação, 189
- E -derivação, 31
- E -prova, 31
- elemento estruturante, 199
 - primitivo, 206
- “embedding”, 20
- equação, 29
 - instância, 29
 - linear, 29
 - orientada, 40
 - permutativa, 252
 - variante de, 157
- erosão, 189
- fecho, 17
- folha, 19
- forma normal, 18
- “full invariance order”, 102
- função, 14
 - atribuição, 37
 - estruturante, 191
 - invariante por translação, 196
 - valoração, 37
- função de terminação, 103
- generalização- E , 49
- “ground term”, 23
- grupo, 34
 - abeliano, comutativo, 35
 - elemento neutro ou origem, 192
- instância- E , 49
- instanciação- E , 49
- interpretação, 43
- interpretação polinomial, 104
- invariância por translação, 193
- k -upla ordenada, 12
- Khoros system, 234
- Lema do diamante, 130
- limitante
 - supremo, 188
 - inferior, 187
 - ínfimo, 188
 - superior, 187
- mapeamento, 14
- “matching”, 27
- matching- E , 49
- Minkowski
 - adição de, 193
 - diferença de, 195
 - soma de, 193
 - subtração de, 195
- modelo, 37
- monóide, 34
- multi-conjuntos, 117

- operação, 14
- operador, 22
 - interseção de, 200
 - morfológico, 205
 - ótimo^p, 235
 - complexidade de, 213
 - complexidade paralela de, 235
 - decomposto, 211
 - equivalência entre, 206
 - seqüencial, 214
 - seqüencial ótimo, 223
 - seqüencial, uniões ou interseções de, 214
 - sobre conjuntos, 189
 - translação de, 192
 - união de, 200
- operador-AC, 57
- operando, 24
- ordem lexicográfica, 98
- ordem monotônica, 101
- ordem parcial estrita estável, 101
- ordem de caminho recursivo, 118
- ordem de terminação, 102
- ordem de Knuth e Bendix, 107
- ordem de simplificação, 110
- ordem *noetheriana* para derivações, 110
- ordens, 16
- palavra, 12
 - comprimento, 12
 - concatenação, 13
- par crítico, 134
 - convergente, 43
 - divergente, 43
 - módulo T , 160
 - especial, 165
- par não ordenado, 12
- par ordenado, 12
- pré-ordem, 112
 - monotônica, 113
 - noetheriana, 112
- produto cartesiano, 13
- R -derivação, 43
- R -prova, 43
- "redex", 42
- regra de reescrita, 41
 - compatibilidade, 42
 - estabilidade, 42
 - extensão de, 153
 - extensão protetora de, 153
 - extensão simples de, 181
 - instância de, 41
 - linear, 41
 - linear à direita, 41
 - linear à esquerda, 41
 - variante de, 157
- relação
 - homeomórfica embutida, 93
 - bem fundada, 15
 - binária, 13
 - de congruência, 24
 - de equivalência, 15
 - inversa, 13
 - noetheriana, 15
- reticulado, 35
 - distributivo, 35
- reticulado completo, 189
- símbolo funcional, 22
 - interpretação, 33
 - realização, 33
- semigrupo, 34
- significado de um termo, 37
- "singleton", 207

- sistema de reescrita
 - auto-embutido, 96
 - propriedade de terminação, 44
- sistema de reescrita
 - canônico, 44
 - compatível com T , 159
 - completo, 44
 - completo módulo T , 156
 - confluyente, 43
 - confluyente módulo T , 154
 - localmente confluyente, 44
 - localmente confluyente módulo T , 156
 - noetheriano, 44
 - noetheriano módulo T , 125, 155
- SRT, 41
- substituição, 24
 - atribuição, 25
 - determinada por, 25
 - mais específica que, 49
 - mais geral que, 49
 - permutação, 24
 - restrição de, 134
 - unificadora, 27
 - unificadora- E , 49
 - independente, 50
 - mais geral, 50
 - congruência- E , 48
- supremo, 188
- Teorema
 - Birkhoff de, 39
 - Knuth e Bendix
 - generalização do, 160
 - Knuth e Bendix de, 137
 - Newman de, 130
- teorema, 31
- teoria básica, 33
- teoria de reescrita, 42
- teoria equacional, 30
 - básica, 33
 - fechada, 33
- teoria fechada, 33
- teoria- AC , 174
- terminação
 - multi-conjuntos, 117
 - derivação auto-embutida, 96
 - função de terminação, 103
 - interpretação polinomial, 104
 - módulo T , 124
 - módulo AC , 125
 - ordem de caminho recursivo, 118
 - ordem de Knuth e Bendix, 107
 - ordem de simplificação, 110
 - ordem de terminação, 102
 - ordem lexicográfica, 98
 - ordem monotônica, 101
 - ordem parcial estrita estável, 101
 - pré-ordem, 112
 - monotônica, 113
 - noetheriana, 112
 - relação homeomórfica embutida, 93
- termo, 22
 - forma normal de, 18
 - canônico, 44
 - aberto, 23
 - básico, 23
 - confluyente, 43
 - domínio estrito, 23
 - extensão de, 153
 - extensão protetora de, 153
 - fechado, 23
 - forma canônica de, 44
 - generalização, 27
 - instância de, 27

- irredutível, 15, 42
- linear, 23
- livre, 23
- localmente convergente, 45
- par convergente, 43
- par divergente, 43
- redutível, 15
- subtermo, 23
 - próprio, 23
- unificáveis, 27
- unificáveis- E , 49
- unificador, 27
- variante, 27
- termo- AC , 57
- translação por um vetor, 192
- transposto, 193
- u.m.g, 27
- unificáveis- E , 49
- unificadora- E , 49
- valoração, 37
- vetor, 192
- “*word problem*”, 47