JOÃO NUNES de SOUZA

LÓGICA para CIÊNCIA da COMPUTAÇÃO

Uma introdução concisa

21 de maio de 2008

1

A linguagem da Lógica Proposicional

Introdução

Alfabeto da Lógica Proposicional

Definição 1.1 (alfabeto) *O alfabeto da Lógica Proposicional é constituído por:*

- *símbolos de pontuação:* $(,)$*;*
- *símbolos de verdade: true, false;*
- *símbolos proposicionais:* $P, Q, R, S, P_1, Q_1, R_1, S_1, P_2, Q_2, \ldots$*;*
- *conectivos proposicionais:* $\neg$,$\vee$,$\wedge$,$\rightarrow$,$\leftrightarrow$.

Fórmulas da Lógica Proposicional

Definição 1.2 (fórmula) *As fórmulas da linguagem da Lógica Proposicional são construídas, de forma indutiva, a partir dos símbolos do alfabeto conforme as regras a seguir. O conjunto das fórmulas é o menor conjunto que satisfaz as regras:*

- *todo símbolo de verdade é uma fórmula;*
- *todo símbolo proposicional é uma fórmula;*
- *se* H *é uma fórmula, então* $(\neg H)$*, a negação de* H*, é uma fórmula;*
- *se* H *e* G *são fórmulas, então a disjunção de* H *e* G*, dada por:* $(H \vee G)$*, é uma fórmula;*
- *se* H *e* G *são fórmulas, então a conjunção de* H *e* G*, dada por:* $(H \wedge G)$*, é uma fórmula;*
- *se* H *e* G *são fórmulas, então a implicação de* H *em* G*, dada por:* $(H \rightarrow G)$*, é uma fórmula. Nesse caso,* H *é o antecedente e* G *o conseqüente da fórmula* $(H \rightarrow G)$*;*
- *se* H *e* G *são fórmulas, então a bi-implicação de* H *e* G*, dada por:* $(H \leftrightarrow G)$*, é uma fórmula. Nesse caso,* H *é o lado esquerdo e* G *o lado direito da fórmula* $(H \leftrightarrow G)$*.*

Notação. Neste livro, os parênteses ou símbolos de pontuação das fórmulas são omitidos quando não há problemas sobre a sua interpretação. Além disso, as fórmulas podem ser escritas em várias linhas para uma melhor leitura. Assim, a fórmula:

$$(((P \vee R) \rightarrow \mathit{true}) \leftrightarrow (Q \wedge S))$$

pode ser escrita como

$$\begin{array}{c}(P \vee R) \rightarrow \mathit{true} \\ \leftrightarrow \\ Q \wedge S\end{array}$$

ou ainda como

$$((P \vee R) \rightarrow \mathit{true}) \leftrightarrow (Q \wedge S).$$

Definição 1.3 (ordem de precedência) *Na Lógica Proposicional, a ordem de precedência dos conectivos proposicionais é definida por:*

- *maior precedência:* $\neg$*;*
- *precedência intermediária:* $\rightarrow$ *,* $\leftrightarrow$*;*
- *menor precedência:* $\wedge$ *,* $\vee$*.*

Linguagem-objeto e Metalinguagem

Variáveis

Notação. Os símbolos proposicionais são representados por variáveis do tipo: $\breve{P}$, com possíveis subíndices. Neste caso, temos a letra P com um pequeno risco na parte de cima. Isso significa, por exemplo, que $\breve{P}_1$ pode representar qualquer um dos símbolos $P, Q, R, S, P_1, Q_1, R_1, S_1, P_2, \ldots$. As variáveis A, B, C, D, E, H com possíveis subíndices representam fórmulas. A variável H_2 pode representar, por exemplo, a fórmula $(P \rightarrow Q)$.

Letras como $\breve{P}, A, B, C, D, E$ e H são elementos da metalinguagem que representam símbolos proposicionais e fórmulas em geral da Lógica Proposicional. Isso significa que, a rigor, $(\breve{P}_1 \rightarrow \breve{P}_2)$ não é uma fórmula da Lógica Proposicional. Essa expressão é a representação de fórmulas do tipo $(P \rightarrow Q)$, $(R \rightarrow S)$ etc. Do mesmo modo, $(H \vee G)$ não é uma fórmula, mas a representação de fórmulas do tipo $((P \rightarrow Q) \vee (R \wedge S))$, onde H é substituída por $(P \rightarrow Q)$ e G por $(R \wedge S)$. Geralmente, expressões do tipo $(\breve{P}_1 \rightarrow \breve{P}_2)$ e $(H \vee G)$ são denominadas esquemas de fórmulas. Os esquemas de fórmulas se transformam em fórmulas quando as metavariáveis são substituídas por símbolos e fórmulas da Lógica. Vale a pena observar nas definições a seguir, a utilização de variáveis que representam símbolos proposicionais e fórmulas.

Alguns Elementos Sintáticos das Fórmulas

Definição 1.4 (comprimento de uma fórmula) *Seja H uma fórmula da Lógica Proposicional. O comprimento de H, denotado por $comp[H]$, é definido como se segue.*

- *Se $H = \breve{P}$ ou é um símbolo de verdade, então $comp[H] = 1$;*
- $comp[\neg H] = comp[H] + 1$;
- $comp[H \vee G] = comp[H] + comp[G] + 1$;
- $comp[H \wedge G] = comp[H] + comp\ [G] + 1$;
- $comp[H \rightarrow G] = comp[H] + comp[G] + 1$;
- $comp[H \leftrightarrow G] = comp[H] + comp[G] + 1$.

Definição 1.5 (subfórmula) *Seja H uma fórmula da Lógica Proposicional, então:*

- *H é uma subfórmula de H;*
- *se H é uma fórmula do tipo $(\neg G)$, então G é uma subfórmula de H;*
- *se H é uma fórmula do tipo: $(G \vee E)$, $(G \wedge E)$, $(G \rightarrow E)$ ou $(G \leftrightarrow E)$, então G e E são subfórmulas de H;*
- *se G é subfórmula de H, então toda subfórmula de G é subfórmula de H.*

Exercícios

Exercícios de Computação

2

A semântica da Lógica Proposicional

Introdução

Interpretação

Definição 2.1 (função binária) *Uma função é binária se seu contradomínio possui apenas dois elementos.*

Definição 2.2 (função total) *Uma função é total se é definida em todos os elementos de seu domínio.*

Definição 2.3 (função interpretação) *Uma interpretação I, na Lógica Proposicional, é uma função binária total na qual,*

- *o domínio de I é constituído pelo conjunto das fórmulas da Lógica Proposicional;*
- *o contradomínio de I é o conjunto $\{T, F\}$.*

Interpretação de Fórmulas

Definição 2.4 (interpretação de fórmulas) *Dadas uma fórmula E e uma interpretação I, então a interpretação de E, indicado por $I[E]$, é determinada pelas regras:*

- *se E é do tipo $\breve{P}$, então $I[E] = I[\breve{P}]$ e $I[\breve{P}] \in \{T, F\}$;*
- *se E é do tipo $true$, então $I[E] = I[true] = T$;*
- *se E é do tipo $false$, então $I[E] = I[false] = F$;*
- *se E é do tipo $\neg H$, então $I[E] = I[\neg H] = T$ se $I[H] = F$ e $I[E] = I[\neg H] = F$ se $I[H] = T$;*
- *se E é do tipo $(H \vee G)$, então $I[E] = I[H \vee G] = T$ se $I[H] = T$ e/ou $I[G] = T$ e $I[E] = I[H \vee G] = F$ se $I[H] = F$ e $I[G] = F$;*
- *se E é do tipo $(H \wedge G)$, então $I[E] = I[H \wedge G] = T$ se $I[H] = T$ e $I[G] = T$ e $I[E] = I[H \wedge G] = F$ se $I[H] = F$ e/ou $I[G] = F$;*
- *se E é do tipo $(H \rightarrow G)$, então $I[E] = I[H \rightarrow G] = T$ se $I[H] = F$ e/ou $I[G] = T$ e $I[E] = I[H \rightarrow G] = F$ se $I[H] = T$ e $I[G] = F$;*
- *se E é do tipo $(H \leftrightarrow G)$, então $I[E] = I[H \leftrightarrow G] = T$ se $I[H] = I[G]$ e $I[E] = I[H \leftrightarrow G] = F$ se $I[H] \neq I[G]$.*

A semântica do conectivo $\neg$.

A semântica do conectivo $\vee$.

A semântica do conectivo $\wedge$.

A semântica do conectivo $\rightarrow$.

A causalidade e a semântica do conectivo $\rightarrow$.

A semântica do conectivo $\leftrightarrow$.

O número de interpretações.

O princípio da composicionalidade

Funções de verdade

Exercícios

Exercícios de Computação

3

Propriedades semânticas da Lógica Proposicional

Introdução

Propriedades Semânticas

Definição 3.1 (propriedades semânticas básicas da Lógica Proposicional) *Sejam $H, G, H_1, H_2, \ldots, H_n$, fórmulas da Lógica Proposicional. As propriedades semânticas básicas da Lógica Proposicional são definidas a seguir.*

- *H é uma tautologia, se, e somente se, para toda interpretação I, $I[H] = T$.*
- *H é satisfatível,[1] se, e somente se, existe uma interpretação I, tal que $I[H] = T$.*
- *H é uma contingência, se, e somente se, existem duas interpretações I_1 e I_2, tais que $I_1[H] = T$ e $I_2[H] = F$.*
- *H é contraditória,[2] se, e somente se, para toda interpretação I, $I[H] = F$.*
- *H implica semanticamente[3] G, ou G é uma conseqüência lógica semântica de H, se, e somente se, para toda interpretação I, se $I[H] = T$, então $I[G] = T$.*
- *H equivale semanticamente[4] G, se e somente se, para toda interpretação I, $I[H] = I[G]$.*
- *Dada uma interpretação I, então I satisfaz H, se $I[H] = T$.*
- *O conjunto*

  $$\beta = \{H_1, H_2, \ldots, H_n, \ldots\}$$

  é satisfatível, se, e somente se, existe uma interpretação I, tal que

  $$I[H_1] = T, I[H_2] = T, \ldots = I[H_n] = T, \ldots.$$

[1] O termo "factível" é também, usualmente, utilizado como sinônimo de "satisfatível".
[2] O termo "contraditório" é também, em geral, usado como sinônimo de "logicamente falso" ou "inconsistente".
[3] A implicação semântica na Lógica Proposicional é também, usualmente, denominada como implicação tautológica.
[4] A equivalência semântica na Lógica Proposicional é também, usualmente, denominada equivalência tautológica.

Nesse caso, I satisfaz o conjunto de fórmulas.

Dado um conjunto de fórmulas vazio, então toda interpretação I satisfaz esse conjunto.

- *O conjunto*

$$\beta = \{H_1, H_2, \ldots, H_n, \ldots\},$$

implica semanticamente[5] *uma fórmula H, se para toda interpretação I; se $I[\beta] = T$, então $I[H] = T$. Nesse caso, também dizemos que H é uma conseqüência lógica semântica de G.*

Notação. Se um conjunto de fórmulas β implica semanticamente H, ou seja, H é conseqüência lógica semântica de G, então tal fato é indicado por $\beta \vDash H$. No caso em que β é vazio, então é utilizada a notação $\vDash H$. O símbolo $\vDash$ é, portanto, utilizado para denotar a implicação semântica ou conseqüência semântica, que relaciona interpretações de fórmulas. No caso em que β não implica semanticamente H, isto é, H não é conseqüência lógica semântica de G, é utilizada a notação: $\beta \nvDash H$

Notação. Da mesma forma, se H implica semanticamente G, isto é, G é uma conseqüência lógica semântica de H, denotamos esse fato por $H \vDash G$. No caso em que H não implica semanticamente G, isto é, G não é uma conseqüência lógica semântica de H, utilizamos a notação: $H \nvDash G$.[6]

Nota. Neste livro, utilizamos, indistintamente, as denominações "implicação semântica" e "conseqüência lógica semântica". Preferimos manter duas denominações para o mesmo conceito devido à tradição do ensino da Lógica em que elas são freqüentemente encontradas. Além disso, quanto do contexto estiver claro, podemos utilizar apenas os termos: "implicação", "conseqüência semântica" ou "conseqüência".

Notação. Se uma interpretação I satisfaz o conjunto de fórmulas β, esse fato é indicado por $I[\beta] = T$.

O princípio do terceiro-excluído.

O princípio da não-contradição.

Nota. Nessa demonstração aparece o símbolo $\Rightarrow$, que é denominado "implica". Esse símbolo pertence à metalinguagem e não deve ser confundido com o símbolo $\rightarrow$ da linguagem da Lógica.

Nota. A seguir, neste livro, as denominações "implicação semântica" e "equivalência semântica" serão denotadas simplesmente por "implicação" e "equivalência" respectivamente, a menos que o contexto não esteja claro.

Observação sobre satisfatibilidade de conjunto de fórmulas.

[5] A implicação semântica, entre um conjunto de fórmulas e uma fórmula, é também denominada, usualmente, de conseqüência tautológica, ou conseqüência lógica na Lógica Proposicional.

[6] É curioso notar que não existe uma unanimidade quanto à denominação do símbolo $\vDash$, em português.

Relações entre as Propriedades Semânticas

Proposição 3.1 (tautologia e contradição) *Dada uma fórmula H, então*

H é tautologia, se, e somente se, $\neg H$ é contraditória.

Proposição 3.2 (tautologia e satisfatibilidade) *Dada uma fórmula H,*

se H é tautologia então H é satisfatível.

Proposição 3.3 (tautologia e contradição) *Dada uma fórmula H, então:*

a. H é tautologia, se, e somente se, $\neg H$ é contraditória;

b. $\neg H$ não é satisfatível, se, e somente se, $\neg H$ é contraditória.

Proposição 3.4 (implicação semântica e o conectivo $\rightarrow$) *Dadas duas fórmulas H e G,*

$H \vDash G$, se, e somente se, $(H \rightarrow G)$ é tautologia.

Proposição 3.5 (equivalência semântica e o conectivo $\leftrightarrow$) *Dadas as fórmulas H e G,*

H equivale a G, se, e somente se, $(H \leftrightarrow G)$ é tautologia.

Proposição 3.6 (equivalência e implicação semânticas) *Dadas duas fórmulas H e G,*

H equivale a G, se, e somente se, $H \vDash G$ e $G \vDash H$.

Proposição 3.7 (transitividade da equivalência semântica) *Dadas as fórmulas E, H e G,*

se E equivale a H e H equivale a G, então E equivale a G.

Proposição 3.8 (satisfatibilidade) *Seja $\{H_1, H_2, \ldots, H_n\}$ um conjunto de fórmulas.*

$\{H_1, H_2, \ldots, H_n\}$ é satisfatível, se, e somente se, $(H_1 \wedge (H_2 \wedge (\ldots \wedge H_n)\ldots))$ é satisfatível.

Equivalências

Conjectura 3.1 (equivalência e tautologia) *Sejam H e G fórmulas da Lógica Proposicional, então*

{ H equivale a G }, se, e somente se, { H é tautologia, se, e somente se, G é tautologia }.

Conclusão: a conjectura indicada anteriormente é falsa, pois ela é composta de duas implicações, sendo uma delas falsa.

Proposição 3.9 (equivalência e tautologia) *Sejam H e G duas fórmulas.*

Se { H equivale a G }, então { H é tautologia, se, e somente se, G é tautologia }.

Lema 3.1 (implicação e tautologia) *Sejam H e G duas fórmulas.*

Se { { $H \vDash G$ } e { H é tautologia } }, então { G é tautologia }.

Lema 3.2 (implicação e conjunção) *Dadas três fórmulas A, B e C, então*

$(A \to (B \to C))$ *equivale a* $((A \wedge B) \to C)$.

Lema 3.3 (implicação e tautologia) *Sejam H e G duas fórmulas.*

Se { $H \vDash G$ }, então { H é tautologia $\Rightarrow$ G é tautologia }.

Teorema 3.1 (teorema da dedução - forma semântica) *Considere β um conjunto de fórmulas e A e B duas fórmulas da Lógica Proposicional.*

$$\beta \cup \{A\} \vDash B, \text{ se, e somente se, } \beta \vDash (A \to B)$$

Demonstração. A ida: "⇒". Temos $\beta \cup \{A\} \vDash B$ e devemos demonstrar que $\beta \vDash (A \to B)$. Mas,

$\beta \cup \{A\} \vDash B \Leftrightarrow$ para toda interpretação I, se $I[\beta \cup \{A\}] = T$, então $I[B] = T$

Por outro lado,

$\beta \vDash (A \to B) \Leftrightarrow$ para toda interpretação I, se $I[\beta] = T$, então $I[(A \to B)] = T$

Portanto, devemos demonstrar que se

$$I[\beta] = T, \text{ então } I[(A \to B)] = T.$$

Para demonstrar essa implicação, devemos supor $I[\beta] = T$ e demonstrar $I[(A \to B)] = T$. Por sua vez, para demonstrar $I[(A \to B)] = T$, devemos supor $I[A] = T$ e demonstrar $I[B] = T$. Além de tudo isso, devemos também utilizar a hipótese: $\beta \cup \{A\} \vDash B$. Portanto, resumindo, devemos supor:

$$I[\beta] = T,\ I[A] = T \text{ e } \beta \cup \{A\} \vDash B$$

e demonstrar $I[B] = T$. Mas, se $I[\beta] = T$, e $I[A] = T$, então $I[\beta \cup \{A\}] = T$. Logo, como temos $\beta \cup \{A\} \vDash B$ e

$$\beta \cup \{A\} \vDash B \Leftrightarrow \text{para toda interpretação } I, \text{ se } I[\beta \cup \{A\}] = T, \text{ então } I[B] = T$$

então concluímos que $I[B] = T$.

A volta: "⇐". Temos $\beta \vDash (A \rightarrow B)$ e devemos demonstrar que $\beta \cup \{A\} \vDash B$.

Portanto, devemos demonstrar que se

$$I[\beta \cup \{A\}] = T, \text{ então } I[B] = T.$$

Para demonstrar essa implicação, devemos supor $I[\beta \cup \{A\}] = T$ e demonstrar $I[B] = T$. Além disso, devemos também utilizar a hipótese: $\beta \vDash (A \rightarrow B)$. Portanto, resumindo, devemos supor:

$$I[\beta \cup \{A\}] = T \text{ e } \beta \vDash (A \rightarrow B)$$

e demonstrar $I[B] = T$. Mas, se $I[\beta \cup \{A\}] = T$, então $I[\beta] = T$. Logo, como $\beta \vDash (A \rightarrow B)$, então concluímos que $I[(A \rightarrow B)] = T$.

Por outro lado, se $I[\beta \cup \{A\}] = T$, temos também que $I[A] = T$. Logo, como $I[(A \rightarrow B)] = T$, concluímos que $I[B] = T$. **cqd**

Exercícios

Exercícios de Computação

4

Métodos para determinação de propriedades semânticas de fórmulas da Lógica Proposicional

Introdução

Método da Tabela-Verdade

Exemplo 4.1 (leis de De Morgan)

P	Q	$\neg(P \wedge Q)$	$(\neg P) \vee (\neg Q)$	$\neg(P \wedge Q) \leftrightarrow (\neg P \vee \neg Q)$
T	T	F	F	T
T	F	T	T	T
F	T	T	T	T
F	F	T	T	T

Tabela 4.1. Tabela-verdade associada à fórmula $\neg(P \wedge Q) \leftrightarrow ((\neg P \vee (\neg Q))$.

P	Q	$\neg(P \vee Q)$	$(\neg P) \wedge (\neg Q)$	$\neg(P \vee Q) \leftrightarrow (\neg P \wedge \neg Q)$
T	T	F	F	T
T	F	F	F	T
F	T	F	F	T
F	F	T	T	T

Tabela 4.2. Tabela-verdade associada à fórmula $\neg(P \vee Q) \leftrightarrow ((\neg P) \wedge (\neg Q))$.

Método da Árvore Semântica

Exemplo 4.2 (lei da contraposição) Este exemplo demonstra que a fórmula

$$(P \to Q) \leftrightarrow (\neg Q \to \neg P)$$

é uma tautologia utilizando o método da árvore semântica.

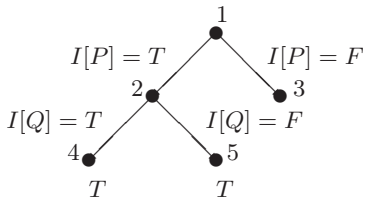

Figura 4.1. Desenvolvimento da árvore semântica.

Método da Negação, ou Redução ao Absurdo

Demonstração da contradição.

Aplicação do método às fórmulas com conectivo $\to$.

Aplicação do método às fórmulas com conectivo $\wedge$.

Aplicação do método às fórmulas com conectivo $\vee$.

Generalização do método.

A ausência do absurdo.

A conseqüência semântica.

A decidibilidade do conjunto das tautologias. Os métodos apresentados neste capítulo constituem algoritmos que decidem se um dada fórmula H é, ou não, uma tautologia.

Os métodos apresentados neste capítulo também são corretos e completos.

- Eles são corretos porque, dada uma fórmula H, que não é um tautologia, tais métodos nunca responderão o contrário, que H é uma tautologia. As respostas dadas pelos métodos são corretas.
- Eles são completos. Isso significa que, dada uma tautologia H, é possível construir uma tabela-verdade, uma árvore semântica ou uma prova por negação, que prove que H é realmente uma tautologia.

Exercícios

Exercícios de Computação

5

Relações semânticas entre os conectivos da Lógica Proposicional

Introdução

Conjuntos de Conectivos Completos

Definição 5.1 (conjunto de conectivos completo) *Seja Ψ um conjunto de conectivos. Ψ é um conjunto completo se as condições a seguir são satisfeitas. Dada uma fórmula H do tipo $\neg\breve{P}$, $(\breve{P}_1 \vee \breve{P}_2)$, $(\breve{P}_1 \wedge \breve{P}_2)$, $(\breve{P}_1 \rightarrow \breve{P}_2)$ ou $(\breve{P}_1 \leftrightarrow \breve{P}_2)$, então é possível determinar uma outra fórmula G, equivalente a H, tal que G contém apenas conectivos do conjunto Ψ e os símbolos $\breve{P}_1$ e $\breve{P}_2$ presentes em H.*

Proposição 5.1 (Equivalência entre $\rightarrow$ e $\neg, \vee$) *O conectivo $\rightarrow$ pode ser expresso semanticamente pelos conectivos $\neg$ e $\vee$.*

Proposição 5.2 (Equivalência entre $\wedge$ e $\neg, \vee$) *O conectivo $\wedge$ pode ser expresso semanticamente pelos conectivos $\neg$ e $\vee$.*

Proposição 5.3 (Equivalência entre $\leftrightarrow$ e $\neg, \vee$) *O conectivo $\leftrightarrow$ pode ser expresso semanticamente pelos conectivos $\neg$ e $\vee$.*

Proposição 5.4 (conjunto de conectivos completo) *O conjunto $\{\neg, \vee\}$ é completo.*

Proposição 5.5 (regra de substituição) *Sejam E_g, E_h, G e H fórmulas da Lógica Proposicional tais que:*

- *G e H são subfórmulas de E_g e E_h respectivamente.*
- *E_h é obtida de E_g substituindo todas as ocorrências da fórmula G em E_g por H.*

Se G equivale a H, então E_g equivale a E_h.

O Princípio da Indução na Lógica[1]

Notação. Para expressar uma asserção qualquer sobre uma fórmula H, escrevemos $B[H]$. Assim, $B[H]$ representa uma sentença que contém a fórmula H. Considere o exemplo:

$$B[H] \text{ representa } \text{"}H \text{ é uma tautologia"}.$$

Nesse caso, a asserção $B[H]$ representa a sentença: "H é tautologia". Neste livro, para representar tal fato, utilizamos a notação:

$$B[H] \equiv \ H \ \text{ é uma tautologia},$$

onde o símbolo $\equiv$ é apenas uma forma sintética de denotar a relação entre $B[H]$ e a sentença que está sendo representada. Um outro exemplo é o seguinte:

$$B[H] \equiv \ \text{o número de abre e fecha parênteses em } H \\ \text{é igual ao dobro do número dos conectivos de } H.$$

Nesse caso, a asserção $B[H]$ representa a sentença: "o número de abre e fecha parênteses em H é igual ao dobro do número dos conectivos de H".

Proposição 5.6 (o princípio da indução na Lógica Proposicional) *Seja $B[E]$ uma asserção que se refere a uma fórmula E da Lógica Proposicional. Se a base e o passo da indução, indicados a seguir, são verdadeiros, então concluímos que $B[E]$ é verdadeira para qualquer fórmula E.*

- *Base da indução na Lógica: $B[\breve{P}]$ é verdadeira para todo símbolo proposicional $\breve{P}$. As asserções $B[false]$ e $B[true]$ são verdadeiras.*
- *Passo da indução na Lógica: Sejam G e H duas fórmulas. Se $B[G]$ e $B[H]$ são verdadeiras, então $B[\neg H]$, $B[G \vee H]$, $B[G \wedge H]$, $B[G \rightarrow H]$ e $B[G \leftrightarrow H]$ são verdadeiras.*

Conclusão: para demonstrar algum resultado utilizando o princípio da indução na Lógica, devemos demonstrar, inicialmente, a base e o passo da indução. Em seguida, utilizar a implicação, determinada pelo princípio da indução, e concluir o resultado desejado, que é igual ao conseqüente de tal implicação.

[1] Esta subseção considera que o leitor já tenha alguma familiaridade com o princípio da indução.

Relação semântica entre conectivos

Proposição 5.7 (relação semântica entre conectivos) *Seja E uma fórmula da Lógica Proposicional. Então existe uma fórmula E_1, equivalente a E, que possui apenas os conectivos $\neg$ e $\vee$ e os símbolos proposicionais e de verdade presentes em E.*

Definição 5.2 (conectivo $nand$) *O conectivo nand é definido pela correspondência:*

$$(P \; nand \; Q) = (\neg(P \wedge Q).$$

Proposição 5.8 (equivalência entre $\neg$ e $\{nand\}$) *O conectivo $\neg$ pode ser expresso semanticamente pelo conectivo nand.*

Proposição 5.9 (equivalência entre $\vee$ e $\{nand\}$) *O conectivo $\vee$ pode ser expresso semanticamente pelo conectivo nand.*

Proposição 5.10 (conjunto de conectivo completo) *O conjunto $\{nand\}$ é completo.*

Proposição 5.11 (relação semântica entre conectivos) *Seja E uma fórmula qualquer da Lógica Proposicional. E pode ser expressa, equivalentemente, utilizando apenas o conectivo nand e os símbolos proposicionais e de verdade presentes em E.*

Definição 5.3 (conectivo nor) *O conectivo nor é definido pela correspondência:*

$$(P \; nor \; Q) = (\neg(P \vee Q).$$

Proposição 5.12 (conjunto de conectivo completo) *O conjunto $\{nor\}$ é completo.*

Proposição 5.13 (relação semântica entre conectivos) *Seja E uma fórmula qualquer da Lógica Proposicional. E pode ser expressa, equivalentemente, utilizando apenas o conectivo nor e os símbolos proposicionais e de verdade presentes em E.*

Redefinição do alfabeto da Lógica Proposicional.

Definição 5.4 (alfabeto na forma simplificada) *O alfabeto da Lógica Proposicional é constituído por:*

- *símbolos de pontuação: (,);*
- *símbolo de verdade:* $false$*;*
- *símbolos proposicionais:* $P, Q, R, S, P_1, Q_1, R_1, S_1, P_2, Q_2 \ldots$*;*
- *conectivos proposicionais:* $\neg$, $\vee$.

Formas normais

Definição 5.5 (literal) *Um literal, na Lógica Proposicional, é um símbolo proposicional ou sua negação.*

Definição 5.6 (forma normal) *Há dois tipos de formas normais:*

- *Uma fórmula* H *está na forma normal disjuntiva* (fnd) *se é uma disjunção de conjunção de literais.*
- *Uma fórmula* H *está na forma normal conjuntiva* (fnc) *se é uma conjunção de disjunção de literais.*

Exercícios

Exercícios de Computação

6

Um sistema axiomático formal na Lógica Proposicional

Introdução

O Sistema Axiomático P_a

Definição 6.1 (sistema axiomático P_a) *O sistema formal axiomático P_a da Lógica Proposicional é definido pela composição dos quatro elementos:*

- *o alfabeto da Lógica Proposicional, na forma simplificada, Definição 5.4, sem o símbolo de verdade $false$;*
- *o conjunto das fórmulas da Lógica Proposicional;*
- *um subconjunto das fórmulas, que são denominadas axiomas;*
- *um conjunto de regras de dedução.*

Definição 6.2 (axiomas do sistema P_a) *Os axiomas*[1] *do sistema P_a são fórmulas da Lógica Proposicional determinadas pelos esquemas indicados a seguir. Nesses esquemas E, G e H são fórmulas quaisquer da Lógica Proposicional.*

- $Ax_1 = \neg(H \vee H) \vee H$,
- $Ax_2 = \neg H \vee (G \vee H)$,
- $Ax_3 = \neg(\neg H \vee G) \vee (\neg(E \vee H) \vee (G \vee E))$.

- $Ax_l = (H \vee H) \rightarrow H$,
- $Ax_2 = H \rightarrow (G \vee H)$,
- $Ax_3 = (H \rightarrow G) \rightarrow ((E \vee H) \rightarrow (G \vee E))$.

Notação. No sistema P_a são consideradas as correspondências a seguir, que definem os conectivos $\rightarrow$, $\leftrightarrow$ e $\wedge$.

[1] Os axiomas de um sistema formal como o P_a são geralmente denominados axiomas lógicos. Isso porque há também os axiomas não-lógicos utilizados no estudo das teorias. Como neste livro não se considera o estudo de teorias, não será feita tal distinção.

$$H \to G \text{ denota } (\neg H \vee G).$$
$$(H \leftrightarrow G) \text{ denota } (H \to G) \wedge (G \to H).$$
$$(H \wedge G) \text{ denota } \neg(\neg H \vee \neg G).$$

Definição 6.3 (regra de inferência do sistema P_a, *modus ponens*) *Dadas as fórmulas H e G, a regra de inferência do sistema P_a, denominada modus ponens (MP), é definida pelo procedimento: tendo H e $(\neg H \vee G)$ deduza G.*

Notação. Para representar o esquema de regra de inferência *modus ponens*, a notação a seguir é considerada

$$MP = \frac{H, (H \to G)}{G}.$$

Nessa notação, o "numerador" da equação, $H, (H \to G)$, é denominado antecedente. O "denominador" é o conseqüente.

Conseqüência lógica sintática em P_a

Definição 6.4 (prova sintática no sistema P_a) *Sejam H uma fórmula e β um conjunto de fórmulas denominadas por hipóteses. Uma prova sintática de H a partir de β, no sistema axiomático P_a, é uma seqüência de fórmulas $H_1, H_2, \ldots, H_n$, onde temos:*

- $H = H_n$.

E para todo i tal que $1 \leq i \leq n$,

- H_i *é um axioma ou*
- $H_i \in \beta$ *ou*
- H_i *é deduzida de H_j e H_k, utilizando a Regra modus ponens, onde $1 \leq j < i$ e $1 \leq k < i$. Isto é,*

$$MP = \frac{H_j \quad H_k}{H_i}$$

Observe que neste caso, necessariamente, $H_k = H_j \to H_i$.

Exemplo 6.1 (prova no sistema P_a) Considere o conjunto de hipóteses $\beta = \{G_1, \ldots, G_9\}$ tal que

$G_1 = (P \wedge R) \to P$;
$G_2 = Q \to P_4$;
$G_3 = P_1 \to Q$;
$G_4 = (P_1 \wedge P_2) \to Q$;
$G_5 = (P_3 \wedge R) \to R$;
$G_6 = P_4 \to P$;

$G_7 = P_1$;

$G_8 = P_3 \rightarrow P$;

$G_9 = P_2$.

A seqüência de fórmulas $H_1, \ldots, H_9$ é uma prova de $(S \vee P)$ a partir de β no sistema axiomático P_a.

$H_1 = G_7$, ou seja: $H_1 = P_1$;

$H_2 = G_3$, ou seja $H_1 = P_1$;

$H_3 = Q$ (resultado de MP em H_1 e H_2);

$H_4 = G_2$, ou seja: $H_4 = Q \rightarrow P_4$;

$H_5 = P_4$ (resultado de MP em H_3 e H_4);

$H_6 = G_6$, ou seja: $H_6 = P_6 \rightarrow P$;

$H_7 = P$ (resultado de MP em H_5 e H_6);

$H_8 = Ax_2$, ou seja: $H_8 = P \rightarrow (S \vee P)$;

$H_9 = (S \vee P)$ (resultado de MP em H_7 e H_8).

Definição 6.5 (conseqüência lógica sintática no sistema P_a) *Dada uma fórmula H e um conjunto de hipóteses β[2], então H é uma conseqüência lógica sintática de β em P_a, se existe uma prova de H a partir de β.*

Definição 6.6 (teorema no sistema P_a) *Uma fórmula H é um teorema em P_a, se existe uma prova de H, em P_a, que utiliza apenas os axiomas. Nesse caso, o conjunto de hipóteses é vazio.*

Notação. Dada uma fórmula H, se H é conseqüência lógica sintática de um conjunto de hipóteses β tal que

$$\beta = \{H_1, H_2, \ldots, H_n, \ldots\},$$

então esse fato é indicado pela notação $\beta \vdash H$ ou

$$\{H_1, H_2, \ldots, H_n, \ldots\} \vdash H.$$

No caso em que H é um teorema, isto é, β é vazio, então utilizamos a notação $\vdash H$.

Proposição 6.1 *Sejam β um conjunto de fórmulas, e A, B e C três fórmulas da Lógica Proposicional. Temos que*

$$\textit{Se } \{\beta \vdash (A \rightarrow B) \text{ e } \beta \vdash (C \vee A)\}, \quad \textit{então} \quad \{\beta \vdash (B \vee C)\}.$$

Proposição 6.2 *Temos que* $\vdash (P \vee \neg P)$.

[2] O conjunto de hipóteses β pode ser finito ou não.

Proposição 6.3 (regra de substituição) *Sejam β um conjunto de fórmulas e H uma fórmula da Lógica Proposicional tais que $\beta \vdash H$.*

Considere $\{P_1, \ldots, P_n\}$ um conjunto de símbolos proposicionais que ocorrem em H, mas não ocorrem nas fórmulas de β .

Seja G a fórmula obtida de H, substituindo os símbolos proposicionais $P_1, \ldots, P_n$ pelas fórmulas $E_1, \ldots, E_n$, respectivamente.

Temos que $\beta \vdash G$.

Proposição 6.4 *Temos que $\vdash (P \to \neg\neg P)$.*

Proposição 6.5 *Temos que $\vdash (P \to P)$.*

Proposição 6.6 *Temos que $\vdash (A \vee B) \to (B \vee A)$.*

Demonstração.

1.	$\vdash (P \to P)$	$pr6.5$
2.	$\vdash (B \to B)$	$pr6.3, 1.$
3.	$\vdash (B \to B) \to ((A \vee B) \to (B \vee A))$	Ax_3
4.	$\vdash (A \vee B) \to (B \vee A)$	$MP, 2., 3.$

cqd

Proposição 6.7 (transitividade) *Se $\beta \vdash (A_1 \to A_2)$ e $\beta \vdash (A_2 \to A_3)$, então $\beta \vdash (A_1 \to A_3)$.*

Demonstração.

1.	$\beta \vdash (\neg A_1 \vee A_2)$	hip
2.	$\beta \vdash (A_2 \to A_3)$	hip
3.	$\beta \vdash (A_3 \vee \neg A_1)$	$pr6.1, 1., 2.$
4.	$\beta \vdash (A_3 \vee \neg A_1) \to (\neg A_1 \vee A_3)$	$pr6.6$
5.	$\beta \vdash (\neg A_1 \vee A_3)$	$MP, 3., 4.$
5.	$\beta \vdash (A_1 \to A_3)$	reescrita de 5.

cqd

Proposição 6.8 *Se $\beta \vdash (A \to C)$ e $\beta \vdash (B \to C)$, então $\beta \vdash ((A \vee B) \to C)$.*

Demonstração.

1.	$\beta \vdash (B \to C)$	hip
2.	$\beta \vdash (B \to C) \to ((A \vee B) \to (C \vee A))$	Ax_3
3.	$\beta \vdash (A \vee B) \to (C \vee A)$	$MP, 1., 2.$
4.	$\beta \vdash (A \to C)$	hip
5.	$\beta \vdash (A \to C) \to ((C \vee A) \to (C \vee C))$	Ax_3
6.	$\beta \vdash (C \vee A) \to (C \vee C)$	$MP, 4., 5.$
7.	$\beta \vdash (A \vee B) \to (C \vee C)$	$pr6.7, 3., 6.$
8.	$\beta \vdash (C \vee C) \to C$	Ax_1
9.	$\beta \vdash (A \vee B) \to C$	$pr6.7, 7., 8.$

cqd

Proposição 6.9 *Se* $\beta \vdash (A \to C)$ *e* $\beta \vdash (\neg A \to C)$, *então* $\beta \vdash C$.

Demonstração.

1.	$\beta \vdash (A \to C)$	hip
2.	$\beta \vdash (\neg A \to C)$	hip
3.	$\beta \vdash (A \vee \neg A) \to C$	$pr6.8, 1., 2.$
4.	$\beta \vdash (A \vee \neg A)$	$pr6.2$
5.	$\beta \vdash C$	$MP, 3., 4.$

cqd

Proposição 6.10 *Se* $\beta \vdash (A \to B)$ *então* $\beta \vdash (A \to (C \vee B))$ *e* $\beta \vdash (A \to (B \vee C))$.

Demonstração.

1.	$\beta \vdash (A \to B)$	hip
2.	$\beta \vdash B \to (C \vee B)$	Ax_2
3.	$\beta \vdash A \to (C \vee B)$	$pr6.7, 1., 2.$
4.	$\beta \vdash (C \vee B) \to (B \vee C)$	$pr6.3, pr6.6$
5.	$\beta \vdash A \to (B \vee C)$	$pr6.7, 3., 4.$

cqd

Proposição 6.11 (associatividade) *Temos que* $\vdash ((A \vee B) \vee C) \to (A \vee (B \vee C))$.

Demonstração.

1.	$\vdash (P \to P)$	$pr6.5$
2.	$\vdash A \to (A \vee (B \vee C))$	$pr6.3, 1., pr6.10$
3.	$\vdash B \to (B \vee C)$	$pr6.3, 1., pr6.10$
4.	$\vdash B \to (A \vee (B \vee C))$	$pr6.10, 3.$
5.	$\vdash (A \vee B) \to (A \vee (B \vee C))$	$pr6.8, 2., 4.$
6.	$\vdash C \to (B \vee C)$	$pr6.3, 2., pr6.10$
7.	$\vdash C \to (A \vee (B \vee C))$	$pr6.10, 6.$
8.	$\vdash ((A \vee B) \vee C) \to (A \vee (B \vee C))$	$pr6.8, 5., 7.$

cqd

Proposição 6.12 *Se* $\beta \vdash ((A \vee B) \vee C)$ *então* $\beta \vdash (A \vee (B \vee C))$.

Demonstração.

1.	$\beta \vdash (A \vee B) \vee C$	hip
2.	$\beta \vdash ((A \vee B) \vee C) \to (A \vee (B \vee C))$	$pr6.11$
3.	$\beta \vdash (A \vee (B \vee C))$	$MP, 1., 2.$

cqd

Proposição 6.13 *Se* $\beta \vdash (A \to B)$ *e* $\beta \vdash (A \to (B \to C))$*, então* $\beta \vdash (A \to C)$.

Demonstração.

1.	$\beta \vdash (A \to B)$	*hip*
2.	$\beta \vdash (\neg A \vee (\neg B \vee C))$	*hip*
3.	$\beta \vdash ((\neg B \vee C) \vee \neg A)$	$pr6.6, 2.$
4.	$\beta \vdash (\neg B \vee (C \vee \neg A))$	$pr12, 3.$
4.	$\beta \vdash (B \to (C \vee \neg A))$	reescrita
5.	$\beta \vdash (A \to (C \vee \neg A))$	$pr6.7, 1., 4.$
5.	$\beta \vdash (\neg A \vee (C \vee \neg A))$	reescrita
6.	$\beta \vdash ((C \vee \neg A) \vee \neg A)$	$pr6.6, 5.$
7.	$\beta \vdash (C \vee (\neg A \vee \neg A))$	$pr12, 6.$
8.	$\beta \vdash (\neg A \vee C)$	$pr6.1, Ax_1, 7.$
8.	$\beta \vdash (A \to c)$	reescrita

cqd

Lema 6.1 *Suponha que*

$$\beta \cup \{A\} \vdash B$$

e que $B \in \beta$*, ou* $B = A$*, ou* B *é axioma. Temos, então, que*

$$\beta \vdash (A \to B).$$

Teorema 6.1 (teorema da dedução - forma sintática) *Se* $\beta \cup \{A\} \vdash B$*, então* $\beta \vdash (A \to B)$.

Proposição 6.14 *Temos que* $\vdash (\neg A \to (\neg B \to \neg(A \vee B)))$.

Demonstração.

1.	$\vdash A \to \neg\neg A$	$pr6.3, pr6.4$
2.	$\vdash B \to \neg\neg B$	$pr6.3, pr6.4$
3.	$\vdash A \to (\neg\neg A \vee \neg\neg B)$	$pr6.10, 1.$
4.	$\vdash B \to (\neg\neg A \vee \neg\neg B)$	$pr6.10, 2.$
5.	$\vdash (A \vee B) \to (\neg\neg A \vee \neg\neg B)$	$pr6.8, 3., 4.$
6.	$\vdash (\neg\neg A \vee \neg\neg B) \vee \neg(A \vee B)$	$pr6.6, 5.$
7.	$\vdash \neg\neg A \vee (\neg\neg B \vee \neg(A \vee B))$	$pr12, 6.$
7.	$\vdash \neg A \to (\neg B \to \neg(A \vee B))$	reescrita

cqd

Proposição 6.15 *Temos que* $\vdash A \rightarrow (A \vee B)$ *e* $\vdash \neg A \rightarrow (\neg A \vee B)$.

Demonstração.

Prova de $\vdash A \rightarrow (A \vee B)$.

1.	$\vdash A \rightarrow A$	$pr6.3, pr6.5$
2.	$\vdash A \rightarrow (A \vee B)$	$pr6.10, 1.$

Prova de $\vdash \neg A \rightarrow (\neg A \vee B)$.

1.	$\vdash \neg A \vee \neg\neg A$	$pr6.3, pr6.4$
2.	$\vdash (\neg A \vee \neg\neg A) \rightarrow (\neg\neg A \vee \neg A)$	$pr6.3, pr6.6$
3.	$\vdash (\neg\neg A \vee \neg A)$	$MP, 1., 2.$
3.	$\vdash \neg A \rightarrow \neg A$	reescrita
4.	$\vdash \neg A \rightarrow (\neg A \vee B)$	$pr6.10, 3.$

cqd

Completude do Sistema Axiomático P_a

Teorema 6.2 (teorema da correção) *Seja* H *uma fórmula da Lógica Proposicional,*

$$se \vdash H \ então \vDash H.$$

Teorema 6.3 (teorema da completude) *Seja* H *uma fórmula da Lógica Proposicional.*

$$Se \vDash H \ então \vdash H.$$

Definição 6.7 (base associada a uma fórmula.) *Seja* H *uma fórmula e* $P_1, \ldots, P_n$ *os símbolos proposicionais contidos em* H.

Dada uma interpretação I, *então a base associada a* H *conforme* I, *denotada por* $B[H, I]$, *é um conjunto de literais, definidos a partir de* $P_1, \ldots, P_n$ *como se segue:*

- *se* $I[P_i] = T$, *então* $P_i \in B[H, I]$;
- *se* $I[P_i] = F$, *então* $\neg P_i \in B[H, I]$.

Lema 6.2 *Seja* H *uma fórmula e* $P_1, ..., P_n$ *os símbolos proposicionais contidos em* H. *Dada uma interpretação* I, *então:*

a) $I[H] = T \Rightarrow B[H, I] \vdash H$.

b) $I[H] = F \Rightarrow B[H, I] \vdash \neg H$.

Definição 6.8 (consistência de um sistema axiomático) *Um sistema axiomático é consistente se, e somente se, dada uma fórmula* H, *não se pode ter* $\vdash H$ *e* $\vdash \neg H$. *Isto é,* H *e* $\neg H$ *não podem ser teoremas ao mesmo tempo.*

Teorema 6.4 (consistência) *O sistema axiomático* P_a *é consistente.*

Definição 6.9 (consistência de um conjunto de fórmulas) *Um conjunto de fórmulas Γ é consistente se, e somente se, não existe fórmula H tal que $\Gamma \vdash H$ e $\Gamma \vdash \neg H$. Isto é, H e $\neg H$ não podem ser provadas a partir de Γ.*

Teorema 6.5 (consistência e satisfatibilidade) *Um conjunto de fórmulas Γ é consistente se, e somente se, é satisfatível.*

Exercícios

Exercícios de Computação

7

Tableaux semânticos e resolução na Lógica Proposicional

Introdução

O Sistema de *tableaux* Semânticos Tb_a

Definição 7.1 (elementos básicos de um sistema de *tableaux* semânticos) *Os elementos básicos do sistema de tableaux semânticos Tb_a, na Lógica Proposicional, são definidos pelos conjuntos:*

- *o alfabeto da Lógica Proposicional, Definição 1.1, sem os símbolos de verdade $false$ e $true$;*
- *o conjunto das fórmulas da Lógica Proposicional;*
- *um conjunto de regras de dedução.*

Definição 7.2 (regras de inferência do *tableau* semântico) *Sejam A e B duas fórmulas da Lógica Proposicional. As regras de inferência do sistema de tableaux semânticos Tb_a, na Lógica Proposicional, são $R_1, \ldots, R_9$ indicadas a seguir.*

$$R_1 = A \wedge B \qquad R_2 = A \vee B \qquad R_3 = A \rightarrow B$$

$$\begin{array}{c} A \\ B \end{array} \qquad \begin{array}{cc} \swarrow & \searrow \\ A & B \end{array} \qquad \begin{array}{cc} \swarrow & \searrow \\ \neg A & B \end{array}$$

$$R_4 = A \leftrightarrow B \qquad R_5 = \neg\neg A \qquad R_6 = \neg A \wedge B$$

$$\begin{array}{cc} \swarrow & \searrow \\ A \wedge B & \neg A \wedge \neg B \end{array} \qquad A \qquad \begin{array}{cc} \swarrow & \searrow \\ \neg A & \neg B \end{array}$$

$$R_7 = \neg(A \vee B) \qquad R_8 = \neg(A \rightarrow B) \qquad R_9 = \neg(A \leftrightarrow B)$$

$$\begin{array}{c} \neg A \\ \neg B \end{array} \qquad \begin{array}{c} A \\ \neg B \end{array} \qquad \begin{array}{cc} \swarrow & \searrow \\ \neg A \wedge B & A \wedge \neg B \end{array}$$

Heurística (aplicação de regras). Aplique preferencialmente as regras R_1, R_5, R_7 e R_8, que não bifurcam o *tableau*.

Definição 7.3 (construção de um *tableau* semântico) Um *tableau* semântico no sistema Tb_a, na Lógica Proposicional, é construído como se segue. Seja

$$\{A_1, \ldots, A_n\}$$

um conjunto de fórmulas.[1]

- A árvore tab_1, a seguir, com apenas um ramo, é um *tableau* iniciado com $\{A_1, \ldots, A_n\}$.

$$\begin{array}{ll} 1. & A_1 \\ 2. & A_2 \\ & . \\ & . \\ & . \\ n. & A_n \end{array}$$

  Nesse *tableau*, as fórmulas $\{A_1, \ldots, A_n\}$ podem ser escritas em qualquer ordem.

- Se tab_2 é a árvore resultante da aplicação de uma das regras $(R_1, \ldots, R_9)$ à árvore tab_1, então tab_2 é também um *tableau* iniciado com $\{A_1, \ldots, A_n\}$.

  Seguindo esse procedimento, expandimos o *tableau* iniciado com $\{A_1, \ldots, A_n\}$.

- Seja $tab_i, i \geq 2$, um *tableau* iniciado com $\{A_1, \ldots, A_n\}$. Se tab_{i+1} é a árvore resultante da aplicação de uma das regras $(R_1, \ldots, R_9)$ à árvore tab_i, então tab_{i+1} é também um *tableau* iniciado com $\{A_1, \ldots, A_n\}$.

Definição 7.4 (ramo) *No sistema Tb_a, um ramo em um tableau é uma seqüência de fórmulas $H_1, \ldots, H_n$, onde H_1 é a primeira fórmula do tableau e, nessa seqüência, H_{i+1} é derivada de H_i, $1 \leq i < n$, utilizando alguma regra de Tb_a.*

Definição 7.5 (ramo fechado) *No sistema Tb_a, um ramo em um tableau é fechado se ele contém uma fórmula A e sua negação $\neg A$.*

Definição 7.6 (ramo saturado) *No sistema Tb_a, um ramo em um tableau é saturado se para toda fórmula A, do ramo:*

- *já foi aplicada alguma regra do sistema Tb_a à fórmula A, ou seja: A já foi expandida por alguma regra; ou*
- *não é possível aplicar nenhuma regra do sistema Tb_a à fórmula A, isto é, A é igual a um literal e não é possível expandi-la por alguma regra.*

Definição 7.7 (ramo aberto) *No sistema Tb_a, um ramo em um tableau é aberto se ele é saturado e não é fechado.*

Definição 7.8 (*tableau* fechado) *No sistema Tb_a, um tableau é fechado quando todos os seus ramos são fechados.*

Definição 7.9 (*tableau* aberto) *No sistema Tb_a, um tableau é aberto se ele possui algum ramo aberto.*

[1] O conjunto de fórmulas $\{A_1, \ldots, A_n\}$ é finito. Para simplificar, a possibilidade de esse conjunto ser infinito não é considerada neste livro.

Conseqüência Lógica no Sistema de *tableaux* Semânticos Tb_a

Definição 7.10 (prova e teorema em *tableaux* semânticos) *Seja H uma fórmula. Uma prova de H, no sistema Tb_a, é um tableau fechado iniciado com a fórmula $\neg H$. Nesse caso, H é um teorema do sistema de tableaux semânticos Tb_a.*

Teorema 7.1 (completude) *Seja H uma fórmula da Lógica Proposicional.*

$$\textit{Se } H \textit{ é uma tautologia, então existe uma prova de } H \textit{ no sistema } Tb_a.$$

Teorema 7.2 (correção) *Seja H uma fórmula da Lógica Proposicional. No sistema Tb_a,*

$$\textit{se } \vdash H, \textit{ então } \vDash H.$$

Notação. Dada uma fórmula H, se H é conseqüência lógica de um conjunto de hipóteses

$$\beta = \{A_1, \ldots, A_n\},$$

no sistema Tb_a, então esse fato é indicado pela notação

$$\beta \vdash H \text{ ou } \{A_1, \ldots, A_n\} \vdash H.$$

Observe que essa notação é análoga àquela utilizada para conseqüência sintática no sistema P_a. O sistema que estiver sendo considerado, P_a ou Tb_a, deve ficar claro no contexto.

O Sistema de Resolução Rs_a

Definição 7.11 (cláusula) *Uma cláusula, na Lógica Proposicional, é uma disjunção de literais. No caso de uma disjunção de zero literal, temos a cláusula vazia.*

Notação. A disjunção de zero literal é a cláusula vazia. Tal cláusula é representada, na notação de conjunto, por $\{\}$.

Definição 7.12 (literais complementares) *Dois literais são complementares se um é a negação do outro. Isto é, $\breve{P}$ e $\neg\breve{P}$ são literais complementares.*

Definição 7.13 (resolvente de duas cláusulas) *Considere duas cláusulas*

$$C_1 = \{A_1, \ldots, A_n\}, \textit{ e } C_2 = \{B_1, \ldots, B_n\},$$

que possuem literais complementares.

Suponha λ um literal em C_1 tal que seu complementar, $\neg\lambda$, pertence a C_2. O resolvente de C_1 e C_2, denominado por

$$Res(C_1, C_2),$$

é definido por:

$$Res(C_1, C_2) = (C_1 - \{\lambda\}) \cup (C_2 - \{\neg\lambda\}).$$

Se $Res(C_1, C_2) = \{\}$, temos um resolvente vazio.

Definição 7.14 (elementos básicos da resolução) *Os elementos básicos do sistema de resolução Rs_a, na Lógica Proposicional, são definidos pelos conjuntos:*

- *o alfabeto da Lógica Proposicional, Definição 1.1, sem os símbolos de verdade $false$ e $true$;*
- *o conjunto das cláusulas da Lógica Proposicional;*
- *a regra de resolução.*

Definição 7.15 (regra de resolução) *No sistema de resolução Rs_a, dadas duas cláusulas*

$$C_1 = \{A_1, \ldots, A_n\}, C_2 = \{B_1, \ldots, B_n\},$$

a regra de resolução aplicada a C_1 e C_2 é definida pelo procedimento a seguir:

$$\textit{tendo } C_1 \textit{ e } C_2, \textit{ deduza } Res(C_1, C_2) \ .$$

Definição 7.16 (construção de uma expansão por resolução) *No sistema de resolução Rs_a, uma expansão por resolução é construída como se segue. Seja $\{A_1, \ldots, A_n\}$ um conjunto de cláusulas.*

- *A estrutura a seguir é uma expansão por resolução sobre $\{A_1, \ldots, A_n\}$.*

  1. A_1
  2. A_2
  .
  .
  .
  n. A_n

  Nessa expansão, as fórmulas $\{A_1, \ldots, A_n\}$ podem ser escritas em qualquer ordem.

- *Seja Exp_2 uma expansão por resolução sobre $\{A_1, \ldots, A_n\}$, obtida pela adição de*

  $$Res(A_i, A_j), i, j \leq n, \ i \neq j,$$

  à expansão Exp_1. A expansão Exp_2 é também uma expansão por resolução sobre $\{A_1, \ldots, A_n\}$.

  Seguindo esse procedimento, a expansão por resolução sobre $\{A_1, \ldots, A_n\}$ é incrementada.

- *Seja $Exp_k, k > 1$, uma expansão por resolução sobre $\{A_1, \ldots, A_n\}$. Considere Exp_{k+1} a expansão por resolução obtida pela adição de*

  $$Res(H_i, H_j), H_i, H_j \in Exp_k, \ i, j \leq k, \ i \neq j,$$

  à expansão Exp_k. A expansão Exp_{k+1} é também uma expansão por resolução sobre $\{A_1, \ldots, A_n\}$.

Conseqüência Lógica na Resolução

Definição 7.17 (forma clausal) *Dada uma fórmula H, uma forma clausal associada a H é uma fórmula H_c tal que H_c é uma conjunção de cláusulas e H_c equivale a H.*

Definição 7.18 (prova por resolução) *Seja H uma fórmula e $\neg H_c$ a forma clausal associada a $\neg H$. No sistema de resolução Rs_a, uma prova de H é uma expansão por resolução fechada sobre o conjunto de cláusulas de $\neg H_c$. Nesse caso, H é um teorema do sistema de resolução.*

Teorema 7.3 (completude) *Seja H uma fórmula da Lógica Proposicional. No sistema de resolução Rs_a,*

se H é uma tautologia, então existe uma prova de H.

Teorema 7.4 (correção) *Seja H uma fórmula da Lógica Proposicional. No sistema de resolução Rs_a,*

se existe uma prova de H, então H é uma tautologia.

Definição 7.19 (conseqüência lógica por resolução) *Dada uma fórmula H e um conjunto*[2] *de hipóteses*

$$\beta = \{A_1, \ldots, A_n\},$$

então H é uma conseqüência lógica de β, no sistema de resolução Rs_a, se existe uma prova de

$$(A_1 \wedge \ldots \wedge A_n) \rightarrow H.$$

Notação. Dada uma fórmula H, se H é conseqüência lógica de um conjunto de hipóteses

$$\beta = \{A_1, \ldots, A_n\},$$

no sistema de resolução Rs_a, então esse fato é indicado pela notação

$$\beta \vdash H \text{ ou } \{A_1, \ldots, A_n\} \vdash H.$$

Exercícios

Exercícios de Computação

[2] Neste livro, consideramos apenas conjuntos de hipóteses finitos.

8

A linguagem da Lógica de Predicados

Introdução

Alfabeto

Definição 8.1 (alfabeto) *O alfabeto da Lógica de Predicados é constituído por:*

- *símbolos de pontuação: (,);*
- *símbolo de verdade:* $false$*;*
- *um conjunto enumerável de símbolos para variáveis:* $x, y, z, w, x_1, y_1, \ldots$;
- *um conjunto enumerável de símbolos para funções:* $f, g, h, f_1, g_1, h_1, f_2, g_2, \ldots$;
- *um conjunto enumerável de símbolos para predicados:* $p, q, r, p_1, q_1, r_1, p_2, q_2, \ldots$;
- *conectivos:* $\neg, \vee, \forall, \exists$.

Associado a cada símbolo para função ou predicado, temos um número inteiro não-negativo k*. Esse número indica a aridade, ou seja, o número de argumentos da função ou predicado.*

Variáveis.

Variáveis e metavariáveis.

Funções e predicados.

Constantes e símbolos proposicionais.

Conectivos.

Elementos Básicos da Linguagem

Definição 8.2 (termo) *O conjunto dos termos da linguagem da Lógica de Predicados é o menor conjunto que satisfaz as regras a seguir:*

- *as variáveis são termos;*
- *se $t_1, t_2, \ldots, t_n$ são termos e $\breve{f}$ é um símbolo para função n-ária, então $\breve{f}(t_1, t_2, \ldots, t_n)$ é um termo.*

Definição 8.3 (átomo) *O conjunto dos átomos da linguagem da Lógica de Predicados é o menor conjunto que satisfaz as regras a seguir:*

- *o símbolo de verdade $false$ é um átomo;*
- *se $t_1, t_2, \ldots, t_n$ são termos e $\breve{p}$ é um símbolo para predicado n-ário, então, $\breve{p}(t_1, t_2, ..., t_n)$ é um átomo.*

Fórmulas

Definição 8.4 (fórmula) *O conjunto das fórmulas da linguagem da Lógica de Predicados é o menor conjunto que satisfaz as regras a seguir.*

- *Todo átomo é uma fórmula.*
- *Se H é uma fórmula, então $(\neg H)$ é uma fórmula.*
- *Se H e G são fórmulas, então $(H \vee G)$ é uma fórmula.*
- *Se H é uma fórmula e $\breve{x}$ uma variável, então $((\forall \breve{x})H)$ e $((\exists \breve{x})H)$ são fórmulas.*

Definição 8.5 (expressão) *Uma expressão da Lógica de Predicados é um termo ou uma fórmula.*

Definição 8.6 (subtermo, subfórmula, subexpressão) *Os elementos a seguir definem as partes de um termo ou fórmula E.*

- *Se $E = \breve{x}$, então a variável $\breve{x}$ é um subtermo de E*
- *Se $E = \breve{f}(t_1, t_2, \ldots, t_n)$, então t_i e $\breve{f}(t_1, t_2, \ldots, t_n)$ são subtermos de E.*
- *Se t_1 é subtermo de t_2 e t_2 é subtermo de E, então t_1 é subtermo de E.*
- *Se $E = (\neg H)$ então H e $(\neg H)$ são subfórmulas de E.*
- *Se E é uma das fórmulas $(H \vee G)$, $(H \wedge G)$, $(H \rightarrow G)$ ou $(H \leftrightarrow G)$, então H, G e E são subfórmulas de E.*
- *Se $\breve{x}$ é uma variável, $\triangle$ um dos quantificadores $\forall$ ou $\exists$ e $E = ((\triangle \breve{x})H)$, então H e $((\triangle \breve{x})H)$ são subfórmulas de E.*
- *Se H_1 é subfórmula de H_2 e H_2 é subfórmula de E, então H_1 é subfórmula de E.*
- *Todo subtermo ou subfórmula é também uma subexpressão.*

Definição 8.7 (literal) *Um literal, na Lógica de Predicados, é um átomo ou a negação de um átomo. Um átomo é um literal positivo. A negação de um átomo é um literal negativo.*

Definição 8.8 (forma normal) *Seja H uma fórmula da Lógica de Predicados.*

- *H está na forma normal conjuntiva, fnc, se é uma conjunção de disjunções de literais.*
- *H está na forma normal disjuntiva, fnd, se é uma disjunção de conjunções de literais.*

Definição 8.9 (ordem de precedência) *Na Lógica de Predicados, a ordem de precedência dos conectivos é a seguinte:*

- *maior precedência: $\neg$;*
- *precedência intermediária superior: $\forall$, $\exists$;*
- *precedência intermediária inferior: $\rightarrow$, $\leftrightarrow$;*
- *precedência inferior: $\vee$, $\wedge$.*

Correspondência entre quantificadores.

Definição 8.10 (comprimento de uma fórmula) *Dada uma fórmula H, da Lógica de Predicados, o comprimento de H, denotado por $comp[H]$, é definido como se segue:*

- *se H é um átomo, então $comp[H] = 1$;*
- *se $H = \neg G$, então $comp[\neg G] = 1 + comp[G]$;*
- *se $H = (E \Diamond G)$, onde $\Diamond$ é um dos conectivos $\vee$,$\wedge$,$\rightarrow$,$\leftrightarrow$ então $comp[E \Diamond G] = 1 + comp[E] + comp[G]$;*
- *se $H = (\triangle \breve{x})G$, onde $\triangle$ é um dos quantificadores $\forall$ ou $\exists$, então $comp[(\triangle \breve{x})G] = 1 + comp[G]$.*

O Princípio da Indução na Lógica de Predicados

Proposição 8.1 (princípio da indução na Lógica de Predicados) *Seja $B[E]$ uma asserção que se refere a uma fórmula E da Lógica de Predicados. Se as duas propriedades a) e b) a seguir são verdadeiras, então concluímos que $B[E]$ é verdadeira para qualquer fórmula E.*

a) Base da Indução. $B[A]$ é verdadeira para todo átomo A.

b) Passo da indução. Sejam G e H duas fórmulas. Se $B[G]$ e $B[H]$ são verdadeiras, então $B[\neg H]$, $B[G \vee H]$ e $B[(\forall x)H]$ são verdadeiras.

Proposição 8.2 (comprimento de uma fórmula) *Sejam H e G duas fórmulas da Lógica de Predicados.*

Se G é uma subfórmula de H, então $comp[G] \leq comp[H]$.

Classificações de variáveis.

Definição 8.11 (escopo de um quantificador) *Seja E uma fórmula da Lógica de Predicados.*

- *Se $(\forall \breve{x})H$ é uma subfórmula de E, então o escopo de $(\forall \breve{x})$ em E é a subfórmula H.*
- *Se $(\exists \breve{x})H$ é uma subfórmula de E, então o escopo de $(\exists \breve{x})$ em E é a subfórmula H.*

Definição 8.12 (ocorrência livre e ligada) *Sejam $\breve{x}$ uma variável e E uma fórmula.*

- *Uma ocorrência de $\breve{x}$ em E é ligada se $\breve{x}$ está no escopo de um quantificador $(\forall \breve{x})$ ou $(\exists \breve{x})$ em E.*
- *Uma ocorrência de $\breve{x}$ em E é livre se não for ligada.*

Definição 8.13 (variável livre e ligada) *Sejam $\breve{x}$ uma variável e E uma fórmula que contém $\breve{x}$.*

- *A variável $\breve{x}$ é ligada em E se existe pelo menos uma ocorrência ligada de $\breve{x}$ em E.*
- *A variável $\breve{x}$ é livre em E se existe pelo menos uma ocorrência livre de $\breve{x}$ em E.*

Definição 8.14 (símbolo livre) *Dada uma fórmula E, os seus símbolos livres são as variáveis que ocorrem livres em E, os símbolos de função e os símbolos de predicado.*

Definição 8.15 (fórmula fechada) *Uma fórmula é fechada quando não possui variáveis livres.*

Definição 8.16 (fecho de uma fórmula) *Seja H uma fórmula da Lógica de Predicados e $\{\breve{x}_1, ..., \breve{x}_n\}$ o conjunto das variáveis livres em H.*

- *O fecho universal de H, indicado por $(\forall *)H$, é dado pela fórmula $(\forall \breve{x}_1)...(\forall \breve{x}_n)H$.*
- *O fecho existencial de H,indicado por $(\exists *)H$, é dado pela fórmula $(\exists \breve{x}_1)...(\exists \breve{x}_n)H$.*

Exercícios

Exercícios de Computação

9

A semântica da Lógica de Predicados

Introdução

Interpretação das Variáveis, Funções e Predicados

Definição 9.1 (interpretação de variáveis, funções e predicados) *Seja U um conjunto não-vazio. Uma interpretação I sobre o domínio U, na lógica de predicados, é uma função tal que:*

- *o domínio da função I é o conjunto dos símbolos de função, de predicados e das expressões da Lógica de Predicados.*

A interpretação das variáveis, funções e predicados é dada por:

- *para toda variável $\breve{x}$, se $I[\breve{x}] = \breve{x}_I$, então $\breve{x}_I \in U$;*
- *para todo símbolo de função $\breve{f}$, n-ário, se $I[\breve{f}] = \breve{f}_I$, então $\breve{f}_I$ é uma função n-ária em U, isto é, $\breve{f}_I : U^n \to U$;*
- *para todo símbolo de predicado $\breve{p}$, n-ário, se $I[\breve{p}] = \breve{p}_I$, então $\breve{p}_I$ é um predicado n-ário em U, isto é, $\breve{p}_I : U^n \to \{T, F\}$;*
- *o caso em que E é uma expressão, $I[E]$ é definida por um conjunto de regras semânticas consideradas mais adiante.*

Regras Semânticas para Interpretação de Expressões

Definição 9.2 (regras semânticas para interpretação de expressões) *Seja E uma expressão e I uma interpretação sobre o domínio U. A interpretação de E conforme I, indicada por $I[E]$, é determinada pelas regras:*

- *se $E = false$, então $I[E] = I[false] = F$;*
- *se $E = \breve{f}(t_1, ..., t_n)$ onde $\breve{f}(t_1, ..., t_n)$ é um termo, então $I[E] = I[\breve{f}(t_1, ..., t_{nI})] = \breve{f}_I(t_{1_I}, ..., t_{n_I})$ onde $I[\breve{f}] = \breve{f}_I$ e para todo termo t_i, $I[t_i] = t_{i_I}$;*
- *se $E = \breve{p}(t_1, ..., t_n)$ onde $\breve{p}(t_1, ..., t_n)$ é um átomo, então $I[E] = I[\breve{p}(t_1, ..., t_n)] = p_I(t_{1_I}, ..., t_{n_I})$ onde $I[\breve{p}] = \breve{p}_I$ e para todo termo t_i, $I[t_i] = t_{i_I}$;*
- *se $E = \neg H$ onde H é uma fórmula, então $I[E] = I[\neg H] = T$ se $I[H] = F$ e $I[E] = I[\neg H] = F$ se $I[H] = T$;*
- *se $E = H \vee G$, onde H e G são duas fórmulas, então $I[E] = I[H \vee G] = T$ se $I[H] = T$ e/ou $I[G] = T$ e $I[E] = I[H \vee G] = F$ se $I[H] = I[G] = F$;*
- *os casos em que $E = (\forall x)H$ e $E = (\exists x)H$ são considerados adiante.*

Regras Semânticas para Interpretação de Fórmulas com Quantificadores

Definição 9.3 (interpretação estendida) *Seja I uma interpretação sobre um domínio U. Considere $\breve{x}$ uma variável da Lógica de Predicados e d um elemento de U.*

Uma extensão de I, conforme $\breve{x}$ e d, é uma interpretação sobre U, denotada por $< \breve{x} \leftarrow d > I$, tal que:

$$< x \leftarrow d > I[\gamma] = \begin{cases} d & se\ \gamma = x \\ I[\gamma] & se\ \gamma \neq x \end{cases}$$

onde γ é uma variável qualquer.

Definição 9.4 (regras semânticas para interpretação de fórmulas com quantificadores) *Sejam H uma fórmula, $\breve{x}$ uma variável e I uma interpretação sobre o domínio U.*

Os valores semânticos de $I[(\forall \breve{x})H]$ e $I[(\exists \breve{x})H]$ são definidos pelas regras:

- $I[(\forall \breve{x})H] = T$ *se, e somente se,* $\forall d \in U,\ < \breve{x} \leftarrow d > I[H] = T$;
- $I[(\forall \breve{x})H] = F$ *se, e somente se,* $\exists d \in U;\ < \breve{x} \leftarrow d > I[H] = F$;
- $I[(\exists \breve{x})H] = T$ *se, e somente se,* $\exists d \in U;\ < \breve{x} \leftarrow d > I[H] = T$;
- $I[(\exists \breve{x})H] = F$ *se, e somente se,* $\forall d \in U,\ < \breve{x} \leftarrow d > I[H] = F$.

Representação de sentenças na lógica de predicados

Exercícios

Exercícios de Computação

10

Propriedades semânticas da Lógica de Predicados

Introdução

Propriedades Semânticas

Definição 10.1 (propriedades semânticas básicas da Lógica de Predicados) *Sejam*

$$H, G, H_1, H_2, \ldots, H_n$$

fórmulas da Lógica de Predicados. As propriedades semânticas básicas da Lógica de Predicados são definidas a seguir.

- *H é válida se, e somente se, para toda interpretação I, $I[H] = T$. No caso em que a análise da interpretação de H não requer a interpretação de quantificadores, então H é tautologicamente válida.*
- *H é satisfatível se, e somente se, existe pelo menos uma interpretação I, tal que $I[H] = T$.*
- *H é uma contingência se, e somente se, existem pelo menos duas interpretações I_1 e I_2, tais que $I_1[H] = T$ e $I_2[H] = F$.*
- *H é contraditória se, e somente se, para toda interpretação I, $I[H] = F$.*
- *H implica semanticamente[1] G se, e somente se, para toda interpretação I, se $I[H] = T$ então $I[G] = T$.*
- *H equivale semanticamente[2] a G se, e somente se, para toda interpretação I, $I[H] = I[G]$.*
- *Uma interpretação I satisfaz H se $I[H] = T$.*
- *O conjunto*

  $$\beta = \{H_1, H_2, \ldots, H_n, \ldots\}$$

  é satisfatível se, e somente se, existe uma interpretação I, tal que

  $$I[H_1] = I[H_2] = \ldots = I[H_n] = \ldots = T.$$

  Nesse caso, I satisfaz o conjunto de fórmulas, o que é indicado por $I[\beta] = T$. Dado um conjunto de fórmulas vazio, então toda interpretação I satisfaz esse conjunto.

[1] A implicação semântica na Lógica de Predicados é também denominada implicação lógica.
[2] A equivalência semântica na Lógica de Predicados é também denominada equivalência lógica.

- *O conjunto*

$$\beta = \{H_1, H_2, \ldots, H_n, \ldots\},$$

implica semanticamente uma fórmula H*, se para toda interpretação* I*; se* $I[\beta] = T$*, então* $I[H] = T$*.*

Notação. Como na Lógica Proposicional, se H é uma conseqüência lógica semântica de um conjunto de fórmulas β, então tal fato é indicado por $\beta \vDash H$.

Notação. Para simplificar, muitas vezes é utilizado neste livro apenas o termo "implicação" no lugar de "implicação semântica", ou "implicação sintática". É o contexto quem determina qual tipo de termo está sendo utilizado. De forma análoga, o termo "equivalência" pode representar "equivalência semântica", ou "equivalência sintática". Se a implicação ou equivalência é uma implicação ou equivalência semântica da Lógica Proposicional ou de Predicados, tal fato também deve estar indicado implicitamente no contexto. Além disso, a notação $\vDash H$ também indica que H é tautologia ou é válida.

Satisfatibilidade de Fórmulas

Validade de fórmulas

Implicações e Equivalências entre Fórmulas

Proposição 10.1 (implicação) *Dada uma fórmula* H *e* $\breve{x}$ *uma variável qualquer da Lógica de Predicados,*

$$\textit{se } H \textit{ é válida, então } (\forall \breve{x})H \textit{ é válida.}$$

Proposição 10.2 (insatisfatibilidade) *Considere as fórmulas*

$$H = (\forall x)(\exists y)E(x, y) \textit{ e } H_s = (\forall x)E(x, f(x)),$$

onde E *é uma fórmula que contém as variáveis livres* x *e* y*; e* f *é uma função qualquer.*

$$\textit{Se } H \textit{ é insatisfatível então } H_s \textit{ é insatisfatível.}$$

Lema 10.1 (interpretação estendida e variável ligada) *Seja* H *uma fórmula na qual a variável* $\breve{x}$ *não ocorre livre. Dada uma interpretação* I *sobre um domínio* U*, então*

$$\forall\ d \in U,\ < \breve{x} \leftarrow d > I[H] = I[H]$$

O conjunto das fórmulas válidas não é decidível.

Exercícios

Exercícios de Computação

11

Programação Lógica

Introdução

Sintaxe da Programação Lógica

Definição 11.1 (cláusula de programa) *Uma cláusula de programa, na Lógica de Predicados, é uma cláusula do tipo*

$$C = (\forall x_1) \ldots (\forall x_n) G,$$

onde G é uma disjunção de literais, que contém exatamente um literal positivo.

Notação. Uma cláusula de programa

$$(\forall *)(B \vee \neg A_1 \vee \ldots \vee \neg A_n)$$

é denotada por

$$B \leftarrow A_1, \ldots, A_n.$$

Nesse caso, B é a cabeça da cláusula e $A_1, \ldots, A_n$ é a cauda.

Definição 11.2 (cláusula unitária) *Uma cláusula de programa unitária é uma cláusula do tipo $B \leftarrow$. Nesse caso, a cláusula não contém literais negativos.*

Definição 11.3 (programa lógico) *Um programa lógico é um conjunto de cláusulas de programa.*

Definição 11.4 (cláusula objetivo) *Uma cláusula objetivo é uma cláusula do tipo $\leftarrow A_1, \ldots, A_n$. Nesse caso a cláusula não contém literal positivo e não é uma cláusula de programa.*

Definição 11.5 (cláusula vazia) *Uma cláusula vazia é uma cláusula que não contém nenhum literal. Uma cláusula vazia C é denotada por $C = \blacksquare$. Observe que a cláusula vazia não é uma cláusula de programa.*

Algoritmo da Unificação

Definição 11.6 (substituição) *Uma substituição na Lógica de Predicados é um conjunto*

$$\theta = \{\breve{x}_1 \hookleftarrow t_1, ..., \breve{x}_n \hookleftarrow t_n\},$$

onde para todo i, $\breve{x}_i$ é variável e t_i é termo tal que $\breve{x}_i \neq t_i$. Além disso, para todo i, j, tem-se que $\breve{x}_i \neq \breve{x}_j$ se $i \neq j$. O conjunto vazio $\{\}$ é a substituição vazia.

Notação. Se o conjunto S é unitário, isto é, $S = \{E\}$, então a aplicação de θ a S é igual à aplicação a E, sendo denotada por: $S\theta = E\theta$.

Definição 11.7 (composição de substituições) *Considere as substituições*

$$\theta_1 = \{x_1 \hookleftarrow t_1, ..., x_n \hookleftarrow t_n\} \text{ e } \theta_2 = \{y_1 \hookleftarrow s_1, ..., y_m \hookleftarrow s_m\}.$$

A composição de θ_1 e θ_2, denotada por $\theta_1\theta_2$, é calculada como se segue:

1. *Construa o conjunto*
   $\phi_1 = \{x_1 \hookleftarrow t_1\theta_2, ..., x_n \hookleftarrow t_n\theta_2, y_1 \hookleftarrow s_1, ..., y_m \hookleftarrow s_m\}$.
2. *Retire de ϕ_1 as ligações $y_i \hookleftarrow s_i$ tal que $y_i = x_j$ para algum j; $1 \leq j \leq n$.*
   Faça ϕ_2 igual ao novo conjunto.
3. *Retire de ϕ_2 as ligações $x_i \hookleftarrow t_i\theta_2$ tal que $x_i = t_i\theta_2$. Faça $\theta_1\theta_2$ igual ao conjunto obtido.*

Proposição 11.1 (composição de substituições) *Considere as substituições θ_1, θ_2 e θ_3 e uma expressão C. Tem-se que*

a) $\theta_1\{\} = \{\}\theta_1 = \theta_1$.

b) $(C\theta_1)\theta_2 = C(\theta_1\theta_2)$.

c) $\theta_1(\theta_2\theta_3) = (\theta_1\theta_2)\theta_3$.

Definição 11.8 (conjunto de diferenças) *Seja $S = \{A_1, ..., A_n\}$ um conjunto finito de expressões. O conjunto de diferenças de S é determinado pelos procedimentos a seguir.*

1. *Aponte para o símbolo mais à esquerda em cada expressão A_i, $1 \leq i \leq m$.*
2. *Enquanto todos os símbolos apontados coincidirem, desloque simultaneamente o apontador para o próximo símbolo, à direita, em cada expressão A_i.*
3. *Se forem encontrados símbolos apontados que não coincidem, então retire a subexpressão E_i de cada expressão A_i, que inicia no símbolo de diferença. Faça o conjunto de diferenças de S igual a $D = \{E_1, ..., E_m\}$. Caso contrário, faça $D = \{\}$.*

Definição 11.9 (expressões unificáveis) *Um conjunto de expressões S é unificável se existe uma substituição θ tal que $|S\theta| = 1$. Nesse caso, θ é denominada unificador de S.*

Definição 11.10 (unificador mais geral) *Seja θ um unificador do conjunto de expressões S. θ é um unificador mais geral de S, umg, se para qualquer unificador φ de S, existe uma substituição ϕ tal que $\varphi = \theta\phi$.*

Definição 11.11 (algoritmo da unificação) *Seja S um conjunto de expressões da Lógica de Predicados. Se S é unificável, o algoritmo a seguir determina um umg de S, caso contrário, ele indica que S não é unificável. Sejam $k \in \mathbb{N}$ e θ_k substituições.*

1. *Faça $k = 0$ e $\theta_0 = \{\}$.*
2. *Se $|S\theta_k| = 1$, então pare! θ_k é um umg de S.*
   Caso contrário, determine o conjunto de diferenças D_k de $S\theta_k$.
3. *Se existe uma variável x e um termo t em D_k tal que x não ocorre em t, então faça $\theta_{k+1} = \theta_k\{x \hookleftarrow t\}, k = k + 1$, vá para o passo 2.*
   Caso contrário, pare! S não é unificável.

Resolução-SLD

Definição 11.12 (regra de computação) *Uma regra de computação é uma função que seleciona um literal a partir de uma lista de literais de uma cláusula objetivo.*

Definição 11.13 (resolvente-SLD) *Considere uma cláusula objetivo*

$$G_i = (\leftarrow A_1, \ldots A_m, \ldots, A_k),$$

uma cláusula de programa

$$C_{i+1} = (A \leftarrow B_1, \ldots, B_q),$$

e Rc uma regra de computação. A cláusula objetivo G_{i+1} é o resolvente-SLD de G_i e C_{i+1} utilizando um unificador mais geral θ_{i+1} via Rc se as condições são satisfeitas.

- $Rc(G_i) = A_m$,
- $A_m\theta_{i+1} = A\theta_{i+1}$ *onde* θ_{i+1} *é um umg de* $\{A_m, A\}$.
- G_{i+1} *é a cláusula objetivo*

  $$\{G_{i+1}\} = \{(\leftarrow A_1, \ldots, A_{m-1}, B_1, \ldots, B_q, A_{m+1}, \ldots, A_k)\}\theta_{i+1}.$$

  Nesse caso, G_{i+1} é denotada por:

  $$G_{i+1} = Res(G_i, C_{i+1}, \theta_{i+1}, Rc).$$

Notação. Dada uma cláusula objetivo $G_i = (\leftarrow A_1, \ldots, A_m, \ldots, A_k)$, uma cláusula de programa $C_{i+1} = (A \leftarrow B_1, \ldots, B_q)$ e Rc uma regra de computação. Se $G_{i+1} = Res(G_i, C_{i+1}, \theta_{i+1}, Rc)$, então a notação da Figura 11.1 é utilizada. □

Definição 11.14 (variações) *Duas cláusulas de programa C_1 e C_2 são variações se existem substituições θ e φ tais que $C_1 = C_2\theta$ e $C_2 = C_1\varphi$. Além disso, as substituições θ e φ só possuem ligações do tipo $x \leftarrow y$, que renomeiam uma variável por outra.*

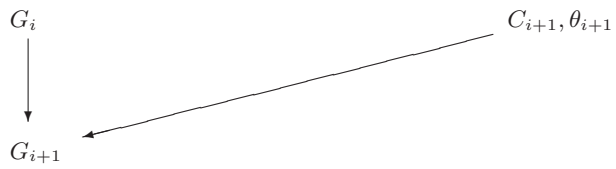

Figura 11.1. Notação para resolvente SLD.

Definição 11.15 (derivação-SLD) *Sejam P_l um programa lógico, G uma cláusula objetivo e Rc uma regra de computação. Uma derivação-SLD de*

$$P_l \cup \{G\}$$

via Rc é uma seqüência

$$G_0, G_1, G_2, \ldots,$$

tal que

$$G = G_0$$

e

$$G_{i+1} = Res(G_i, C_{i+1}, \theta_{i+1}, Rc).$$

Cada cláusula C_i é uma variação de uma cláusula de P_l, sendo denominada cláusula de entrada. Veja Figura 11.2.

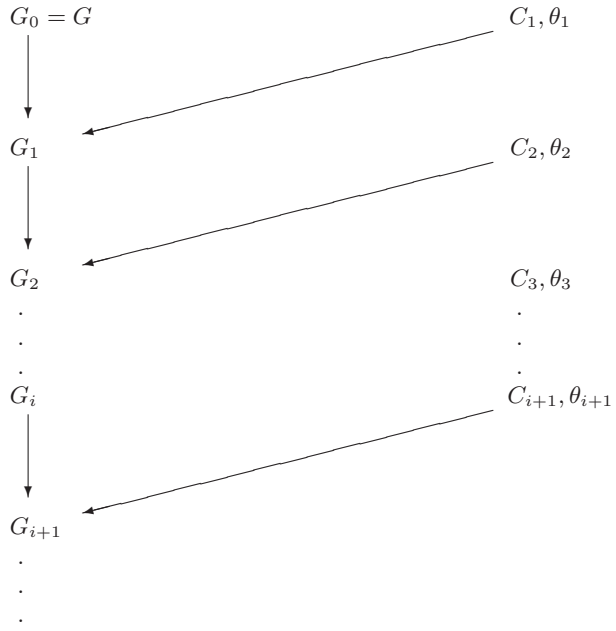

Figura 11.2. Notação para resolvente-SLD.

Definição 11.16 (refutação-SLD) *Sejam P_l um programa lógico, G uma cláusula objetivo e Rc uma regra de computação.*

- *Uma derivação-SLD de $P_l \cup \{G\}$ via Rc é fechada se é finita e a última cláusula é vazia; caso contrário, ela é aberta.*
- *Uma refutação-SLD de $P_l \cup \{G\}$ de comprimento n é uma derivação-SLD de $P_l \cup \{G\}$ via Rc dada pela seqüência $G_0 = G, G_1, \ldots, G_n$ onde $G_n = \{\}$. Como $G_n = \{\}$, a derivação é fechada.*

Definição 11.17 (substituição resposta associada a uma refutação-SLD) *Sejam P_l um programa lógico, G uma cláusula objetivo e Rc uma regra de computação tais que existe uma refutação-SLD de $P_l \cup \{G\}$ via Rc. Se a seqüência de substituições utilizadas na refutação-SLD é $\theta_1, \ldots, \theta_n$, então a composição $\theta_1 \ldots \theta_n$ é a substituição resposta associada a refutação-SLD via Rc.*

Definição 11.18 (prova por refutação-SLD) *Seja G uma cláusula objetivo e P_l um programa lógico. Uma prova de G, por resolução-SLD a partir de P_l, é uma refutação-SLD de $P_l \cup \{G\}$.*

Definição 11.19 (conseqüência lógica por resolução-SLD) *Seja G uma cláusula objetivo e P_l um programa lógico. G é uma conseqüência lógica por resolução-SLD de P_l se existe uma refutação-SLD de $P_l \cup \{G\}$.*

Teorema 11.1 (teorema da completude) *Sejam P_l um programa lógico e $G = (\leftarrow A_1, \ldots, A_n)$ uma cláusula objetivo. Se $P_l \rightarrow (A_1 \wedge \ldots \wedge A_n)$ é válida, então existe uma refutação-SLD de $P_l \cup \{G\}$.*

Teorema 11.2 (teorema da correção) *Sejam P_l um programa lógico e $G = (\leftarrow A_1, \ldots, A_n)$ uma cláusula objetivo. Se existe uma refutação-SLD de $P_l \cup \{G\}$, então $P_l \rightarrow (A_1 \wedge \ldots \wedge A_n)$ é válida.*

Procedimentos de Refutação-SLD

Definição 11.20 (árvore-SLD) *Sejam P_l um programa lógico, G uma cláusula objetivo e Rc uma regra de computação. A árvore-SLD associada a $P_l \cup \{G\}$, via Rc, é definida por:*

- *cada nó da árvore é rotulado por uma cláusula objetivo (pode-se ter a cláusula vazia);*
- *a raiz da árvore é rotulada pela cláusula $G_0 = G$;*
- *suponha um nó rotulado pela cláusula*

  $$G_i = (\leftarrow A_1, \ldots, A_m, \ldots, A_n)$$

  e pela regra de computação Rc tal que

  $$Rc(A_1, \ldots, A_m, \ldots, A_n) = A_m.$$

  Para cada cláusula de entrada

  $$C_{i+1} = (A \leftarrow B_1, \ldots, B_q),$$

  pertencente a P_l, tal que A_m e A são unificáveis por θ_{i+1}, considere

  $$G_{i+1} = Res(G_i, C_{i+1}, \theta_{i+1}, Rc).$$

  Nesse caso, o nó rotulado por G_i tem um nó descendente rotulado pela cláusula G_{i+1}. A aresta que liga o nó G_i ao nó G_{i+1} é rotulada pela cláusula de entrada C_{i+1} e pela substituição θ_{i+1}.
- *Nós rotulados pela cláusula vazia $\{\}$ não possuem descendentes. Os ramos da árvore com folhas rotulados pela cláusula vazia são denominados ramos fechados. Os outros tipos de ramos são abertos. A composição das substituições associadas a cada ramo da árvore forma a substituição resposta associada ao ramo.*

Exercícios

Exercícios de Computação

Referências

[Ait-Kaci, 1991] H. Ait-Kaci, *Warren's Abstract Machine: A Tutorial Reconstruction,* Mit-Press, 1991.

[Amble, 1987] T. Amble, *Logic Programming and knowledge Engineering,* Addison Wesley, 1987.

[Alencar, 1986] E. Alencar Filho, *Iniciação à Lógica Matemática,* Editora Nobel, 1986.

[Andrews, 1986] P. B. Andrews, *An Introduction to Mathematical Logic and Type Theory: To Truth Through Proof,* Academic Press, 1986.

[Barwise, 1977] J. Barwise, *Handbook of Mathematical Logic,* North-Holland, 1977.

[Bratko, 1990] I. Bratko, *PROLOG, Programming for Artificial Inteligence,* 2ª ed., Addison Wesley, 1990.

[Caravaglia, 1987] S. Caravaglia, *PROLOG, Programming Techniques and Applications,* Harper and Row Publishers, 1987.

[Casanova, 1987] M. A. Casanova, *Programando em Lógica e a linguagem PROLOG,* Edgard Blücher, 1987.

[Causey, 2001] R. L. Causey, *Logic, Sets, and Recursion,* Jones and Bartlett Publishers, 2001.

[Ceri, 1990] S. Ceri, G. Gottlob, L. Tranca, *Logic Programming and Database,* Springer Verlag, 1990.

[Chang, 1973] C. L. Chang, R. C. T. Lee, *Symbolic Logic and Mechanical Theorem Proving,* Academic Press, 1973.

[Chauí, 2002] M. Chauí, *Convite à Filosofia,* Editora Ática, 2002.

[Clocksin, 1984] W. F. Clocksin, C. S. Mellish, *Programming in PROLOG,* Springer Verlag, 1984.

[Coelho, 1988] H. Coelho, J. C. Cotta, *PROLOG by Example,* Springer Verlag, 1988.

[Cormen, 2002] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, *Algoritmos: Teoria e Prática,* Editora Campus, 2002.

[Costa, 1988] N. C. A. Costa, R. Cerrion, *Introdução a Lógica Elementar,* Editora da Universidade Federal do Rio Grande do Sul, 1988.

[Dalen, 1989] D. Dalen, *Logic and Structure,* Springer-Verlag, 1989.

[DeMasi, 2002] D. De Masi, *Criatividade e Grupos Criativos,* Editora Sextante, 2002.

[DeMasi, 2001] D. De Masi, *O Ócio Criativo,* Editora Sextante, 2002.

[Deyi, 1984] L. Deyi, *A PROLOG Database System,* John Wiley and Sons, 1984.

[Dijkstra, 1984] E. W. Dijkstra, *A Discipline of Programming,* Prentice-Hall, 1976.

[Dybvig, 1996] R. K. Dybvig, *Scheme Programming Language. The ANSI Scheme,* Prentice-Hall, 1996.

[Enderton, 1972] H. B. Enderton, *A Mathematical Introduction to Logic,* Academic Press, 1972.

[Epstein, 1999] R. L. Epstein, *Critical Thinking,* Wadsworth Publishing Company, 1999.

[Fitting, 1990] M. Fitting, *First-Order Logic and Automated Theorem Proving,* Springer-Verlag, 1990.

[Francez, 1992] Francez, N., *Program Verification,* Addison Wesley, 1992.

[Gabbay, 1994] D. Gabbay, F. Gunthner, *Handbook of Philosophical Logic,* Kluwer Academic Publishing, 1994.

[Goldstein, 2007] L. Goldstein, A. Brennan, M. Deutsch, J. Y. F. Lau, *Lógica, Conceitos-Chave em Filosofia,* Artmed, 2007.

[Haak, 1998] S. Haak, *A Filosofia da Lógica,* Editora Unesp, 1998.

[Hurley, 2000] P. J. Hurley, R. W. Burch, *A Consise Introduction to Logic,* Wadsworth, 2000.

[Kelly, 1997] J. Kelly, *The Essence of Logic,* Prentice Hall, 1997.

[Kowalski, 1979] R. Kowalski, *Logic for Problem Solving,* North-Holland, 1979.
[Le, 1993] T. V. Le, *PROLOG Programming,* John Wiley and Sons, 1993.
[Lloyd, 1984] J. W. Lloyd, *Foundations of Logic Programming,* Springer-Verlag, 1984.
[Marker, 2002] D. Marker, *Model Theory: An Introduction,* Springer Verlag, 2002
[McDonald, 1990] C. McDonald, M. Yazdani, *PROLOG Programming: A Tutorial Introduction,* Blackwell Scientific Publications, 1990.
[Manna, 1985] Z. Manna, R. Waldinger, *The Logical Basis for Computer Programming,* Vol. 1, Addison Wesley, 1985.
[Manna, 1990] Z. Manna, R. Waldinger, *The Logical Basis for Computer Programming,* Vol. 2, Addison Wesley, 1990.
[Mendelson, 1987] E. Mendelson, *Introduction to Mathematical Logic,* Wadsworth and Brook, 1987.
[Merritt, 1990] D. Merritt, *Adventure in PROLOG,* Springer Verlag, 1990.
[Mortari, 2001] C. A. Mortari, *Introdução à Lógica,* Editora Unesp, 2001.
[Nolt, 1988] J. Nolt, D. Rohatyn, *Lógica,* Editora Makron Books do Brasil, 1988.
[Palazzo, 1997] L. A. M., Palazzo, *Introdução à Programação PROLOG,* Editora da Universidade Católica de Pelotas, EDUCAT, 1997.
[Robinson, 1965] J. A. Robinson, "A Machine-Oriented Logic Based on Resolution Principle", *Journal of the ACM,* Janeiro, 1965.
[Richards, 1989] T. Richards, *Clausal Form Logic: An Introduction to the Logic of Computer Reasoning,* Addison Wesley, 1989.
[Ruth, 2000] M. R. A. Ruth, M. D. Ryan, *Modelling an Reasoning about Systems,* Cambridge University Press, 2000.
[Saint-Dizier, 1990] P. Saint-Dizier, *An Introduction to Programming in PROLOG,* Springer Verlag, 1990.
[Salmon, 1984] W. C. Salmon, *Lógica,* Editora Prentice Hall do Brasil, 1984.
[Shoenfield, 1967] J. R. Shoenfield, *Mathematical Logic,* Addison-Wesley, 1967.
[Shoup, 2005] V. Shoup, *A Computational Introduction to Number Theory an Algebra,* Cambridge University Press, 2005.
[Silva, 2006] Silva, F. S. C., Finger, M., Melo, A. C. V., *Lógica para Computação,* Thomson Pioneira, 2006.
[Sipser, 1997] M. Sipser, *Introduction to the Theory of Computation,* PWS Publishing Co, 1997.
[Souza, 2002] J. N. de Souza, *Lógica para Ciência da Computação,* Editora Campus, 2002.
[Sterling, 1988] L. Sterling, E. Shapiro, *The Art of PROLOG: Advanced Programming Techniques,* Mit-Press, 1988.
[Velleman, 1994] D. J. Velleman, *How to Prove It, A Structured Approach,* Cambridge University Press, 1994.
[Winston, 1989] P. Winston, B. Horn, *Lisp,* 3ª ed., Addison-Wesley Publishing Co, 1989.