Lógica Computacional (CC2003)

Nelma Moreira

Lógica Computacional 21

Conteúdo

1 Mais Teorias (decidíveis)

Teorias de Lógica de Primeira Ordem

Uma teoria $\mathcal{T}$ é um conjunto fórmulas fechadas.

Uma teoria é finitamente axiomatizável se existe um conjunto finito $\mathcal{A} \subseteq \mathcal{T}$ (axiomas) tal que

$$\forall \varphi, \varphi \in \mathcal{T} \text{ sse } \varphi \vdash \mathcal{A}.$$

1. Teoria da igualdade e funções não interpretadas
2. Teoria da aritmética - Axiomas de Peano
3. Teoria de Conjuntos de Zermelo-Frankle
4. Teoria da Geometria
5. Teoria de inteiros
6. Teoria de Grupos

Axiomas de Peano

Os axiomas são factos básicos dos números naturais:

1. $\forall x(x + 1 \neq 0)$

2. $\forall x \forall y (x + 1 = y + 1 \rightarrow x = y)$
3. $0 + 1 = 1$
4. $\forall x \; x + 0 = x$
5. $\forall x \forall y \; x + (y + 1) = (x + y) + 1$
6. $\forall x \; x \times 0 = 0$
7. $\forall x \forall y \; x \times (y + 1) = (x \times y) + x$
8. (princípio da indução) $(Q(0) \; \wedge \; (\forall x (Q(x) \rightarrow Q(x + 1)) \rightarrow \forall x Q(x)$

Não é decidível (Teorema da Incompletitude de Gödel)

Teoria da igualdade $\mathcal{T}_E$

$$\begin{aligned}
\forall x.x = x \\
\forall x, y.x = y \rightarrow y = x \\
\forall x, y, z.x = y \wedge y = z \rightarrow x = z \\
\forall \bar{x}, \bar{y}.x_1 = y_1 \wedge \cdots x_n = y_n \rightarrow f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n) \\
\forall \bar{x}, \bar{y}.x_1 = y_1 \wedge \cdots x_n = y_n \rightarrow P(x_1, \ldots, x_n) \rightarrow P(y_1, \ldots, y_n)
\end{aligned}$$

Indecidível. Fragmentos decidíveis:

- Predicados monádicos
- Sem Predicados
- Sem quantificadores
- Conjunções de fórmulas atómicas

Teoria dos *Arrays*, $\mathcal{T}_A$

$read(a, i)$ and $write(a, i, v)$

$$\begin{aligned}
\mathcal{T}_E \\
\forall a, i, j.i = j \rightarrow read(a, i) = read(a, j) \\
\forall a, i, j, v.i = j \rightarrow read(write(a, i, v), j) = v \\
\forall a, i, j, v.\neg(i = j) \rightarrow read(write(a, i, v), j) = read(a, j) \\
\forall a, b.(\forall i.read(a, i) = read(b, i)) \rightarrow a = b
\end{aligned}$$

Indecídivel. Fragmento sem quantificadores decidível.

Procedimentos de decisão

- São especifícos a uma determinada teoria
- Determinam se uma determinada fórmula é inconsistente, satisfazível ou válida
- Podem trabalhar sobre conjunções de fórmulas ou determinar se uma fórmula é consequência de outras
- Podem utilizar heurísticas para acelarar procedimentos, mas têm sempre que dar uma resposta correcta e terminar (ou seja têm de ser íntegros e completos)

Teorias decidíveis

- Existem muitas teorias úteis decidíveis (ou pelo menos fragmentos):
  - Igualdade com símbolos funcionais não interpretados

    $$x = y \wedge f(f(f(x))) = f(x) \rightarrow f(f(f(f(f(y))))) = f(x)$$

  - Updates de funções, registos e tuplos
  - Aritmética linear sobre inteiros e racionais

    $$x \leq y \wedge x \leq 1 - y \wedge 2 \times x \geq 1 \rightarrow 4 \times x = 2$$

  - Lógica diferencial (caso especial rápido)

    $$x - y < c$$

  - Vectores de bits: operações aritméticas módulo
  - Listas e outras estruturas de dados recursivas.
- Combinações de teorias decidíceis são em geral também decidíceis

Resolutores SMT

- Utilizam-se procedimentos individuais ou combinados para decidir conjunções de fórmulas com base nas suas teorias decidíveis (`TSolver`)
- SMT permite estrutura proposicional geral
- Devem explorar estratégias de procura em resolutores SAT modernos
- Os termos são substituidos por variáveis proposicionais
- Encontrar uma solução num resolutor SAT
- Se encontrar, devolve às variáveis a sua interpretação e envia a fórmula para o procedimento de decisão adequado para ela.

Resolutores SMT

Seja $\mathsf{prop}(\varphi)$ uma função que mapeia φ numa fórmula proposicional (substituinto fórmulas atómicas por variávies proposicionais) e unprop a função inversa. Dada uma valorização ρ para $\mathsf{prop}(\varphi)$ seja

$$\varphi(\rho) = \{\mathsf{unprop}(p_i) \mid \rho(p_i) = \mathbf{V}\} \cup \{\neg\mathsf{unprop}(p_i) \mid \rho(p_i) = \mathbf{F}\}$$

```
1  SMT-Solver(φ) {
2      A := prop(φ)
3      loop
4          (r, ρ) := SAT(A)
5          if r= unsat then return unsat
6          (r, θ):=TSolver(φ(ρ))
7          if r=sat then return sat
8          C := ⋁_{B∈θ} ¬prop(B)
9          A := A ∧ C
10 }
```

Programação em Lógica

Um programa é um conjunto de fórmulas lógicas e a sua execução corresponde a uma demonstração de que uma fórmula é um teorema.

Os sistemas que vamos ver baseiam-se em:

- considerar fórmulas em forma prenexa e em que a matriz está em normal conjuntiva: cláusulas
- como sistema dedutivo usar variantes da resolução
- o sistema dedutivo é íntegro e completo, para uma estrutura determinada.
- computacionalmente "universal", i.e., equivalente a máquinas de Turing ou uma qualquer linguagem de programação de uso geral... C, Java, Python, Haskell

1.1 Resolução para a lógica proposicional

Resolução para a lógica proposicional

Um literal é uma fórmula atómica ou a sua negação: p, $\neg p$

Uma cláusula é uma disjunção de literais: $p \vee \neg q \vee \neg p \vee s$ e pode representar-se por um conjunto

$$\{p, \neg q, \neg p, s\}$$

Então uma fórmula da lógica proposicional em FNC, p.e.

$$\neg p \ \wedge \ (q \ \vee \ r \ \vee \ q) \ \wedge \ (\neg r \ \vee \ \neg s) \ \wedge \ (p \ \vee \ s) \ \wedge \ (\neg q \ \vee \ \neg s)$$

pode ser vista como um conjunto de cláusulas:

$$\{\{\neg p\}, \{q, r\}, \{\neg r, \neg s\}, \{p, s\}, \{\neg q, \neg s\}\}$$

Sistema dedutivo por Resolução (LP)

É um sistema dedutivo por refutação: para deduzir φ, deduz-se que $\neg\varphi$ é uma contradição.

Seja $\mathcal{C}$ um conjunto de cláusulas C e representamos por **F** a cláusula vazia.

O sistema dedutivo por *resolução* não tem axiomas e apenas uma regra de inferência:

Regra de inferência da Resolução

$$\frac{C \cup \{p\} \qquad C' \cup \{\neg p\}}{C \cup C'}$$

A conclusão $C \cup C'$ diz-se a resolvente das premissas.

Uma dedução de **F** a partir de um conjunto $\mathcal{C}$ de cláusulas diz-se uma refutação de $\mathcal{C}$.

Uma dedução por resolução de uma fórmula φ é uma refutação de cláusulas que correspondem à FNC de $\neg\varphi$.

Dado

$$\{\{\neg p\}, \{q, r\}, \{\neg r, \neg s\}, \{p, s\}, \{\neg q, \neg s\}\}$$

temos

$$\dfrac{\dfrac{\{p,s\} \ \{\neg p\}}{\{s\}} \ \dfrac{\dfrac{\{q,r\} \ \{\neg r,\neg s\}}{\{q,\neg s\}} \ \{\neg q,\neg s\}}{\{\neg s\}}}{\mathbf{F}}$$

Integridade e Completude da Resolução para LP

Teorema 21.1. *(Integridade) Seja $\mathcal{C} = \{C_1, \ldots, C_n\}$ um conjunto não vazio de cláusulas da lógica proposicional.*

i. Sejam C_i e C_j cláusulas de $\mathcal{C}$ e R uma resolvente de C_i e C_j. Então $\mathcal{C} \models R$.

*ii. Se $\mathcal{C} \vdash_R \mathbf{F}$,isto é, existe uma dedução de **F** a partir de $\mathcal{C}$ usando apenas a regra da Resolução, então $\mathcal{C}$ não é satisfazível.*

Teorema 21.2. *(Completude) Se um conjunto de cláusulas $\mathcal{C}$ é não satisfazível então existe uma dedução $\mathcal{C} \vdash_R \mathbf{F}$.*

Notação clausal

Um cláusula

$$\{\alpha_1, \ldots, \alpha_k, \neg\beta_1, \ldots, \neg\beta_l\}$$

é equivalente a

$$(\beta_1 \ \wedge \ \cdots \ \wedge \ \beta_l) \rightarrow (\alpha_1 \ \vee \ \cdots \ \vee \ \alpha_k)$$

também se pode representar na notação clausal:

$$\alpha_1, \ldots, \alpha_k \leftarrow \beta_1, \ldots, \beta_l$$

1.2 Cláusulas

Cláusulas para LPO

Seja $\mathcal{L}$ uma LPO com igualdade e pelo menos uma constante.
Um literal positivo (ou átomo) é uma fórmula atómica : $P(x, y)$, $f(x) = a$
Um literal negativo é a negação de uma fórmula atómica: $\neg P(x, y)$
Uma cláusula é uma fórmula φ da forma:

$$\forall x_1 \ldots \forall x_s(\alpha_1 \ \vee \ \ldots \ \vee \ \alpha_k \ \vee \ \neg\beta_1 \ \vee \ \ldots \ \vee \ \neg\beta_n)$$

onde α_i, β_i são átomos e $x_1, \ \ldots, \ x_s$ todas as variáveis que ocorrem em φ.
Também se pode escrever φ como:

$$\forall x_1 \ldots \forall x_s((\beta_1 \ \wedge \ \ldots \ \wedge \ \beta_n) \rightarrow (\alpha_1 \ \vee \ \ldots \ \vee \ \alpha_k))$$

ou ainda representá-la em notação clausal:

$$\alpha_1, \ldots, \alpha_k \leftarrow \beta_1, \ldots, \beta_n$$

Literais e Cláusulas Fechadas

Um literal diz-se fechado se todos os seus termos são fechados, isto é, não tem variáveis.

$$\neg P(f(a), c),\ Q(g(f(a), f(b)),\ h(a, g(b)) = f(g(a, f(a)),\ \neg a = b\ ,\ a = f(a)$$

Uma cláusula diz-se fechada se todos os seus literais são fechados.

$$\{\neg P(f(a), c), Q(g(f(a), f(b)), \neg h(a, g(b)) = f(g(a, f(a))\}$$

1.3 Conversão para forma clausal

Conversão em forma clausal

Para qualquer fórmula φ existe um conjunto de cláusulas que é satisfazível se e só se φ é satisfazível.

Para converter φ para forma clausal primeiro converte-se para forma prenexa:

$$Q_1 x_1 Q_2 x_2 \dots Q_n x_n \psi$$

onde cada Q_i é ou $\forall$ ou $\exists$, $1 \leq i \leq n$, e ψ é uma fórmula sem quantificadores.

Depois temos de *eliminar* quantificadores existenciais.

Conversão em forma clausal

Proposição 21.1. *Seja $\forall y_1 \dots y_n \exists x \varphi$ uma proposição de uma linguagem de $1^{\underline{a}}$ ordem $\mathcal{L}$ e seja $\mathcal{L}'$ uma linguagem com os símbolos de $\mathcal{L}$ e com mais um símbolo n-ário f. Então,*

$$\forall y_1 \dots y_n \exists x \varphi$$

é satisfazível em $\mathcal{L}$ se e só se

$$\forall y_1 \dots y_n \varphi[f(y_1, \dots, y_n)/x]$$

é satisfazível em $\mathcal{L}'$.

Nota: as duas fórmulas não são semanticamente equivalentes: apenas a satisfazibilidade de uma é garantida pela satisfazibilidade da outra.

Conversão em forma clausal

Dem.

Temos que mostrar que existe uma estrutura $\mathcal{A} = (A, .^{\mathcal{A}})$ tal que

$$\mathcal{A} \models \forall y_1 \dots y_n \exists x \varphi$$

sse existe $\mathcal{A}'$ (estrutura de $\mathcal{L}'$) tal que

$$\mathcal{A}' \models \forall y_1 \dots y_n \varphi[f(y_1, \dots, y_n)/x]$$

($\Rightarrow$) Para todos os $a_1, \dots a_n \in A$, existe $b \in A$ tal que

$$\mathcal{A} \models_{s[a_1/y_1]\dots[a_n/y_n][b/x]} \varphi$$

b depende dos $a_1, \dots, a_n$. Então a estrutura $\mathcal{A}'$ pode ser igual a $\mathcal{A}$ e tal que o valor de $f^{\mathcal{A}'}$ seja dado, para todo $a_1, \dots, a_n \in A$ por :

$$f^{\mathcal{A}'}(a_1, \dots, a_n) = b$$

Conversão em forma clausal

Dem.

($\Leftarrow$)

Em $\mathcal{A}'$, consideremos os valores de

$$f^{\mathcal{A}'}(a_1, \ldots, a_n).$$

Estes são os valores de cada um dos b.

Skolemização

É um algoritmo que converte uma proposição φ de uma linguagem de 1ª ordem, numa conjunção (ou conjunto) de cláusulas $\varphi = \varphi_1 \wedge \ldots \wedge \varphi_n$ (duma linguagem alargada $\mathcal{L}'$) tal que:

- cada φ_i é da forma $\forall x_1 \ldots \forall x_n(\lambda_1 \vee \ldots \vee \lambda_n)$, onde cada λ_i é um literal
- a fórmula φ é satisfazível se e só se φ é satisfazível

Algoritmo de Skolemização

1. Converter φ em forma normal prenexa $Q_1x_1Q_2x_2 \ldots Q_nx_n\psi$ onde cada Q_i é ou $\forall$ ou $\exists$ e ψ não tem quantificadores (e denomina-se a *matriz*).
2. Para $1 \leq i \leq n$ se Q_i é um quantificador existencial $\exists$ e $x_{i_1}, \ldots x_{i_{m_i}}$ as variáveis quantificadas universalmente de índice menor que i então substitui-se:

   $$\exists x_i\theta \text{ por } \theta[f_i(x_{i_1}, \ldots x_{i_{m_i}})/x_i]$$

   onde f_i é um novo símbolo funcional de aridade m_i.

Algoritmo de Skolemização

3. Converter a matriz da fórmula resultante para forma normal conjuntiva aplicando sucessivamente as seguintes transformações:

   $$\begin{array}{lcl} \neg\neg\varphi & \longrightarrow & \varphi \\ \neg(\varphi \wedge \psi) & \longrightarrow & \neg\varphi \vee \neg\psi \\ \neg(\varphi \vee \psi) & \longrightarrow & \neg\varphi \wedge \neg\psi \\ \varphi \vee (\psi \wedge \theta) & \longrightarrow & (\varphi \vee \psi) \wedge (\varphi \vee \theta) \end{array}$$

   :

4. Aplicar a seguinte transformação:

$$\forall x(\varphi \wedge \psi) \quad \longrightarrow \quad \forall x\varphi \wedge \forall x\psi$$

Exercício 21.1. *Aplica o algoritmo à seguinte fórmula:*

$$\forall x(\forall y(P(y) \to R(y,x)) \to Q(x))$$

$\diamond$

Resolução 21.1

Para forma normal prenexa basta passar para fora os quantificadores:

$$\forall x\forall y((P(y) \to R(y,x)) \to Q(x))$$

Como não tem quantificadores existênciais basta, converter a matriz para forma normal conjuntiva:

$$\forall x\forall y(\neg(\neg P(y) \vee R(y,x)) \vee Q(x)) \tag{1}$$
$$\forall x\forall y((P(y) \wedge \neg R(y,x)) \vee Q(x)) \tag{2}$$
$$\forall x\forall y((P(y) \vee Q(x)) \wedge (\neg R(y,x) \vee Q(x))) \tag{3}$$
$$\forall x\forall y((P(y) \vee Q(x))) \wedge \forall x\forall y(\neg R(y,x) \vee Q(x)) \tag{4}$$

Resolução para cláusulas fechadas

A regra da resolução para cláusulas fechadas é praticamente igual à da lógica proposicional.

Sejam $C \cup \{l\}$ e $C' \cup \{\neg l\}$ duas cláusulas fechadas.

Regra de inferência da Resolução

$$\frac{C \cup \{l\} \qquad C' \cup \{\neg l\}}{C \cup C'}$$

$C \cup C'$ é a **resolvente** de $C \cup \{l\}$ e $C' \cup \{\neg l\}$,

Exemplo 21.1. *Sendo* $\{P(f(a),g(c)), Q(a,c)\}$ *e* $\{\neg P(f(a),g(c)), Q(f(a),g(c))\}$. *Então, uma resolvente é*

$$\{Q(a,c), Q(f(a),g(c))\}$$