Lógica de Primeira Ordem: resolução

Alexandre Rademaker

March 30, 2017

Linguagem

Símbolos lógicos:

- "(", ")", $\rightarrow$, $\neg$, $\wedge$, $\vee$.
- Variáveis
- Símbolo de igualdade

Parâmetros:

- Símbolos quantificadores: $\forall$ e $\exists$
- Símbolos predicativos de aridade *n*. Exemplo: pai^2.
- Símbolos de constantes (aridade zero). Exemplo: z^0
- Símbolos de funções de aridade *n*. Exemplo: $+^2$.

Semântica

Se σ é uma sentença. Como dizer que "σ é verdade em $\mathfrak{A}$"? Sem a necessidade de traduzir σ para português?

$$\models_{\mathfrak{A}} \sigma$$

Para uma WFF qualquer, precisamos de:

$$s : V \to |\mathfrak{A}|$$

Para então, informalmente definir "$\mathfrak{A}$ satisfaz σ com s" representado por:

$$\models_{\mathfrak{A}} \sigma[s]$$

se e somente se da tradução de σ determinada por $\mathfrak{A}$, onde a variável x é traduzida por $s(x)$ se x é livre, é verdade.

Interpretação de termos

Definimos a função:

$$\overline{s} : T \to |\mathfrak{A}|$$

que mapea termos para elementos do universo de $\mathfrak{A}$. Como:

1. Para cada variável x, $\overline{s}(x) = s(x)$.
2. Para cada constante c, $\overline{s}(c) = c^{\mathfrak{A}}$.
3. Se $t_1, \ldots, t_n$ são termos e f é uma fução, então

$$\overline{s}(f(t_1, \ldots, t_n)) = f^{\mathfrak{A}}(\overline{s}(t_1), \ldots, \overline{s}(t_n))$$

$\overline{s}$ depende de $\mathfrak{A}$ e s. Notação alternativa para $\overline{s}(t)$ poderia ser $t^{\mathfrak{A}}[s]$.

Interpretação de fórmulas

Fórmulas atômicas. Definimos explicitamente, dois casos:

1. Igualdade onde $=$ significa $=$, não é um parâmetro aberto à interpretações.

$$\models_{\mathfrak{A}} t_1 = t_2\ [s] \text{ sse } \overline{s}(t_1) = \overline{s}(t_2)$$

2. Para um predicado n-ário P:

$$\models_{\mathfrak{A}} P(t_1, \ldots, t_n)\ [s] \text{ sse } \langle \overline{s}(t_1), \ldots, \overline{s}(t_n) \rangle \in P^{\mathfrak{A}}$$

Interpretação de fórmulas

Outras WFF. Definimos recursivamente:

1. $\models_{\mathfrak{A}} \neg\phi\ [s]$ sse $\not\models_{\mathfrak{A}} \phi\ [s]$
2. $\models_{\mathfrak{A}} \phi \to \psi\ [s]$ sse ou $\not\models_{\mathfrak{A}} \phi\ [s]$ ou $\models_{\mathfrak{A}} \psi\ [s]$ ou ambos.
3. $\models_{\mathfrak{A}} \phi \wedge \psi\ [s]$ sse $\models_{\mathfrak{A}} \phi\ [s]$ e $\models_{\mathfrak{A}} \psi\ [s]$.
4. $\models_{\mathfrak{A}} \phi \vee \psi\ [s]$ sse $\models_{\mathfrak{A}} \phi\ [s]$ ou $\models_{\mathfrak{A}} \psi\ [s]$.
5. $\models_{\mathfrak{A}} \forall x\psi\ [s]$ sse para todo $d \in |\mathfrak{A}|$, temos $\models_{\mathfrak{A}} \psi\ [s(x|d)]$.

Onde $s(x|d)$ é a função s com uma diferença, para a variável x, ela retorna d.

$$s(x|d)(y) = \begin{cases} s(y) & \text{se } y \neq x \\ d & \text{se } y = x \end{cases}$$

Pragmatics

- Em geral, não lidamos diretamente com a interpretação, mas com <u>teorias</u> que limitem as interpretações que estamos interessados.
- Seja α e β duas sentenças quaisquer e γ a setença $\neg(\beta \wedge \neg\alpha)$. Suponha $\mathcal{I}$ uma interpretação que torne α verdadeira, em $\mathcal{I}$ a fórmula γ também será verdadeira, por que?
- Não precisamos para isso entender nenhum dos símbolos não lógicos de α ou γ.
- Dizemos que $\alpha \models \gamma$ (γ é consequência lógica de α).
- As letras α, γ e β são 'esquemas' de fórmulas.

Consequência Lógica

$S \models \alpha$ onde S é um conjunto de sentenças. S logically entails α. Se e somente se (sss)

para toda interpretação $\mathcal{I}$ se $\mathcal{I} \models S$ então $\mathcal{I} \models \alpha$. Em outras palavras, todo modelo de S satisfaz α.

De outra forma, não existe interpretação $\mathcal{I}$ tal que $\mathcal{I} \models S \cup \{\neg\alpha\}$. Dizemos que $S \cup \{\neg\alpha\}$ é insatisfatível (unsatisfiable) neste caso.

Valid é um caso especial de entailment: Uma sentença é válida quando $\models \alpha$, ou seja, é consequência lógica de um conjunto vazio. Neste caso, para toda interpretação $\mathcal{I}$, temos $\mathcal{I} \models \alpha$. Ou $\neg\alpha$ é unsat.

Entailment se reduz para valid: if $S = \{\alpha_1, \ldots, \alpha_n\}$ então $S \models \alpha$ sss a sentença $s_1 \wedge \ldots \wedge \alpha_n \rightarrow \alpha$ é válida.

Resolução

Dada uma KB e uma sentença α, queremos um procedimento para decidir se $KB \models \alpha$. Se $\beta(x_1, \ldots, x_n)$ é uma fórmula com variáveis livres, queremos ainda um procedimento para para achar termos t_i, se eles existirem, tal que $KB \models \beta(t_1, \ldots, t_n)$.

Mas como veremos, nenhum método computacional poderá sempre nos dar a resposta desejada. Mas queremos um método o mais correto e completo possível.

Lembrando que $KB \models \alpha$ iff $\models (\alpha_1 \land \ldots \land \alpha_n) \to \alpha$ iff $KB \cup \{\neg\alpha\}$ não é 'satisfatível' iff $KB \cup \{\neg\alpha\} \models \neg TRUE$.

Caso Proposicional

Toda fórmula α pode ser convertida em uma fórmula α' equivalente, na forma de uma conjunção de dijunções de literais, onde literais são átomos ou negação de átomos.

Dizemos que α e α' são equivalentes logicamente e α' está na forma CNF (conjuntive normal form).

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

Procedimento para CNF

1. Eliminar $\rightarrow, \equiv$ usando o fato destas serem abreviações para fórmulas expressas com $\vee, \wedge, \neg$.
2. mover $\neg$ para dentro, até que aparece apenas em frente a um átomo, usando equivalências:

$$\models \neg\neg\alpha \equiv \alpha$$
$$\models \neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$$
$$\models \neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta)$$

3. Distribuir $\wedge$ em $\vee$ usando

$$\models (\alpha \vee (\beta \wedge \gamma)) \equiv ((\beta \wedge \gamma) \vee \alpha) \equiv ((\alpha \vee \beta) \vee (\alpha \vee \gamma))$$

4. Coletar termos usando

$$\models (\alpha \wedge \alpha) \equiv \alpha$$
$$\models (\alpha \vee \alpha) \equiv \alpha$$

Example

$$(P \to Q) \to R$$

Formula Clausal

Forma abreviada de CNF. Conjunto finito de clausulas onde clausulas são conjuntos finitos de átomos.

Entendidas como a conjunção das clausulas. Onde as clausulas são a dijunção dos literais.

Se p é um literal, usamos $\overline{p}$ para seu complemento. $\overline{p} = \neg p$ e $\neg\overline{p} = p$.

Examples: $[p, \neg q, r]$ e $\{[p, \neg q, r], [q]\}$ e $[\neg p]$ (unit clause)

Importante: $\{\}$ (fórmula clausal vazia) é diferente de $\{[]\}$ (fórmula contendo apenas uma cláusula vazia). A [] (clausula vazia) é entendida como $\neg TRUE$ (dijunção não possível) e logo, $\{[]\}$ é $\neg TRUE$. Mas $\{\}$ é uma conjunção sem 'constraints', logo $TRUE$.

Procedimento

Colocar as fórmulas de KB e a fórmula α na forma CNF.

Determinar se o conjunto resultante é SAT.

Entailment então se reduz a SAT do conjunto de fórmulas em CNF.