Classical Logic

Predicate Logic

Knowledge Representation and Reasoning

October 31, 2005

Outline

Why First-Order Logic (FOL)?

- In propositional logic, the only building blocks are atomic propositions.
- We cannot talk about the internal structures of these propositions.
- Example:
  - *All CS students know formal logic*
  - *Peter is a CS student*
  - Therefore, *Peter knows formal logic*
  - Not possible in propositional logic
- Idea: We introduce predicates, functions, object variables and quantifiers.

Syntax

- variable symbols: $x, y, z, \ldots$
- n-ary function symbols: $f, g, \ldots$
- constant symbols: $a, b, c, \ldots$
- n-ary predicate symbols: $P, Q, \ldots$
- logical symbols: $\forall$, $\exists$, $=$, $\neg$, $\wedge$, $\ldots$

Terms	t	$\longrightarrow$	x	variable
		$\mid$	$f(t_1, \ldots, t_n)$	function application
		$\mid$	a	constant
Formulae	φ	$\longrightarrow$	$P(t_1, \ldots, t_n)$	atomic formula
		$\mid$	$t = t'$	identity formulae
		$\mid$	$\ldots$	propositional connectives
		$\mid$	$\forall x(\varphi')$	universal quantification
		$\mid$	$\exists x(\varphi')$	existential quantification

ground term, etc.: term, etc. without variable occurrences

Semantics: Idea

- In FOL, the universe of discourse consists of objects, functions over these objects, and relations over these objects.
- Function symbols are mapped to functions, predicate symbols are mapped to relations, and terms to objects.
- Notation: Instead of $I(x)$ we write x^I.
- Note: Usually one considers all possible non-empty universes. (However, sometimes the interpretations are restricted to particular domains, e.g. integers or real numbers.)
- Satisfiability and validity is then considered wrt all these universes.

Formal Semantics: Interpretations

Interpretations: $I = \langle \mathcal{D}, \cdot^I \rangle$ with $\mathcal{D}$ being an arbitrary non-empty set and I being a function which maps

- n-ary function symbols f to n-ary functions $f^I \in [\mathcal{D}^n \to \mathcal{D}]$,
- constant symbols a to objects $a^I \in \mathcal{D}$, and
- n-ary predicates P to n-ary relations $P^I \subseteq \mathcal{D}^n$.

Interpretation of ground terms:

$$(f(t_1, \ldots, t_n))^I \quad = \quad f^I({t_1}^I, \ldots, {t_n}^I) \ (\in \mathcal{D})$$

Truth of ground atoms:

$$I \models P(t_1, \ldots, t_n) \quad \text{iff} \quad \langle {t_1}^I, \ldots, {t_n}^I \rangle \in P^I$$

Examples

$$\begin{aligned}
\mathcal{D} &= \{d_1, \ldots, d_n\}, n \geq 2 \\
\mathrm{a}^I &= d_1 \\
\mathrm{b}^I &= d_2 \\
\mathrm{eye}^I &= \{d_1\} \\
\mathrm{red}^I &= \mathcal{D} \\
I &\models \mathrm{red(b)} \\
I &\not\models \mathrm{eye(b)}
\end{aligned}$$

$$\begin{aligned}
\mathcal{D} &= \{1, 2, 3, \ldots\} \\
1^I &= 1 \\
2^I &= 2 \\
&\vdots \\
\mathrm{even}^I &= \{2, 4, 6, \ldots\} \\
\mathrm{succ}^I &= \{(1 \mapsto 2), (2 \mapsto 3), \ldots\} \\
I &\not\models \mathrm{even}(3) \\
I &\models \mathrm{even}(\mathrm{succ}(3))
\end{aligned}$$

Formal Semantics: Variable Maps

V is the set of variables. Function $\alpha \colon V \to \mathcal{D}$ is a variable map.
Notation: $\alpha[x/d]$ is identical to α except for x where $\alpha[x/d](x) = d$.
Interpretation of terms under I, α:

$$\begin{aligned}
x^{I,\alpha} &= \alpha(x) \\
a^{I,\alpha} &= a^I \\
(f(t_1, \ldots, t_n))^{I,\alpha} &= f^I({t_1}^{I,\alpha}, \ldots, {t_n}^{I,\alpha})
\end{aligned}$$

Truth of atomic formulae:

$$I, \alpha \models P(t_1, \ldots, t_n) \quad \text{iff} \quad \langle {t_1}^{I,\alpha}, \ldots, {t_n}^{I,\alpha} \rangle \in P^I$$

Example (cont.):

$$\alpha = \{x \mapsto d_1, y \mapsto d_2\} \qquad I, \alpha \models \mathrm{red}(x) \qquad I, \alpha[y/d_1] \models \mathrm{eye}(y)$$

Formal Semantics: Truth

Truth of φ by I under α ($I,\alpha \models \varphi$) is defined as follows.

$$
\begin{array}{rcl}
I,\alpha \models P(t_1,\ldots,t_n) & \text{iff} & \langle {t_1}^{I,\alpha},\ldots,{t_n}^{I,\alpha}\rangle \in P^I \\
I,\alpha \models t_1 = t_2 & \text{iff} & {t_1}^{I,\alpha} = {t_2}^{I,\alpha} \\
I,\alpha \models \neg\varphi & \text{iff} & I,\alpha \not\models \varphi \\
I,\alpha \models \varphi\wedge\psi & \text{iff} & I,\alpha \models \varphi \text{ and } I,\alpha \models \psi \\
I,\alpha \models \varphi\vee\psi & \text{iff} & I,\alpha \models \varphi \text{ or } I,\alpha \models \psi \\
I,\alpha \models \varphi\rightarrow\psi & \text{iff} & \text{if } I,\alpha \models \varphi, \text{then } I,\alpha \models \psi \\
I,\alpha \models \varphi\leftrightarrow\psi & \text{iff} & I,\alpha \models \varphi, \text{ iff } I,\alpha \models \psi \\
I,\alpha \models \forall x\,\varphi & \text{iff} & I,\alpha[x/d] \models \varphi \text{ for all } d \in \mathcal{D} \\
I,\alpha \models \exists x\,\varphi & \text{iff} & I,\alpha[x/d] \models \varphi \text{ for some } d \in \mathcal{D}
\end{array}
$$

Examples

$$
\begin{array}{rcl}
\Theta & = & \left\{ \begin{array}{l} \text{eye}(a), \text{eye}(b) \\ \forall x(\text{eye}(x) \rightarrow \text{red}(x)) \end{array} \right\} \\
\mathcal{D} & = & \{d_1,\ldots,d_n,\}\ n > 1 \\
\text{a}^I & = & d_1 \\
\text{b}^I & = & d_1 \\
\text{eye}^I & = & \{d_1\} \\
\text{red}^I & = & \mathcal{D} \\
\alpha & = & \{(x \mapsto d_1),(y \mapsto d_2)\}
\end{array}
$$

Questions:

- $I,\alpha \models \text{eye}(b) \vee \neg\text{eye}(b)$? Yes
- $I,\alpha \models \text{eye}(x) \rightarrow$ $\text{eye}(x) \vee \text{eye}(y)$? Yes
- $I,\alpha \models \text{eye}(x) \rightarrow \text{eye}(y)$? No
- $I,\alpha \models \text{eye}(a) \wedge \text{eye}(b)$? Yes
- $I,\alpha \models \forall x(\text{eye}(x) \rightarrow \text{red}(x))$? Yes
- $I,\alpha \models \Theta$? Yes

Terminology

I,α is a model of φ iff

$$I,\alpha \models \varphi.$$

A formula can be satisfiable, unsatisfiable, falsifiable, valid.
Two formulae φ and ψ are logically equivalent ($\varphi \equiv \psi$) iff for all I,α:

$$I,\alpha \models \varphi \text{ iff } I,\alpha \models \psi.$$

Note: $\mathrm{P(x)} \not\equiv \mathrm{P(y)}$!
Logical Implication is also similar to propositional logic:

$$\Theta \models \varphi \quad \text{iff} \quad \text{for all } I,\alpha \text{ s.t. } I,\alpha \models \Theta \text{ also } \ I,\alpha \models \varphi.$$

Free and Bound Variables

Variables can be free or bound (by a quantifier) in a formula:

$$
\begin{array}{rcl}
\text{free}(x) & = & \{x\} \\
\text{free}(f(t_1,\ldots,t_n)) & = & \text{free}(t_1) \cup \ldots \cup \text{free}(t_n) \\
\text{free}(t_1 = t_2) & = & \text{free}(t_1) \cup \text{free}(t_2) \\
\text{free}(P(t_1,\ldots,t_n)) & = & \text{free}(t_1) \cup \ldots \cup \text{free}(t_n) \\
\text{free}(\neg\varphi) & = & \text{free}(\varphi) \\
\text{free}(\varphi * \psi) & = & \text{free}(\varphi) \cup \text{free}(\psi) \quad * = \vee,\wedge,\rightarrow,\leftrightarrow \\
\text{free}(\Xi x \varphi) & = & \text{free}(\varphi) - \{x\} \quad \Xi = \forall,\exists
\end{array}
$$

Example: $\forall \mathrm{x}\ (\mathrm{R}(\boxed{\mathrm{y}},\boxed{\mathrm{z}}) \wedge \exists\ \mathrm{y}\ (\neg\mathrm{P(y,x)} \vee \mathrm{R}(\mathrm{y},\boxed{\mathrm{z}})))$
Framed occurrences are free, all others are bound.

Open & Closed Formulae

- Formulae without free variables are called closed formulae or sentences. Formulae with free variables are called open formulae.
- Closed formulae are all we need when we want to state something about the world. Open formulae (and variable maps) are only necessary for technical reasons (semantics of $\forall$ and $\exists$).
- Note that *logical equivalence*, *satisfiability*, and entailment are independent from variable maps if we consider only closed formulae.
- For closed formulae, we omit α in connection with $\models$:

$$I \models \varphi.$$

Important Theorems

Theorem (Compactness)

Let $\Phi \cup \{\psi\}$ be a set of closed formulae.

(a) $\Phi \models \psi$ *iff there exists a finite subset* $\Phi' \subset \Phi$ *s. t.* $\Phi' \models \psi$.

(b) Φ *is satisfiable iff each finite subset* $\Phi' \subset \Phi$ *is satisfiable.*

Theorem (Löwenheim-Skolem)

Each countable set of closed formuale that is satisfiable is satisfiable on a countable domain.

Literature

Harry R. Lewis and Christos H. Papadimitriou.
Elements of the Theory of Computation.
Prentice-Hall, Englewood Cliffs, NJ, 1981 (Chapters 8 & 9).

Volker Sperschneider and Grigorios Antoniou.
Logic – A Foundation for Computer Science.
Addison-Wesley, Reading, MA, 1991 (Chapters 1–3).

H.-P. Ebbinghaus, J. Flum, and W. Thomas.
Einführung in die mathematische Logik.
Wissenschaftliche Buchgesellschaft, Darmstadt, 1986.

U. Schöning.
Logik für Informatiker.
Spektrum-Verlag.