Logic in Computer Science

Lecture 05_1

一种命题逻辑的优化.扩展.

Semantics of Predicate Logic

Li Qin

Associate Professor

School of Software Engineering

Semantics of predicate logic

- ★ Models
- ★ Semantic entailment
- ★ Semantics of equality
- ★ Undecidability of predicate logic

In propositional logic, given the formula

$$(p \vee \neg q) \to (q \to p)$$

we can give it a truth value (T or F) based on a given valuation (assumed truth values for p and q).

What about the predicate logic formula

谓词 predicate

$$\forall x \exists y\, ((P(x) \vee \neg Q(y)) \to (Q(x) \to P(y)))$$

term { variable
constant
function

We could assign truth values to $P(x)$ and $Q(y)$ and, based on that, compute a truth value for the entire formula. However, in general, the variables express relationships between predicates, and the assignment of truth values to atoms cannot be done randomly.

Dealing with Quantifiers

$$\phi ::= P(t_1, \ldots, t_n) \mid \phi \ \substack{\wedge \\ \vee \\ \neg \\ \rightarrow} \ \phi \mid \forall x . \phi \mid \exists x . \phi$$

Variables are placeholders for ***any***, or ***some***, unspecified concrete value.

$\exists x \Phi$ We try to find some instance of x (some concrete value) such that Φ holds for that particular instance of x. If this succeeds, then $\exists x \Phi$ evaluates to T; otherwise (i.e. there is no concrete value of x that realizes Φ) the formula evaluates to F.

$\forall x \Phi$ We try to show that for all possible instances of x, Φ evaluates to T. If this is successful, $\forall x \Phi$ evaluates to T; otherwise (i.e. if there exists some instance of x that does not realize Φ), the formula evaluates to F.

Language Signature = (F, P)
Model $\mathcal{M}$ = $(\mathcal{F}, \mathcal{P})$

Definition: Let $\mathcal{F}$ be a set of function symbols and $\mathcal{P}$ a set of predicate symbols, each symbol with a fixed number of required arguments. A ***model*** $\mathcal{M}$ of the pair $(\mathcal{F}, \mathcal{P})$ consists of the following set of data:

论域

1. A non-empty set A, the ***universe of concrete values***;
2. for each $f \in \mathcal{F}$ with n arguments, a concrete function $f^{\mathcal{M}} : A^n \to A$; and
3. for each $P \in \mathcal{P}$ with n arguments, a subset $P^{\mathcal{M}} \subseteq A^n$ of tuples over A.

The distinction between f and $f^{\mathcal{M}}$, and between P and $P^{\mathcal{M}}$ is most important. f is a ***symbol***, whereas $f^{\mathcal{M}}$ denotes a ***concrete function***. Similarly, P is a ***symbol***, whereas $P^{\mathcal{M}}$ is a ***concrete subset*** of A^n, for some natural number n.

Example — Real Numbers

Let $\mathcal{F} \stackrel{def}{=} \{+, *, -\}$ and $\mathcal{P} \stackrel{def}{=} \{=, \leq, <, \texttt{zero}\}$, where $+$, $*$, $-$ take 2 arguments, and where $=$, $\leq$, $<$ are predicates with 2 arguments, and $\texttt{zero}$ is a predicate with 1 argument.

The model $\mathcal{M}$:

1. The non-empty set A is the set of real numbers.
2. The function $+^{\mathcal{M}}$, $*^{\mathcal{M}}$, and $-^{\mathcal{M}}$ take two real numbers as arguments and return their sum, product, and difference, respectively.
3. The predicates $=^{\mathcal{M}}$, $\leq^{\mathcal{M}}$, and $<^{\mathcal{M}}$ model the relations equal to, less than, and strictly less than, respectively. The predicate $\texttt{zero}^{\mathcal{M}}$ holds for r iff r equals to 0.

$=^{M} =_{df} \{(x,y) \mid x=y,\ x \in \mathbb{R},\ y \in \mathbb{R}\}$ $\quad$ $zero =_{df} \{0\}$

Example formula:

n元谓词是一种n元关系

$$\forall x \forall y \, \underline{(\texttt{zero}(y) \rightarrow x * y = y)}$$

$M, l[y \to 0]$

Example — Bit Strings

Let $\mathcal{F} \stackrel{def}{=} \{e, \cdot\}$, and $\mathcal{P} \stackrel{def}{=} \{\leq\}$, where e is a constant, $\cdot$ is a function of 2 arguments and $\leq$ is a predicate with 2 arguments.

The model $\mathcal{M}$:

1. A is the set of binary strings over the alphabet $\{0, 1\}$, including the empty string ε.
2. The interpretation of $\cdot^{\mathcal{M}}$ is the concatenation of strings.
3. $\leq^{\mathcal{M}}$ is the prefix ordering of strings, that is the set $\{(s_1, s_2) \mid s_1 \text{ is a prefix of } s_2\}$.

Bit String Formulas

$\forall x\,((x \le x \cdot e) \wedge (x \cdot e \le x))$

Every word is a prefix of itself concatenated with the empty word

$\exists y \forall x\,(y \le x)$

There exists a word s that is the prefix of every word (in fact it is ε).

$\forall x \exists y\,(y \le x)$

Every word has a prefix.

$\forall x \forall y \forall z\,((x \le y) \to (x \cdot z \le y \cdot z))$

If s_1 is a prefix of s_2, then $s_1 s_2$ is a prefix of $s_1 s_3$ (doesn't hold).

$\neg \exists x \forall y\,((x \le y) \to (y \le x))$

There is no word s such that whenever s is a prefix of some other word s_1, it is the case that s_1 is a prefix of s as well.

Given a formula $\forall x \Phi$, or $\exists x \Phi$, we intend to check whether Φ holds for all, respectively some, value a in our model. We have no way of expressing this in our syntax.

We are forced to interpret formulas relative to an ***environment (look-up table)***, that is, a mapping from variable symbols to concrete values.

$$l : \mathbf{var} \mapsto A$$

Definition (Updated Look-Up Tables): Let l be a look-up table $l : \mathbf{var} \mapsto A$, and let $a \in A$. We denote by $l[x \mapsto a]$ the look-up table which maps x to a and any other variable y to $l(y)$.

The Satisfaction Relation

Definition: Given a model $\mathcal{M}$ for a pair $(\mathcal{F}, \mathcal{P})$ and given an environment l, we define the ***satisfaction relation***

$$\mathcal{M} \models_l \Phi$$

for each formula Φ over the pair $(\mathcal{F}, \mathcal{P})$by structural induction on Φ. The denotation $\mathcal{M} \models_l \Phi$ says that Φ computes to T in the model $\mathcal{M}$ wrt the environment l.

- P: If Φ is of the form $P(t_1, t_2, \ldots, t_n)$, then we interpret the terms $t_1, t_2, \ldots, t_n$ in our set A by replacing all variables with their values according to l. In this way we compute concrete values $a_1, a_2, \ldots, a_n$ of A for each of these terms, where we interpret any function symbol $f \in \mathcal{F}$ by $f^{\mathcal{M}}$. Now $\mathcal{M} \models_l P(t_1, \ldots, t_n)$ holds iff $(a_1, \ldots, a_n) \in P^{\mathcal{M}}$.
- $\forall x$: The relation $\mathcal{M} \models_l \forall x \Psi$ holds iff $\mathcal{M} \models_{l[x \mapsto a]} \Psi$ holds for all $a \in A$.
- $\exists x$: The relation $\mathcal{M} \models_l \exists x \Psi$ holds iff $\mathcal{M} \models_{l[x \mapsto a]} \Psi$ holds for some $a \in A$.
- $\neg$: The relation $\mathcal{M} \models_l \neg\Psi$ holds iff it is not the case that $\mathcal{M} \models_l \Psi$ holds.
- $\vee$: The relation $\mathcal{M} \models_l \Psi_1 \vee \Psi_2$ iff $\mathcal{M} \models_l \Psi_1$ or $\mathcal{M} \models_l \Psi_2$ holds.
- $\wedge$: The relation $\mathcal{M} \models_l \Psi_1 \wedge \Psi_2$ iff $\mathcal{M} \models_l \Psi_1$ and $\mathcal{M} \models_l \Psi_2$ holds.
- $\rightarrow$: The relation $\mathcal{M} \models_l \Psi_1 \rightarrow \Psi_2$ iff $\mathcal{M} \models_l \Psi_2$ holds whenever $\mathcal{M} \models_l \Psi_1$ holds.

Example

Let $\mathcal{F} \stackrel{def}{=} \{\mathtt{alma}\}$ and $\mathcal{P} \stackrel{def}{=} \{\mathtt{loves}\}$, where $\mathtt{alma}$ is a constant and $\mathtt{loves}$ is a predicate with two arguments. The model $\mathcal{M}$ we choose here consists of the set $A \stackrel{def}{=} \{a, b, c\}$, the constant function $\mathtt{alma}^{\mathcal{M}} \stackrel{def}{=} a$ and the predicate $\mathtt{loves}^{\mathcal{M}} \stackrel{def}{=} \{(a, a), (b, a), (c, a)\}$. We want to check whether the model $\mathcal{M}$ satisfies

None of Alma's lovers' lovers love her.

Translation into predicate logic:

$$\forall x \forall y \, (\mathtt{loves}(x, \mathtt{alma}) \land \mathtt{loves}(y, x) \rightarrow \neg \mathtt{loves}(y, \mathtt{alma}))$$

The model $\mathcal{M}$ does not satisfy the formula. However, if we change the interpretation of $\mathtt{loves}$ to be $\mathtt{loves}^{\mathcal{M}} = \{(b, a), (c, b)\}$, then the new model satisfies the formula above.

Definition: Let $\Phi_1, \Phi_2, \ldots, \Phi_n, \Psi$, be formulas in predicate logic. Then, $\Phi_1, \Phi_2, \ldots, \Phi_n \models \Psi$ denotes that, whenever $\mathcal{M} \models_l \Phi_i$, $1 \leq i \leq n$, then $\mathcal{M} \models_l \Phi$, for all models $\mathcal{M}$ and look-up tables l.

The $\models$ symbol is overloaded.

$\mathcal{M} \models \Phi$ denotes ***satisfiability***

$\Phi_1, \ldots, \Phi_n \models \Psi$ denotes ***semantic entailment***

$\mathcal{M}, l \Vdash \forall x\,(P(x) \to Q(x))$ $\mathcal{M}, l \Vdash \psi_1 \to \psi_2$

$(\forall c \in A, \mathcal{M}, l[x \to c] \Vdash P(x) \to Q(x))$ $\mathcal{M}, l \Vdash \psi_1 \to \mathcal{M}, l \Vdash \psi_2$

$\wedge\ (\forall a \in A, \mathcal{M}, l[x \to a] \Vdash P(x)) \to (\forall b \in A, \mathcal{M}, l[x \to b] \Vdash Q(x))$

1. $\neg \forall x P(x)$
2. $\forall x P(x) \to \forall x Q(x)$

论域A $\mathcal{M} \Vdash \forall x\,(P(x) \to Q(x))$; $\mathcal{M} \Vdash \forall x P(x)$

$$\forall x\,(P(x) \to Q(x)) \models \forall x P(x) \to \forall x Q(x)$$

$\phi_1, \phi_2, \ldots, \phi_n$ iff $\forall \mathcal{M}, l$ $\mathcal{M}, l \Vdash \phi_1, \phi_2, \ldots \phi_n \to \mathcal{M}, l \Vdash \psi$

Let $\mathcal{M}$ be a model satisfying $\forall x\,(P(x) \to Q(x))$. We need to show that $\mathcal{M}$ satisfies $\forall x P(x) \to \forall x Q(x)$ as well. On inspecting the definition of $\mathcal{M} \models \Psi_1 \to \Psi_2$, we see that we are done if not every element of A satisfies P. Otherwise, every element does satisfy P. But since $\mathcal{M}$ satisfies $\forall x\,(P(x) \to Q(x))$, the latter forces every element of our model to satisfy Q as well. By combining these 2 cases (i.e. either all elements or $\mathcal{M}$ satisfy $\mathcal{P}$, or not), we have shown that $\mathcal{M}$ satisfies $\forall x P(x) \to \forall x Q(x)$.

Semantic Entailment — Example 2

$$\forall x\, P(x) \to \forall x\, Q(x) \models \forall x\,(P(x) \to Q(x))$$

This sequent doesn't hold. Indeed, let $\mathcal{M}'$ be a model that satisfies $\forall x\, P(x) \to \forall x\, Q(x)$. If A' is its underlying set and $P^{\mathcal{M}'}$ and $Q^{\mathcal{M}'}$ are the corresponding interpretations of P and Q, then $\mathcal{M}' \models \forall x\, P(x) \to \forall x\, Q(x)$ simply says that, if $P^{\mathcal{M}'}$ equals A', then $Q^{\mathcal{M}'}$ must equal A' as well. However, if $P^{\mathcal{M}'}$ does not equal A', then this implication is vacuously true. It is now easy to construct a counterexample.

$A' \stackrel{def}{=} \{a, b\}$, $P^{\mathcal{M}'} \stackrel{def}{=} \{a\}$, and $Q^{\mathcal{M}'} \stackrel{def}{=} \{b\}$. Then

$$\mathcal{M}' \models \forall x\, P(x) \to \forall x\, Q(x)$$

holds, while

$$\mathcal{M}' \models \forall x\,(P(x) \to Q(x))$$

doesn't hold.

Most models have natural interpretations, but semantic entailment

$$\Phi_1, \ldots, \Phi_n \models \Psi$$

一个确定性的解释.

确定论域A，谓词的真假在论域上任一点是确定的

really depends on all the <u>possible models</u>, even those that do not make sense. This means that a predicate may have any interpretation.

However, there is a famous exception: ***equality***. The equality predicate must always be interpreted as the equality relation on the set A. If, for example, $A = \{a, b, c\}$, then $=^{\mathcal{M}}$ is $\{(a,a),(b,b),(c,c)\}$.

Decidability:

- Given a sequent $\Phi_1, \ldots, \Phi_n \models \Psi$, is it possible to know whether there is a proof for it. ***Answer:*** NO.
- Given a semantic entailment sequent $\Phi_1, \ \ldots \ , \Phi_n \models \Psi$, is it possible to know if it holds? ***Answer:*** NO.

Soundness:

- If we have a proof of $\Phi_1, \ldots \Phi_n \vdash \Psi$ hold? ***Answer:*** YES.

Correctness:

- If we know that $\Phi_1, \ldots, \Phi_n \models \Psi$ holds, is there a proof of $\Phi_1, \ \ldots \ , \Phi_n \vdash \Psi$? ***Answer:*** YES.

Completeness = Corectness + Decidability. Predicate logic is undecidable, and therefore incomplete.