

Mathematical and Logical Foundations of Computer Science

Predicate Logic (Semantics)

Vincent Rahli

(some slides were adapted from Rajesh Chitnis' slides)

University of Birmingham

Where are we?

- ▶ Symbolic logic
- ▶ Propositional logic
- ▶ **Predicate logic**
- ▶ Intuitionistic vs. Classical logic
- ▶ Type theory

Today

- ▶ Semantics of Predicate Logic
- ▶ Models
- ▶ Variable valuations
- ▶ Satisfiability & validity

Further reading:

- ▶ Chapter 10 of
http://leanprover.github.io/logic_and_proof/

Recap: Syntax

The syntax of predicate logic is defined by the following grammar:

$$t ::= x \mid f(t, \dots, t)$$

$$P ::= p(t, \dots, t) \mid \neg P \mid P \wedge P \mid P \vee P \mid P \rightarrow P \mid \forall x.P \mid \exists x.P$$

where:

- ▶ x ranges of variables
- ▶ f ranges over function symbols
- ▶ $f(t_1, \dots, t_n)$ is a well-formed term only if f has arity n
- ▶ p ranges over predicate symbols
- ▶ $p(t_1, \dots, t_n)$ is a well-formed formula only if p has arity n

The pair of a collection of function symbols, and a collection of predicate symbols, along with their arities, is called a **signature**.

The scope of a quantifier extends as far right as possible. E.g., $P \wedge \forall x.p(x) \vee q(x)$ is read as $P \wedge \forall x.(p(x) \vee q(x))$

Recap: Substitution

Substitution is defined recursively on terms and formulas:
 $P[x \backslash t]$ substitute all the free occurrences of x in P with t .

$$\begin{array}{ll} x[x \backslash t] & = t \\ x[y \backslash t] & = x \\ (f(t_1, \dots, t_n))[x \backslash t] & = f(t_1[x \backslash t], \dots, t_n[x \backslash t]) \\ (p(t_1, \dots, t_n))[x \backslash t] & = p(t_1[x \backslash t], \dots, t_n[x \backslash t]) \\ \hline (\neg P)[x \backslash t] & = \neg P[x \backslash t] \\ (P_1 \wedge P_2)[x \backslash t] & = P_1[x \backslash t] \wedge P_2[x \backslash t] \\ (P_1 \vee P_2)[x \backslash t] & = P_1[x \backslash t] \vee P_2[x \backslash t] \\ (P_1 \rightarrow P_2)[x \backslash t] & = P_1[x \backslash t] \rightarrow P_2[x \backslash t] \\ \hline (\forall x. P)[x \backslash t] & = \forall x. P \\ (\exists x. P)[x \backslash t] & = \exists x. P \\ (\forall y. P)[x \backslash t] & = \forall y. P[x \backslash t], \text{ if } y \notin \text{fv}(t) \\ (\exists y. P)[x \backslash t] & = \exists y. P[x \backslash t], \text{ if } y \notin \text{fv}(t) \end{array}$$

The additional **conditions** ensure that **free variables do not get captured**.

These conditions can always be met by silently renaming bound variables before substituting.

Recap: \forall & \exists elimination and introduction rules

Natural Deduction rules for quantifiers:

$$\begin{array}{c}
 \frac{P[x \backslash y]}{\forall x.P} \quad [\forall I] \qquad \frac{\forall x.P}{P[x \backslash t]} \quad [\forall E] \qquad \frac{P[x \backslash t]}{\exists x.P} \quad [\exists I] \qquad \frac{\exists x.P \quad \begin{array}{c} \overline{P[x \backslash y]}^1 \\ \vdots \\ Q \end{array}}{Q}^1 \quad [\exists E]
 \end{array}$$

Condition:

- ▶ for $[\forall I]$: y must not be free in any not-yet-discharged hypothesis or in $\forall x.P$
- ▶ for $[\forall E]$: $\mathbf{fv}(t)$ must not clash with $\mathbf{bv}(P)$
- ▶ for $[\exists I]$: $\mathbf{fv}(t)$ must not clash with $\mathbf{bv}(P)$
- ▶ for $[\exists E]$: y must not be free in Q or in not-yet-discharged hypotheses or in $\exists x.P$

Recap: \forall & \exists left and right rules

Sequent Calculus rules for quantifiers:

$$\frac{\Gamma \vdash P[x \backslash y]}{\Gamma \vdash \forall x.P} [\forall R] \qquad \frac{\Gamma, P[x \backslash t] \vdash Q}{\Gamma, \forall x.P \vdash Q} [\forall L]$$

$$\frac{\Gamma \vdash P[x \backslash t]}{\Gamma \vdash \exists x.P} [\exists R] \qquad \frac{\Gamma, P[x \backslash y] \vdash Q}{\Gamma, \exists x.P \vdash Q} [\exists L]$$

Conditions:

- ▶ for $[\forall R]$: y must not be free in Γ or $\forall x.P$
- ▶ for $[\forall L]$: $\mathbf{fv}(t)$ must not clash with $\mathbf{bv}(P)$
- ▶ for $[\exists R]$: $\mathbf{fv}(t)$ must not clash with $\mathbf{bv}(P)$
- ▶ for $[\exists L]$: y must not be free in Γ , Q , or $\exists x.P$

Interpretation of Predicate & Function Symbols

Semantics: Assigning meaning/interpretations to formulas

Earlier in the module: a **particular semantics** for propositional logic

- ▶ Each proposition has a meaning (a **truth value**) of **T** or **F**
- ▶ Used truth tables to check **semantic validity**

We now **extend** this particular semantics to predicate logic

- ▶ Propositional logic constructs are interpreted similarly
- ▶ In addition, we need to interpret
 - ▶ **predicate & function symbols**
 - ▶ **quantifiers**

Predicate symbols: for example, given the domain \mathbb{N} and a unary predicate symbol **even**, what is the meaning of **even**?

- ▶ to state that a number is $0, 2, 4, \dots$?
- ▶ is it always obvious?
- ▶ what if we had a predicate symbol **small**?
- ▶ what does that mean?

Interpretation of Predicate & Function Symbols

Given a domain D and a predicate symbol p of arity n

- ▶ p is interpreted by a n -ary relation \mathcal{R}_p
- ▶ of the form $\{\langle d_1^1, \dots, d_n^1 \rangle, \langle d_1^2, \dots, d_n^2 \rangle, \dots\}$
- ▶ where each d_j^i is in D
- ▶ we write: $\mathcal{R}_p \in 2^{D^n}$ or $\mathcal{R}_p \subseteq D^n$

For example

- ▶ a meaningful interpretation for **even** would be
 - ▶ $\{\langle 0 \rangle, \langle 2 \rangle, \langle 4 \rangle, \dots\}$
- ▶ a meaningful interpretation for **odd** would be
 - ▶ $\{\langle 1 \rangle, \langle 3 \rangle, \langle 5 \rangle, \dots\}$
- ▶ a meaningful interpretation for **prime** would be
 - ▶ $\{\langle 2 \rangle, \langle 3 \rangle, \langle 5 \rangle, \dots\}$

Interpretation of Predicate & Function Symbols

Function symbols: for example, given the domain \mathbb{N} and a binary function symbol `add`, what is the meaning of `add`?

- ▶ is it addition?
- ▶ is it always obvious?
- ▶ what if we had a binary function symbol `combine`?
- ▶ what does that mean?


Given a domain D and a function symbol f of arity n

- ▶ f is interpreted by a function \mathcal{F}_f from D^n to D
- ▶ we write: $\mathcal{F}_f \in D^n \rightarrow D$

For example

- ▶ a meaningful interpretation for `add` would be
 - ▶ $+$
- ▶ a meaningful interpretation for `mult` would be
 - ▶ \times

Interpretation of Predicate & Function Symbols

WARNING : sometimes for convenience we will use the same symbol for a function symbol and its interpretation

For example:

1. we have used 0 in our examples as a **constant symbol**, which has no meaning on its own
2. this constant symbol would be interpreted by the natural number 0 , which is an **object of the domain** \mathbb{N}

Even though we used the same symbols, these symbols stand for different entities:

1. a **constant symbol**
2. an **object of the domain**

If we want to distinguish them, we might use:

1. $\bar{0}$ for the **constant symbol**
2. 0 for the **object of the domain**

Models

Models: a model provides the interpretation of all symbols

Given a **signature** $\langle\langle f_1^{k_1}, \dots, f_n^{k_n} \rangle, \langle p_1^{j_1}, \dots, p_m^{j_m} \rangle\rangle$

- ▶ of function symbols f_i of arity k_i , for $i < n$
- ▶ of predicate symbols p_i of arity j_i , for $i < m$

a **model** is a structure $\langle D, \langle \mathcal{F}_{f_1}, \dots, \mathcal{F}_{f_n} \rangle, \langle \mathcal{R}_{p_1}, \dots, \mathcal{R}_{p_m} \rangle \rangle$

- ▶ of a non-empty domain D
- ▶ interpretations \mathcal{F}_{f_i} for function symbols f_i
- ▶ interpretations \mathcal{R}_{p_i} for function symbols p_i

Models of predicate logic replace **truth assignments** for propositional logic

For example:

- ▶ we might interpret the signature $\langle\langle \text{add} \rangle, \langle \text{even} \rangle\rangle$
 - ▶ where **add** is a binary function symbol
 - ▶ and **even** is a unary predicate symbol
- ▶ by the model $\langle \mathbb{N}, \langle \langle + \rangle, \langle \{ \langle 0 \rangle, \langle 2 \rangle, \langle 4 \rangle, \dots \} \rangle \rangle \rangle$

Models

A **model** assigns meaning to function and predicate symbols

Variable valuations: In addition, we need to assign meaning to variables:

- ▶ this is done using a partial function v
- ▶ that maps variables to D
- ▶ i.e., a mapping of the form $x_1 \mapsto d_1, \dots, x_n \mapsto d_n$
- ▶ which maps each x_i to d_i , i.e., to $v(x_i)$
- ▶ $\text{dom}(v) = \{x_1, \dots, x_n\}$
- ▶ let \cdot be the empty mapping
- ▶ we write $v, x \mapsto d$ for the mapping that
 - ▶ maps x to d
 - ▶ and maps each $y \in \text{dom}(v)$ such that $x \neq y$ to $v(y)$

For example

- ▶ $(x_1 \mapsto d_1), x_2 \mapsto d_2$ maps x_1 to $?d_1$ and x_2 to $?d_2$
- ▶ $(x_1 \mapsto d_1, x_2 \mapsto d_2), x_1 \mapsto d_3$ maps x_1 to $?d_3$ and x_2 to $?d_2$

Semantics of Predicate Logic

Given a **model** M with domain D and a **variable valuation** v , to assign **meaning** to Predicate Logic formulas, we define two operations:

- ▶ $\llbracket t \rrbracket_v^M$, which gives meaning to the term t w.r.t. M and v
- ▶ $\models_{M,v} P$, which gives meaning to the formula P w.r.t. M and v

Meaning of terms:

- ▶ $\llbracket x \rrbracket_v^M = v(x)$
- ▶ $\llbracket f(t_1, \dots, t_n) \rrbracket_v^M = \mathcal{F}_f(\langle \llbracket t_1 \rrbracket_v^M, \dots, \llbracket t_n \rrbracket_v^M \rangle)$

Semantics of Predicate Logic

Given a **model** M with domain D and a **variable valuation** v , to assign **meaning** to Predicate Logic formulas, we define two operations:

- ▶ $\llbracket t \rrbracket_v^M$, which gives meaning to the term t w.r.t. M and v
- ▶ $\models_{M,v} P$, which gives meaning to the formula P w.r.t. M and v

Meaning of formulas:

- ▶ $\models_{M,v} p(t_1, \dots, t_n)$ iff $\langle \llbracket t_1 \rrbracket_v^M, \dots, \llbracket t_n \rrbracket_v^M \rangle \in \mathcal{R}_p$
- ▶ $\models_{M,v} \neg P$ iff $\not\models_{M,v} P$
- ▶ $\models_{M,v} P \wedge Q$ iff $\models_{M,v} P$ and $\models_{M,v} Q$
- ▶ $\models_{M,v} P \vee Q$ iff $\models_{M,v} P$ or $\models_{M,v} Q$
- ▶ $\models_{M,v} P \rightarrow Q$ iff $\models_{M,v} Q$ whenever $\models_{M,v} P$
- ▶ $\models_{M,v} \forall x.P$ iff for every $d \in D$ we have $\models_{M,(v,x \mapsto d)} P$
- ▶ $\models_{M,v} \exists x.P$ iff there exists a $d \in D$ such that $\models_{M,(v,x \mapsto d)} P$

Semantics of Predicate Logic

For example:

- ▶ consider the signature $\langle\langle\text{zero}, \text{succ}, \text{add}\rangle, \langle\text{even}, \text{odd}\rangle\rangle$
- ▶ the model M : $\langle\mathbb{N}, \langle 0, +1, + \rangle, \langle \{\langle 0 \rangle, \langle 2 \rangle, \langle 4 \rangle, \dots \}, \{\langle 1 \rangle, \langle 3 \rangle, \langle 5 \rangle, \dots \} \rangle\rangle$
- ▶ we write $+1$ for the function that given a number increments it by 1
- ▶ $+(n, m)$ stands for $n + m$

What is $\models_{M, \cdot} \text{even}(\text{succ}(\text{zero})) \vee \text{odd}(\text{succ}(\text{zero}))$?

- ▶ iff $\models_{M, \cdot} \text{even}(\text{succ}(\text{zero}))$ or $\models_{M, \cdot} \text{odd}(\text{succ}(\text{zero}))$
- ▶ iff $\langle \llbracket \text{succ}(\text{zero}) \rrbracket^M \rangle \in \{\langle 0 \rangle, \langle 2 \rangle, \langle 4 \rangle, \dots\}$ or $\langle \llbracket \text{succ}(\text{zero}) \rrbracket^M \rangle \in \{\langle 1 \rangle, \langle 3 \rangle, \langle 5 \rangle, \dots\}$
- ▶ iff $\langle 1 \rangle \in \{\langle 0 \rangle, \langle 2 \rangle, \langle 4 \rangle, \dots\}$ or $\langle 1 \rangle \in \{\langle 1 \rangle, \langle 3 \rangle, \langle 5 \rangle, \dots\}$
- ▶ iff True

Semantics of Predicate Logic

For example:

- ▶ consider the signature $\langle\langle\text{zero}, \text{succ}, \text{add}\rangle, \langle\text{even}, \text{odd}\rangle\rangle$
- ▶ the model M : $\langle\mathbb{N}, \langle 0, +1, + \rangle, \langle \{\langle 0 \rangle, \langle 2 \rangle, \langle 4 \rangle, \dots \}, \{\langle 1 \rangle, \langle 3 \rangle, \langle 5 \rangle, \dots \} \rangle\rangle$
- ▶ we write $+1$ for the function that given a number increments it by 1
- ▶ $+(n, m)$ stands for $n + m$

What is $\models_M. \forall x.\text{even}(x)$?

- ▶ iff for all $n \in \mathbb{N}$, $\models_{M, x \mapsto n} \text{even}(x)$
- ▶ iff for all $n \in \mathbb{N}$, $\langle \llbracket x \rrbracket_{x \mapsto n}^M \rangle \in \{\langle 0 \rangle, \langle 2 \rangle, \langle 4 \rangle, \dots\}$
- ▶ iff for all $n \in \mathbb{N}$, $\langle n \rangle \in \{\langle 0 \rangle, \langle 2 \rangle, \langle 4 \rangle, \dots\}$
- ▶ iff False, because $1 \notin \{0, 2, 4, \dots\}$

Semantics of Predicate Logic

For example:

- ▶ consider the signature $\langle\langle\text{zero}, \text{succ}, \text{add}\rangle, \langle\text{even}, \text{odd}\rangle\rangle$
- ▶ the model M : $\langle\mathbb{N}, \langle 0, +1, + \rangle, \langle \{\langle 0 \rangle, \langle 2 \rangle, \langle 4 \rangle, \dots \}, \{\langle 1 \rangle, \langle 3 \rangle, \langle 5 \rangle, \dots \} \rangle\rangle$
- ▶ we write $+1$ for the function that given a number increments it by 1
- ▶ $+(n, m)$ stands for $n + m$

What is $\models_M. \forall x. \text{even}(x) \rightarrow \neg \text{odd}(x)$?

- ▶ iff for all $n \in \mathbb{N}$, $\models_{M, x \mapsto n} \text{even}(x) \rightarrow \neg \text{odd}(x)$
- ▶ iff for all $n \in \mathbb{N}$, $\models_{M, x \mapsto n} \neg \text{odd}(x)$ whenever $\models_{M, x \mapsto n} \text{even}(x)$
- ▶ iff for all $n \in \mathbb{N}$, $\neg \models_{M, x \mapsto n} \text{odd}(x)$ whenever $\models_{M, x \mapsto n} \text{even}(x)$
- ▶ iff for all $n \in \mathbb{N}$, $\langle [x]_{x \mapsto n}^M \rangle \notin \{\langle 1 \rangle, \langle 3 \rangle, \langle 5 \rangle, \dots\}$ whenever $\langle [x]_{x \mapsto n}^M \rangle \in \{\langle 0 \rangle, \langle 2 \rangle, \langle 4 \rangle, \dots\}$
- ▶ iff for all $n \in \mathbb{N}$, $\langle n \rangle \notin \{\langle 1 \rangle, \langle 3 \rangle, \langle 5 \rangle, \dots\}$ whenever $\langle n \rangle \in \{\langle 0 \rangle, \langle 2 \rangle, \langle 4 \rangle, \dots\}$
- ▶ iff for all $n \in \mathbb{N}$, $n \notin \{1, 3, 5, \dots\}$ whenever $n \in \{0, 2, 4, \dots\}$
- ▶ iff True

Semantics of Predicate Logic

For example:

- ▶ consider the signature $\langle\langle\text{zero}, \text{succ}, \text{add}\rangle, \langle\text{lt}, \text{ge}\rangle\rangle$
- ▶ the model M :
 $\langle\mathbb{N}, \langle 0, +1, + \rangle, \langle \{\langle 0, 1 \rangle, \langle 0, 2 \rangle, \langle 1, 2 \rangle, \dots \}, \{\langle 0, 0 \rangle, \langle 1, 1 \rangle, \langle 1, 0 \rangle, \dots \} \rangle\rangle$
- ▶ we write $+1$ for the function that given a number increments it by 1
- ▶ $+(n, m)$ stands for $n + m$

What is $\models_{M, \cdot} \forall x. \forall y. \text{lt}(x, y) \rightarrow \text{ge}(y, x)$?

- ▶ iff for all $n, m \in \mathbb{N}$, $\models_{M, x \mapsto n, y \mapsto m} \text{lt}(x, y) \rightarrow \text{ge}(y, x)$
- ▶ iff for all $n, m \in \mathbb{N}$, $\models_{M, x \mapsto n, y \mapsto m} \text{ge}(y, x)$ whenever $\models_{M, x \mapsto n, y \mapsto m} \text{lt}(x, y)$
- ▶ iff for all $n, m \in \mathbb{N}$,
 $\langle \llbracket y \rrbracket_{x \mapsto n, y \mapsto m}^M, \llbracket x \rrbracket_{x \mapsto n, y \mapsto m}^M \rangle \in \{\langle 0, 0 \rangle, \langle 1, 1 \rangle, \langle 1, 0 \rangle, \dots\}$ whenever
 $\langle \llbracket x \rrbracket_{x \mapsto n, y \mapsto m}^M, \llbracket y \rrbracket_{x \mapsto n, y \mapsto m}^M \rangle \in \{\langle 0, 1 \rangle, \langle 0, 2 \rangle, \langle 1, 2 \rangle, \dots\}$
- ▶ iff for all $n, m \in \mathbb{N}$, $\langle m, n \rangle \in \{\langle 0, 0 \rangle, \langle 1, 1 \rangle, \langle 1, 0 \rangle, \dots\}$ whenever $\langle n, m \rangle \in \{\langle 0, 1 \rangle, \langle 0, 2 \rangle, \langle 1, 2 \rangle, \dots\}$
- ▶ iff True

Satisfiability & Validity

We write $\models_M P$ for $\models_{M, \cdot} P$

Truth: P is **true** in the model M if $\models_M P$

We also say that M is a model of P

Satisfiability: P is **satisfiable** if there is a model M such that P is true in M , i.e., $\models_M P$

Validity: P is **valid** if for all model M , P is true in M

Example: $\models_{M, \cdot} \forall x. \text{even}(x) \rightarrow \neg \text{odd}(x)$ is satisfiable (see above) but not valid because not true for example in the model $\langle \mathbb{N}, \langle 0, +1, + \rangle, \langle \{ \langle 0 \rangle, \langle 2 \rangle, \langle 4 \rangle, \dots \}, \{ \langle 0 \rangle, \langle 2 \rangle, \langle 4 \rangle, \dots \} \rangle \rangle$

Decidability: Validity is not decidable for predicate logic, i.e., there is no algorithm that given a formula P either returns “yes” if P is valid, and otherwise returns “no”, while it is decidable for propositional logic

Recap: Soundness & Completeness

Given a deduction system such as Natural deduction, a formula is said to be **provable** if there is a proof of it in that deduction system

- ▶ This is a **syntactic** notion
- ▶ it asserts the existence of a syntactic object: a proof
- ▶ typically written $\vdash A$

A formula A is **valid** if for all model M , A is true in M , i.e., $\models_M A$

- ▶ it is a **semantic** notion
- ▶ it is checked w.r.t. valuations/models that give meaning to formulas
- ▶ written $\models A$

Soundness: a deduction system is sound w.r.t. a semantics if every provable formula is valid

- ▶ i.e., if $\vdash A$ then $\models A$

Completeness: a deduction system is complete w.r.t. a semantics if every valid formula is provable

- ▶ i.e., if $\models A$ then $\vdash A$

Soundness & Completeness

Natural Deduction for Predicate Logic is

- ▶ **sound** and
- ▶ **complete**

w.r.t. the **model semantics of Predicate Logic**

Proving those properties is done within the **metatheory**

We will not prove them here

Conclusion

What did we cover today?

- ▶ Semantics of Predicate Logic
- ▶ Models
- ▶ Variable valuations
- ▶ Satisfiability & validity

Further reading:

- ▶ Chapter 10 of
http://leanprover.github.io/logic_and_proof/

Next time?

- ▶ Equivalences in Predicate Logic