CS 745 / ECE 725
Computer Aided Verification

Lecture 3: Predicate Logic

Jo Atlee

DC 2337, jmatlee@uwaterloo.ca

Office Hours: Mon 1:00-2:00, Wed 1:00-2:00

http://www.student.cs.uwaterloo.ca/~cs745

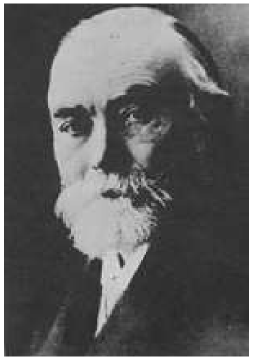
Today's Agenda

- Predicate Logic
- Equality

Predicate Logic

Invented by Gottlob Frege (1848–1925).

Predicate Logic is also called "first order logic".



"Every good mathematician is at least half a philosopher, and every good philosopher is at least half a mathematician."

Motivation

There are some kinds of descriptions and reasoning that we can't do in propositional logic. For example:

Every person likes ice cream.
Billy is a person.
Therefore, Billy likes ice cream.

In propositional logic, the best we can do is $A \wedge B \Rightarrow C$, which isn't a tautology. It doesn't capture the relationships between propositions.

Motivation

We want to be able to

- refer to objects (e.g., Billy) and collections (e.g., person)
- indicate that objects have characteristics (e.g., likes ice cream)
- express relations between objects (e.g., Billy is older than Bryon)
- specify that collections or members of collections have characteristics (e.g., every member or some member)

The predicates and quantifiers of predicate logic allow us to capture these concepts.

Predicates

A predicate (also called a propositional function) defines an attribute (property, characteristic) in terms of the objects that possess that attribute. The syntax is functional, where the result of the function (T or F) indicates whether the predicate's argument(s) possess the attribute.

Billy is a person.
$person(Billy)$

Billy like ice cream.
$likesIceCream(Billy)$

We can have n-ary predicates:
$older(Billy, Bryon)$

For the moment, we aren't dealing with types.

Variables

Variables represent arbitrary or unnamed object instances. They allow us to express properties without being specific about which object possess the property.

Variables act a placeholders for specific objects.

$person(x)$

$likesIceCream(x)$

$older(x, y)$

Quantifiers

Quantifiers are used to express properties about the members of a collection (domain). A quantifier will introduce a variable that refers to an arbitrary member of the collection.

- Universal quantification ($\forall$) applies to all members of a collection.
- Existential quantification ($\exists$) asserts something about some member of a collection.

Everybody with money goes to the movies:

Some student likes logic:

Quantifiers

More formally:

Universal quantification ($\forall x \bullet \phi$) corresponds to finite or infinite conjunction of the formula ϕ instantiated with every element of the collection (domain).

Existential quantification ($\exists x \bullet \phi$) corresponds to finite or infinite disjunction of the formula ϕ instantiated with every element of the domain.

$\forall$ and $\exists$ are duals of each other:

$\exists x \bullet P(x)$ is the same as $\neg \forall x \bullet \neg P(x)$
$\forall x \bullet P(x)$ is the same as $\neg \exists x \bullet \neg P(x)$

Free and Bound Variables

Given that quantifiers introduce variables, a predicate logic formula can have two types of variables:

- bound variables, which are introduced by quantifiers; these variables represent all possible values
- free variables, whose values are determined by the formula's valuation.

A formula is closed if it contains no free occurrences of any variable.

Predicate Logic

1. syntax (well-formed formulas)
2. semantics
3. proof theory
   - axiom systems
   - natural deduction

Predicate Logic: Syntax

The syntax of predicate logic consists of:

1. constants
2. variables $x, y, \ldots$
3. functions
4. predicates
5. logical connectives
6. quantifiers ($\forall, \exists$)
7. punctuation: , • ( )

Predicate Logic: Syntax

A term is an expression whose value is **not** a truth value. Terms are objects.

Definition. Terms are defined as follows:

1. Every constant is a term.
2. Every variable is a term.
3. If $t_1, t_2, t_3, \ldots t_n$ are terms then $f(t_1, t_2, t_3, \ldots t_n)$ is a term, where f is an n-ary function.
4. Nothing else is a term.

Predicate Logic: Syntax

A formula is an expression whose value **is** a truth value.

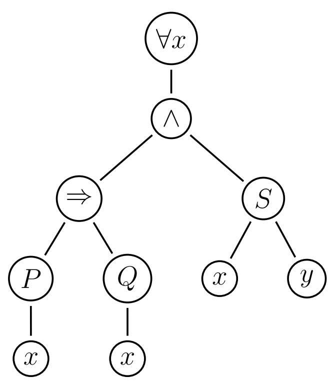
Definition. Well-formed formulas are defined as follows:

1. $P(t_1, t_2, t_3, \ldots t_n)$ is a wff, where t_i is a term, and P is an n-ary predicate. These are called atomic formulas.
2. If A, and B are wffs, then so are $(\neg A)$, $(A \wedge B)$, $(A \vee B)$, $(A \Rightarrow B)$, and $(A \Leftrightarrow B)$.
3. If A is a wff, the so are ($\forall x \bullet A$), ($\exists x \bullet A$).
4. Nothing else is a wff.

We often omit the brackets using the same precedence rules as propositional logic for the logical connectives.

Scope and Binding of Variables

One can determine whether a variable is bound, and what the scope of a bound variable is, from a wff's parse tree.



- A leaf node x is a bound variable if it has an ancestor node labelled $\forall x$ or $\exists x$.
- A leaf node x is a free variable if has no such ancestor node.
- The scope of a bound variable x is the subtree whose root is the quantifier that introduces x, minus any subtrees whose roots are $\forall x$ or $\exists x$.

Substitution

Variables are place holders. Given a variable x, a term t and a formula P, we define $P[t/x]$ to be the formula obtained by replacing all free occurrences of variable x in P with t.

Warning: In substitution $P[t/x]$, both t and x must be "free for x in P". It is a problem if t includes some free variable y and if y is bound at some occurrence of x in P.

Example:
A is $\forall y \bullet P(x) \wedge Q(y)$
t is $f(y)$
$P[t/x]$ is $\forall y \bullet P(f(y)) \wedge Q(y)$ $\quad\therefore t$ is NOT free for x in A.

Predicate Logic

1. syntax (well-formed formulas)
2. semantics
3. proof theory
   - axiom systems
   - natural deduction

Predicate Logic: Semantics

Recall that a semantics is a mapping between two worlds.

A model for predicate logic consists of:

1. a non-empty domain of objects: D
2. a mapping I, called an interpretation that associates the terms of the syntax with objects in a domain

It's important that D be non-empty, otherwise some tautologies wouldn't hold such as $(\forall x.A(x)) \Rightarrow (\exists x.A(x))$.

Interpretations

An interpretation assigns:

1. a fixed element $c' \in D$ to each constant c of the syntax
2. an n-ary function $f' : D^n \rightarrow D$ to each n-ary function, f, of the syntax
3. an n-ary relation $P' \subseteq D^n$ to each n-ary predicate, P, of the syntax

Example of a Model

Let's say our syntax has the constant c, the function f (unary), and two predicates P and Q (both binary).

And suppose that the domain is the natural numbers. Let the model have the following interpretation:

- $I(c)$ is 0
- $I(f)$ is <u>suc</u>, the successor function
- $I(P)$ is $<$
- $I(Q)$ is $=$

Valuations

Definition. A valuation v of a formula ϕ is an assignment of each term variable to a value in the object domain D.

Such evaluations are also called environments.

Evaluating Formulae

We evaluate a formula ϕ with respect to

- a model $\mathcal{M}$ that maps each constant c, function f, and predicate p respectively to c', f', P' in domain D
- a valuation v that maps terms to objects in domain D

Evaluation is defined by structural induction on formula ϕ:

$$\begin{array}{ll} v(c) = c' & v(\phi \vee \psi) = v(\phi) \vee v(\psi) \\ v(f(t_1 ... t_n)) = f'(v(t_1) ... v(t_n)) & v(\phi \wedge \psi) = v(\phi) \wedge v(\psi) \\ v(P(t_1 ... t_n)) = P'(v(t_1) ... v(t_n)) & v(\phi \Rightarrow \psi) = v(\phi) \Rightarrow v(\psi) \\ v(\neg \phi) = \neg(v(\phi)) & v(\phi \Leftrightarrow \psi) = v(\phi) \Leftrightarrow v(\psi) \end{array}$$

$$v(\forall x \bullet A) = \bigwedge_{d \in D} v(A[d/x]) \qquad v(\exists x \bullet A) = \bigvee_{d \in D} v(A[d/x])$$

Example Evaluation

Let

- D be the set of natural numbers
- g be the function $+$
- h be the function suc
- c (constant) be 3
- y (variable) be 1

$$\begin{aligned} v(g(h(c), y)) &= v(h(c)) + v(y) \\ &= \text{suc}(v(c)) + 1 \\ &= \text{suc}(3) + 1 \\ &= 5 \end{aligned}$$

Terminology

Definition. A predicate logic formula is satisfiable iff there is **some** valuation in **some** model such that the formula evaluates to T.

Definition. A predicate logic formula is logically valid (tautology) iff it evaluates to T for **every** valuation in **every** model.

Definition. A predicate logic formula is a contradiction iff it evaluates to F for **every** valuation in **every** model.

Semantic Entailment

Semantic entailment has a similar meaning in predicate logic as it has in propositional logic.

$$\phi_1, \phi_2, \phi_3 \models \psi$$

means that for **every** valuation v in **every** model, if $v(\phi_1) = v(\phi_2) = v(\phi_3) =$ T, then $v(\psi) =$ T.

Which is equivalent to saying that $(\phi_1 \wedge \phi_2 \wedge \phi_3) \Rightarrow \psi$ is a tautology:

$$\boxed{(\phi_1, \phi_2, \phi_3 \models \psi) \equiv (\models (\phi_1 \wedge \phi_2 \wedge \phi_3) \Rightarrow \psi)}$$

Proof by Refutation

A closed formula is valid (a tautology) iff its negation is not satisfiable.

Counterexamples

Definition. A counterexample for a closed formula is a model in which the formula does not evaluate to T.

We can “prove” that a formula is not a tautology by providing a counterexample.

Counterexamples

Show that $(\forall x \bullet P(x) \vee Q(x)) \iff ((\forall x \bullet P(x)) \vee (\forall x \bullet Q(x)))$ is not a tautology.

Predicate Logic

1. syntax (well-formed formulas)
2. semantics
3. proof theory
   - axiom systems
   - natural deduction

An Axiomatic System for Predicate Logic

An extension of the axiomatic system for propositional logic.
Uses only: $\Rightarrow$, $\neg$, $\forall$

Five axiom (schemes):

1. $A \Rightarrow (B \Rightarrow A)$
2. $(A \Rightarrow (B \Rightarrow C)) \Rightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C))$
3. $(\neg A \Rightarrow \neg B) \Rightarrow (B \Rightarrow A)$
4. $(\forall x \bullet A(x)) \Rightarrow A(t)$, where t is free for x in A
5. $\forall x \bullet (A \Rightarrow B(x)) \Rightarrow (A \Rightarrow (\forall x \bullet B(x)))$, where A contains no free occurrences of x

Rules of Inference

Two rules of inference:

1. (modus ponens - MP) From A and $A \Rightarrow B$, B can be derived, where A and B are any well-formed formulas.
2. (generalization) From A, $\forall x \bullet A$ can be derived, where A is any well-formed formula and x is any variable.

Example

Prove $\forall x \bullet \forall y \bullet A \vdash_{ph} \forall y \bullet \forall x \bullet A$

1	$\forall x \bullet \forall y \bullet A$	premise
2	$(\forall x \bullet \forall y \bullet A) \Rightarrow \forall y \bullet A$	Ax4
3	$\forall y \bullet A$	MP 1, 2
4	$(\forall y \bullet A) \Rightarrow A$	Ax4
5	A	MP 3, 4
6	$\forall x \bullet A$	Gen of 5
7	$\forall y \bullet \forall x \bullet A$	Gen of 6

Deduction Theorem

Theorem. If $S \cup \{A\} \vdash_{ph} B$ by a derivation containing no application of generalization to a variable that occurs free in A, then $S \vdash_{ph} A \Rightarrow B$.

Corollary. If A is closed and if $S \cup \{A\} \vdash_{ph} B$, then $S \vdash_{ph} (A \Rightarrow B)$.

Soundness and Completeness

First-order axiomatic logic is sound and complete.

Completeness was proven by Kurt Gödel in 1929 in his doctoral dissertation.

Predicate Logic

1. syntax (well-formed formulas)
2. semantics
3. proof theory
   - axiom systems
   - natural deduction

Predicate Logic: Natural Deduction

Extend the set of rules we use for propositional logic with ones to handle quantifiers.

Rules for Universal Quantification

forall-elimination

$$\frac{\forall x \bullet P}{P[t/x]}\ \forall e$$

t must be free for x in P.

forall-introduction

$$\frac{\left[\begin{array}{l} x_0 \\ \quad\vdots \\ \quad P[x_0/x] \end{array}\right]}{\forall x \bullet P}\ \forall i$$

x_0 must be arbitrary, meaning it doesn't appear outside the subproof.

Predicate Logic: Natural Deduction

Rules for Existential Quantification

exists-introduction

$$\frac{P[t/x]}{\exists x \bullet P}\ \exists i$$

t must be free for x in P.

exists-elimination

$$\frac{\exists x \bullet P \qquad \left[\begin{array}{ll} x_0 & P[x_0/x] \quad \text{assumption} \\ & \vdots \\ & Q \end{array}\right.}{Q}\ \exists e$$

Ideally, we would derive Q from all substitutions for x in P (proof by cases). But as with the proof rules for universal quantification, deriving Q from an arbitrary substitution is sufficient.

Example

Show $\forall x \bullet P(x) \Rightarrow Q(x), \forall x \bullet P(x) \vdash_{ND} \forall x \bullet Q(x)$

1		$\forall x \bullet P(x) \Rightarrow Q(x)$	premise
2		$\forall x \bullet P(x)$	premise
3	x_0		
4		$P(x_0) \Rightarrow Q(x_0)$	$\forall$e 1
5		$P(x_0)$	$\forall$e 2
6		$Q(x_0)$	$\Rightarrow$e $4, 5$
7		$\forall x \bullet Q(x)$	$\forall$i $3-6$

Exercises

1. $P(a), \forall x \bullet P(x) \Rightarrow \neg Q(x) \vdash_{ND} \neg Q(a)$
2. $\neg \forall x \bullet P(x) \vdash_{ND} \exists x . \neg P(x)$

Theory of Equality

So far we've made no restrictions on models (as long as they provide **some** interpretation for each constant, function, and predicate in our logic). This is a very liberal notion of models.

Sometimes we want to assume at least something about the semantics of our logic.

The least common denominator to all sensible models is the notion of equality. That is, there is a distinguished predicate $=$ whose meaning is defined to relate equivalent terms.

Example:
Given $D = \{c_1, c_2\}$, the semantics of predicate $=$ is defined to be

$$\{(c_1, c_1), (c_2, c_2)\}$$

An Axiomatic System with Equality

To the previous axioms and rules of inference, we add:

EAx1 $\forall x \bullet x = x$
EAx2 $\forall x \bullet \forall y \bullet x = y \Rightarrow (A(x,x) \Rightarrow A(x,y))$
EAx3 $\forall x \bullet \forall y \bullet x = y \Rightarrow f(x) = f(y)$

Natural Deduction Rules for Equality

Reflexivity

$$\frac{}{t = t} \; = \mathrm{i}$$

This inference rule is called an axiom, because it has no premises.

Natural Deduction Rules for Equality

Substitution

$$\frac{t_1 = t_2 \quad P[t_2/x]}{P[t_1/x]} \; = \mathrm{e} \qquad \frac{t_1 = t_2 \quad P[t_1/x]}{P[t_2/x]} \; = \mathrm{e}$$

where t_1 and t_2 are free in x in P.

Examples

From these two inference rules, we can derive two other properties that we expect equality to have:

1. Symmetry: $\vdash_{ND} \forall x, y \bullet (x = y) \Rightarrow (y = x)$
2. Transitivity $\vdash_{ND} \forall x, y, z \bullet (x = y) \wedge (y = z) \Rightarrow (x = z)$

Example

$\vdash_{\text{ND}} \forall x, y \bullet (x = y) \Rightarrow (y = x)$

1	x_0	
2	y_0	
3	$x_0 = y_0$	assumption
4	$x_0 = x_0$	$=$i
5	$y_0 = x_0$	$=$e $3, 4$
6	$x_0 = y_0 \Rightarrow y_0 = x_0$	$\Rightarrow$i $3 - 5$
7	$\forall y \bullet (x_0 = y) \Rightarrow (y = x_0)$	$\forall$i $2 - 6$
8	$\forall x, y \bullet (x = y) \Rightarrow (y = x)$	$\forall$i $1 - 7$

Example

$\vdash_{\text{ND}} \forall x, y, z \bullet (x = y) \wedge (y = z) \Rightarrow (x = z)$

1	x_0	
2	y_0	
3	z_0	
4	$(x_0 = y_0) \wedge (y_0 = z_0)$	assumption
5	$x_0 = y_0$	$\wedge$e 3
6	$y_0 = z_0$	$\wedge$e 3
7	$x_0 = z_0$	$=$e $4, 5$
8	$(x_0 = y_0) \wedge (y_0 = z_0) \Rightarrow (x_0 = z_0)$	$\Rightarrow$i $4 - 7$
9	$\forall z \bullet (x_0 = y_0) \wedge (y_0 = z) \Rightarrow (x_0 = z)$	$\forall$i $3 - 8$
10	$\forall y, z \bullet (x_0 = y) \wedge (y = z) \Rightarrow (x_0 = z)$	$\forall$i $2 - 9$
11	$\forall x, y, z \bullet (x = y) \wedge (y = z) \Rightarrow (x = z)$	$\forall$i $1 - 10$

Extensional Equality

Equality in the domain is extensional, meaning it is equality in meaning rather than form.

This is in contrast to intensional equality which is equality in form rather than meaning.

In logic, we are interested in whether two terms represent the same object, not whether they are the same symbols.

If two terms are intensionally equal then they are also extensionally equal, but not necessarily the other way around.

What to Remember

Predicate Logic

- motivation
- syntax (well-formed formulae)
- semantics (models, valuations)

Axiomatic proofs

- 5 axioms
- 2 inference rules (*modus ponens* and generalization)
- sound and complete (but not decidable)

Natural deduction

- 1 axiom (for equality)
- inference rules eliminate or introduce logical operator, quantifier, equality
- sound and complete (not decidable)

Summary

Predicate Logic

- motivation
- syntax
- semantics
- proof procedures
  - axiom system (with equality)
  - natural deduction (with equality)

Next Lecture: Introduction to Model Checking
References: Model Checking, Chapters 1-3

References

[HR04] Michael R. A. Huth and Mark D. Ryan. Logic in Computer Science. Cambridge University Press, Cambridge, 2004. Second Edition.