Predicate Logic

CS 3234: Logic and Formal Systems

Martin Henz and Aquinas Hobor

September 2, 2010

Generated on Tuesday 14 September, 2010, 11:29

1 Syntax of Predicate Logic

1.1 Need for Richer Language

Propositional logic can easily handle simple declarative statements such as:

> Student Peter Lim enrolled in CS3234.

Propositional logic can also handle combinations of such statements such as:

> Student Peter Lim enrolled in Tutorial 1, *and* student Julie Bradshaw is enrolled in Tutorial 2.

However, statements involving formulations such as *"there exists..."* or *"every..."* or *"among..."* are difficult to express in propositional logic. A statement of the form

> *Every* student is younger than *some* instructor.

talks about concepts such as

- being a student,
- being an instructor, and
- being younger than somebody else

These are *properties* of elements of a *set* of objects. We express them in predicate logic using *predicates*.

Example 1. *The statement*

> Every *student is younger than* some *instructor.*

is expressed using the following predicates.

- S*: For example,* $S(andy)$ *could denote that Andy is a student.*
- I*: For example,* $I(paul)$ *could denote that Paul is an instructor.*
- Y*: For example,* $Y(andy, paul)$ *could denote that Andy is younger than Paul.*

A practical problem arises when such predicates are used to express statements such as

> *Every* student is younger than *some* instructor.

How do we express *"every student"*? We need *variables* that can stand for constant values, and a *quantifier* symbol that denotes *"every"*. Using variables and quantifiers, we can write:

$$\forall x(S(x) \rightarrow (\exists y(I(y) \wedge Y(x, y)))).$$

Literally: For every x, if x is a student, then there is some y such that y is an instructor and x is younger than y.

Example 2. *Consider the following statement.*

> *Not all birds can fly.*

Using the following predicates,

$B(x)$**:** *x is a bird*

$F(x)$**:** *x can fly*

we can express the sentence as follows:

$$\neg(\forall x(B(x) \rightarrow F(x)))$$

Example 3. *Consider the following statement.*

> *Every girl is younger than her mother.*

Using the following predicates,

$G(x)$**:** *x is a girl*

$M(x, y)$**:** *x is y's mother*

$Y(x, y)$**:** *x is younger than y*

we can express the sentence as follows:

$$\forall x \forall y(G(x) \wedge M(y, x) \rightarrow Y(x, y))$$

Note that in the previous example, the variable y is only introduced to denote the mother of x. If everyone has exactly one mother, the predicate $M(y, x)$ is a function, when read from right to left.

We introduce a function symbol m that can be applied to variables and constants as in

$$\forall x(G(x) \to Y(x, m(x)))$$

Example 4. *Consider the following statement.*

Andy and Paul have the same maternal grandmother.

Without function symbols, we would have to write

$$\begin{aligned}\forall x\forall y\forall u\forall v(&M(x, y) \land M(y, andy) \land \\ &M(u, v) \land M(v, paul) \to x = u)\end{aligned}$$

However, with the function symbol m, we can simply write:

$$m(m(andy)) = m(m(paul))$$

2 Predicate Logic as a Formal Language

At any point in time, we want to describe the features of a particular "world", using predicates, functions, and constants. Thus, we introduce for this world:

- a set of predicate symbols $\mathcal{P}$
- a set of function symbols $\mathcal{F}$

Every function symbol in $\mathcal{F}$ and predicate symbol in $\mathcal{P}$ comes with a fixed arity, denoting the number of arguments the symbol can take. Function symbols with arity 0 are called *constants*.

Definition 1. *The set of* terms *in predicate logic is given by the BNF:*

$$t ::= \ x \mid c \mid f(t, \ldots, t)$$

where x ranges over a given set of variables $\mathcal{V}$, c ranges over nullary function symbols in $\mathcal{F}$, and f ranges over function symbols in $\mathcal{F}$ with arity $n > 0$.

Example 5. *If n is a nullary function symbol (constant), f is a unary function symbol, and g is a binary function symbol, then examples of terms are:*

- $g(f(n), n)$
- $f(g(n, f(n)))$

Example 6. *If $0, 1, 2, 3$ are nullary functions (constants), s is unary, and $+, -$ and $*$ are binary, then*

$$*(-(2, +(s(x), y)), x)$$

is a term.

Occasionally, we allow ourselves to use infix notation for function symbols as in

$$(2 - (s(x) + y)) * x$$

Definition 2. *The set of* formulas *in predicate logic is defined by the BNF:*

$$\begin{array}{rcl} \phi & ::= & P(t_1, t_2, \ldots, t_n) \mid \bot \mid \top \mid (\neg\phi) \mid (\phi \wedge \phi) \mid (\phi \vee \phi) \mid \\ & & (\phi \to \phi) \mid (\forall x \phi) \mid (\exists x \phi) \end{array}$$

where $P \in \mathcal{P}$ is a predicate symbol of arity $n \geq 0$, t_i are terms over $\mathcal{F}$ and x is a variable.

We allow for nullary predicate symbols. The predicates that they denote do not depend on any arguments, and as such are similar to propositional atoms in propositional logic.

Convention 1. *Just like for propositional logic, we introduce convenient conventions to reduce the number of parentheses:*

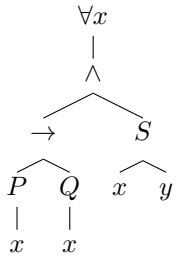
- $\neg, \forall x$ *and* $\exists x$ *bind most tightly;*
- *then* $\wedge$ *and* $\vee$*;*
- *then* $\to$*, which is right-associative.*

We extend the the notion of a *parse tree*, to provide for functions, predicates and quantifiers.

Example 7.

$$\forall x((P(x) \to Q(x)) \wedge S(x, y))$$

has parse tree



2.1 Equality

Equality is a common predicate, usually used in infix notation.

$$= \in \mathcal{P}$$

Example 8. *Instead of the formula*

$$= (f(x), g(x))$$

we usually write the formula

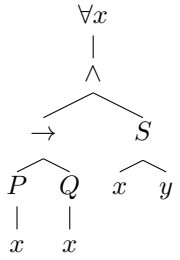
$$f(x) = g(x)$$

2.2 Free and Bound Variables

Consider the formula

$$\forall x((P(x) \to Q(x)) \wedge S(x, y))$$

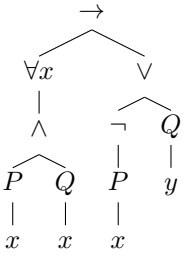
with the following syntax tree:



The quantifier $\forall x$ refers to all *occurrences* of x below it in the syntax tree. We say that the quantifier *binds* the variable occurrence. The variable occurrence x is said to be *bound* by $\forall x$. A variable that is not bound by any quantifier is called *free*. For example, the variable y is a free variable in the formula above. Consider the formula

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

with the following syntax tree:



Here, the occurrences of x in $P(x) \wedge P(x)$ are bound by $\forall x$, whereas the occurrence of x in $\neg P(x)$ is free.

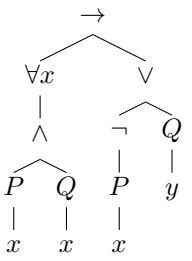
In order to define the semantics of quantifiers, we need to be able to replace free occurrences of variables systematically by terms, using an operation called *substitution*.

Definition 3. *Given a variable x, a term t and a formula ϕ, we define $[x \Rightarrow t]\phi$ to be the formula obtained by replacing each free occurrence of variable x in ϕ with t.*

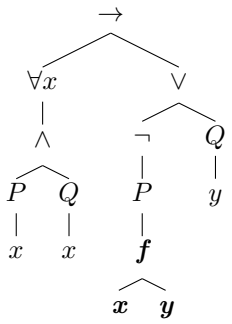
Example 9.

$$[x \Rightarrow f(x, y)]((\forall x(P(x) \wedge Q(x))) \to (\neg P(x) \vee Q(y)))$$

$$= \forall x(P(x) \wedge Q(x))) \to (\neg P(f(x, y)) \vee Q(y))$$



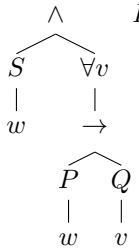
becomes



The notion of substitution of x by t in ϕ, denoted $[x \Rightarrow t]\phi$ poses a technical difficulty when t contains a variable y and x occurs under the scope of $\forall y$ in ϕ.

Example 10.

$$[w \Rightarrow f(v,v)](S(w) \wedge \forall v(P(w) \to Q(v)))$$



Here the variable v occurs in the term that is to be substituted for w. However, there is an occurrence of w under a $\forall w$. Thus, a naive execution of the substitution would "slip" occurrences of w "under" the scope of $\forall w$. This is to be avoided; any variable in t needs to be free in $[x \Rightarrow t]\phi$.

Definition 4. *Given a term t, a variable x and a formula ϕ, we say that t* is free for x in ϕ, *if no free x leaf in ϕ occurs in the scope of $\forall y$ or $\exists y$ for any variable y occurring in t.*

In order to compute $[x \Rightarrow t]\phi$, we demand that t is free for x in ϕ. If this condition does not hold, we consistently rename bound variables in ϕ.

Example 11.

$$[w \Rightarrow f(v,v)](S(w) \wedge \forall v(P(w) \to Q(v)))$$

$$\Downarrow$$

$$[w \Rightarrow f(v,v)](S(w) \wedge \forall z(P(w) \rightarrow Q(z)))$$

$$\Downarrow$$

$$S(f(v,v)) \wedge \forall z(P(f(v,v)) \rightarrow Q(z))$$

3 Semantics of Predicate Logic

3.1 Models

Definition 5. *Let $\mathcal{F}$ contain function symbols and $\mathcal{P}$ contain predicate symbols. A model $\mathcal{M}$ for $(\mathcal{F}, \mathcal{P})$ consists of:*

1. *A non-empty set U, the* universe*;*
2. *for each nullary function symbol $f \in \mathcal{F}$ a concrete element $f^{\mathcal{M}} \in U$;*
3. *for each $f \in F$ with arity $n > 0$, a concrete function $f^{\mathcal{M}} : U^n \rightarrow U$;*
4. *for each $P \in \mathcal{P}$ with arity $n > 0$, a function $P^{\mathcal{M}} : U^n \rightarrow \{F, T\}$.*
5. *for each $P \in \mathcal{P}$ with arity $n = 0$, a value from $\{F, T\}$.*

Example 12. *Let $\mathcal{F} = \{e, \cdot\}$ and $\mathcal{P} = \{\leq\}$.*
Let model $\mathcal{M}$ for $(\mathcal{F}, \mathcal{P})$ be defined as follows:

1. *Let U be the set of binary strings over the alphabet $\{0, 1\}$;*
2. *let $e^{\mathcal{M}} = \epsilon$, the empty string;*
3. *let $\cdot^{\mathcal{M}}$ be defined such that $s_1 \cdot^{\mathcal{M}} s_2$ is the concatenation of the strings s_1 and s_2; and*
4. *let $\leq^{\mathcal{M}}$ be defined such that $s_1 \leq^{\mathcal{M}} s_2$ iff s_1 is a prefix of s_2.*

Examples of elements of U are ϵ and 10001. The term $1010 \cdot 1100$ is given the meaning $1010 \cdot^{\mathcal{M}} 1100 = 10101100$ in $\mathcal{M}$, whereas the term $000 \cdot \epsilon$ is given the meaning $000 \cdot^{\mathcal{M}} \epsilon = 000$.

3.2 Equality Revisited

Usually, we require that the equality predicate $=$ is interpreted as same-ness. This means that allowable models are restricted to those in which $a =^{\mathcal{M}} b$ holds if and only if a and b are the same elements of the model's universe.

Example 13. *Continuing Example 12, we require in every model $\mathcal{M}$ that $000 =^{\mathcal{M}} 000$ holds and that $001 =^{\mathcal{M}} 100$ does not hold. We write $001 \neq^{\mathcal{M}} 100$ to denote the latter.*

Example 14. *Let $\mathcal{F} = \{z, s\}$ and $\mathcal{P} = \{\leq\}$.*
Let model $\mathcal{M}$ for $(\mathcal{F}, \mathcal{P})$ be defined as follows:

1. *Let U be the set of natural numbers;*
2. *let $z^{\mathcal{M}} = 0$;*
3. *let $s^{\mathcal{M}}$ be defined such that $s(n) = n + 1$; and*
4. *let $\leq^{\mathcal{M}}$ be defined such that $n_1 \leq^{\mathcal{M}} n_2$ iff the natural number n_1 is less than or equal to n_2.*

With the above restriction on equality, we can see that the relation $=^{\mathcal{M}}$ is a subset of $\leq^{\mathcal{M}}$; we write $=^{\mathcal{M}} \subseteq \leq^{\mathcal{M}}$.

3.3 Free Variables and the Satisfaction Relation

We can give meaning to formulas with free variables by providing an environment (lookup table) that assigns variables to elements of our universe:

$$l : \mathcal{V} \to U.$$

We define environment extension such that $l[x \mapsto a]$ is the environment that maps x to a and any other variable y to $l(y)$. Using this definition, we can now define when a model satisfies a formula.

Definition 6. *The model $\mathcal{M}$ satisfies ϕ with respect to environment l, written $\mathcal{M} \models_l \phi$:*

- *in case ϕ is of the form $P(t_1, t_2, \ldots, t_n)$, if $a_1, a_2, \ldots, a_n$ are the results of evaluating $t_1, t_2, \ldots, t_n$ with respect to l, and if $P^{\mathcal{M}}(a_1, a_2, \ldots, a_n) = T$;*
- *in case ϕ is of the form P, if $P^{\mathcal{M}} = T$;*
- *in case ϕ has the form $\forall x\psi$, if the $\mathcal{M} \models_{l[x \mapsto a]} \psi$ holds for all $a \in U$;*
- *in case ϕ has the form $\exists x\psi$, if the $\mathcal{M} \models_{l[x \mapsto a]} \psi$ holds for some $a \in U$;*
- *in case ϕ has the form $\neg\psi$, if $\mathcal{M} \models_l \psi$ does not hold;*
- *in case ϕ has the form $\psi_1 \vee \psi_2$, if $\mathcal{M} \models_l \psi_1$ holds or $\mathcal{M} \models_l \psi_2$ holds;*

- *in case* ϕ *has the form* $\psi_1 \wedge \psi_2$*, if* $\mathcal{M} \models_l \psi_1$ *holds and* $\mathcal{M} \models_l \psi_2$ *holds; and*
- *in case* ϕ *has the form* $\psi_1 \to \psi_2$*, if* $\mathcal{M} \models_l \psi_2$ *holds whenever* $\mathcal{M} \models_l \psi_1$ *holds.*

If a formula ϕ has no free variables, we call ϕ a *sentence*. In this case, $\mathcal{M} \models_l \phi$ holds or does not hold regardless of the choice of l. Thus for sentences ϕ, we leave out the environment, and write $\mathcal{M} \models \phi$ or $\mathcal{M} \not\models \phi$.

Definition 7. *Let* Γ *be a possibly infinite set of formulas in predicate logic and* ψ *a formula. We say that* Γ entails ψ*, written* $\Gamma \models \psi$*, iff for all models* $\mathcal{M}$ *and environments* l*, whenever* $\mathcal{M} \models_l \phi$ *holds for all* $\phi \in \Gamma$*, then* $\mathcal{M} \models_l \psi$*.*

Definition 8. *We say that a formula* ψ *is* satisfiable*, iff there is some model* $\mathcal{M}$ *and some environment* l *such that* $\mathcal{M} \models_l \psi$ *holds.*

Definition 9. *A set of formulas* Γ *is called* satisfiable*, iff there is some model* $\mathcal{M}$ *and some environment* l *such that* $\mathcal{M} \models_l \phi$*, for all* $\phi \in \Gamma$*.*

Definition 10. *Let* Γ *be a possibly infinite set of formulas in predicate logic and* ψ *a formula. The formula* ψ *is called* valid*, iff for all models* $\mathcal{M}$ *and environments* l*, we have* $\mathcal{M} \models_l \psi$*.*

Note that both validity and entailment require to consider all possible models. Not only are we free to choose the universe, we are also free to decide the interpretation of every function and predicate symbol. As a result, the number of models is usually infinite (and usually not countably infinite). This makes it very hard to prove validity and entailment, using semantic techniques.

The question arises how to effectively argue about all possible models. Would it be possible to define a method of natural deduction that allows us to answer the questions of entailment and validity?