Function and Predicate Symbols

We next extend the language of propositional logic by *function* and *predicate symbols*.

We use the letters $f, g, h, \ldots$ to denote function symbols, and the letters $P, Q, R, \ldots$ to denote predicate symbols.

We also associate with each function and predicate symbol a non-negative integer, called its *arity*.

Function symbols, as we shall see, denote functions over a certain domain; predicate symbols, relations. The arity indicates the number of arguments a function or relation takes.

A (function or predicate) symbol of arity 0 is called a *constant*. Sometimes we use superscripts to indicate the arity of a symbol, e.g., we may write f^2 for a binary function symbol.

Finally, we use the letters $x, y, z, \ldots$ to denote (individual) *variables*, ranging over elements of a specified domain.

Terms and Atoms

Let $\mathcal{F}$ and $\mathcal{P}$ be sets of function and predicate symbols, respectively, and $\mathcal{X}$ be a set of variables.

The set $\mathcal{T}(\mathcal{F}, \mathcal{V})$ of *terms* (over $\mathcal{F}$ and $\mathcal{X}$) is defined inductively by:

- every variable x in $\mathcal{X}$ is a term, and
- if f is a function symbol in $\mathcal{F}$ of arity n, and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$ is also a term.

For example, if f is a binary function symbol, a is a constant, and x is a variable, then a, $f(a, x)$, and $f(a, a)$ are all terms.

Similarly, we define:

- if P is a predicate symbol in $\mathcal{P}$ of arity n, and $t_1, \ldots, t_n$ are terms, then $P(t_1, \ldots, t_n)$ is an *atomic formula*, or *atom* for short.

Thus, if P is a binary predicate symbol, then $P(x, x)$ and $P(a, f(x, a))$ are atomic formulas.

Parentheses are not necessary, but increase the readability.

If a term or atom contains no variables, it is said to be *variable-free* or *ground*.

First-Order Languages

The language of predicate logic uses the propositional connectives as well as additional logical operators called *quantifiers*, more specifically, a *universal* quantifier $\forall$ and an *existential* quantifier $\exists$.

The symbols of *(predicate) logic* are thus the following:

connectives: $\wedge, \vee, \neg, \rightarrow$

quantifiers: $\forall, \exists$

function symbols: $f, g, \ldots$

predicate symbols: $P, Q, R, \ldots$

variables: $x, y, z, \ldots$

A *first-order language* $\mathcal{L}$ is specified by its sets of function and predicate symbols and variables.

Syntax of Predicate Logic

In predicate logic there are two kinds of expressions: We have already given (recursive) definitions of the sets of terms and atomic formulas.

The definition of arbitrary formulas extends the definition of propositional formulas:

$$
\begin{array}{rcl}
\langle \text{formula} \rangle & ::= & \bot \mid \top \mid \langle \text{atom} \rangle \\
& & \mid (\neg \langle \text{formula} \rangle) \\
& & \mid (\langle \text{formula} \rangle \wedge \langle \text{formula} \rangle) \\
& & \mid (\langle \text{formula} \rangle \vee \langle \text{formula} \rangle) \\
& & \mid (\langle \text{formula} \rangle \rightarrow \langle \text{formula} \rangle) \\
& & \mid (\forall \langle \text{variable} \rangle \, \langle \text{formula} \rangle) \\
& & \mid (\exists \langle \text{variable} \rangle \, \langle \text{formula} \rangle) \\
\langle \text{variable} \rangle & ::= & x \mid y \mid z \mid \ldots
\end{array}
$$

We usually use Greek letters, $\phi, \psi, \ldots$ to denote predicate logic formulas.

If ϕ is a quantified formula $\forall x \, \psi$ or $\exists x \, \psi$ we say that the quantifier *binds* the variable x and called ϕ the *scope* of the quantifier expression $\forall x$ or $\exists x$, respectively.

For example, in $\forall x \exists y\, P(x, y)$ the first quantifier binds x and the second binds y. The scope of the second quantifier is the formula $\exists y\, P(x, y)$.

Free and Bound Variables

The same variable may occur several times in a formula. We distinguish between *free* and *bound* occurrences of variables.

Each occurrence of a variable x that is in the scope of a quantifier expression $\forall x$ or $\exists x$ is said to be *bound*. An occurrence of x that is not bound is said to be *free*.

For example, in $\forall x \exists y \, P(x, y)$ all variable occurrences are bound, whereas in $\exists y \, P(x, y)$ the occurrence of x is free.

The same variable may have both free and bound occurrences in a formula, e.g., the variable x in $Q(x) \vee \exists x \, \neg R(x)$.

Formulas without free occurrences of variables are called *sentences*. Thus, $\forall x \exists y \, P(x, y)$ is a sentence but $\exists y \, P(x, y)$ is not.

Semantically, sentences are formulas that can be true or false, whereas the truth value of a formula with free occurrences of variables depends on the assignment of values to these variables.

Substitution

Terms may be substituted for variables, but only for *free* occurrences of variables in a formula.

Definition.

> If ϕ is a formula, t is a term, and x is a variable, then we denote by $\phi[t/x]$ the formula obtained by replacing each free occurrence of x in ϕ by t.

We also use the letters σ and τ to denote substitutions.

For example, if σ is the substitution $[f(a)/x]$, then $P(x,x)\sigma$ is $P(f(a), f(a))$ and $(Q(x) \vee \exists x \neg R(x))\sigma$ is $Q(f(a)) \vee \exists x \neg R(x)$.

In general, if ϕ contains only bound occurrences of x, then $\phi[t/x]$ is identical to ϕ.

Unfortunately, substitutions may have undesired side effects, and hence their application needs to be constrained.

Definition.

> We say that a term t is *free for* a variable x in a formula ϕ if no free occurrence of x is in the scope of a quantifier that binds a variable y occuring in t.

Is $f(x,y)$ free for x in $\forall x\, [P(x) \wedge Q(y)]$?

Is $f(x,y)$ free for y in $\forall x\, [P(x) \wedge Q(y)]$?

Semantics of Predicate Logic

Let $\mathcal{F}$ be a set of function symbols and $\mathcal{P}$ a set of predicate symbols.

A *model* $\mathcal{M}$ of $(\mathcal{F}, \mathcal{P})$ consists of the following set of data:

1. a non-empty set A (the *universe of concrete values*)

2. for each n-ary function symbol $f \in \mathcal{F}$ an n-ary function $f^{\mathcal{M}} : A^n \to A$

3. for each n-ary predicate symbol $P \in \mathcal{P}$ an n-ary relation $P^{\mathcal{M}} \subseteq A^n$

For example, let $\mathcal{F}$ be the set $\{0, 1, +, *, -\}$ and $\mathcal{P}$ the set $\{=, \leq, <\}$ We may take the set of real numbers as universe and define $0^{\mathcal{M}}$ as the real number 0, $1^{\mathcal{M}}$ as the real number 1, $+^{\mathcal{M}}$ as addition, $*^{\mathcal{M}}$ as multiplication, $-^{\mathcal{M}}$ as subtraction, $=^{\mathcal{M}}$ as the equality predicate, $\leq^{\mathcal{M}}$ as the less-than-or-equal-to relation, and $<^{\mathcal{M}}$ as the less-than relation.

Environments

In evaluating quantified formulas, $\forall x \phi$ or $\exists x \phi$, intuitively one has to determine whether a formula ϕ is true for some or all values a of the specified universe.

This intuition can not be expressed in the syntax of predicate logic. Therefore we have to interpret formulas relative to an "environment."

Definition.

> Let $\mathcal{V}$ be a set of variables and A a non-empty set. By an *environment*, or *look-up table*, for $\mathcal{V}$ and A we simply mean a function
>
> $$l : \mathcal{V} \rightarrow A.$$

If l is a look-up table (for $\mathcal{V}$ and A) and $a \in A$, we denoted by $l[x \mapsto a]$ the "updated" look-up table l' for which $l'(x) = a$ and, for all variables $y \neq x$, $l'(y) = l(y)$.

Interpretation of Terms

Terms of a first-order language can be interpreted as denoting elements of the universe A. More specifically, a model $\mathcal{M}$ for $(\mathcal{F}, \mathcal{P})$ (over universe A) and an environment l for $\mathcal{V}$ and A induce a mapping

$$t_l^{\mathcal{M}} : \mathcal{T}(\mathcal{F}, \mathcal{V}) \to A$$

from the set of terms $\mathcal{T}(\mathcal{F}, \mathcal{V})$ to the set A defined by:

$$t_l^{\mathcal{M}} = \begin{cases} l(x) & \text{if } t \text{ is a variable } x \\ f^{\mathcal{M}}((t_1)_l^{\mathcal{M}}, \ldots, (t_k)_l^{\mathcal{M}}) & \text{if } t = f(t_1, \ldots, t_k) \end{cases}$$

More generally, one may view a term t as inducing a mapping $t^{\mathcal{M}}$ from environments to the given universe, by defining $t^{\mathcal{M}}(l)$ as $t_l^{\mathcal{M}}$.

Example.

> Take the term $t = x \cdot (y + (x + 1) \cdot z)$. If $\mathcal{M}$ is the model above and $l(x) = 3$, $l(y) = 2$, and $l(z) = 1$, then $t_l^{\mathcal{M}} = 18$.

The Satisfaction Relation

Let $\mathcal{M}$ be a model for $(\mathcal{F}, \mathcal{P})$ over universe A and l be an environment for $\mathcal{V}$ and A.

We define a relation

$$\mathcal{M} \models_l \phi$$

by structural induction on formulas ϕ over the first-order language defined by $\mathcal{F}$, $\mathcal{P}$, and $\mathcal{V}$:

(i) If $\phi = P(t_1, \ldots, t_k)$, then $\mathcal{M} \models_l \phi$ holds iff $(t_1)_l^{\mathcal{M}}, \ldots, (t_n)_l^{\mathcal{M}}) \in P^{\mathcal{M}}$.

(ii) If $\phi = \neg\psi$, then $\mathcal{M} \models_l \phi$ holds iff $\mathcal{M} \models_l \psi$ does not hold.

(iii) If $\phi = \psi_1 \wedge \psi_2$, then $\mathcal{M} \models_l \phi$ holds iff both $\mathcal{M} \models_l \psi_1$ and $\mathcal{M} \models_l \psi_2$ hold.

(iv) If $\phi = \psi_1 \vee \psi_2$, then $\mathcal{M} \models_l \phi$ holds iff at least one of $\mathcal{M} \models_l \psi_1$ and $\mathcal{M} \models_l \psi_2$ holds.

(v) If $\phi = \forall x \psi$, then $\mathcal{M} \models_l \phi$ holds iff $\mathcal{M} \models_{l[x \mapsto a]} \psi$ holds for all elements $a \in A$.

(vi) If $\phi = \exists x \psi$, then $\mathcal{M} \models_l \phi$ holds iff $\mathcal{M} \models_{l[x \mapsto a]} \psi$ holds for some element $a \in A$.

If ϕ is a sentence we often write

$$\mathcal{M} \models \phi$$

since the choice of an environment l is then irrelevant.

Semantic Entailment

Definition

Let $\phi_1, \ldots, \phi_n, \psi$ be predicate logic formulas. We write

$$\phi_1, \ldots, \phi_n \models \psi$$

to indicate that, whenever $\mathcal{M} \models_l \phi_i$ for all i with $1 \leq i \leq n$, then $\mathcal{M} \models_l \psi$, for all models $\mathcal{M}$ and environments l.

If $\models \psi$, then the formula ψ is said to be *valid*.

Note that a formula ψ is valid if $\mathcal{M} \models_l \psi$, for all models $\mathcal{M}$ and environments l.

We will discuss proof systems, such as an extended natural deduction calculus, so that the corresponding notion of provability captures the semantic concept of entailment.