Predicate Logic

Predicate logic, also called *first-order logic*, or *FOPC*, is motivated by the recognition that it is difficult to formulate interesting statements in propositional logic.

The problem is that propositional logic does not have any way to distinguish between *individuals* and *properties* as the object of study. A model is just a set of truth values.

You might represent the assertion that "Sam is a man" by the variable sm, and that "Bill is a man" by bm. But by compressing the identity of the individual and the property of the individual into one symbol, it is difficult to discern (and impossible to use in a proof) that we are talking about a shared property, and to deduce, for example, that both Sam and Bill are mortal.

In predicate logic we distinguish between the property (called the *predicate*) and the individual, writing expressions such as $man(bill)$ and $man(sam)$. We also distinguish between *constants* such as *bill* which refer to a particular individual, and *variables*, like x which range over all individuals.

The Syntax of First-Order Logic

As with propositional logic, we will fix an unambiguous notation:

Definitions: Fix the following disjoint, countably infinite sets of *non-logical* symbols:

1. $\mathcal{A} = \{a, b, c, \ldots\}$, the set of *constant symbols*
2. $\mathcal{F} = \{f, g, h, \ldots\}$, the set of *function symbols*
3. $\mathcal{P} = \{p, q, r, \ldots\} \cup \{\doteq\}$, the set of *predicate symbols*

as well as two total functions $arity_{\mathcal{F}} : \mathcal{F} \longrightarrow \mathcal{N}_{+}$ and $arity_{\mathcal{P}} : \mathcal{P} \longrightarrow \mathcal{N}$ determining the *arity* of each function and predicate symbol.

We will call $\mathcal{L} = \mathcal{A} \cup \mathcal{F} \cup \mathcal{P}$ the set of non-logical symbols of the language.

The Syntax of First-Order Logic

Further, fix the following sets of *logical* symbols:

1. $\mathcal{X} = \{x, y, z, \ldots\}$, the (countably infinite) set of *variables*
2. $Conn = \{\wedge, \vee, \Rightarrow, \Leftarrow, \neg, \equiv\}$ the set of *connectives*.
3. $Const = \{\top, \bot\}$ the set of *logical constants*.
4. $Quant = \{\forall, \exists\}$
5. $Aux = \{),,,(\}$ the set of *auxilliary symbols*.

The *first-order language over* $\mathcal{L}$ has as its alphabet the set

$$\Sigma = \mathcal{L} \cup \mathcal{X} \cup Conn \cup Const \cup Quant \cup Aux$$

The Syntax of First-Order Logic

Definition: The set *TERMS* of first-order terms is the smallest set of strings from Σ^* such that:

1. For every $x \in \mathcal{X}$, $x \in TERMS$.
2. For every $c \in \mathcal{A}$, $c \in TERMS$.
3. For every $f \in \mathcal{F}$, if $arity_{\mathcal{F}}(f) = n$ and $t_1, \ldots, t_n \in TERMS$, then $f\mathbf{(}t_1\mathbf{,}\ldots\mathbf{,}t_n\mathbf{)} \in TERMS$.

Assuming $a, b \in \mathcal{A}$, $x, y \in \mathcal{X}$, $f, g \in \mathcal{F}$, $arity_{\mathcal{F}}(f) = 1$, and $arity_{\mathcal{F}}(g) = 2$, then the following are examples of well-formed terms:

- a
- $f(a)$
- $g(x, y)$
- $g(\underline{f(\underline{a})}, \underline{g(\underline{x}, \underline{f(\underline{g(\underline{f(\underline{b})}, \underline{y})})})})$

The Syntax of First-Order Logic

Definition: The set $ATOMS$ of first-order atomic formulas is the smallest set of strings from Σ^* such that:

1. $\bot \in ATOMS$, and $\top \in ATOMS$.
2. For every $p \in \mathcal{P}$, if $arity_{\mathcal{P}}(p) = n$ and $t_1, \ldots, t_n \in TERMS$, then $p\mathbf{(}t_1\mathbf{,}\ldots\mathbf{,}t_n\mathbf{)} \in ATOMS$.

Assuming $a, b \in \mathcal{A}$, $x, y \in \mathcal{X}$, $f, g \in \mathcal{F}$, $arity_{\mathcal{F}}(f) = 1$, $arity_{\mathcal{F}}(g) = 2$, $arity_{\mathcal{P}}(p) = 1$, and $arity_{\mathcal{P}}(q) = 2$ then the following are examples of well-formed terms:

- $p(a)$
- $p(f(a))$
- $q(g(x, y), f(a))$
- $q(f(a), g(f(a), g(x, f(g(f(b), y)))))$

Note that predicates cannot occur among the arguments to a predicate. While the string $g(f(a), y) \in TERMS$, the string $q(p(a), y) \notin ATOMS$.

The Syntax of First-Order Logic

Definition: Given $\mathcal{L}$ and its extension Σ as defined above, and the sets $TERMS$ and $ATOMS$ as defined above, the first-order language of formulas over $\mathcal{L}$ (abbreviated $FORM$), also called the set of *well-formed-formulas* (or *WFF*'s) of the language, is the smallest subset of Σ^* such that:

1. For all $A \in ATOM$, $A \in FORM$.
2. If $\Phi \in FORM$ then $(\neg\Phi) \in FORM$.
3. If $\Phi, \Psi \in FORM$, and $\bullet \in \{\wedge, \vee, \Rightarrow, \Leftarrow, \equiv\}$, then $(\Phi \bullet \Psi) \in FORM$
4. if $\Phi \in FORM$ and $x \in \mathcal{X}$ then $\forall x(\Phi) \in FORM$ and $\exists x(\Phi) \in FORM$

For all $A \in ATOMS$, the formulas $A, (\neg A) \in FORM$, are called *literals*.

Informal Semantics of First-Order Logic

In propositional logic semantics came in the simple form of a mapping (valuation) of propositional letters to truth values. The semantics of first-order formulas are naturally more complex, since the goal is to reason about more complex structures.

Informally:

- Constants stand in for individuals. So, we might have the constant sam that corresponds to some person named Sam.
- Function symbols stand in for functions that map from one or more constants to other constants. So, $son(sam)$ would assume that we interpret the function symbol son as the mapping that takes us from a person to their son. Having applied son to sam we are now, in the compound expression, talking about another person.

Informal Semantics of First-Order Logic

- Predicate symbols stand for truth-valued functions, that correspond to whether the arguments they are applied to have some property (i.e., belong to some relation). Thus we might assume that $serialKiller$ corresponds to the property of being an deranged homicidal maniac, and thus $serialKiller(son(sam))$ has the value true. We could imagine other predicates, such as $alias$ that holds on a pair of constants if the police has found that the two people are the same, so that $alias(son(sam), davidBerkowitz)$ is true.