Predicate Logic

Predicate logic expands the language of *propositional logic* . For example, the assertion "x is less than 5", where x is a variable, is not a proposition because you cannot evaluate it as true or false, unless x is given a value. We cannot make general statements in propositional logic, but we can in predicate logic.

Predicate logic introduces the concept of *quantification*, that allows us to make statements like "some integers are greater than 7".

Quantification Symbols

Two quantification symbols are introduced:

- *Universal quantification*: $\forall x$. ("for all x, it is the case that ..."),
- *Existential quantification*: $\exists x$. ("there exists an x, such that ...").

For example:

$\exists x$. Greater(x,7) which means that there exists an x, such that x is greater than 7.

$\forall x$.(Greater(x,7) $\vee$ Greater(7,x)) which means that for all x, it is the case that x is greater than 7 or 7 is greater than x.

$\forall x$. $\exists y$. Parent(x,y) which means that every person has a parent.

Where in the examples Greater(x,y) means that x is greater than y and Parent(x,y) means that x is the parent of y. Notice the context implies that x is an integer or person – this can be stated if necessary.

Deduction and Semantics

Predicate logic allows us to determine whether statements are true or false. Let us introduce a new notation:

$p \vdash q$,

meaning "p proves q".

A typical deduction rule in predicate logic is called *modus ponens*:

$p, p \Rightarrow q \vdash q$,

which means that if we know p and we know that p implies q then we can deduce q.

The *semantics* (ie context) of the language is given by the *domain of discourse* and an *interpretation function*. The domain of discourse is the values that variables can range over and the interpretation function maps symbols on to the domain.

For example Greater(x,y) could presume that x and y are integers and Parent(x,y) could presume that x and y are people.

Continuing with the Parent(x,y) example:

Parent(*x,y*) means *x* is the parent of *y,*

Male(x) means that *x* is male,

Female(x) means that *x* is female.

We can now define other family relationships in terms of these:

Mother (x,y) ⇔ Parent(x,y) ∧ Female(x) means that x is the mother of y,

Father(x,y) ⇔Parent(x,y) ∧ Male(x) means that x is the father of y,

Daughter(x,y) ⇔Parent(y,x) ∧ Female(x) means that x is the daughter of y,

Son(x,y) ⇔Parent(y,x) ∧ Male(x) means that x is the daughter of y,

Grandparent(x,y) ⇔ Parent(x,z) ∧ Parent(z,y) means that *x* is the grandparent of *y*.

And so on.

Once structures have been defined, particular instances can be stated.

Male(andrew)

Female(barbara)

Parent(andrew, barbara)

Male(colin)

Parent(colin, andrew)

If we query Daughter(barbara,andrew) of Granparent(colin, Barbara) then both would evaluate to true.