Lecture 15: Semantics of First Order Logic

1 Review

The *vocabulary* of predicate calculus consists of predicate symbols (with arity), function symbols (with arity) and variables. The predicate symbols denote relations. The syntax of first order logic is defined on this vocabulary:

- Variables + Function symbols $\longrightarrow$ Terms
- Terms + Predicate symbols $\longrightarrow$ Formulas

Predicate calculus can be used to model mathematical structures. A *mathematical structure* A consists of a tuple $(D, f_1^A, \ldots, f_n^A, P_1^A, \ldots, P_m^A)$ where

- D is the universe or domain of 'atomic' objects.
- f_i^A is the interpretation of f_i in A.
- $f_i : D^k \to D$ where k is the arity of f_i.
- P_i^A is the interpretation of P_i in A.
- $P_i \subseteq D^k$ where k is the arity of P_i.

For example, we can model the mathematical concept of graphs as a pair $G = (V, E)$ where V is the universe of vertices and E is the binary edge relation.

2 Examples of First-Order Logic

- "There is a mother to all children" –

  $$(\forall x)(child(x) \to (\exists y)(mother(y, x)))$$

  Note that this is different from

  $$(\exists y)(\forall x)(child(x) \to (mother(y, x)))$$

  Contrast this with "There is a president for all Americans", which we'd express as

  $$(\exists y)(\forall x)(American(x) \to (president(y, x)))$$

- “For every girl, there is a boy who loves only her” –

  $(\forall y)(girl(x) \rightarrow (\exists x)(loves(x,y) \wedge (\forall z)((girl(z)) \wedge loves(x,z)) \rightarrow (z \approx y))))$

- It depends on the definition os ‘is’:
  - “Leebron is a university president” –
  $$UniversityPresident(Leebron)$$
  - “Leebron is the President of Rice” –
  $$Leebron \approx President(Rice)$$
  - “Rice has a president” –
  $$(\exists x)(president(x, Rice))$$

3 Semantics of First-Order logic

Semantics relates the syntax to the world (relational structure). $A \models \varphi$ denotes that formula φ is true in the world A. Here ‘$\models$’ is the semantical relation.

Consider the statement $A \models (x = 2)$. Does it make sense to ask whether the formula ‘$x = 2$’ is true in the world A? The truth of the formula depends on the value of x, but x is a variable and can take any value. Some values of x might make the formula true, while others might falsify it. Thus, we need to qualify our answer by saying that the formula is true when x has a certain value. To do this we can use a function α that assigns values to variables. Such a function is called a ‘binding’. Thus, statements about semantics have the form $A, \alpha \models \varphi$ where A is a structure, φ is a formula and α is a variable assignment, i.e., $\alpha : Var \rightarrow D$. Note that our notion of truth is now a ternary relation, as opposed to the binary relation of propositional logic.

We next define this notion formally. First we extend the notion of bindings to terms.

Definition 1 *Given structure A and $\alpha : Var \rightarrow D$, we define $\overline{\alpha} : Term \rightarrow D$ as follows:*

1. $\overline{\alpha}(x) = \alpha(x)$*, where x is a variable*
2. $\overline{\alpha}(f(t_1, \ldots, t_k)) = f^A(\overline{\alpha}(t_1), \ldots, \overline{\alpha}(t_k))$*, where k is the arity of f, and* $t_1, \ldots, t_k \in$ TERMS.

Note that if f is a 0-ary function then $\overline{\alpha}(f) = f^A$.

We now define $A, \alpha \models \varphi$. (note: other notations in use include $A \models_\alpha \varphi$ and $A \models \varphi[\alpha]$).

1. $A, \alpha \models P(t_1, \ldots, t_k)$ if $(\overline{\alpha}(t_1), \ldots, \overline{\alpha}(t_k)) \in P^A$.

2. $A, \alpha \models (\theta \wedge \psi)$ if $A, \alpha \models \theta$ and $A, \alpha \models \psi$.

3. $A, \alpha \models (\neg\theta)$ if $A, \alpha \not\models \theta$.

4. $A, \alpha \models (\exists x)\varphi$ if there is some $a \in D$ such that $A, \alpha[x \mapsto a] \models \varphi$.

5. $A, \alpha \models (\forall x)\varphi$ if for all $a \in D$ we have $A, \alpha[x \mapsto a] \models \varphi$.

We can think of an assignment as an array. If $\alpha(x_i) = a_i$, then an assignment to $x_1, \ldots, x_n$ is the array $[a_1, \ldots, a_n]$. Then $\alpha[x_i \mapsto a_i']$ simply updates the ith entry of the array.

4 Distinction between free and bound variables

It is clear that variables used in quantifiers are different from the other variables in that the quantifier prevents the assignment from affecting it.
For example, consider the formula:

$$(\exists x)(\exists x)p(x)$$

This formula will be satisfiable if the following are satisfiable:

- $A, \alpha \models (\exists x)(\exists x)p(x)$ or
- $A, \alpha[x \mapsto a] \models (\exists x)p(x)$ or
- $A, \alpha[x \mapsto a][x \mapsto b] \models p(x)$

That means if there is an $a \in D$ and there is a $b \in D$, we can find a structure A and an assignment α so that $A, \alpha[x \mapsto a][x \mapsto b] \models p(b)$. From the definition of mapping on assignments, this becomes $A, \alpha[x \mapsto b] \models p(b)$. So the outermost quantifier does not affect the formula at all because the inner quantifier somehow makes x "disappear" from the formula.

Let's formalize this. First we define the set of variables that occur in a formula:

Definition 2 *The set of variables in a term t, denoted by $Vars(t)$, can be thought of as a function $Vars : Terms \rightarrow 2^{Variables}$, and is defined as:*

1. $Vars(x) = \{x\}$
2. $Vars(f(t_1, \ldots, t_k)) = \bigcup_i Vars(t_i)$

Definition 3 *The set of variables in a formula φ, denoted by $Vars(\varphi)$, can also be thought of as a function $Vars : Terms \rightarrow 2^{Variables}$ and equals:*

1. $Vars(P(t_1, \ldots, t_k)) = \bigcup_i Vars(t_i)$
2. $Vars(\neg\varphi) = Vars(\varphi)$

3. $Vars(\theta \wedge \psi) = Vars(\theta) \cup Vars(\psi)$
4. $Vars((\exists x)\varphi) = Vars((\forall x)\varphi) = Vars(\varphi) \cup \{x\}$

We can see from examples that variables occurring with a quantifier are treated differently in the assignment. To distinguish them, we call quantified variables **bound**, and unquantified variables **free**.

Definition 4 *The set of free variables in a formula φ, denoted by $FVars(\varphi)$, can be thought of as a function $FVars : Form \rightarrow 2^{Variables}$ and is defined as:*

1. $FVars(P(t_1, \ldots, t_k)) = \bigcup_i Vars(t_i)$
2. $FVars(\neg\varphi) = FVars(\varphi)$
3. $FVars(\theta \circ \psi) = FVars(\theta) \cup FVars(\psi)$
4. $FVars((\exists x)\varphi) = FVars((\forall x)\varphi) = FVars(\varphi) - \{x\}$