Lecture notes 12.0

First-order logic: syntax

COMP 2411, session 1, 2004

Introduction (1)

First-order logic is based on a formal language with considerably more expressive power than the language of propositional logic.

For instance, *Tom likes Jerry*, *Jerry likes Tom*, *someone likes Tom*, *Jerry likes everyone* can only be 'translated' into propositional logic as $p,\ q.\ r,\ s$.

In first-order logic, they can be translated as:

- $\mathrm{likes}(\mathrm{tom}, \mathrm{jerry})$
- $\mathrm{likes}(\mathrm{jerry}, \mathrm{tom})$
- $\exists x\ \mathrm{likes}(x, \mathrm{tom})$
- $\forall x\ \mathrm{likes}(\mathrm{jerry}, x)$

henceforth revealing formal relationships between those statements.

Introduction (2)

Whereas the syntax of propositional logic is defined from propositional atomic formulas and boolean operators, the syntax of first-order logic is defined from:

- a fixed set of logical symbols encompassing
  - (first-order) variables,
  - boolean operators,
  - quantifiers,
  - possibly the predicate symbol $=$;
- a set of nonlogical symbols encompassing
  - possibly functions symbols,
  - possibly predicate symbols distinct from $=$.

Introduction (3)

The formal meaning of a logical symbol is fixed.

- A variable will be an arbitrary name for 'something.'
- We already know the meaning of the boolean operators.
- There are two quantifiers:
  - $\exists$, the existential quantifier, whose meaning is *there exists* or *some*;
  - $\forall$, the universal quantifier, whose meaning is *all* or *every*;
- $=$ denotes identity; its meaning is *is the same as*.

The formal meaning of a nonlogical symbol is not fixed: these symbols enable to describe various families of 'worlds,' and each such description determines the formal meaning of the nonlogical symbols.

Function and predicate symbols (1)

Function symbols are meant to be used to denote entities, things, objects, people, etc., whereas predicate symbols are meant to be used to denote properties or relations.

Both function and predicate symbols have an arity (number of arguments). Predicate symbols of arity 0 would correspond to propositional atomic formulas, enabling to directly embed propositional logic into first-order logic.

Still in practice, predicate symbols are assumed to be of nonnull arity.

On the other hand, function symbols can be nullary (have a null arity), in which case they are called constants.

Function and predicate symbols (2)

For instance:

- john, π, $\mathrm{father\_of}$, $\mathrm{middle\_point}$ could be examples of nullary, nullary, unary, and binary function symbols, respectively.
- $\mathrm{is\_a\_boy}$, likes, $\mathrm{graduated\_from\_in}$ could be examples of unary, binary, and ternary predicate symbols, respectively.

Though such a choice of function and predicate symbols suggests an intended interpretation, we often prefer symbols like a, a' f, R, R_1, whether we have an intended interpretation in mind or not for these symbols.

We will see that it is essential to consider unintended interpretations as well as intended ones, and from this point of view, abstract symbols help a lot. . .

Function and predicate symbols (3)

Nonnullary function symbols and predicate symbols are used to denote entities and relations, but they do not denote entities and relations by themselves; terms and formulas do.

Terms and formulas are built from the (logical and nonlogical) symbols, according to certain formation rules.

Terms will denote individuals in a "world" (abstract picture of “things” in the broader sense: human beings, animals, objects, events, mathematical entities, etc.). They correspond to nominal expressions in English.

Formulas will denote declarative statements, involving properties of individuals or relations between individuals. They correspond to verbal expressions in English.

Function and predicate symbols (4)

Terms do not need formulas, but formulas need terms. *E.g.*, ‘Go out!’ contains no nominal expression, but it is not a declarative statement, contrary to ‘John is going out.’

Terms do not need formulas, but formulas need terms. *E.g.*, ‘Go out!’ contains no nominal expression, but it is not a declarative statement, contrary to ‘John is going out.’ More precisely:

- Terms are built from the variables and the function symbols.
- Formulas are built from the terms, the predicate symbols, the boolean operators, and the quantifiers.

Vocabularies (1)

The vocabulary of a first-order language *without* equality consists of:

- a denumerable set of (first-order) variables;
- the boolean operators $\neg$, $\vee$, $\wedge$, $\rightarrow$ and $\leftrightarrow$;
- the quantifiers $\exists$ and $\forall$;
- a (possibly empty) countable set of function symbols;
- a nonempty, countable set of predicate symbols.

Vocabularies (2)

The vocabulary of a first-order language *with* equality consists of:

- a denumerable set of (first-order) variables;
- the boolean operators $\neg$, $\vee$, $\wedge$, $\rightarrow$ and $\leftrightarrow$;
- the quantifiers $\exists$ and $\forall$;
- the binary predicate symbol $=$;
- a (possibly empty) countable set of function symbols;
- a (possibly empty) countable set of predicate symbols.

Terms: definition

Given a vocabulary V, the set of terms over V is inductively as the smallest set such that:

- every variable is a term over V;
- for all $n \in \mathbb{N}$, n-ary function symbols f in V and terms $t_1, \ldots, t_n$ over V, $f(t_1, \ldots, t_n)$ is a term over V.

In particular, every constant in V is a term over V.

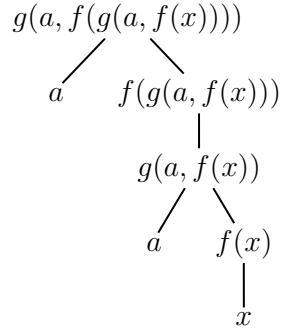
A term is *closed* iff it contains no variable. Note that:

- there exists denumerably many terms over V;
- there exists a closed term over V iff V contains a constant.

Terms: examples (1)

Suppose for instance that V contains the function symbols $a/0, f/1, g/2$ (we use h/n to express that the arity of h is equal to n). Then $g(a, f(g(a, f(x))))$ is a term over V.

Like any term, it can be represented by a parse tree:



Terms: examples (2)

Suppose the world we have in mind is $\mathbb{N}$. To denote its individuals, we have two options:

- include infinitely many constants in the vocabulary, like zero, one, two, $\mathrm{three}\ldots$;
- include one constant and one unary function symbol in the vocabulary, like $\overline{0}/0$ and $s/1$.

We would denote 3 by three in the first case, and by $s(s(s(\overline{0})))$ in the second case.

When a vocabulary V contains a unary function symbol f, we often use $f^n(t)$ as an abbreviation for $\overbrace{f(\ldots(f(}^{n}t\overbrace{)\ldots)}^{n}$, for all $n \in \mathbb{N}$ and terms t over V. *E.g.*, $s^0(\overline{0})$ is $\overline{0}$, $s^1(\overline{0})$ is $s(\overline{0})$, and $s^4(\overline{0})$ is $s(s(s(s(\overline{0}))))$.

Terms: examples (3)

Suppose that the world we have in mind is the Australian population today. We could use a vocabulary V that contains a few constants for the VIP, *e.g.*, $\mathrm{howard}/0$ and $\mathrm{warne}/0$, plus other function symbols like $\mathrm{father}/1$, $\mathrm{son}/2$, etc.

Then for all terms t, nonnull $n \in \mathbb{N}$ and terms t, t_1, t_2 over V, $\mathrm{father}^n(t)$ is the abbreviation of a term over V and $\mathrm{son}(t_1, t_2)$ is a term over V.

Some terms do not have any intuitive meaning w.r.t. the world we have in mind, *e.g.*:

- $\mathrm{father}(\mathrm{father}(\mathrm{father}(\mathrm{father}(\mathrm{warne}))))$, or
- $\mathrm{son}(\mathrm{howard}, \mathrm{warne})$.

Formulas: definition

Given a vocabulary V, the set of formulas over V is inductively as the smallest set such that:

- for all $n > 0$, n-ary predicate symbols P in V distinct from $=$ and for all terms $t_1, \ldots, t_n$ over V, $P(t_1, \ldots, t_n)$ is a formula over V;
- if V contains $=$ then for all terms t_1, t_2 over V, $t_1 = t_2$ is a formula over V;
- for all formulas φ over V, $\neg\varphi$ is a formula over V;
- for all formulas φ, ψ over V, $(\varphi \vee \psi)$, $(\varphi \wedge \psi)$, $(\varphi \rightarrow \psi)$ and $(\varphi \leftrightarrow \psi)$ are formulas over V;
- for all formulas φ over V and for all variables x, $\exists x\varphi$ and $\forall x\varphi$ are formulas over V.

Formulas of the first kind are called atomic formulas or atoms.

Formulas: examples (1)

Suppose for instance that V contains the function symbols $a/0, f/1, g/2$ and the predicate symbols $P/1, Q/2$. The following are examples of formulas over V:

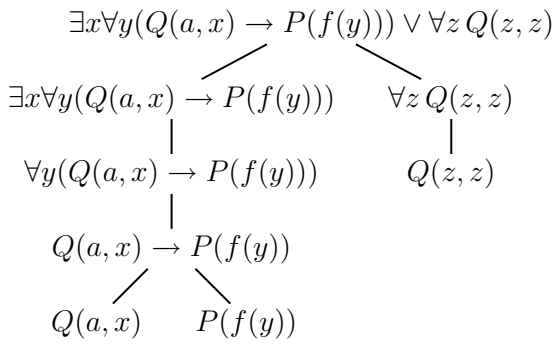
- $Q(a, f(x))$
- $\exists x \forall y (Q(a, x) \rightarrow P(f(y)))$
- $\exists x \forall y (Q(a, x) \rightarrow P(f(y))) \vee \forall z\, Q(z, z)$
- $\exists x \forall y \exists x \forall x\, Q(a, y)$

We use the same precedence and associativity rules for boolean operators to omit parentheses as we do in propositional logic.

Formulas: examples (2)

Like terms, formulas can be represented by parse-trees. Below is the parse-tree for

$$\exists x \forall y (Q(a,x) \rightarrow P(f(y))) \vee \forall z\, Q(z,z) :$$



Formulas: examples (3)

To describe a family where a woman called Mary has a child called Tom, we could use a vocabulary containing two constants mary and tom, a unary function symbol father, and two binary predicate symbols loves and $\mathrm{is\_child\_of}$.

The following are examples of formulas.

- *Mary loves Tom*: $\mathrm{loves}(\mathrm{mary}, \mathrm{tom})$
- *Tom does not loves Mary*: $\neg\mathrm{loves}(\mathrm{tom}, \mathrm{mary})$
- *Mary's father loves Tom*: $\mathrm{loves}(\mathrm{father}(\mathrm{mary}), \mathrm{tom})$
- *Tom loves himself*: $\mathrm{loves}(\mathrm{tom}, \mathrm{tom})$
- *Tom is Mary's child*: $\mathrm{is\_child\_of}(\mathrm{tom}, \mathrm{mary})$
- *Mary has a child*: $\exists \mathrm{x}\ \mathrm{is\_child\_of}(\mathrm{x}, \mathrm{mary})$
- *Not everybody has a child*: $\neg\forall \mathrm{x} \exists \mathrm{y}\ \mathrm{is\_child\_of}(\mathrm{y}, \mathrm{x})$

Formulas: examples (4)

Suppose that the vocabulary also contains a unary predicate symbol $\mathrm{is\_a\_mother}$, and that the language contains equality.

The following are examples of formulas.

- *Every mother loves her children*:
  $\forall \mathrm{x} \forall \mathrm{y}((\mathrm{is\_a\_mother}(\mathrm{x}) \wedge \mathrm{is\_child\_of}(\mathrm{y}, \mathrm{x})) \rightarrow \mathrm{loves}(\mathrm{x}, \mathrm{y}))$
- *Some mothers have more than two children*:
  $\exists \mathrm{x} \exists \mathrm{y}_1 \exists \mathrm{y}_2 \exists \mathrm{y}_3 (\mathrm{is\_a\_mother}(\mathrm{x}) \wedge$
  $is\_child\_of(y_1, x) \wedge is\_child\_of(y_2, x) \wedge \mathrm{is\_child\_of}(\mathrm{y}_3, \mathrm{x})$
  $\wedge \mathrm{y}_1 \neq \mathrm{y}_2 \wedge \mathrm{y}_1 \neq \mathrm{y}_3 \wedge \mathrm{y}_2 \neq \mathrm{y}_3)$

Everybody loves one of Mary's children CANNOT be represented by $\forall \mathrm{x} \exists \mathrm{y}\ \mathrm{loves}(\mathrm{x}, \mathrm{is\_child\_of}(\mathrm{y}, \mathrm{mary}))$, because that is NOT a formula.

Formulas: examples (5)

If the vocabulary also contains a unary function symbol $\mathrm{child\_of}$, then *everybody loves Mary's only child* can be represented by $\forall \mathrm{x}\, \mathrm{loves}(\mathrm{x}, \mathrm{child\_of}(\mathrm{mary}))$.

If Mary has many children, then $\forall \mathrm{x}\, \mathrm{loves}(\mathrm{x}, \mathrm{child\_of}(\mathrm{mary}))$ does not represent *everybody loves one of Mary's children*: two different people might not love the same child.

If Mary has many children, then $\forall \mathrm{x}\, \mathrm{loves}(\mathrm{x}, \mathrm{child\_of}(\mathrm{mary}))$ does not represent *one of Mary's children is loved by everybody*: there is no guarantee that $\mathrm{child\_of}(\mathrm{mary})$ 'picks up' the beloved child.

Formulas: examples (6)

To represent families, where couples have a variable number of children, we can use a vocabulary consisting a constant none, a constant for every person, a binary function symbol child, and a ternary function symbol family.

The family consisting of the parents Bill and Mary and their children Tom and Alice can then be represented by the term $\mathrm{family}(\mathrm{bill}, \mathrm{mary}, \mathrm{child}(\mathrm{tom}, \mathrm{child}(\mathrm{alice}, \mathrm{none})))$.

As often when the vocabulary contains 'meaningful' function symbols whose arity is nonnull, many terms, like $\mathrm{child}(\mathrm{family}(\mathrm{mary}, \mathrm{mary}, \mathrm{mary}), \mathrm{mary})$, are 'meaningless.'

Function symbols have a fixed arity, but families have a variable number of members. Hence a construction of this kind cannot be avoided.

Subformulas

A subformula of a formula φ is any formula involved in the inductive definition of φ being a formula, *i.e.*, any formula that labels one of the nodes of φ's parse tree. For example, the subformulas of $\exists x \forall y (Q(a, x) \to P(f(y))) \vee \forall z\, Q(z, z)$ are:

- $\exists x \forall y (Q(a, x) \to P(f(y))) \vee \forall z\, Q(z, z)$
- $\exists x \forall y (Q(a, x) \to P(f(y)))$
- $\forall y (Q(a, x) \to P(f(y)))$
- $Q(a, x) \to P(f(y))$
- $Q(a, x)$
- $P(f(y))$
- $\forall z\, Q(z, z)$
- $Q(z, z)$

Bound and free occurrences

An occurrence of a variable x in a formula φ is bound if it occurs in a subformula of φ of the form $\exists x \psi$ or $\forall x \psi$.

An occurrence of a variable that is not bound is free.

For instance, in $\exists x P(x, y) \wedge \neg \exists z \forall y (Q(x, y, z) \vee \exists x R(x, z))$, the bound occurrences of variables are:

- the first, second, fourth and fifth occurrences of x;
- the second and third occurrences of y;
- the first, second and third occurrences of z.

x and y have both bound and free occurrences in φ.

Closures

A formula is closed if it contains no free occurrence of a variable.

Let $x_1, \ldots, x_n$ be all variables that occur free in a formula φ.
A formula of the form $\forall x_1 \ldots \forall x_n \varphi$ is called a universal closure of φ and is denoted $\forall \varphi$.
A formula of the form $\exists x_1 \ldots \exists x_n \varphi$ is called an existential closure of φ and is denoted $\exists \varphi$.

Let $\varphi = \exists x P(x, y) \wedge \neg \exists z \forall y (Q(x, y, z) \vee \exists x R(x, z))$. Existential and universal closures of φ are, respectively:

- $\exists x \exists y (\exists x P(x, y) \wedge \neg \exists z \forall y (Q(x, y, z) \vee \exists x R(x, z)))$
- $\forall x \forall y (\exists x P(x, y) \wedge \neg \exists z \forall y (Q(x, y, z) \vee \exists x R(x, z)))$

Universal and existential closures of formulas are obviously closed.

Substitutions (1)

Given a formula φ, a variable x and a term t, t is said to be substituble for x in φ iff all free occurrences of x in φ can be replaced by t and no occurrence of a variable in t becomes bound in the resulting formula.

Let $\varphi = \exists x P(x,y) \wedge \neg\exists z \forall y (Q(x,y,z) \vee \exists x R(x,z))$, $t_1 = f(a,y,u)$ and $t_2 = g(v,x)$.

- t_1 is not substituble for x in φ
- t_1 is substituble for y in φ
- t_1 is substituble for z in φ
- t_2 is substituble for x in φ
- t_2 is not substituble for y in φ
- t_2 is substituble for z in φ

Substitutions (2)

Given a formula φ, a variable x and a term t, if t is substituble for x in φ, then $\varphi[t/x]$ denotes the formula obtained from φ by substituting all free occurrences of x in φ by t.

With the previous example:

- $\varphi[t_1/y] = \exists x P(x, f(a,y,u)) \wedge \neg\exists z \forall y (Q(x,y,z) \vee \exists x R(x,z))$
- $\varphi[t_2/x] = \exists x P(x,y) \wedge \neg\exists z \forall y (Q(g(v,x),y,z) \vee \exists x R(x,z))$
- $\varphi[t_1/z] = \varphi[t_2/z] = \varphi$