FIRST ORDER LOGIC

based on

Huth & Ruan
Logic in Computer Science:
Modelling and Reasoning about Systems
Cambridge University Press, 2004

First order logic

(also called predicate logic)

- Essentially, first order logic adds variables in logic formulas

Assume we have three cats (Anna, Bella, Cat), and cats have tails.

In **propositional logic**, we could write:
iscatAnna, iscatBella, iscatCat, iscatAnna → hastailAnna, iscatBella → hastailBella, iscatCat → hastailCat.

In **first order logic**, we would write:
iscat(anna), iscat(bella), iscat(cat), $\forall X\ (\text{iscat}(X) \rightarrow \text{hastail}(X))$

Terms

- **<u>Terms</u>** are defined as follows:
  - any variable is a term
  - if $c \in \mathcal{F}$ is a nullary function (no parameters), then c is a term
  - if $t_1, t_2, \ldots, t_n$ are terms and f is a function of arity *n*>0 then $f(t_1, t_2, \ldots, t_n)$ is a term
  - nothing else is a term

Terms

- Examples of well-formed terms, assuming f is a function of arity 2, g is a function of arity 1, c is a function of arity 0:
  - $f(g(c),g(g(c)))$
  - $f(f(g(c),c),g(c))$
  - $g(g(g(f(c,c))))$
- Examples of badly-formed terms, for the above functions:
  - $f(c)$
  - $f(c,c) \rightarrow g(c)$

First order logic

- **<u>(Well-formed) formulas</u>** in first order logic for a set of functions symbols $\mathcal{F}$ and predicate symbols $\mathcal{P}$ are obtained by using the following construction rules, and only these rules, a finite number of times:
  - If P is a predicate symbol of arity *n* and $t_1, \ldots, t_n$ are terms over $\mathcal{F}$, then $P(t_1, \ldots, t_n)$ is a well-formed formula.
  - if ϕ is a well-formed formula, then so is $(\neg\phi)$
  - if ϕ and ψ are well-formed formulas, then so is $(\phi \wedge \psi)$
  - if ϕ and ψ are well-formed formulas, then so is $(\phi \vee \psi)$
  - if ϕ and ψ are well-formed formulas, then so is $(\phi \rightarrow \psi)$
  - if ϕ is a formula and *x* is a variable, then $(\forall x\phi)$ and $(\exists x\phi)$ are formulas

Universal Quantifier

- $\forall$ denotes the **universal quantifier**
- It can be read as "for all"

$$\forall X \,(\, \text{iscat}(X) \rightarrow \text{hastail}(X)\,)$$

"for all *X* it is true that if *X* is a cat, then *X* has a tail"

<u>Confusion about capitals</u>

$\forall X \,(\, \text{iscat}(X) \rightarrow \text{hastail}(X)\,)$

$\forall x \,(\, \text{Iscat}(x) \rightarrow \text{Hastail}(x)\,)$

both notations can be used, as long as you do this consistently!

Existential Quantifier

- $\exists$ denotes the **existential quantifier**
- It can be read as “there is”

$$(\exists X\ \text{student}(X)) \rightarrow (\exists Y\ \text{university}(Y))$$

“if there is an *X* which is a student, then there is an *Y* which is a university”

First order logic

- Given the following predicate symbols:
  - *S(x,y)*: *x* is a son of *y*
  - *F(x,y)*: *x* is the father of *y*
  - *B(x,y)*: *x* is a brother of *y*

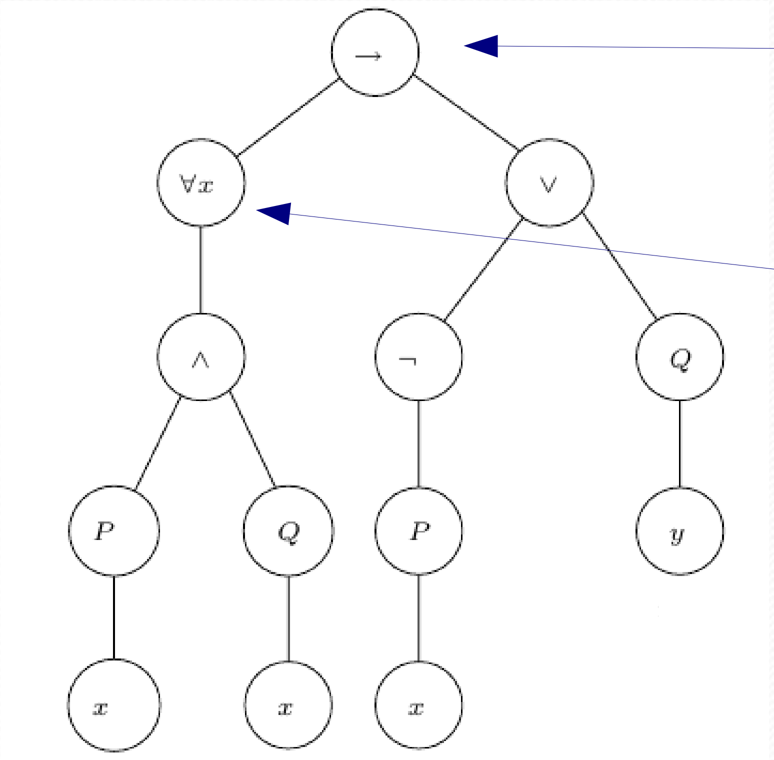
  the following are well-formed formulas:
  - $\forall x \forall y \forall z (F(x,y) \wedge S(y,z) \rightarrow B(x,z))$
  - $\forall x \forall y (S(x,y) \rightarrow F(y,x))$
  - $\forall x \forall y (F(x,y) \rightarrow S(x,y))$
  - $\forall x ((\exists y S(x,y)) \rightarrow (\exists z F(x,z)))$

Note: formulas are well-formed if their syntax is correct

Free & bound variables

- We can build parse trees for formulas

$$(\forall x\,(P(x) \wedge Q(x))) \rightarrow (\neg P(x) \vee Q(y))$$

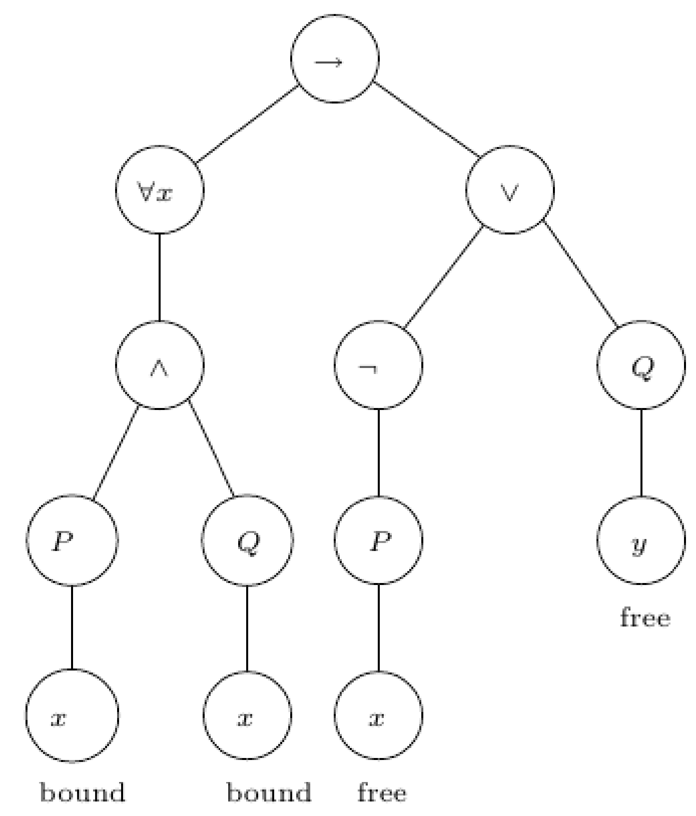


binary node for binary connective

unary node for quantifiers, unary connective

Free & bound variables

- A quantifier for variable *x* **binds** all variables *x* occurring below its corresponding node in the parse tree; a variable which is not bound is **free**
- If there is no free variable, the formula is **closed**



Interpretations

- Let $\mathcal{F}$ be a set of function symbols and $\mathcal{P}$ a set of predicate symbols, each symbol with a fixed number of arguments. An **<u>interpretation</u>** $\mathcal{I}$ of the pair $(\mathcal{F}, \mathcal{P})$ consists of the following data:
  - A non-empty set *A*, the universe of values
  - For each nullary function symbol $f \in \mathcal{F}$ a, a concrete element $f^{\mathcal{I}}$ of *A*
  - for each $f \in \mathcal{F}$ with arity *n*>0, a concrete function $\mathcal{F}^{\mathcal{I}} : A^n \to A$ from A^n, the set of *n*-tuples over *A*, to *A*
  - for each $P \in \mathcal{P}$ with arity *n*>0, a subset $P^{\mathcal{I}} \subseteq A^n$ of *n*-tuples over *A*

Interpretations: Example

- Assuming *f* is a function of arity 2, *g* is a function of arity 1, *c* is a function of arity 0, and *P* is unary
- A possible interpretation is:
  - $A = \{0, 1, 2\}$
  - $c^{\mathcal{I}} = 0$
  - $g^{\mathcal{I}}(0) = 1, g^{\mathcal{I}}(1) = 2, g^{\mathcal{I}}(2) = 2$

    $f^{\mathcal{I}}(x, y) = \min(2, x + y)$
  - $P^{\mathcal{I}} = \{0, 2\}$

For a given formula, we will define when the interpretation makes it true

Interpretations

- Given an interpretation:
  - $A = \{0, 1, 2\}$
  - $c^{\mathcal{I}} = 0$
  - $g^{\mathcal{I}}(0) = 1, g^{\mathcal{I}}(1) = 2, g^{\mathcal{I}}(2) = 2$

    $f^{\mathcal{I}}(x, y) = \min(2, x + y)$
  - $P^{\mathcal{I}} = \{0, 2\}$
- Examples of formulas that are true for this interpretation:
  - $P(c) \wedge P(g(g(c)))$
  - $\exists X \; P(g(g(X)))$

Look-up Tables

- A look-up table for a universe *A* of values and variables *var* is a function: *l: var* → *A* from the set of variables *V* to *A*

$l(x)$ may be undefined for some x

- We denote by $l[x \mapsto a]$ the look-up table in which variable *x* in *var* is mapped to value *a* in *A*, and all other variables *y* are mapped to *l(y)*

Given $l(X)=1$, $l(Y)=2$.
The look-up table denoted by $l[X \mapsto 3]$ is the look-up table in which $l(X)=3$, $l(Y)=2$

Satisfaction of Formulas

- Given an interpretation $\mathcal{I}$ for a pair $(\mathcal{F}, \mathcal{P})$ and given a look-up table for all free variables in formula φ , we define the satisfaction relation $\mathcal{I} \models_l \varphi$ as follows:
  - If φ is of the form $P(t_1, t_2, \ldots, t_n)$, then we interpret the terms $t_1, \ldots, t_n$ by replacing all variables with their values according to *l*. In this way we compute values $a_1, \ldots a_n$ of *A*, where we interpret any function symbol $f \in \mathcal{F}$ by $f^{\mathcal{I}}$. Now $\mathcal{I} \models_l P(t_1, \ldots, t_n)$ holds iff $(a_1, \ldots, a_n)$ is in the set $P^{\mathcal{I}}$.
  - …

Satisfaction of Formulas

- Given an interpretation $\mathcal{I}$ for a pair $(\mathcal{F}, \mathcal{P})$ and given a look-up table for all free variables in formula ϕ, we define the satisfaction relation $\mathcal{I} \models_l \phi$ as follows:
  - If ϕ is of the form $\forall x \ \psi$, then $\mathcal{I} \models_l \forall x \ \psi$ holds iff $\mathcal{I} \models_{l[x \mapsto a]} \psi$ holds for **all** *a* in *A*
  - If ϕ is of the form $\exists x \ \psi$, then $\mathcal{I} \models_l \exists x \ \psi$ holds iff $\mathcal{I} \models_{l[x \mapsto a]} \psi$ holds for **some** *a* in *A*
  - If ϕ is of the form $\neg\psi$, then $\mathcal{I} \models_l \neg\psi$ holds iff $\mathcal{I} \models_l \psi$ does not hold
  - If ϕ is of the form $\psi_1 \wedge \psi_2$, then $\mathcal{I} \models_l \psi_1 \wedge \psi_2$ holds if both $\mathcal{I} \models_l \psi_1$ and $\mathcal{I} \models_l \psi_2$ hold

and similar for $\vee$ and $\rightarrow$

Satisfaction of Formulas

- If φ is a closed formula, then interpretation $\mathcal{I}$ is a **model** for φ, denoted by $\mathcal{I} \models \varphi$, iff $\mathcal{I} \models_l \varphi$ (where *l* does not define an image for any of the variables)
- "a model is an interpretation which makes the formula true"

Entailment

- First order logic formula ϕ **semantically entails** first order logic formula ψ, denoted by $\phi \models \psi$, iff all models of formula ϕ are also models for formula ψ.

- Natural deduction rules can also be defined for first-order logic (but will not be discussed here)

Bad news
$\phi \models \psi$ is undecidable: no algorithm can exist to decide this relation for any pair of formulas

Horn Clauses

- A first-order logic formula is a Horn clause iff
  - it is closed
  - it is a formula of the form

    $$\forall x_1 \cdots \forall x_n (l_1 \vee \cdots \vee l_m)$$

    i.e., it is a disjunction of literals, and all variables are universally quantified
  - it has at most one positive literal

$$\forall x \forall y \forall z (\neg F(x, y) \vee \neg S(y, z) \vee B(x, z))$$
$$\forall x \forall y \forall z (F(x, y) \wedge S(y, z) \rightarrow B(x, z))$$
$$\forall x \forall z ((\exists y \ (F(x, y) \wedge S(y, z))) \rightarrow B(x, z))$$

Logic Programming

- Resolution can also be defined for clauses in first order logic and is the basis of logic programming

$$\forall x \forall y \forall z (F(x,y) \wedge S(y,z) \rightarrow B(x,z))$$

In the Prolog language:

```
b(X,Z) :- f(X,Y), s(Y,Z)
f(anna,bill).
s(bill,jack).
```

Given knowledge

```
:- b(anna,jack)
True.
```

Query

Answer