




First Order Logic

CIS*3700 (Winter 2007)



First Order Logic (FOL)

- Propositional logic assumes that the world contains facts.
- First-Order Logic (like natural language) assumes that the world contains:
 - Objects: people, houses, numbers, colors, baseball games, wars, ...
 - Relations: red, round, prime, brother of, bigger than, part of, comes between, ...
 - Functions: father of, best friend, one more than, plus, ...

Syntax

<Sentence> → <AtomicSentence>
 | (<Sentence> <Connective> <Sentence>)
 | <Quantifier> <Variable>, ... <Sentence>
 | ¬ <Sentence>

<AtomicSentence> → <Predicate>(<Term>, ...) | <Term> = <Term> | TRUE | FALSE

<Term> → <Function>(<Term>, ...) | <Constant> | <Variable>

<Connective> → ⇒ | ∧ | ∨ | ⇔



<Quantifier> → ∀ | ∃

<Constant> → A | X_i | John | ...

<Variable> → a | x | s | ...



<Predicate> → Before | HasColor | Raining | ...

<Function> → Mother | LeftLeg | ...

Terms vs Predicates

- Terms:
 - John: a particular individual
 - Father(John): the father of John
 - Father(Father(John)): the grandfather of John
- Predicates:
 - Love(Fred, Mary)
 - Father(Richard, John)
 - Married(Father(John), Mother(John))

Quantifiers



- Quantifiers have scopes: an occurrence of a variable is bound iff it is in the scope of a quantifier; otherwise, it is free.

$\forall x P(x, y)$ --- x is bound and y is free

$\forall x P(x, y) \vee \exists y Q(y)$ --- y is both free and bound



$\forall x (\text{Man}(x) \Rightarrow \text{Mortal}(x))$ --- All men are mortal

$\exists x (\text{Man}(x) \wedge \text{Wise}(x))$ --- There exists a wise man

Quantifiers

- Typically, \Rightarrow is the main connective with \forall
 $\forall x (\text{At}(x, \text{Guelph}) \Rightarrow \text{Smart}(x))$ --- Everyone at Guelph is smart
 - Common mistake:
 $\forall x (\text{At}(x, \text{Guelph}) \wedge \text{Smart}(x))$ --- Everyone is at Guelph and everyone is smart
- Typically, \wedge is the main connective with \exists
 $\exists x (\text{At}(x, \text{Guelph}) \wedge \text{Smart}(x))$ --- Someone at Guelph is smart
 - Common mistake:
 $\exists x (\text{At}(x, \text{Guelph}) \Rightarrow \text{Smart}(x))$ --- Anyone not at Guelph is also smart

Properties of Quantifiers

- $\forall x \forall y$ is the same as $\forall y \forall x$
- $\exists x \exists y$ is the same as $\exists y \exists x$
- $\exists x \forall y$ is not the same as $\forall y \exists x$
 $\exists x \forall y \text{ Loves}(x,y)$
 - "There is a person who loves everyone in the world"
- $\forall y \exists x \text{ Loves}(x,y)$
 - "Everyone in the world is loved by at least one person"



Interpretations

- An interpretation of a sentence consists of a non-empty domain D and the following assignments:
 - To each constant, we assign an element in D
 - To each n-place function, we assign a mapping from D^n to D
 - To each n-place predicate, we assign a mapping from D^n to {TRUE, FALSE}.



Example

- Given sentence: $\forall x (P(x) \Rightarrow Q(f(x), a))$, we can have an interpretation as follows:

$D = \{1, 2\}$ $a = 1$ $f(1) = 2$ and $f(2) = 1$

P(1)	P(2)	Q(1,1)	Q(1,2)	Q(2,1)	Q(2,2)
F	T	T	T	F	T

If $x = 1$, $(P(1) \Rightarrow Q(f(1), a)) = (P(1) \Rightarrow Q(2,1)) = (F \Rightarrow F) = T$.
 If $x = 2$, $(P(2) \Rightarrow Q(f(2), a)) = (P(2) \Rightarrow Q(1,1)) = (T \Rightarrow T) = T$.

Thus, the sentence is true in the given interpretation.



Equality

- $term_1 = term_2$ is true under an interpretation if and only if $term_1$ and $term_2$ refer to the same object
- E.g., defining *Sibling* in terms of *Parent*:

$$\forall x,y (Sibling(x,y) \Leftrightarrow \neg(x = y) \wedge \exists m,f (\neg(m = f) \wedge Parent(m,x) \wedge Parent(f,x) \wedge Parent(m,y) \wedge Parent(f,y)))$$



Theoretical Results for FOL

- There exists no decision procedure to check the validity or unsatisfiability of sentences in FOL
- However, there are procedures which can verify valid sentences. For invalid sentences, these procedures may never terminate.



Proof Procedure

- Convert sentences into Prenex Normal Form
- Eliminate all quantifiers
- Convert the result into Conjunctive Normal Form and break the result into a set of clauses
- Apply unification-based resolution



Prenex Normal Form (PNF)

- A sentence is in a Prenex Normal Form iff it has the following format:

$$Q_1 x_1, Q_2 x_2, \dots, Q_n x_n M$$

where Q_i is either \forall or \exists , and M is a sentence without any quantifiers.

- The set of quantifiers is called the prefix and M is called the matrix.

Transformation to PNF

- Step 1: eliminate connectives (\Rightarrow) and (\Leftrightarrow):

$$\begin{aligned}\alpha \Leftrightarrow \beta &= (\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha) \\ \alpha \Rightarrow \beta &= \neg \alpha \vee \beta\end{aligned}$$

- Step 2: bring negations to the front of predicates:

$$\begin{aligned}\neg(\neg \alpha) &= \alpha \\ \neg(\alpha \vee \beta) &= \neg \alpha \wedge \neg \beta & \neg(\alpha \wedge \beta) &= \neg \alpha \vee \neg \beta \\ \neg(\forall x \alpha[x]) &= \exists x (\neg \alpha[x]) \\ \neg(\exists x \alpha[x]) &= \forall x (\neg \alpha[x])\end{aligned}$$

Transformation to PNF

- Step 3: rename bound variables if necessary
- Step 4: move quantifiers to the front:

$$\begin{aligned}Q x \alpha[x] \vee \beta &= Q x (\alpha[x] \vee \beta) \\ Q x \alpha[x] \wedge \beta &= Q x (\alpha[x] \wedge \beta)\end{aligned}$$

$$\begin{aligned}\forall x \alpha[x] \wedge \forall x \beta[x] &= \forall x (\alpha[x] \wedge \beta[x]) \\ \exists x \alpha[x] \vee \exists x \beta[x] &= \exists x (\alpha[x] \vee \beta[x])\end{aligned}$$

$$\begin{aligned}Q_1 x \alpha[x] \vee Q_2 x \beta[x] &= Q_1 x, Q_2 y (\alpha[x] \vee \beta[y]) \\ Q_3 x \alpha[x] \wedge Q_4 x \beta[x] &= Q_3 x, Q_4 y (\alpha[x] \wedge \beta[y])\end{aligned}$$

Transformation Example

$$\begin{aligned}\forall x, \forall y (\exists z (P(x,z) \wedge P(y,z)) \Rightarrow \exists u Q(x,y,u)) \\ &= \forall x, \forall y (\neg(\exists z (P(x,z) \wedge P(y,z))) \vee \exists u Q(x,y,u)) \\ &= \forall x, \forall y (\forall z (\neg P(x,z) \vee \neg P(y,z)) \vee \exists u Q(x,y,u)) \\ &= \forall x, \forall y, \forall z ((\neg P(x,z) \vee \neg P(y,z)) \vee \exists u Q(x,y,u)) \\ &= \forall x, \forall y, \forall z, \exists u ((\neg P(x,z) \vee \neg P(y,z)) \vee Q(x,y,u)) \\ &= \forall x, \forall y, \forall z, \exists u (\neg P(x,z) \vee \neg P(y,z) \vee Q(x,y,u))\end{aligned}$$

Eliminating Quantifiers

- Skolemization: given a sentence in its prenex normal form, eliminate an existential quantifier Q_r as follows:

- If no universal quantifier appears before Q_r , replace all x_r by a constant c and delete $Q_r x_r$ from the prefix.
- If Q_{s_1}, \dots, Q_{s_m} are universal quantifiers before Q_r , then replace all x_r by a m -place function $f(x_{s_1}, \dots, x_{s_m})$ and delete $Q_r x_r$ from the prefix.

Eliminating Quantifiers

- Once all existential quantifiers are eliminated, the remaining universal quantifiers can also be eliminated, since we can assume that all variables are universally quantified.

$$\begin{aligned}\exists x, \forall y, \forall z, \exists u, \forall v, \exists w P(x, y, z, u, v, w) \\ &= \forall y, \forall z, \forall v P(a, y, z, f(y,z), v, g(y,z,v)) \\ &= P(a, y, z, f(y,z), v, g(y,z,v))\end{aligned}$$

- Theoretical result: the unsatisfiability of a sentence is not affected by the skolemization process.

Substitution

- A substitution is a finite set $\{t_1/v_1, \dots, t_n/v_n\}$, where v_i is a variable; t_i is a term; and no two variables are the same.

- Examples:

- All variables are replaced in parallel, not sequential.

$\{f(z)/x, y/z\}$

$\{a/x, g(y)/y, f(g(b))/z\}$

Unification

- A substitution is called a unifier for a set of sentences $\{E_1, \dots, E_k\}$ iff E_1, \dots, E_k become the same after the substitution.

- The set $\{E_1, \dots, E_k\}$ is said to be unifiable if there is a unifier for it.

- Example: $\{P(a,y), P(x,f(b))\}$ is unifiable since the substitution $\{a/x, f(b)/y\}$ is a unifier.

Unification Algorithm

function UNIFY(x, y, θ) returns a substitution to make x and y identical

inputs: x , a variable, constant, list, or compound
 y , a variable, constant, list, or compound
 θ , the substitution built up so far

if $\theta = \text{failure}$ then return failure

else if $x = y$ then return θ

else if VARIABLE?(x) then return UNIFY-VAR(x, y, θ)

else if VARIABLE?(y) then return UNIFY-VAR(y, x, θ)

else if COMPOUND?(x) and COMPOUND?(y) then

return UNIFY(ARGs[x], ARGs[y], UNIFY(OP[x], OP[y], θ))

else if LIST?(x) and LIST?(y) then

return UNIFY(REST[x], REST[y], UNIFY(FIRST[x], FIRST[y], θ))

else return failure

Resolution Example

- Problem:

- Some patients like all doctors.
- No patient likes any quack.
- Therefore, no doctor is a quack.

- Sentences:

(1) $\exists x (P(x) \wedge \forall y (D(x) \Rightarrow L(x,y)))$

(2) $\forall x (P(x) \Rightarrow \forall y (Q(y) \Rightarrow \neg L(x,y)))$

(3) $\forall x (D(x) \Rightarrow \neg Q(x))$

Resolution Example

- Transforming sentences into clauses:

(4) $P(a)$ --- from (1) with a as a skolem constant

(5) $\neg D(y_2) \vee L(a,y_2)$ --- from (1)

(6) $\neg P(x_3) \vee \neg Q(y_3) \vee \neg L(x_3,y_3)$ --- from (2)

(7) $D(b)$ --- from negation of (3)

(8) $Q(b)$ --- from negation of (3)

Resolution Example

- Resolution process:

(9) $r[7,5a](y_5=b) L(a, b)$

(10) $r[8a,1](x_6=a) \neg Q(y_{10}) \vee \neg L(a,y_{10})$

(11) $r[8,10a](y_{10}=b) \neg L(a,b)$

(12) $r[9,11] \{ \}$

Factoring Clauses

- Given these two clauses:
 - $P(x) \vee P(y)$
 - $\neg P(u) \vee \neg P(w)$
- Obviously, we have a contradiction (just unifying all the variables to x). Using resolution, however, we get:
 - $r[1a,2a] (u=x) P(y) \vee \neg P(w)$
 - $r[2b,3a] (y=w) \neg P(u) \vee \neg P(w_1)$
 - $r[3a,4a] (y=u) \neg P(w) \vee P(w_1)$



Factoring Clauses

- If two or more predicates of a clause can be unified, then the resulting clause with all duplicates removed is called a factor of the clause:
 - $f[1ab] (y=x) P(x)$
 - $f[2ab] (w=u) \neg P(u)$
 - $r[6,7] (u=x) \{ \}$
- Resolution with factoring is both sound and complete.



Extracting Answers

- Given the following clauses:
 - $Father(Art, Jon)$
 - $Father(Bob, Kim)$
 - $\neg Father(y, z) \vee Parent(y, z)$
- Add an additional predicate to the negation of the question: Who is Jon's parent?
 - $\neg Parent(x, Jon) \vee Ans(x)$



Extracting Answers

- Applying resolution, we get:
 - $r[4,3b] (y=x, z=Jon) \neg Father(x, Jon) \vee Ans(x)$
 - $r[5a,1] (x=Art) Ans(Art)$
- Replacing (1) by (7) below, we get:
 - $Father(Art, Jon) \vee Father(Bob, Jon)$
 - $r[5a,7a] (x=Art) Father(Bob, Jon) \vee Ans(Art)$
 - $r[5a,8a] (x=Bob) Ans(Art) \vee Ans(Bob)$



Monkey-Banana Problem

- Problem:** A monkey wants to eat a banana that is hung from the ceiling of a room. The monkey is too short to reach the banana. However, the monkey can walk around the room, carry a chair in the room, and climb the chair to reach the banana.
- Predicates:**
 - $P(x, y, z, s)$ denotes a state s in which the monkey is at x , the banana at y , and the chair at z .
 - $R(s)$ denotes a state s in which the monkey can reach the banana.



Monkey-Banana Problem

- Functions:**
 - $Walk(y, z, s)$: the monkey walks from y to z in state s
 - $Carry(y, z, s)$: the monkey carries the chair from y to z in state s
 - $Climb(s)$: the monkey climbs the chair in state s .
- Initial state s_1 :** the monkey is at a , the banana is at b , and the chair is at c .



Monkey-Banana Problem

- Rules for actions and the initial state:

- (1) $\neg P(x, y, z, s) \vee P(z, y, z, \text{walk}(x, z, s))$
- (2) $\neg P(x, y, x, s) \vee P(y, y, y, \text{carry}(x, y, s))$
- (3) $\neg P(b, b, b, s) \vee R(\text{climb}(s))$
- (4) $P(a, b, c, s_1)$

- The clause for the question:

- (5) $\neg R(s) \vee \text{ANS}(s)$



Monkey-Banana Problem

- Resolution process:

$$(6) \neg P(b, b, b, s) \vee \text{ANS}(\text{climb}(s)) \quad (5)+(3)$$

$$(7) \neg P(x, b, x, s) \vee \text{ANS}(\text{climb}(\text{carry}(x, b, s))) \quad (6)+(2)$$

$$(8) \neg P(x, b, z, s) \vee \text{ANS}(\text{climb}(\text{carry}(z, b, \text{walk}(x, z, s)))) \quad (7)+(1)$$

$$(9) \text{ANS}(\text{climb}(\text{carry}(c, b, \text{walk}(a, c, s_1)))) \quad (8)+(4)$$



Wumpus KB

- Representing Percepts:

- $$\forall b, g, m, c, t \text{ Percept}([S, b, g, m, c], t) \Rightarrow \text{Stench}(t)$$
- $$\forall s, g, m, c, t \text{ Percept}([s, B, g, m, c], t) \Rightarrow \text{Breeze}(t)$$
- $$\forall s, b, m, c, t \text{ Percept}([s, b, G, m, c], t) \Rightarrow \text{Glitter}(t)$$
- ...

- Connecting percepts and actions in reflex agents:

$$\forall t \text{ Glitter}(t) \Rightarrow \text{BestAction}(\text{Grab}, t)$$

...



Wumpus KB

- Linking neighboring squares:

$$\forall x, y, a, b \text{ Adjacent}([x, y], [a, b]) \Leftrightarrow [a, b] \in \{[x+1, y], [x-1, y], [x, y+1], [x, y-1]\}$$

- Properties of squares:

$$\forall s, t \text{ At}(\text{Agent}, s, t) \wedge \text{Breeze}(t) \Rightarrow \text{Breezy}(s)$$

- Squares are breezy near a pit:

- Diagnostic rule --- infer cause from effect

$$\forall s \text{ Breezy}(s) \Rightarrow \exists r \text{ Adjacent}(r, s) \wedge \text{Pit}(r)$$

- Causal rule --- infer effect from cause

$$\forall r \text{ Pit}(r) \Rightarrow (\forall s \text{ Adjacent}(r, s) \Rightarrow \text{Breezy}(s))$$



Limitations with FOL

- Complete: all the sentences necessary to solve a problem must be available or can be derived by inferences.

- Consistent: there are no contradictory sentences in KB.

- Monotonic: new sentences can be added, but they are always consistent with the existing sentences. Nothing will ever be retracted from the set of sentences that are known.



Limitations with FOL

- The previous requirements are too strong for commonsense reasoning:

Birds fly. Tweety is a bird. Tweety flies.

Emu are birds, but emu don't fly.

Tweety is an emu. Tweety doesn't fly.

$$\forall x (\text{Bird}(x) \wedge \neg \text{Emu}(x) \wedge \neg \text{Penguin}(x) \wedge \dots \Rightarrow \text{Flies}(x))$$

- Our knowledge is often incomplete, inconsistent, and nonmonotonic.

