

# CmpE 540

## Principles of Artificial Intelligence

Pınar Yolum  
[pinar.yolum@boun.edu.tr](mailto:pinar.yolum@boun.edu.tr)

Department of  
Computer Engineering  
Boğaziçi University

# First-Order Logic

Chapter 8

(Based mostly on the course slides  
from <http://aima.cs.berkeley.edu/> and  
<http://www.cmpe.boun.edu.tr/~akin/>)

## Outline

- Why FOL?
- Syntax and semantics of FOL
- Using FOL
- Wumpus world in FOL
- Knowledge engineering in FOL

## Pros and cons of propositional logic

- ☺ Propositional logic is **declarative**
- ☺ Propositional logic allows partial/disjunctive/negated information
  - (unlike most data structures and databases)
- ☺ Propositional logic is **compositional**:
  - meaning of  $B_{1,1} \wedge P_{1,2}$  is derived from meaning of  $B_{1,1}$  and of  $P_{1,2}$
- ☺ Meaning in propositional logic is **context-independent**
  - (unlike natural language, where meaning depends on context)
- ☺ Propositional logic has very limited expressive power
  - (unlike natural language)
  - E.g., cannot say "pits cause breezes in adjacent squares"
    - except by writing one sentence for each square

## First-order logic

- Whereas propositional logic assumes the world contains **facts**,
- first-order logic (like natural language) assumes 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 of FOL: Basic elements

- Constants KingJohn, 2, NUS,...
- Predicates Brother, >,...
- Functions Sqrt, LeftLegOf,...
- Variables x, y, a, b,...
- Connectives  $\neg, \Rightarrow, \wedge, \vee, \Leftrightarrow$
- Equality =
- Quantifiers  $\forall, \exists$

## Atomic sentences

Atomic sentence = *predicate* ( $term_1, \dots, term_n$ )  
or  $term_1 = term_2$

Term = *function* ( $term_1, \dots, term_n$ )  
or *constant* or *variable*

- E.g., *Brother*(KingJohn, RichardTheLionheart)
- *>* (*Length*(*LeftLegOf*(Richard)), *Length*(*LeftLegOf*(KingJohn)))

## Complex sentences

- Complex sentences are made from atomic sentences using connectives

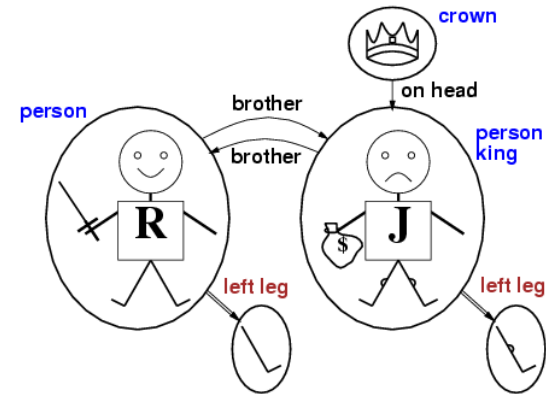
$\neg S, S_1 \wedge S_2, S_1 \vee S_2, S_1 \Rightarrow S_2, S_1 \Leftrightarrow S_2,$

E.g. *Sibling*(KingJohn, Richard)  $\Rightarrow$   
*Sibling*(Richard, KingJohn)

## Truth in first-order logic

- Sentences are true with respect to a **model** and an **interpretation**
- Model contains objects (**domain elements**) and relations among them
- Interpretation specifies referents for
  - constant symbols** → **objects**
  - predicate symbols** → **relations**
  - function symbols** → **functional relations**
- An atomic sentence  $predicate(term_1, \dots, term_n)$  is true iff the **objects** referred to by  $term_1, \dots, term_n$  are in the **relation** referred to by  $predicate$

## Models for FOL: Example



## Universal quantification

- $\forall \langle \text{variables} \rangle \langle \text{sentence} \rangle$

All kings are persons:

$$\forall x \text{ King}(x) \Rightarrow \text{Person}(x)$$

- $\forall x P$  is true in a model  $m$  iff  $P$  is true with  $x$  being each possible object in the model

- Roughly speaking, equivalent to the **conjunction of instantiations** of  $P$

$$\begin{aligned} & \text{King}(\text{Richard}) \Rightarrow \text{Person}(\text{Richard}) \\ \wedge & \text{King}(\text{John}) \Rightarrow \text{Person}(\text{John}) \\ \wedge & \text{King}(\text{John's-left-leg}) \Rightarrow \text{Person}(\text{John's-left-leg}) \\ \wedge & \dots \end{aligned}$$

## A common mistake to avoid

- Typically,  $\Rightarrow$  is the main connective with  $\forall$
- Common mistake: using  $\wedge$  as the main connective with  $\forall$ :

$$\forall x \text{ King}(x) \wedge \text{Person}(x)$$

means "Everyone is a King and everyone is a person"

## Existential quantification

- $\exists \langle \text{variables} \rangle \langle \text{sentence} \rangle$
- John has a crown on his head:
- $\exists x \text{Crown}(x) \wedge \text{OnHead}(x, \text{John})$
- $\exists x P$  is true in a model  $m$  iff  $P$  is true with  $x$  being some possible object in the model
- Roughly speaking, equivalent to the **disjunction** of **instantiations** of  $P$ 
  - $\text{Crown}(\text{Richard}) \wedge \text{OnHead}(\text{Richard}, \text{John})$
  - $\vee \text{Crown}(\text{Richard's-left-leg}) \wedge \text{OnHead}(\text{Richard's-left-leg}, \text{John})$
  - $\vee \text{Crown}(\text{Crown1}) \wedge \text{OnHead}(\text{Crown1}, \text{John})$
  - $\vee \dots$

## Another common mistake to avoid

- Typically,  $\wedge$  is the main connective with  $\exists$
- Common mistake: using  $\Rightarrow$  as the main connective with  $\exists$ :  
 $\exists x \text{Crown}(x) \Rightarrow \text{OnHead}(x, \text{John})$   
is true if there is anything that is not a crown!

## 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”
- **Quantifier duality**: each can be expressed using the other
- $\forall x \text{Likes}(x, \text{IceCream}) \quad \neg \exists x \neg \text{Likes}(x, \text{IceCream})$
- $\exists x \text{Likes}(x, \text{Broccoli}) \quad \neg \forall x \neg \text{Likes}(x, \text{Broccoli})$

## Equality

- $\text{term}_1 = \text{term}_2$  is true under a given interpretation if and only if  $\text{term}_1$  and  $\text{term}_2$  refer to the same object
- E.g., definition of *Sibling* in terms of *Parent*:  
 $\forall x,y \text{Sibling}(x,y) \Leftrightarrow [\neg(x = y) \wedge \exists m,f \neg (m = f) \wedge \text{Parent}(m,x) \wedge \text{Parent}(f,x) \wedge \text{Parent}(m,y) \wedge \text{Parent}(f,y)]$

## Using FOL

The kinship domain:

- Brothers are siblings  
 $\forall x,y \text{ Brother}(x,y) \Leftrightarrow \text{Sibling}(x,y)$
- One's mother is one's female parent  
 $\forall m,c \text{ Mother}(c) = m \Leftrightarrow (\text{Female}(m) \wedge \text{Parent}(m,c))$
- "Sibling" is symmetric  
 $\forall x,y \text{ Sibling}(x,y) \Leftrightarrow \text{Sibling}(y,x)$

## Using FOL

The set domain:

- $\forall s \text{ Set}(s) \Leftrightarrow (s = \{\}) \vee (\exists x,s_2 \text{ Set}(s_2) \wedge s = \{x|s_2\})$
- $\neg \exists x,s \{x|s\} = \{\}$
- $\forall x,s \ x \in s \Leftrightarrow s = \{x|s\}$
- $\forall x,s \ x \in s \Leftrightarrow [\exists y,s_2 (s = \{y|s_2\} \wedge (x = y \vee x \in s_2))]$
- $\forall s_1,s_2 \ s_1 \subseteq s_2 \Leftrightarrow (\forall x \ x \in s_1 \Rightarrow x \in s_2)$
- $\forall s_1,s_2 \ (s_1 = s_2) \Leftrightarrow (s_1 \subseteq s_2 \wedge s_2 \subseteq s_1)$
- $\forall x,s_1,s_2 \ x \in (s_1 \cap s_2) \Leftrightarrow (x \in s_1 \wedge x \in s_2)$
- $\forall x,s_1,s_2 \ x \in (s_1 \cup s_2) \Leftrightarrow (x \in s_1 \vee x \in s_2)$

## Interacting with FOL KBs

- Suppose a wumpus-world agent is using an FOL KB and perceives a smell and a breeze (but no glitter) at  $t=5$ :

```
Tell(KB,Percept([Smell,Breeze,None],5))
Ask(KB,∃a BestAction(a,5))
```

- I.e., does the KB entail some best action at  $t=5$ ?
- Answer: Yes,  $\{a/Shoot\}$  ← substitution (binding list)
- Given a sentence  $S$  and a substitution  $\sigma$ ,
- $S\sigma$  denotes the result of plugging  $\sigma$  into  $S$ ; e.g.,  
 $S = \text{Smarter}(x,y)$   
 $\sigma = \{x/Hillary,y/Bill\}$   
 $S\sigma = \text{Smarter}(Hillary,Bill)$
- $\text{Ask}(KB,S)$  returns some/all  $\sigma$  such that  $KB \models \sigma$

## Deducing hidden properties

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

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:

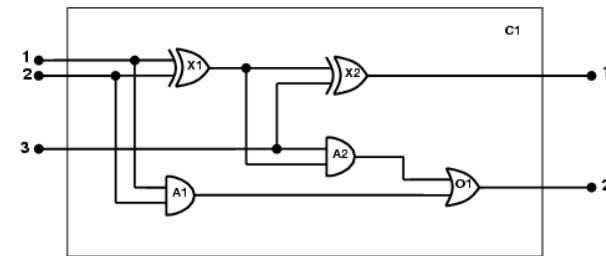
- Diagnostic rule---infer cause from effect  
 $\forall s \text{ Breezy}(s) \Rightarrow \exists r \text{ Adjacent}(r,s) \wedge \text{Pit}(r)$
- Causal rule---infer effect from cause  
 $\forall r \text{ Pit}(r) \Rightarrow [\forall s \text{ Adjacent}(r,s) \Rightarrow \text{Breezy}(s)]$

## Knowledge engineering in FOL

1. Identify the task
2. Assemble the relevant knowledge
3. Decide on a vocabulary of predicates, functions, and constants
4. Encode general knowledge about the domain
5. Encode a description of the specific problem instance
6. Pose queries to the inference procedure and get answers
7. Debug the knowledge base

## The electronic circuits domain

### One-bit full adder



## The electronic circuits domain

1. Identify the task
  - Does the circuit actually add properly? (circuit verification)
2. Assemble the relevant knowledge
  - Composed of wires and gates; Types of gates (AND, OR, XOR, NOT)
  - Irrelevant: size, shape, color, cost of gates
3. Decide on a vocabulary
  - Alternatives:
    - Type( $X_1$ ) = XOR
    - Type( $X_1$ , XOR)
    - XOR( $X_1$ )

## The electronic circuits domain

4. Encode general knowledge of the domain
  - $\forall t_1, t_2 \text{ Connected}(t_1, t_2) \Rightarrow \text{Signal}(t_1) = \text{Signal}(t_2)$
  - $\forall t \text{ Signal}(t) = 1 \vee \text{Signal}(t) = 0$
  - $1 \neq 0$
  - $\forall t_1, t_2 \text{ Connected}(t_1, t_2) \Rightarrow \text{Connected}(t_2, t_1)$
  - $\forall g \text{ Type}(g) = \text{OR} \Rightarrow \text{Signal}(\text{Out}(1, g)) = 1 \Leftrightarrow \exists n \text{ Signal}(\text{In}(n, g)) = 1$
  - $\forall g \text{ Type}(g) = \text{AND} \Rightarrow \text{Signal}(\text{Out}(1, g)) = 0 \Leftrightarrow \exists n \text{ Signal}(\text{In}(n, g)) = 0$
  - $\forall g \text{ Type}(g) = \text{XOR} \Rightarrow \text{Signal}(\text{Out}(1, g)) = 1 \Leftrightarrow \text{Signal}(\text{In}(1, g)) \neq \text{Signal}(\text{In}(2, g))$
  - $\forall g \text{ Type}(g) = \text{NOT} \Rightarrow \text{Signal}(\text{Out}(1, g)) \neq \text{Signal}(\text{In}(1, g))$

## The electronic circuits domain

### 5. Encode the specific problem instance $\square$

Type( $X_1$ ) = XOR            Type( $X_2$ ) = XOR  
Type( $A_1$ ) = AND            Type( $A_2$ ) = AND  
Type( $O_1$ ) = OR

Connected(Out(1, $X_1$ ),In(1, $X_2$ ))      Connected(In(1, $C_1$ ),In(1, $X_1$ ))  
Connected(Out(1, $X_1$ ),In(2, $A_2$ ))      Connected(In(1, $C_1$ ),In(1, $A_1$ ))  
Connected(Out(1, $A_2$ ),In(1, $O_1$ ))      Connected(In(2, $C_1$ ),In(2, $X_1$ ))  
Connected(Out(1, $A_1$ ),In(2, $O_1$ ))      Connected(In(2, $C_1$ ),In(2, $A_1$ ))  
Connected(Out(1, $X_2$ ),Out(1, $C_1$ ))      Connected(In(3, $C_1$ ),In(2, $X_2$ ))  
Connected(Out(1, $O_1$ ),Out(2, $C_1$ ))      Connected(In(3, $C_1$ ),In(1, $A_2$ ))

$\square$

## The electronic circuits domain

### 6. Pose queries to the inference procedure

What are the possible sets of values of all the terminals for the adder circuit?

$\exists i_1, i_2, i_3, o_1, o_2$  Signal(In(1, $C_1$ )) =  $i_1 \wedge$  Signal(In(2, $C_1$ )) =  $i_2 \wedge$  Signal(In(3, $C_1$ )) =  $i_3 \wedge$  Signal(Out(1, $C_1$ )) =  $o_1 \wedge$  Signal(Out(2, $C_1$ )) =  $o_2$

### 7. Debug the knowledge base

May have omitted assertions like  $1 \neq 0$

## Summary

- First-order logic:
  - objects and relations are semantic primitives
  - syntax: constants, functions, predicates, equality, quantifiers
- Increased expressive power: sufficient to define wumpus world