Chapter 8: First Order Logic

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)

First-order logic

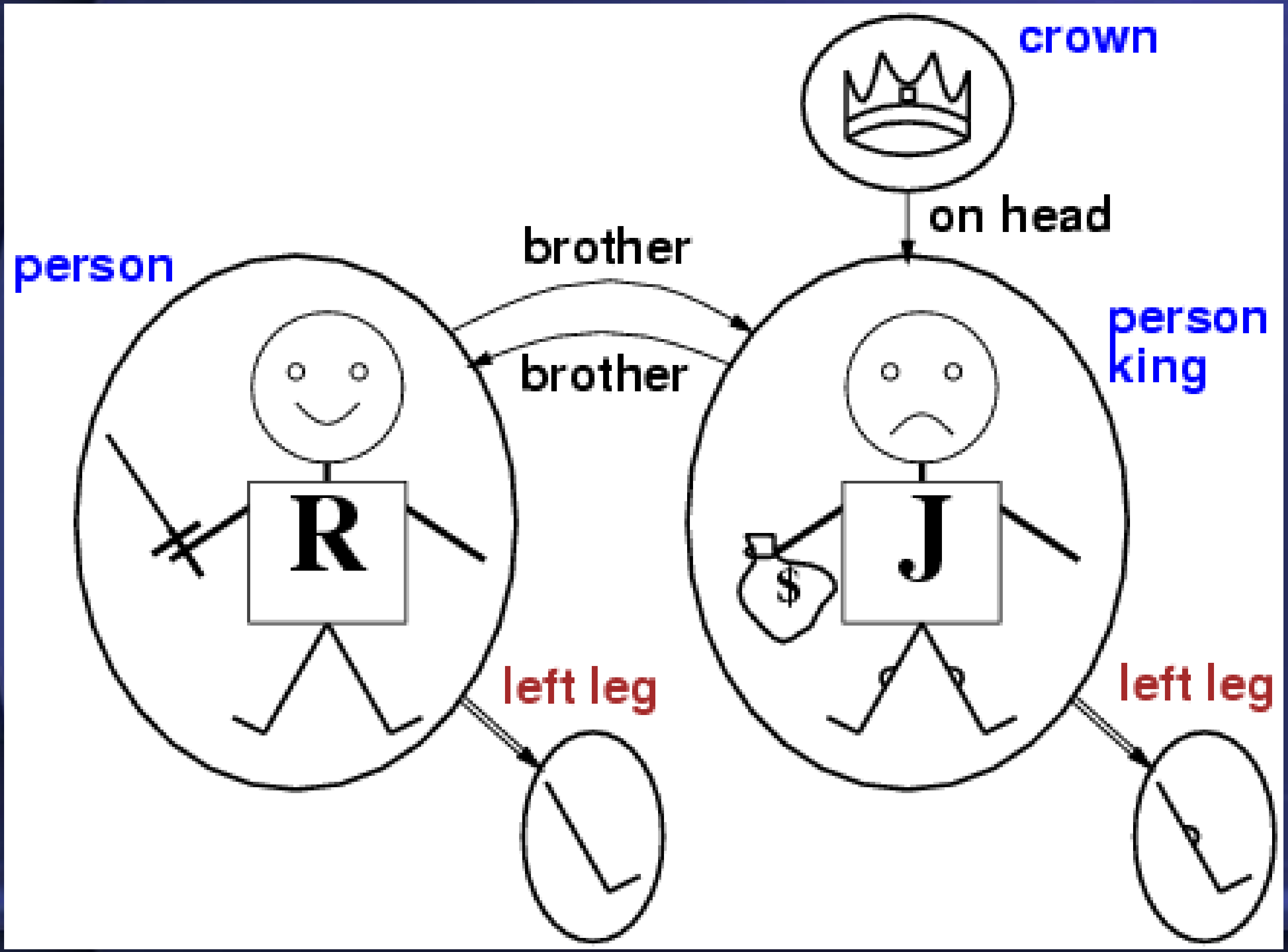
- Whereas propositional logic assumes the world contains facts,

- 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, …

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 $\forall, \exists$

Models for FOL: Example



Universal quantification

- $\forall$<*variables*> <*sentence*>

Everyone at NUS is smart:

$\forall x\ At(x,NUS) \Rightarrow Smart(x)$

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

$$
\begin{aligned}
& At(KingJohn,NUS) \Rightarrow Smart(KingJohn) \\
\wedge\ & At(Richard,NUS) \Rightarrow Smart(Richard) \\
\wedge\ & At(NUS,NUS) \Rightarrow Smart(NUS)
\end{aligned}
$$

Existential quantification

- $\exists$<*variables*> <*sentence*>

- Someone at NUS is smart:
- $\exists x$ At(x,NUS) $\wedge$ Smart(x)

- Roughly speaking, equivalent to the disjunction of instantiations of *P*
-

  At(KingJohn,NUS) $\wedge$ Smart(KingJohn)
  $\vee$ At(Richard,NUS) $\wedge$ Smart(Richard)
  $\vee$ At(NUS,NUS) $\wedge$ Smart(NUS)
  $\vee$ ...

Properties of quantifiers

- ∀x ∀y is the same as ∀y ∀x
-
- ∃x ∃y is the same as ∃y ∃x
-
- ∃x ∀y is not the same as ∀y ∃x
-
- ∃x ∀y Loves(x,y)
  - "There is a person who loves everyone in the world"
  -
- ∀y ∃x Loves(x,y)
  - "Everyone in the world is loved by at least one person"
  -
- Quantifier duality: each can be expressed using the other

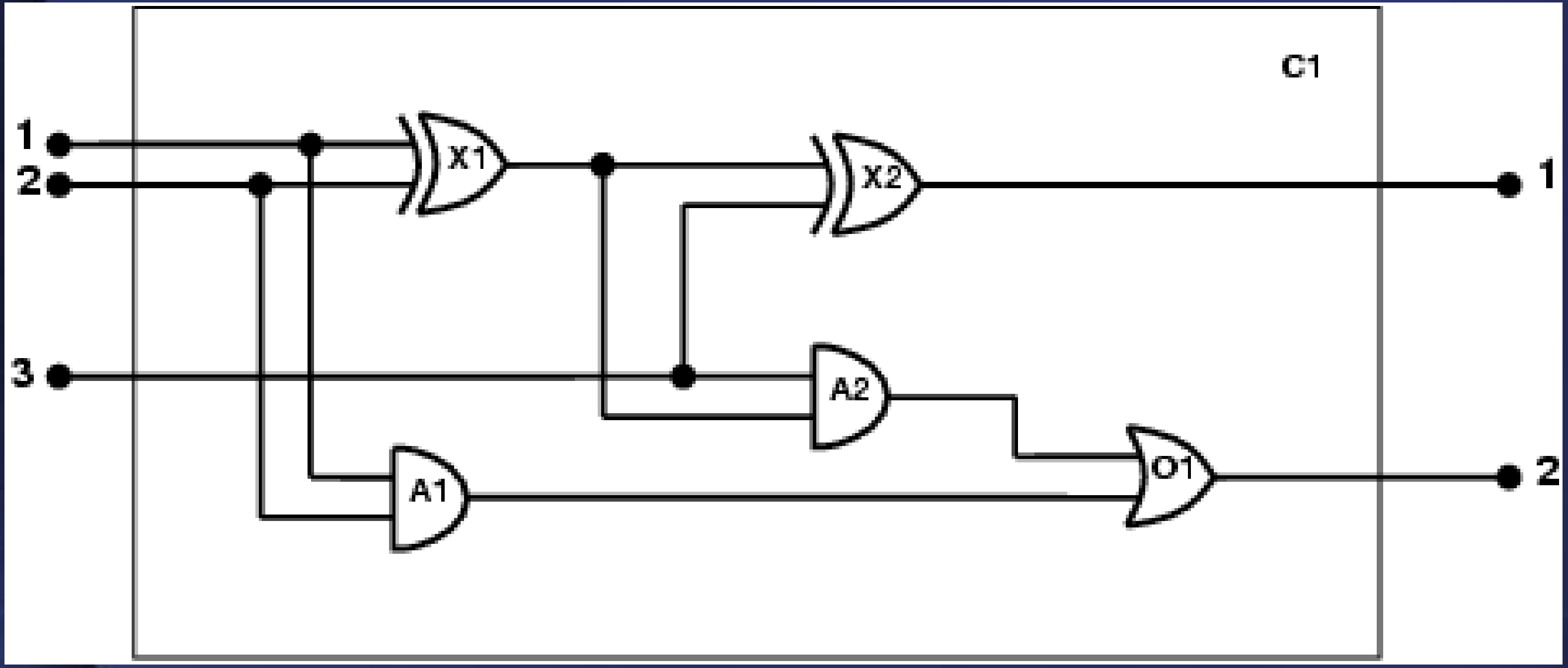
Using FOL

The kinship domain:

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

The electronic circuits domain

One-bit full adder



The electronic circuits domain

1. Identify the task
2.
   - Does the circuit actually add properly? (circuit verification)
   -
2. Assemble the relevant knowledge
3.
   - Composed of wires and gates; Types of gates (AND, OR, XOR, NOT)
   -
   - Irrelevant: size, shape, color, cost of gates
   -
3. Decide on a vocabulary
4.

The electronic circuits domain

4. Encode general knowledge of the domain
5.

– $\forall t_1,t_2$ Connected(t_1, t_2) $\Rightarrow$ Signal(t_1) = Signal(t_2)
– $\forall t$ Signal(t) = 1 $\vee$ Signal(t) = 0
–
– $1 \neq 0$
–
– $\forall t_1,t_2$ Connected(t_1, t_2) $\Rightarrow$ Connected(t_2, t_1)
–
– $\forall g$ Type(g) = OR $\Rightarrow$ Signal(Out(1,g)) = 1 $\Leftrightarrow$ $\exists n$ Signal(In(n,g)) = 1

The electronic circuits domain

4. Encode general knowledge of the domain
5.
   - …
   -
   - $\forall g\ Type(g) = AND \Rightarrow Signal(Out(1,g)) = 0 \Leftrightarrow \exists n\ Signal(In(n,g)) = 0$
   -
   - $\forall g\ Type(g) = XOR \Rightarrow Signal(Out(1,g)) = 1 \Leftrightarrow Signal(In(1,g)) \neq Signal(In(2,g))$
   -
   - $\forall g\ Type(g) = NOT \Rightarrow Signal(Out(1,g)) \neq Signal(In(1,g))$
   -

The electronic circuits domain

5. Encode the specific problem instance
6.

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

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

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

The electronic circuits domain

6. Pose queries to the inference procedure
7.

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

Summary

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