First-Order Logic

Chapter 8
Outline

• Why FOL?
• Syntax and semantics of FOL
• Using FOL
• Wumpus world in FOL
• Knowledge engineering in FOL
Pros/cons of propositional logic

😊 Propositional logic is **declarative** (recall TELL/ASK)

😊 Propositional logic allows partial/disjunctive/negated information (unlike most data structures and databases)

😊 Propositional logic is **compositional**:
  – meaning of $B_{1,1} \land P_{1,2}$ 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    UGA, GaTech, 2,...
- Predicates   Brother, Greater, President...
- Functions    Sqrt, NumStudents, Length...
- Variables    x, y, a, b,...
- Connectives  ¬, ⇒, ∧, ∨, ⇔
- Equality     =
- Quantifiers  ∀, ∃
Atomic sentences

Atomic sentence = $\textit{predicate} \ (\textit{term}_1,\ldots,\textit{term}_n)$

or $\textit{term}_1 = \textit{term}_2$

Term = $\textit{function} \ (\textit{term}_1,\ldots,\textit{term}_n)$

or constant or variable

• E.g., $\textit{President}(\text{UGA, Adams})$,
  $\textit{Greater}(\text{NumStudents}(\text{UGA}), \text{NumStudents}(\text{GaTech}))$
Complex sentences

- Complex sentences are made from atomic sentences using connectives
  \[ \neg S, S_1 \land S_2, S_1 \lor S_2, S_1 \Rightarrow S_2, S_1 \Leftrightarrow S_2, \]

E.g. \( \text{Sibling}(\text{KingJohn}, \text{Richard}) \Rightarrow \text{Sibling}(\text{Richard}, \text{KingJohn}) \)

\[ \text{Greater}(2,1) \lor \leq (1,2) \]
\[ \text{Greater}(2,1) \land \neg \text{Greater}(2,1) \]
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 $\rightarrow$ objects
  - Predicate symbols $\rightarrow$ relations
  - Function symbols $\rightarrow$ functional relations

- An atomic sentence $\text{predicate}(\text{term}_1,\ldots,\text{term}_n)$ is true iff the objects referred to by $\text{term}_1,\ldots,\text{term}_n$ are in the relation referred to by $\text{predicate}$ e.g., $\text{USG}(\text{UGA, GaTech, UWG, VSU, MCG})$
Models for FOL: Example
Universal quantification

• $\forall <variables> <sentence>$

Everyone at UGA is smart (note UoD: $x \in$ Students):
$\forall x \text{At}(x, \text{UGA}) \Rightarrow \text{Smart}(x)$

• $\forall x \, P$ 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 $P$
  \[ \text{At}(\text{John, UGA}) \Rightarrow \text{Smart}(\text{John}) \]
  \[ \land \text{At}(\text{Richard, UGA}) \Rightarrow \text{Smart}(\text{Richard}) \]
  \[ \land \text{At}(\text{Jim, UGA}) \Rightarrow \text{Smart}(\text{Jim}) \]
  \[ \land \ldots \]
A common mistake to avoid

• Typically, \( \Rightarrow \) is the main connective with \( \forall \)

• Common mistake: using \( \land \) as the main connective with \( \forall \):

\[
\forall x \ At(x, UGA) \land Smart(x)
\]

means “Everyone is at UGA and everyone is smart”
Existential quantification

• $\exists \langle variables \rangle \ <sentence>$

• Someone at GaTech is smart:
  • $\exists x \; \text{At}(x, \text{GaTech}) \land \text{Smart}(x)$

• $\exists x \; P$ 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 $P$
  • $\\text{At}(\text{John}, \text{GaTech}) \land \text{Smart}(\text{John})$
  $\lor \text{At}(\text{Richard}, \text{GaTech}) \land \text{Smart}(\text{Richard})$
  $\lor \text{At}(\text{Jim}, \text{GaTech}) \land \text{Smart}(\text{Jim})$
  $\lor \ldots$
Another common mistake to avoid

- Typically, $\land$ is the main connective with $\exists$

- Common mistake: using $\Rightarrow$ as the main connective with $\exists$:

  $$\exists x \text{ At}(x, \text{UGA}) \Rightarrow \text{Smart}(x)$$

  is true if there is anyone who is not at UGA! (recall implication is true if premise is false)
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
- $\forall x \text{ Likes}(x,\text{IceCream})$ $\rightarrow \exists x \neg \text{ Likes}(x,\text{IceCream})$
- $\exists x \text{ Likes}(x,\text{Broccoll})$ $\rightarrow \forall x \neg \text{ Likes}(x,\text{Broccoll})$
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 $\text{Sibling}$ in terms of $\text{Parent}$:

$$\forall x,y \ Sibling(x,y) \iff \neg (x = y) \land \exists m,f \ (m = f) \land \text{Parent}(m,x) \land \text{Parent}(f,x) \land \text{Parent}(m,y) \land \text{Parent}(f,y)$$
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
  \[ \forall m, c \ Mother(c) = m \Leftrightarrow (Female(m) \land Parent(m, c)) \]

• “Sibling” is symmetric (brother isn’t)
  \[ \forall x, y \ Sibling(x, y) \Leftrightarrow Sibling(y, x) \]
Using FOL

The set domain (x is an element or object, could be a set):

- $\forall s : \text{Set}(s) \iff (s = \{\}) \lor (\exists x, s_2 \text{ Set}(s_2) \land s = \{x|s_2\})$
- $\neg \exists x, s : \{x|s\} = \{\}$ (can’t decompose empty set)
- $\forall x, s : x \in s \iff s = \{x|s\}$ (no dups)
- $\forall x, s : x \in s \iff [\ \exists y, s_2 : (s = \{y|s_2\} \land (x = y \lor x \in s_2)))]$
- $\forall s_1, s_2 : s_1 \subseteq s_2 \iff (\forall x : x \in s_1 \Rightarrow x \in s_2)$
- $\forall s_1, s_2 : (s_1 = s_2) \iff (s_1 \subseteq s_2 \land s_2 \subseteq s_1)$
- $\forall x, s_1, s_2 : x \in (s_1 \cap s_2) \iff (x \in s_1 \land x \in s_2)$
- $\forall x, s_1, s_2 : x \in (s_1 \cup s_2) \iff (x \in s_1 \lor x \in s_2)$

- $\{x|s\}$ means x is adjoined to set s
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 $t=5$:
  
  $\text{Tell}(\text{KB,Percept([Smell,Breeze,None],5)})$
  $\text{Ask}(\text{KB,}\exists a \text{ BestAction}(a,5))$

• 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.,
  
  $S = \text{Smarter}(x,y)$
  $\sigma = \{x/\text{Hillary}, y/\text{Bill}\}$
  $S\sigma = \text{Smarter}(\text{Hillary},\text{Bill})$

• $\text{Ask}(\text{KB,S})$ returns some/all $\sigma$ such that $\text{KB} \models \sigma$
Knowledge base for the wumpus world

• **Perception**
  – $\forall t, s, b \ \text{Percept}([s, b, \text{Glitter}], t) \Rightarrow \text{Glitter}(t)$

• **Reflex**
  – $\forall t \ \text{Glitter}(t) \Rightarrow \text{BestAction(Grab, } t)$
Deducing hidden properties

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

Properties of squares:
- \( \forall s,t \ At(Agent,s,t) \land Breeze(t) \Rightarrow Breezy(s) \)

Squares are breezy near a pit:
- **Diagnostic** rule---infer cause from effect
  \( \forall s \ Breezy(s) \Rightarrow \exists r \ Adjacent(r,s) \land Pit(r) \)
- **Causal** rule---infer effect from cause
  \( \forall r \ Pit(r) \Rightarrow [\forall s \ Adjacent(r,s) \Rightarrow Breezy(s) ] \)
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 circuit 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
   - \( \forall t_1, t_2 \) Connected\( (t_1, t_2) \Rightarrow \text{Signal}(t_1) = \text{Signal}(t_2) \)
   - \( \forall t \) Signal\( (t) = 1 \lor \text{Signal}(t) = 0 \)
   - \( 1 \neq 0 \)
   - \( \forall t_1, t_2 \) Connected\( (t_1, t_2) \Rightarrow \text{Connected}(t_2, t_1) \)
   - \( \forall g \) Type\( (g) = \text{OR} \Rightarrow \text{Signal}(\text{Out}(1,g)) = 1 \iff \exists n \text{Signal}(\text{In}(n,g)) = 1 \)
   - \( \forall g \) Type\( (g) = \text{AND} \Rightarrow \text{Signal}(\text{Out}(1,g)) = 0 \iff \exists n \text{Signal}(\text{In}(n,g)) = 0 \)
   - \( \forall g \) Type\( (g) = \text{XOR} \Rightarrow \text{Signal}(\text{Out}(1,g)) = 1 \iff \text{Signal}(\text{In}(1,g)) \neq \text{Signal}(\text{In}(2,g)) \)
   - \( \forall g \) Type\( (g) = \text{NOT} \Rightarrow \text{Signal}(\text{Out}(1,g)) \neq \text{Signal}(\text{In}(1,g)) \)
   - Etc.
The electronic circuits domain

5. Encode the specific problem instance

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,$A_2$))
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?

\[ \exists i_1, i_2, i_3, o_1, o_2 \text{ Signal(In}(1,C_1)) = i_1 \land \text{Signal(In}(2,C_1)) = i_2 \land \text{Signal(In}(3,C_1)) = i_3 \land \text{Signal(Out}(1,C_1)) = o_1 \land \text{Signal(Out}(2,C_1)) = o_2 \]

7. Debug the knowledge base

May have omitted assertions like \( 1 \neq 0 \)
The electronic circuits domain

\[ \exists i_1, i_2, i_3, o_1, o_2 \text{ Signal(In}(1, C_1)) = i_1 \land \text{Signal(In}(2, C_1)) = i_2 \land \text{Signal(In}(3, C_1)) = i_3 \land \text{Signal(Out}(1, C_1)) = o_1 \land \text{Signal(Out}(2, C_1)) = o_2 \]

<table>
<thead>
<tr>
<th></th>
<th>i_1</th>
<th>i_2</th>
<th>i_3</th>
<th>O_1 (sum)</th>
<th>O_2 (carry)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Summary

• First-order logic:
  – objects and relations are semantic primitives
  – syntax: constants, functions, predicates, equality, quantifiers

• Increased expressive power: sufficient to define wumpus world and lots of other cool stuff too 😊