First-order logic

Chapter 7, AIMA2e Chapter 8

Outline

◊ Why FOL?
◊ Syntax and semantics of FOL
◊ Fun with sentences
◊ Wumpus world in FOL
Pros and cons of propositional logic

- Propositional logic is declarative: pieces of syntax correspond to facts
- 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 world contains facts, first-order logic (like natural language) assumes the world contains

- **Objects**: people, houses, numbers, theories, Ronald McDonald, colors, baseball games, wars, centuries . . .
- **Relations**: red, round, bogus, prime, multistoried . . ., brother of, bigger than, inside, part of, has color, occurred after, owns, comes between, . . .
- **Functions**: father of, best friend, third inning of, one more than, beginning of . . .
Logics in general

<table>
<thead>
<tr>
<th>Language</th>
<th>Ontological Commitment</th>
<th>Epistemological Commitment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Propositional logic</td>
<td>facts</td>
<td>true/false/unknown</td>
</tr>
<tr>
<td>First-order logic</td>
<td>facts, objects, relations</td>
<td>true/false/unknown</td>
</tr>
<tr>
<td>Temporal logic</td>
<td>facts, objects, relations, times</td>
<td>true/false/unknown</td>
</tr>
<tr>
<td>Probability theory</td>
<td>facts</td>
<td>degree of belief ∈ [0, 1]</td>
</tr>
<tr>
<td>Fuzzy logic</td>
<td>degree of truth ∈ [0, 1]</td>
<td>known interval value</td>
</tr>
</tbody>
</table>

Syntax of FOL: Basic elements

- **Constants**: KingJohn, 2, UCB, ...
- **Predicates**: Brother, >, ...
- **Functions**: Sqrt, LeftLegOf, ...
- **Variables**: x, y, a, b, ...
- **Connectives**: ∧, ∨, ¬, ⇒, ⇔
- **Equality**: =
- **Quantifiers**: ∀, ∃
Atomic sentences

Atomic sentence = \( \text{predicate}(\text{term}_1, \ldots, \text{term}_n) \)
or \( \text{term}_1 = \text{term}_2 \)

Term = \( \text{function}(\text{term}_1, \ldots, \text{term}_n) \)
or constant or variable

E.g., \( \text{Brother}(\text{KingJohn}, \text{RichardTheLionheart}) \)
\( > (\text{Length}(\text{LeftLegOf}(\text{Richard})), \)
\( \text{Length}(\text{LeftLegOf}(\text{KingJohn}))) \)

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. \( \text{Sibling}(\text{KingJohn}, \text{Richard}) \Rightarrow \text{Sibling}(\text{Richard}, \text{KingJohn}) \)
\( > (1, 2) \lor \leq (1, 2) \)
\( > (1, 2) \land \neg > (1, 2) \)
Truth in first-order logic

Sentences are true with respect to a model and an interpretation

Model contains $\geq 1$ 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 $\text{predicate}$

Models for FOL: Example

[Diagram showing relationships between R, J, crown, on head, brother, left leg, and person]
 Models for FOL: Lots!

We can enumerate the models for a given KB vocabulary:

For each number of domain elements $n$ from 1 to $\infty$
  For each $k$-ary predicate $P_k$ in the vocabulary
    For each possible $k$-ary relation on $n$ objects
      For each constant symbol $C$ in the vocabulary
        For each choice of referent for $C$ from $n$ objects . . .

Computing entailment by enumerating models is not going to be easy!

 Universal quantification

$\forall <variables> <sentence>$

Everyone at GMU is smart:
$\forall x \ At(x, GMU) \Rightarrow Smart(x)$
$\forall x \ P$ 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 $P$

$$At(KingJohn, GMU) \Rightarrow Smart(KingJohn)$$
$$\land At(Richard, GMU) \Rightarrow Smart(Richard)$$
$$\land At(Mason, GMU) \Rightarrow Smart(Mason)$$
$$\land \ldots$$
**A common mistake to avoid**

Typically, ⇒ is the main connective with ∀

Common mistake: using ∧ as the main connective with ∀:

\[ ∀x \ At(x, GMU) ∧ Smart(x) \]

means “Everyone is at GMU and everyone is smart”

---

**Existential quantification**

∃ <variables> <sentence>

Someone at Madison is smart:

∃x  At(x, Madison) ∧ Smart(x)

∃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

\[ At(KingJohn, Madison) ∧ Smart(KingJohn) \]
\[ ∨ At(Richard, Madison) ∧ Smart(Richard) \]
\[ ∨ At(Madison, Madison) ∧ Smart(Madison) \]
\[ ∨ ... \]
Another common mistake to avoid

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

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

$$\exists x \ At(x, Madison) \Rightarrow Smart(x)$$

is true if there is anyone who is not at Madison!

Properties of quantifiers

$\forall x \ \forall y \text{ is the same as } \forall y \ \forall x$  \text{(why??)}

$\exists x \ \exists y \text{ is the same as } \exists y \ \exists x$  \text{(why??)}

$\exists x \ \forall y \text{ is not the same as } \forall y \ \exists x$

$\exists x \ \forall y \ Loves(x, y)$

“There is a person who loves everyone in the world”

$\forall y \ \exists x \ Loves(x, y)$

“Everyone in the world is loved by at least one person”

Quantifier duality: each can be expressed using the other

$\forall x \ Likes(x, IceCream) \quad \neg \exists x \ \neg Likes(x, IceCream)$

$\exists x \ Likes(x, Broccoli) \quad \neg \forall x \ \neg Likes(x, Broccoli)$
Fun with sentences

Brothers are siblings

∀ x, y  \( \text{Brother}(x, y) \Rightarrow \text{Sibling}(x, y) \).

“Sibling” is symmetric
Brothers are siblings
\[ \forall x, y \quad Brother(x, y) \Rightarrow Sibling(x, y). \]
“Sibling” is symmetric
\[ \forall x, y \quad Sibling(x, y) \Leftrightarrow Sibling(y, x). \]
One’s mother is one’s female parent
\[ \forall x, y \quad Mother(x, y) \Leftrightarrow (Female(x) \wedge Parent(x, y)). \]
A first cousin is a child of a parent’s sibling
Fun with sentences

Brothers are siblings
∀ x, y  Brother(x, y) ⇒ Sibling(x, y).

“Sibling” is symmetric
∀ x, y  Sibling(x, y) ⇔ Sibling(y, x).

One’s mother is one’s female parent
∀ x, y  Mother(x, y) ⇔ (Female(x) ∧ Parent(x, y)).

A first cousin is a child of a parent’s sibling
∀ x, y  FirstCousin(x, y) ⇔
∃ p, ps  Parent(p, x) ∧ Sibling(ps, p) ∧ Parent(ps, y)

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.,  
\[
\begin{align*}
1 = 2 & \text{ and } \forall x \times (\sqrt{x}, \sqrt{x}) = x \text{ are satisfiable} \\
2 = 2 & \text{ is valid}
\end{align*}
\]

E.g., definition of (full) Sibling in terms of Parent:
∀ x, y  Sibling(x, y) ⇔ \[
\neg (x = y) ∧ \exists m, f \neg (m = f) ∧ \\
\text{Parent}(m, x) ∧ \text{Parent}(f, x) ∧ \text{Parent}(m, y) ∧ \text{Parent}(f, y)
\]
**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 \( t = 5 \):

\[
\text{Tell}(KB, \text{Percept}([\text{Smell}, \text{Breeze}, \text{None}], 5)) \\
\text{Ask}(KB, \exists a \; \text{Action}(a, 5))
\]

I.e., does the KB entail any particular actions at \( t = 5 \)?

Answer: Yes, \{a/\text{Shoot}\} ← substitution (binding list)

Given a sentence \( S \) and a substitution \( \sigma \),

\( S\sigma \) denotes the result of plugging \( \sigma \) into \( S \); e.g.,

\[
S = \text{Smarter}(x, y) \\
\sigma = \{x/\text{Hillary, y/Bill}\} \\
S\sigma = \text{Smarter}(\text{Hillary, Bill})
\]

\( \text{Ask}(KB, S) \) returns some/all \( \sigma \) such that \( KB \models S\sigma \)

---

**Knowledge base for the wumpus world**

“Perception”

\[
\forall b, g, t \; \text{Percept}([\text{Smell, b, g}], t) \Rightarrow \text{Smelt}(t) \\
\forall s, b, t \; \text{Percept}([s, b, \text{Glitter}], t) \Rightarrow \text{AtGold}(t)
\]

**Reflex:** \( \forall t \; \text{AtGold}(t) \Rightarrow \text{Action(Grab, t)} \)

**Reflex with internal state:** do we have the gold already?

\[
\forall t \; \text{AtGold}(t) \land \neg \text{Holding}(\text{Gold, t}) \Rightarrow \text{Action(Grab, t)}
\]

\( \text{Holding(\text{Gold, t})} \) cannot be observed

\( \Rightarrow \) keeping track of change is essential
Deducing hidden properties

Properties of locations:
\( \forall x, t \ At(Agent, x, t) \land Smelt(t) \Rightarrow Smelly(x) \)
\( \forall x, t \ At(Agent, x, t) \land Breeze(t) \Rightarrow Breezy(x) \)

Squares are breezy near a pit:

**Diagnostic** rule—infer cause from effect
\( \forall y \ Breezy(y) \Rightarrow \exists x \ Pit(x) \land Adjacent(x, y) \)

**Causal** rule—infer effect from cause
\( \forall x, y \ Pit(x) \land Adjacent(x, y) \Rightarrow Breezy(y) \)

Neither of these is complete—e.g., the causal rule doesn’t say whether squares far away from pits can be breezy

**Definition** for the *Breezy* predicate:
\( \forall y \ Breezy(y) \Leftrightarrow [\exists x \ Pit(x) \land Adjacent(x, y)] \)

Keeping track of change

Facts hold in situations, rather than eternally
E.g., *Holding(Gold, Now)* rather than just *Holding(Gold)*

**Situation calculus** is one way to represent change in FOL:

Adds a situation argument to each non-eternal predicate
E.g., *Now* in *Holding(Gold, Now)* denotes a situation

Situations are connected by the *Result* function
*Result*(a, s) is the situation that results from doing a in s
Describing actions I

“Effect” axiom—describe changes due to action
\[ \forall s \; AtGold(s) \Rightarrow Holding(Gold, Result(Grab, s)) \]

“Frame” axiom—describe non-changes due to action
\[ \forall s \; HaveArrow(s) \Rightarrow HaveArrow(Result(Grab, s)) \]

Frame problem: find an elegant way to handle non-change
(a) representation—avoid frame axioms
(b) inference—avoid repeated “copy-overs” to keep track of state

Qualification problem: true descriptions of real actions require endless caveats—what if gold is slippery or nailed down or . . .

Ramification problem: real actions have many secondary consequences—what about the dust on the gold, wear and tear on gloves, . . .

Describing actions II

Successor-state axioms solve the representational frame problem

Each axiom is “about” a predicate (not an action per se):

P true afterwards \[\Leftrightarrow\] [an action made P true
\[\lor\] P true already and no action made P false]

For holding the gold:
\[ \forall a, s \; Holding(Gold, Result(a, s)) \Leftrightarrow \]
\[ [(a = Grab \land AtGold(s)) \lor (Holding(Gold, s) \land a \neq Release)] \]
Making plans

Initial condition in KB:
\[ At(Agent, [1, 1], S_0) \]
\[ At(Gold, [1, 2], S_0) \]

Query: \[ Ask(KB, \exists s \ Holding(Gold, s)) \]
i.e., in what situation will I be holding the gold?

Answer: \[ \{ s/Result(Grab, Result(Forward, S_0)) \} \]
i.e., go forward and then grab the gold

This assumes that the agent is interested in plans starting at \( S_0 \) and that \( S_0 \)
is the only situation described in the KB

Making plans: A better way

Represent plans as action sequences \([a_1, a_2, \ldots, a_n]\)

\[ PlanResult(p, s) \] is the result of executing \( p \) in \( s \)

Then the query \[ Ask(KB, \exists p \ Holding(Gold, PlanResult(p, S_0))) \]
has the solution \[ \{ p/[Forward, Grab] \} \]

Definition of \( PlanResult \) in terms of \( Result \):
\[ \forall s \ PlanResult([], s) = s \]
\[ \forall a, p, s \ PlanResult([a|p], s) = PlanResult(p, Result(a, s)) \]

Planning systems are special-purpose reasoners designed to do this type of inference more efficiently than a general-purpose reasoner
Summary

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

Increased expressive power: sufficient to define wumpus world

Situation calculus:
– conventions for describing actions and change in FOL
– can formulate planning as inference on a situation calculus KB