**First-order logic**

**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, end 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>unknown degree of belief</td>
</tr>
<tr>
<td>Fuzzy logic</td>
<td>facts + degree of truth</td>
<td>known interval value</td>
</tr>
</tbody>
</table>
Syntax of FOL: Basic elements

Constants  \( KingJohn, 2, UCB, \ldots \)
Predicates  \( Brother, >, \ldots \)
Functions  \( Sqrt, LeftLegOf, \ldots \)
Variables  \( x, y, a, b, \ldots \)
Connectives  \( \land, \lor, \neg, \Rightarrow, \Leftrightarrow \)
Equality  \( = \)
Quantifiers  \( \forall, \exists \)
Atomic sentences

Atomic sentence \(= \) predicate\((term_1, \ldots, term_n)\)
\(\text{or } term_1 = term_2\)

Term \(= \) function\((term_1, \ldots, term_n)\)
\(\text{or constant or variable}\)

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

Complex sentences are made from atomic sentences using connectives

$$\neg S, \quad S_1 \land S_2, \quad S_1 \lor S_2, \quad S_1 \Rightarrow S_2, \quad S_1 \Leftrightarrow S_2$$

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

\(>(1, 2) \lor \leq (1, 2)\)

\(>(1, 2) \land \neg>(1, 2)\)
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 $\text{predicate}$.
Entailment in propositional logic can be computed by enumerating models.

We *can* enumerate the FOL 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 FOL models is not easy!
Universal quantification

\(\forall \langle \text{variables} \rangle \langle \text{sentence} \rangle\)

Everyone at Berkeley is smart:
\(\forall x \ At(x, Berkeley) \Rightarrow Smart(x)\)

\(\forall x \ P\) 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 \(P\)

\((At(KingJohn, Berkeley) \Rightarrow Smart(KingJohn))\)
\(\land (At(Richard, Berkeley) \Rightarrow Smart(Richard))\)
\(\land (At(Berkeley, Berkeley) \Rightarrow Smart(Berkeley))\)
\(\land \ldots\)
A common mistake to avoid

Typically, ⇒ is the main connective with ∀

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

$$\forall x \; At(x, Berkeley) \land Smart(x)$$

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

\[ \exists \langle \text{variables} \rangle \langle \text{sentence} \rangle \]

Someone at Stanford is smart:
\[ \exists x \ At(x, \text{Stanford}) \land \text{Smart}(x) \]

\[ \exists x \ P \text{ is true in a model } m \text{ iff } P \text{ is true with } x \text{ being some possible object in the model} \]

**Roughly** speaking, equivalent to the disjunction of instantiations of \( P \)

\[ (At(KingJohn, Stanford) \land \text{Smart}(KingJohn)) \lor (At(Richard, Stanford) \land \text{Smart}(Richard)) \lor (At(Stanford, Stanford) \land \text{Smart}(Stanford)) \lor \ldots \]
Another common mistake to avoid

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

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

$$\exists x \; At(x, Stanford) \Rightarrow Smart(x)$$

is true if there is anyone who is not at Stanford!
Properties of quantifiers

\(\forall x\ \forall y\) is the same as \(\forall y\ \forall x\) (why??)

\(\exists x\ \exists y\) is the same as \(\exists y\ \exists x\) (why??)

\(\exists x\ \forall y\) 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

\(\forall x\ Likes(x, \text{IceCream})\quad \neg\exists x\ \neg Likes(x, \text{IceCream})\)

\(\exists x\ Likes(x, \text{Broccoli})\quad \neg\forall x\ \neg Likes(x, \text{Broccoli})\)
Fun with sentences

Brothers are siblings
Fun with sentences

Brothers are siblings

\[ \forall x, y \; \text{Brother}(x, y) \Rightarrow \text{Sibling}(x, y). \]

“Sibling” is symmetric
Fun with sentences

Brothers are siblings

\[ \forall x, y \ Brother(x, y) \Rightarrow Sibling(x, y). \]

“Sibling” is symmetric

\[ \forall x, y \ Sibling(x, y) \Leftrightarrow Sibling(y, x). \]

One’s mother is one’s female parent
Brothers are siblings

\( \forall x, y \ Brother(x, y) \implies Sibling(x, y). \)

“Sibling” is symmetric

\( \forall x, y \ Sibling(x, y) \iff Sibling(y, x). \)

One’s mother is one’s female parent

\( \forall x, y \ Mother(x, y) \iff (Female(x) \land Parent(x, y)). \)

A first cousin is a child of a parent’s sibling
Brothers are siblings

\[ \forall x, y \quad \text{Brother}(x, y) \Rightarrow \text{Sibling}(x, y). \]

“Sibling” is symmetric

\[ \forall x, y \quad \text{Sibling}(x, y) \Leftrightarrow \text{Sibling}(y, x). \]

One’s mother is one’s female parent

\[ \forall x, y \quad \text{Mother}(x, y) \Leftrightarrow (\text{Female}(x) \land \text{Parent}(x, y)). \]

A first cousin is a child of a parent’s sibling

\[ \forall x, y \quad \text{FirstCousin}(x, y) \Leftrightarrow \exists p, ps \quad \text{Parent}(p, x) \land \text{Sibling}(ps, p) \land \text{Parent}(ps, y) \]
Equality

$term_1 = term_2$ is true under a given interpretation if and only if $term_1$ and $term_2$ refer to the same object.

E.g., $1 = 2$ and $\forall x \times (Sqrt(x), Sqrt(x)) = x$ are satisfiable.

$2 = 2$ is valid.

E.g., definition of (full) $Sibling$ in terms of $Parent$:

$\forall x, y Sibling(x, y) \iff [\neg(x = y) \land \exists m, f \neg(m = f) \land Parent(m, x) \land Parent(f, x) \land Parent(m, y) \land Parent(f, y)]$
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 \( t = 5 \):

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

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

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

Given a sentence \( S \) and a substitution \( \sigma \),
\( S\sigma \) denotes the result of plugging \( \sigma \) into \( S \); e.g.,
\[
S = \text{Smarter}(x, y)
\]
\[
\sigma = \{x/\text{Hillary}, y/\text{Bill}\}
\]
\[
S\sigma = \text{Smarter}(\text{Hillary, Bill})
\]

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

“Perception”
∀ b, g, t Percept([Smell, b, g], t) ⇒ Smelt(t)
∀ s, b, t Percept([s, b, Glitter], t) ⇒ AtGold(t)

Reflex: ∀ t AtGold(t) ⇒ Action(Grab, t)

Reflex with internal state: do we have the gold already?
∀ t AtGold(t) ∧ ¬Holding(Gold, t) ⇒ Action(Grab, t)

Holding(Gold, t) cannot be observed
⇒ 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., $\text{Holding}(\text{Gold}, \text{Now})$ rather than just $\text{Holding}(\text{Gold})$

**Situation calculus** is one way to represent change in FOL:
   Adds a situation argument to each non-eternal predicate
E.g., $\text{Now}$ in $\text{Holding}(\text{Gold}, \text{Now})$ denotes a situation

Situations are connected by the $\text{Result}$ function
$\text{Result}(a, s)$ is the situation that results from doing $a$ in $s$
Describing actions I

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

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

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):

\[ \text{P true afterwards} \iff \left[ \text{an action made P true} \right. \\
\left. \lor \quad \text{P true already and no action made P false} \right] \]

For holding the gold:

\[ \forall a, s \quad \text{Holding}(\text{Gold}, \text{Result}(a, s)) \iff \\
\left[ (a = \text{Grab} \land \text{AtGold}(s)) \right. \\
\left. \lor \quad (\text{Holding}(\text{Gold}, s) \land a \neq \text{Release}) \right] \]
Making plans

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

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

Answer: \( \{s/Result(\text{Grab}, \text{Result}(\text{Forward}, S_0))\} \)
   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
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:**
\[
\begin{align*}
\forall s \; PlanResult([], s) &= s \\
\forall a, p, s \; PlanResult([a|p], s) &= PlanResult(p, Result(a, s))
\end{align*}
\]

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