Propositional logic

Propositional logic is declarative

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 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

Examples of things we can say:

- All men are mortal:
  \[ \forall x \text{ Man}(x) \implies \text{Mortal}(x) \]

- Everybody loves somebody
  \[ \forall x \exists y \text{ Loves}(x, y) \]

- The meaning of the word “above”
  \[ \forall x \forall y \text{ above}(x, y) \iff (\text{on}(x, y) \lor \exists z (\text{on}(x, z) \land \text{above}(z, y))) \]
First Order logic

- Whereas propositional logic assumes the world contains **facts**

- first-order logic has
  - **Objects**: people, houses, numbers, colors, …
  - **Relations**: red, round, prime, brother-of, bigger-than, part-of, …
  - **Functions**: father-of, plus, …
Logics in General

- Ontological commitment: What exists in the world
  - PL: facts that hold or do not hold.
  - FOL: objects with relations between them that hold or do not hold

- Epistemological commitment: state of knowledge allowed with respect to a fact

<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 $\in [0, 1]$</td>
</tr>
<tr>
<td>Fuzzy logic</td>
<td>degree of truth $\in [0, 1]$</td>
<td>known interval value</td>
</tr>
</tbody>
</table>
Syntax of FOL

- User defines these primitives:
  - Constant symbols (i.e., the "individuals" in the world) E.g., Mary, 3
  - Function symbols (mapping individuals to individuals) E.g., father-of(Mary) = John, color-of(Sky) = Blue
  - Relation/predicate symbols (mapping from individuals to truth values) E.g., greater(5,3), green(Grass), color(Grass, Green)
FOL supplies these primitives:

- Variable symbols. E.g., x, y
- Connectives. Same as in PL: \( \neg, \Rightarrow, \land, \lor, \leftrightarrow \)
- Equality =
- Quantifiers: Universal (\( \forall \)) and Existential (\( \exists \))
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 \text{constant} or \text{variable}

Examples:

\( \text{Brother(KingJohn,RichardTheLionheart)} \)
\( \text{Greater(Length(LeftLegOf(Richard)), Length(LeftLegOf(KingJohn)))} \)
Complex sentences

Complex sentences are made from atomic sentences using connectives

\[ \neg S, \ S_1 \land S_2, \ S_1 \lor S_2, \ S_1 \Rightarrow S_2, \ S_1 \Leftrightarrow S_2 \]

and by applying quantifiers.

Examples:

- \( Sibling(KingJohn, Richard) \Rightarrow Sibling(Richard, KingJohn) \)
- \( greater(1,2) \lor less-or-equal(1,2) \)

- \( \forall x, y \ Sibling(x, y) \Rightarrow Sibling(y, x) \)
Truth in first-order logic

- Need to specify
  - constant symbols $\rightarrow$ objects
  - predicate symbols $\rightarrow$ relations
  - function symbols $\rightarrow$ functional relations

- An atomic sentence $\text{predicate}(term_1,\ldots,term_n)$ is true iff the objects referred to by $term_1,\ldots,term_n$ are in the relation referred to by $\text{predicate}$.
Models for FOL: Example
Quantifiers

- Allow us to express properties of collections of objects instead of enumerating objects by name
- Universal: “for all” $\forall$
- Existential: “there exists” $\exists$
Universal quantification

∀<variables> <sentence>

Everyone at CSU is smart:
∀x At(x, CSU) ⇒ Smart(x)

- ∀x P(x) is true iff P is true for every object x
- Roughly speaking, equivalent to the conjunction of instantiations of P
  At(KingJohn,CSU) ⇒ Smart(KingJohn)
  ∧ At(Richard,CSU) ⇒ Smart(Richard)
  ∧ At(CSU,CSU) ⇒ Smart(CSU)
  ∧ ...
Using universal quantifiers

- Typically $\Rightarrow$ is the main connective with $\forall$

- Do not make the following mistake:
  $\forall x \ At(x, \text{CSU}) \land \text{Smart}(x)$
  means “Everyone is at CSU and everyone is smart”
Existential quantification

\[ \exists \text{<variables>} \ <\text{sentence}> \]

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

- \[ \exists x P(x) \text{ is true iff P is true for some object } x \]
- Roughly speaking, equivalent to the disjunction of instantiations of P
  \[ \text{At(KingJohn,CSU)} \land \text{Smart(KingJohn)} \]
  \[ \lor \text{At(Richard, CSU)} \land \text{Smart(Richard)} \]
  \[ \lor \text{At(CSU, CSU)} \land \text{Smart(CSU)} \]
  \[ \lor \ldots \]
Existential quantification (cont.)

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

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

When is this true?
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
  ∀x Likes(x,IceCream)  ¬∃x ¬Likes(x,IceCream)
  ∃x Likes(x,Broccoli)  ¬∀x ¬Likes(x,Broccoli)
Equality

- $term_1 = term_2$ is true if and only if $term_1$ and $term_2$ refer to the same object.

- E.g., definition of 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 smell and a breeze (but no glitter) at position [i,j]:

  \[ \text{Tell}(KB, \text{Percept}([\text{Smell}, \text{Breeze}, \text{NoGlitter}],[i,j])) = \text{assertion} \]

  \[ \text{Ask}(KB, \exists a \text{ BestAction}(a,[i,j])) \]
  \[ (= \text{query}) \]

  I.e., does the KB entail some best action?

- Answering yes without the best action is not very helpful so:
  Answer: Yes, \( \{a/\text{Shoot}\} \) : substitution (binding list)

- \( \text{Ask}(KB, \alpha) \) returns some/all s such that KB entails \( \text{SUBST}(s, \alpha) \).
- To determine good course of action: \( \text{Ask}(KB, \text{Safe}([1,3])) \)
KB for wumpus world

- **Perception**
  - $\forall t,s,g,m,c \ \text{Percept}([s,\text{Breeze},g,m,c],t) \implies \text{Breeze}(t)$

- **Action**
  - $\forall t \ \text{Glitter}(t) \implies \text{BestAction}(\text{Grab}, t)$
The wumpus world

Squares are breezy near a pit:

- First define the concept of adjacency:
  \[ \forall x, y, a, b \ \text{Adjacent}([x,y],[a,b]) \iff (x = a \land (y = b - 1 \lor y = b + 1)) \lor (y = b \land (x = a - 1 \lor x = a + 1)) \]

- \( At(Agent, s, t) \) means Agent at square s at time t

- Infer properties
  \[ \forall s, t \ \ At(Agent, s, t) \land Breeze(t) \Rightarrow Breezy(s) \]
The wumpus world

Squares are breezy near a pit:
\[ \forall s \text{ Breezy}(s) \iff \exists r \text{ Adjacent}(s, r) \land \text{Pit}(r) \]

What is the following saying?
\[ \forall x,r,s,t \, \text{At}(x,r,t) \land \text{At}(x,s,t) \Rightarrow s = r \]

Recall that \( \text{At}(\text{Agent},s,t) \) means Agent at square \( s \) at time \( t \)
Creating a KB using FOL

1. Identify the task (what will the KB be used for)
2. Assemble the relevant knowledge
   Knowledge acquisition.
3. Decide on a vocabulary of predicates, functions, and constants
   Translate domain-level knowledge into logic-level names.
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
Examples

The *kinship* domain

- Basic predicates: Female, Parent...

Other predicates in this domain:

- One's mother is one's female parent
  \[ \forall x,c \ (\text{Mother}(c) = x) \iff (\text{Female}(x) \land \text{Parent}(x,c)) \]

- Sibling
  \[ \forall x,y \ \text{Sibling}(x,y) \iff [\neg (x = y) \land \exists p \ \text{Parent}(p,x) \land \text{Parent}(p,y)] \]

- These are the axioms of the domain (they are also definitions since they use biconditionals).

- Some sentences are “theorems” -- they can be derived from the axioms:
  - “Sibling” is symmetric
    \[ \forall x,y \ \text{Sibling}(x,y) \iff \text{Sibling}(y,x) \]
Examples (cont)

The set domain

Notation: \{x|s\} is the set resulting from adding x to the set s.

- \( \forall s \ \text{Set}(s) \iff (s = \{\} \lor (\exists x, s_2 \ \text{Set}(s_2) \land s = \{x|s_2\})) \)
- \( \neg \exists x, s \ {x|s} = \{} \)
- \( \forall x, s \ x \in s \iff [ \exists y, s_2 (s = \{y|s_2\} \land (x = y \lor x \in s_2))] \)
- \( \forall s_1, s_2 \ s_1 \subseteq s_2 \iff (\forall x \ x \in s_1 \implies x \in s_2) \)
- \( \forall s_1, s_2 (s_1 = s_2) \iff (s_1 \subseteq s_2 \land s_2 \subseteq s_1) \)
- \( \forall x, s_1, s_2 \ x \in (s_1 \cap s_2) \iff (x \in s_1 \land x \in s_2) \)
- \( \forall x, s_1, s_2 \ x \in (s_1 \cup s_2) \iff (x \in s_1 \lor x \in s_2) \)
Examples (cont)

The *natural numbers* domain

- 0 is a natural number:
  \[ \text{NatNum}(0) \]

- The successor of a natural number is a natural number:
  \[ \forall n \text{NatNum}(n) \Rightarrow \text{NatNum}(S(n)) \]

- Constraints on the successor function:
  \[ \forall n \neg (0 = S(n)) \]
  \[ \forall m,n \neg (m = n) \Rightarrow \neg (S(m) = S(n)) \]

- Defining addition:
  \[ \forall n \text{NatNum}(n) \Rightarrow +(0, n) = n \]
  \[ \forall m,n \text{NatNum}(m) \land \text{NatNum}(n) \Rightarrow +(S(m), n) = S(+m,n)) \]