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 \( B_1 \land B_2 \) is derived from meaning of \( B_1 \) and of \( B_2 \).
Meaning in propositional logic is context-independent (unlike natural languages, where meaning depends on context).
Propositional logic has very limited expressive power (unlike natural languages).

First-order logic

Whereas propositional logic assumes world contains facts, first-order logic (the original language) assumes the world contains entities.

Syntax of FOL: Basic elements

Constants: KingJohn; 2; UCB;
Predicates: Brother; >;
Functions: Sqrt; LeftLegOf;
Variables: x; y; a; b;
Connectives: ^, _;
Equality: =
Quantiﬁers: \( \forall \), \( \exists \)

First-order logic

 Whereas propositional logic assumes world contains facts, first-order logic (the original language) assumes the world contains entities.
Atomic sentences

An atomic sentence is a predicate applied to terms:

\[ \text{predicate} \left( \text{term}_1; \ldots; \text{term}_n \right) \]

or

\[ \text{term}_1 = \text{term}_2 \]

A term is a function applied to terms:

\[ \text{function} \left( \text{term}_1; \ldots; \text{term}_n \right) \]

or

a constant or a variable.

Example:

\[ \text{Brother}(\text{KingJohn};\text{RichardTheLionheart}) > \left(\text{Length}(\text{LeftLegOf}(\text{Richard});\text{Length}(\text{LeftLegOf}(\text{KingJohn})))\right) \]

Chapter 8

Complex sentences

Complex sentences are made from atomic sentences using connectives:

\[ S_1 \land S_2 \]

\[ S_1 \lor S_2 \]

\[ S_1 \rightarrow S_2 \]

\[ S_1 \leftrightarrow S_2 \]

Example:

\[ \text{Sibling}(\text{KingJohn};\text{Richard}) \]

\[ \text{Sibling}(\text{Richard};\text{KingJohn}) \]

\[ (1, 2) \land (1, 2) \]

\[ (1, 2) \lor (1, 2) \]

\[ (1, 2) \leftrightarrow (1, 2) \]

Truth in first-order logic

Sentences are true with respect to a model and an interpretation.

A model contains objects (domain elements) and relations among them.

An interpretation specifies referents for constant symbols, predicate symbols, function symbols, and variables.

A model can be enumerated by enumerating its objects, relations, and interpretations.

A sentence is true if the objects referred to by its terms are in the relation referred to by its predicate in the model.

Example:

Consider the interpretation in which

\[ \text{Richard} \mapsto \text{Richard the Lionheart} \]

\[ \text{John} \mapsto \text{the evil King John} \]

\[ \text{Brother} \mapsto \text{the brotherhood relation} \]

Under this interpretation, \( \text{Brother}(\text{Richard};\text{John}) \) is true just in case \( \text{Richard the Lionheart} \) and \( \text{the evil King John} \) are in the brotherhood relation in the model.

Chapter 8

Models for FOL: Lots!

Entailment in propositional logic can be computed by enumerating models:

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

$\forall x \text{At}(x; \text{Berkeley}) \land \text{Smart}(x)$

Roughly speaking, equivalent to the conjunction of instantiations of $P$

Typically, a common mistake to avoid

$\exists x \text{At}(x; \text{Berkeley}) \land \text{Smart}(x)$

Common mistake: using $\land$ as the main connective with $\forall$

Another common mistake to avoid

$\exists x \text{At}(x; \text{Berkeley}) \land \text{Smart}(x)$

Typically, a common mistake to avoid

Some possible objects in the model $x \in M$ such that $\text{At}(x; \text{Berkeley})$ and $\text{Smart}(x)$
Fun with sentences

Brothers are siblings.

8 x; y Brother(x; y) \implies Sibling(x; y).

Sibling(x; y) is symmetric.

8 x; y Sibling(x; y) \implies Sibling(y; x).

One's mother is one's female parent.

8 x; y Mother(x; y) \implies Female(x) \land Parent(x; y).

A first cousin is a child of a parent's sibling.

8 x; y FirstCousin(x; y) \implies Parent(p; x) \land Parent(p; 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 \sqrt{x} = \sqrt{x} are satisfiable.

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 time t = 5.

Tell(KB; Percept(\{Smell; Breeze; None\}; 5))

Ask(KB; a; 5)

I.e., does KB entail any particular actions at time t = 5?

Answer: Yes; f a = Shoot g substitution (binding list)

Given a sentence S and a substitution , S denotes the result of plugging into S; e.g., S = Smarter(x; y) = f x = Hillary; y = Bill g S = Smarter(Hillary; Bill).

Ask(KB; S) returns some/all such that KB \models S.

Equality

\forall x; y \forall \exists x' \exists y' (F(x; x') \land F(y; y') \land \forall z (F(z; z') \implies z = x \lor z = y)) \implies x = y.

A first cousin is a child of a parent's sibling.

A x; y FirstCousin(x; y) \implies Parent(p; x) \land Parent(p; y).

One's mother is one's female parent.

A x; y Mother(x; y) \implies Female(x) \land Parent(x; y).

Each term is a (full) Sibling.

A x; y Sibling(x; y) \implies Sibling(y; x).

A x; y Brother(x; y) \implies Sibling(y; x).

Brothers are siblings.
Knowledge base for the Wumpus world

Chapter 8 25

Deducing hidden properties

Properties of locations:

- At(x) ∧ Smell(x, t) ⇒ Smell(x, t)
- Breeze(x) ⇒ Breezy(x)

Squares are breezy near a pit:

Diagnostic rule: infer cause from effect

- Breezy(y) ⇒ 9 x Pit(x) ∧ Adjacent(x, y)

Causal rule: infer effect from cause

- Pit(x) ∧ Adjacent(x, y) ⇒ 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:

- Breezy(y) ⇔ 9 x Pit(x) ∧ Adjacent(x, y)

Chapter 8 26

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)

Situations are connected by the Result function

- Result(a; s) is the situation that results from doing a in s

PIT PIT PIT Gold PIT Gold S 0 Forward S 1

Chapter 8 27

Describing actions I

- Efect axiom: describe changes due to action
- Result(a; s) ⇒ Holding(Gold; Result(Grab; s))

- Frame axiom: describe non-changes due to action
- Result(a; s) ⇒ Result(Grab; s)

Frame problem: find an elegant way to handle non-change.

- Representation: avoid frame axioms.
- 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, ...

Chapter 8 28

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, [an action made P true]_P true already and no action made P false

For holding the gold:

- a; s Holding(Gold; Result(a; s)), [a = Grab ∧ AtGold(s) \¬ (Holding(Gold; s) ∧ a \ne Release)]

Chapter 8 29

Making plans

Initial condition in KB:

- At(Agent; [1; 1]; S0)
- At(Gold; [1; 2]; S0)

Query:

- Ask(KB; 9 s Holding(Gold; s))

i.e., in what situation will I be holding the gold?

Answer:

- f s(Result(Grab(Result(Forward; S0)))) g

i.e., go forward and then grab the gold.

This assumes that the agent is interested in plans starting at S0 and that S0 is the only situation described in the KB.
Making plans: A better way

Represent plans as action sequences

PlanResult(p; s) is the result of executing p in s.

Then the query Ask(KB; 9p Holding(Gold; PlanResult(p; S0)))) has the solution f p = [Forward; Grab] g.

Definition of PlanResult in terms of Result:

\[
\begin{align*}
\text{PlanResult}(\text{a}_1 \ldots \text{a}_n; s) &= s \\
\text{PlanResult}(\text{a}_1 \ldots \text{a}_n; s) &= \text{PlanResult}(\text{p}; \text{Result}(\text{a}; s))
\end{align*}
\]

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

Chapter 8

Increasing 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