Inference in first-order logic:
Reducing first-order inference to propositional inference

AIMA
Chapter 9.1-9.2, 9.5

Outline
- Reducing first-order inference to propositional inference
- Unification

Universal instantiation (UI)
- Every instantiation of a universally quantified sentence is entailed by it:
  \[ \forall v \alpha \vdash \text{Subst}(\{v/g\}, \alpha) \]
  for any variable \( v \) and ground term \( g \)
- E.g., \( \forall x \text{King}(x) \land \text{Greedy}(x) \Rightarrow \text{Evil}(x) \) yields any or all of:
  \[ \forall x, \exists x \text{King}(x) \land \text{Greedy}(x) \Rightarrow \text{Evil}(x) \]
  \[ \text{Subst}(\{v/\text{John}\}, \alpha) \]
  where \( v \) is replaced by \( \text{John} \)
- Instantiating the universal sentence in all possible ways, we have:
  \[ \text{King}(\text{John}) \land \text{Greedy}(\text{John}) \Rightarrow \text{Evil}(\text{John}) \]
  \[ \text{King}(\text{Richard}) \land \text{Greedy}(\text{Richard}) \Rightarrow \text{Evil}(\text{Richard}) \]
  \[ \text{King}(\text{GoldCrown}) \land \text{Greedy}(\text{GoldCrown}) \Rightarrow \text{Evil}(\text{GoldCrown}) \]

Existential instantiation (EI) (simple case)
- For any sentence \( \alpha \), variable \( v \), and constant symbol \( k \) that does not appear elsewhere in the knowledge base:
  \[ \exists v \alpha \vdash \text{Subst}(\{v/k\}, \alpha) \]
- E.g., \( \exists x \text{Crown}(x) \land \text{OnHead}(x, \text{John}) \) yields:
  \[ \text{Crown}(\text{C1}) \land \text{OnHead}(\text{C1}, \text{John}) \]
  provided \( \text{C1} \) is a new constant symbol, called a Skolem constant

Existential Instantiation continued
- UI can be applied several times to add new sentences
  - the new KB is logically equivalent to the old
- EI can be applied once to replace the existential sentence
  - the new KB is not equivalent to the old
  - but it is satisfiable iff the old KB was satisfiable

Reduction to propositional inference
Suppose the KB contains just the following:
\[ \forall x \text{King}(x) \land \text{Greedy}(x) \Rightarrow \text{Evil}(x) \]
\[ \text{King}(\text{John}) \]
\[ \text{Greedy}(\text{John}) \]
\[ \text{Brother}(\text{Richard}, \text{John}) \]
Instantiating the universal sentence in all possible ways, we have:
\[ \text{King}(\text{John}) \land \text{Greedy}(\text{John}) \Rightarrow \text{Evil}(\text{John}) \]
\[ \text{King}(\text{Richard}) \land \text{Greedy}(\text{Richard}) \Rightarrow \text{Evil}(\text{Richard}) \]
\[ \text{King}(\text{John}) \land \text{Greedy}(\text{John}) \Rightarrow \text{Evil}(\text{Richard}) \]
\[ \text{Brother}(\text{Richard}, \text{John}) \]
- The new KB is propositionalized: atomic proposition symbols are
  \[ \text{King}(\text{John}), \text{Greedy}(\text{John}), \text{Evil}(\text{John}), \text{King}(\text{Richard}), \text{etc.} \]
Reduction contd.

- **Claim:** Every FOL KB can be propositionalized so as to preserve entailment
  - (A ground sentence is entailed by new KB iff entailed by original KB)
- **Idea:** propositionalize KB and query, apply resolution, return result
- **Problem:** with function symbols, there are infinitely many ground terms,
  - e.g., Father(Father(Father(John)))

Problems with propositionalization

1. Propositionalization generates lots of irrelevant sentences.
2. With \( p \)-ary predicates and \( n \) constants, there are \( p \cdot n^p \) instantiations.
   - With function symbols, doing this naïvely can result in an unbounded number of sentences!

Substitution and Unification

- **Subst(\( \theta \), \( p \))**: the result of substituting \( \theta \) into sentence \( p \)
  - If \( \theta = \{x/Jane\} \) (i.e. “substitute “Jane” for “x”), then \( \text{Subst}(\theta, \text{Happy}(x)) = \text{Happy}(\text{Jane}) \)
- **Unify algorithm:** takes 2 sentences \( p \) and \( q \) and returns a unifier \( \theta \) if one exists
  \[ \text{Unify}(p,q) = \theta \text{ where } \text{Subst}(\theta, p) = \text{Subst}(\theta, q) \]
- **Example:**
  \[ p = \text{Knows}(\text{John},x) \]
  \[ q = \text{Knows}(\text{John},\text{Jane}) \]
  \[ \text{Unify}(p,q) = \{x/\text{Jane}\} \]

Unification

More Examples:
\[
\begin{array}{ccc}
p & q & \theta \\
\text{Knows}(\text{John},x) & \text{Knows}(\text{John},\text{Jane}) & \{y/\text{John}\} \\
\text{Knows}(\text{John},x) & \text{Knows}(\text{y},\text{OJ}) & \{x/\text{OJ},y/\text{John}\} \\
\text{Knows}(\text{John},x) & \text{Knows}(\text{y},\text{Mother}(y)) & \{y/\text{Mother}(\text{John})\} \\
\text{Knows}(\text{John},x) & \text{Knows}(\text{x},\text{OJ}) & \text{fail} \\
\end{array}
\]

Most General Unifier

- To unify \( \text{Knows}(\text{John},x) \) and \( \text{Knows}(\text{y},\text{z}) \),
  \[ \theta = \{y/\text{John}, x/\text{z}\} \]
  \[ \alpha \theta = \{y/\text{John}, x/\text{John}, z/\text{John}\} \]
- The first unifier is more general than the second.
- There is a single most general unifier (MGU) that is unique up to renaming of variables.
  \[ \text{MGU} = \{y/\text{John}, x/\text{z}\} \]
### Generalized Modus Ponens (GMP)

\[
P_1, P_2, \ldots, P_n \quad \text{where } P_i \theta = P_i \text{ for all } i
\]

Example:

\[
\text{King}(John), \text{Greedy}(y) \quad \text{(King}(x) \land \text{Greedy}(x) \Rightarrow \text{Evil}(x))
\]

\[
\text{Evil}(John) \quad \theta = \{y/John, y/John\}
\]

- GMP used with KB of definite clauses (special case of Horn clause with exactly one positive literal)
- All variables assumed universally quantified

### Resolution for FOL: brief summary

- Full first-order version:
  \[
  \xi \lor \ldots \lor \xi_i \land \ldots \land \xi_j \lor \ldots \lor \xi_k \land \ldots \land \xi_m \Rightarrow \emptyset
  \]
  
  where \(\text{Unify}(\xi, \neg\xi) = \emptyset\).

  - The two clauses are assumed to be “standardized apart” so that they share no variables.

  - For example,

\[
\neg\text{Rich}(x) \lor \text{Unhappy}(x) \quad \text{Rich}(Ken)
\]

- Apply resolution steps to CNF(KB \land \neg\neg); complete for FOL

### Conversion to CNF for FOL

- Everyone who loves all animals is loved by someone:
  \[\forall x \forall y \text{Animal}(y) \Rightarrow \text{Loves}(x,y) \Rightarrow [\exists y \text{Loves}(x,y)]\]

1. Eliminate biconditionals and implications

\[
\forall x \forall y \neg \text{Animal}(y) \land \text{Loves}(x,y) \Rightarrow [\exists y \text{Loves}(x,y)]
\]

2. Move \(\neg\) inwards:

\[
\forall x \forall y p \equiv \exists y p = \neg p
\]

\[
\forall x \forall y \neg \text{Animal}(y) \land \neg \text{Loves}(x,y) \Rightarrow [\exists y \text{Loves}(x,y)]
\]

\[
\forall x \forall y \text{Animal}(y) \land \neg \text{Loves}(x,y) \Rightarrow [\exists y \text{Loves}(x,y)]
\]

- Skolemize: a more general form of existential instantiation.

  - Each existential variable is replaced by a Skolem function of the enclosing universally quantified variables:

\[
\forall x \forall y \text{Animal}(F(x)) \land \neg \text{Loves}(x,F(x)) \Rightarrow \text{Loves}(G(x),x)
\]

5. Drop universal quantifiers:

\[
[\text{Animal}(F(x)) \land \neg \text{Loves}(x,F(x))] \lor \text{Loves}(G(x),x)
\]

6. Distribute \(\lor\) over \(\land\):

\[
[\text{Animal}(F(x)) \lor \text{Loves}(G(x),x)] \lor [\neg \text{Loves}(x,F(x)) \lor \text{Loves}(G(x),x)]
\]

### Conversion to CNF contd.

### Example: Did Curiosity kill the cat?

- Jack owns a dog. Every dog owner is an animal lover. No animal lover kills an animal. Either Jack or Curiosity killed the cat, who is named Tuna. Did Curiosity kill the cat?

- The axioms can be represented as follows:

<table>
<thead>
<tr>
<th>Axiom</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\exists x) Dog(x) \land \text{Owns}(Jack,x)</td>
</tr>
<tr>
<td>(\forall x) (\exists y) Dog(y) \land \text{Owns}(x,y) \Rightarrow \text{AnimalLover}(x)</td>
</tr>
<tr>
<td>(\forall x) \text{AnimalLover}(x) \Rightarrow (\forall y) \text{AnimalLover}(y) \Rightarrow \neg \text{Kills}(x,y)</td>
</tr>
<tr>
<td>\text{Kills}(Jack, Tuna) \land \neg \text{Kills}(Curiosity, Tuna)</td>
</tr>
<tr>
<td>\text{Cat}(Tuna)</td>
</tr>
<tr>
<td>(\forall x) \text{Cat}(x) \Rightarrow \text{Animal}(x)</td>
</tr>
</tbody>
</table>

(Taken and next slide adapted from CMSC421 – Fall 2006, Univ. of Maryland)