Inference in first-order logic

CHAPTER 9, SECTIONS 1–4

Outline

◇ Proofs
◇ Unification
◇ Generalized Modus Ponens
◇ Forward and backward chaining
Proofs

Sound inference: find $\alpha$ such that $KB \models \alpha$.
Proof process is a search, operators are inference rules.

E.g., Modus Ponens (MP)

$$\frac{\alpha, \; \alpha \Rightarrow \beta}{\beta \; \text{At}(Joe, UCB) \; \text{At}(Joe, UCB) \Rightarrow \text{OK}(Joe)} \; \text{OK}(Joe)$$

E.g., And-Introduction (AI)

$$\begin{array}{c}
\alpha \; \beta \\
\alpha \land \beta
\end{array} \; \begin{array}{c}
\text{OK}(Joe) \; \text{CSMajor}(Joe) \\
\text{OK}(Joe) \land \text{CSMajor}(Joe)
\end{array}$$

E.g., Universal Elimination (UE)

$$\begin{array}{c}
\forall x \; \alpha \\
\forall x \; \text{At}(x, UCB) \Rightarrow \text{OK}(x)
\end{array} \; \begin{array}{c}
\alpha[x/\tau] \\
\text{At}(Pat, UCB) \Rightarrow \text{OK}(Pat)
\end{array}$$

$\tau$ must be a ground term (i.e., no variables)

---

Example proof

Bob is a buffalo
Pat is a pig
Buffaloes outrun pigs
Bob outruns Pat

<table>
<thead>
<tr>
<th>Step</th>
<th>Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Buffalo(Bob)</td>
</tr>
<tr>
<td>2</td>
<td>Pig(Pat)</td>
</tr>
<tr>
<td>3</td>
<td>$\forall x, y ; \text{Buffalo}(x) \land \text{Pig}(y) \Rightarrow \text{Faster}(x, y)$</td>
</tr>
<tr>
<td>4</td>
<td>Buffalo(Bob) \land Pig(Pat)</td>
</tr>
<tr>
<td>5</td>
<td>Buffalo(Bob) \land Pig(Pat) \Rightarrow Faster(Bob, Pat)</td>
</tr>
<tr>
<td>6</td>
<td>Faster(Bob, Pat)</td>
</tr>
</tbody>
</table>

---

Al 1 & 2
UE 3, $\{x/\text{Bob, } y/\text{Pat}\}$
MP 6 & 7
Search with primitive inference rules

Operators are inference rules
States are sets of sentences
Goal test checks state to see if it contains query sentence

AI, UE, MP is a common inference pattern

Problem: branching factor huge, esp. for UE

Idea: find a substitution that makes the rule premise match some known facts

⇒ a single, more powerful inference rule

Unification

A substitution σ unifies atomic sentences p and q if pσ = qσ

<table>
<thead>
<tr>
<th>p</th>
<th>q</th>
<th>σ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knows(John,x)</td>
<td>Knows(John, Jane)</td>
<td>{x/Jane}</td>
</tr>
<tr>
<td>Knows(John,x)</td>
<td>Knows(y, OJ)</td>
<td>{x/John, y/OJ}</td>
</tr>
<tr>
<td>Knows(John,x)</td>
<td>Knows(y, Mother(y))</td>
<td>{y/John, x/Mother(John)}</td>
</tr>
</tbody>
</table>

Idea: Unify rule premises with known facts, apply unifier to conclusion
E.g., if we know q and Knows(John, x) ⇒ Likes(John, x) then we conclude Likes(John, Jane)
Likes(John, OJ)
Likes(John, Mother(John))
Section 9.4
Forward and Backward Chaining

---

**Generalized Modus Ponens (GMP)**

\[ p_1^i, \ p_2^i, \ldots, \ p_n^i, \ (p_1 \land p_2 \land \ldots \land p_n \Rightarrow q) \]

\[ q^\sigma \quad \text{where } p_i^\sigma = p_i \sigma \text{ for all } i \]

E.g. \( p_1^i = \text{Faster}(\text{Bob, Pat}) \)
\( p_2^i = \text{Faster}(\text{Pat, Steve}) \)
\( p_1 \land p_2 \Rightarrow q = \text{Faster}(x, y) \land \text{Faster}(y, z) \Rightarrow \text{Faster}(x, z) \)
\( \sigma = \{x/\text{Bob}, y/\text{Pat}, z/\text{Steve}\} \)
\( q^\sigma = \text{Faster}(\text{Bob, Steve}) \)

GMP used with KB of definite clauses (exactly one positive literal):
either a single atomic sentence or
(conjunction of atomic sentences) \( \Rightarrow \) (atomic sentence)
All variables assumed universally quantified
Completeness in FOL

Procedure $i$ is complete if and only if

$$KB \vdash \alpha \text{ whenever } KB \models \alpha$$

Forward and backward chaining are complete for Horn KBs but incomplete for general first-order logic.

E.g., from

- $PhD(x) \Rightarrow HighlyQualified(x)$
- $\neg PhD(x) \Rightarrow EarlyEarnings(x)$
- $HighlyQualified(x) \Rightarrow Rich(x)$
- $EarlyEarnings(x) \Rightarrow Rich(x)$

should be able to infer $Rich(Mr)$, but FC/BC won’t do it.

Does a complete algorithm exist?

Resolution

Entailment in first-order logic is only semidecidable:

- can find a proof of $\alpha$ if $KB \models \alpha$
- cannot always prove that $KB \not\models \alpha$

Cf. Halting Problem: proof procedure may be about to terminate with success or failure, or may go on for ever.

Resolution is a refutation procedure:

- to prove $KB \models \alpha$, show that $KB \land \neg \alpha$ is unsatisfiable.

Resolution uses $KB$, $\neg \alpha$ in CNF (conjunction of clauses).

Resolution inference rule combines two clauses to make a new one:

$$C_1 \lor C_2$$

Inference continues until an empty clause is derived (contradiction).
Resolution inference rule

Basic propositional version:

\[
\begin{align*}
\alpha \lor \beta, \quad \neg \beta \lor \gamma & \quad \text{or equivalently} \quad \neg \alpha \Rightarrow \beta, \quad \beta \Rightarrow \gamma \\
\alpha \lor \gamma
\end{align*}
\]

Full first-order version:

\[
\begin{align*}
p_1 \lor \ldots \lor p_{m-1} \lor p_m, \\
q_1 \lor \ldots \lor q_{n-1} \lor q_n
\end{align*}
\]

\[
\begin{align*}
(p_1 \lor \ldots \lor p_{m-1} \lor p_{m+1} \ldots \lor p_m \lor q_1 \ldots \lor q_{n-1} \lor q_{n+1} \ldots \lor q_n)\sigma
\end{align*}
\]

where \( p_i\sigma = \neg q_i\sigma \)

For example,

\[
\begin{align*}
\neg \text{Rich}(x) \lor \text{Unhappy}(x) \\
\text{Rich}(Me) \\
\text{Unhappy}(Me)
\end{align*}
\]

with \( \sigma = \{ x/\text{Me} \} \)

Conjunctive Normal Form

Literal = (possibly negated) atomic sentence, e.g., \( \neg \text{Rich}(\text{Me}) \)

Clause = disjunction of literals, e.g., \( \neg \text{Rich}(\text{Me}) \lor \text{Unhappy}(\text{Me}) \)

The KB is a conjunction of clauses

Any FOL KB can be converted to CNF as follows:

1. Replace \( P \Rightarrow Q \) by \( \neg P \lor Q \)
2. Move \( \neg \) inwards, e.g., \( \neg \forall x P \) becomes \( \exists x \neg P \)
3. Standardize variables apart, e.g., \( \forall x P \lor \exists x Q \) becomes \( \forall \tilde{x} P \lor \exists \tilde{y} Q \)
4. Move quantifiers left in order, e.g., \( \forall x P \lor \exists x Q \) becomes \( \forall \tilde{x} P \lor Q \)
5. Eliminate \( \exists \) by Skolemization (next slide)
6. Drop universal quantifiers
7. Distribute \( \land \) over \( \lor \), e.g., \( (P \land Q) \lor R \) becomes \( (P \lor Q) \land (P \lor R) \)
Skolemization

\[ \exists x \text{Rich}(x) \text{ becomes } \text{Rich}(G1) \text{ where G1 is a new "Skolem constant"} \]

\[ \exists k \ \delta_\exists(k^\alpha) = k^\beta \text{ becomes } \delta_\exists(e^\beta) = e^\gamma \]

More tricky when \( \exists \) is inside \( \forall \)

E.g., "Everyone has a heart"

\[ \forall x \ \text{Person}(x) \Rightarrow \exists y \ \text{Heart}(y) \land \text{Has}(x, y) \]

Incorrect:

\[ \forall x \ \text{Person}(x) \Rightarrow \text{Heart}(H1) \land \text{Has}(x, H1) \]

Correct:

\[ \forall x \ \text{Person}(x) \Rightarrow \text{Heart}(H(x)) \land \text{Has}(x, H(x)) \]

where \( H \) is a new symbol ("Skolem function")

Skolem function arguments: all enclosing universally quantified variables

---

Resolution proof

To prove \( \alpha \):

- negate it
- convert to CNF
- add to CNF KB
- infer contradiction

E.g., to prove \( \text{Rich}(me) \), add \( \neg\text{Rich}(me) \) to the CNF KB

\[ \neg\text{PhD}(x) \lor \text{HighlyQualified}(x) \]

\[ \text{PhD}(x) \lor \text{EarlyEarnings}(x) \]

\[ \neg\text{HighlyQualified}(x) \lor \text{Rich}(x) \]

\[ \neg\text{EarlyEarnings}(x) \lor \text{Rich}(x) \]