Inference in first-order logic

Chapter 9

Outline

♦ Reducing first-order inference to propositional inference
♦ Unification
♦ Generalized Modus Ponens
♦ Forward and backward chaining
♦ Logic programming
♦ Resolution

Universal instantiation (UI)

Every instantiation of a universally quantified sentence is entailed by it:

\[
\forall v \alpha \quad \text{SUBST}(\{v/g\}, \alpha)
\]

for any variable \(v\) and ground term \(g\)

E.g., \(\forall x \text{ King}(x) \land \text{Greedy}(x) \Rightarrow \text{Evil}(x)\) yields

King(Johann) \land Greedy(Johann) \Rightarrow Evil(Johann)

King(Richard) \land Greedy(Richard) \Rightarrow Evil(Richard)

King(Father(Johann)) \land Greedy(Father(Johann)) \Rightarrow Evil(Father(Johann))

Existential instantiation (EI)

For any sentence \(\alpha\), variable \(v\), and constant symbol \(k\) that does not appear elsewhere in the knowledge base:

\[
\exists v \alpha \quad \text{SUBST}(\{v/k\}, \alpha)
\]

E.g., \(\exists x \text{ Crown}(x) \land \text{OnHead}(x, Johann)\) yields

Crown(C1) \land \text{OnHead}(C1, Johann)

provided \(C1\) is a new constant symbol, called a Skolem constant

Another example: from \(\exists x \ d(x^y)/dy = x^y\) we obtain

\(d(e^y)/dy = e^y\)

provided \(e\) is a new constant symbol

Existential instantiation contd.

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 is satisfiable iff the old KB was satisfiable

Reduction to propositional inference

Suppose the KB contains just the following:

\[
\forall x \text{ King}(x) \land \text{Greedy}(x) \Rightarrow \text{Evil}(x)
\]

\(\text{King}(Johann)\)

\(\text{Greedy}(Johann)\)

\(\text{Brother}(Richard, Johann)\)

Instantiating the universal sentence in all possible ways, we have

\(\text{King}(Johann) \land \text{Greedy}(Johann) \Rightarrow \text{Evil}(Johann)\)

\(\text{King}(Richard) \land \text{Greedy}(Richard) \Rightarrow \text{Evil}(Richard)\)

\(\text{King}(Johann)\)

\(\text{Greedy}(Johann)\)

\(\text{Brother}(Richard, Johann)\)

The new KB is propositionalized: proposition symbols are

\(\text{King}(Johann), \text{Greedy}(Johann), \text{Evil}(Johann), \text{King}(Richard)\) etc.
Claim: a ground sentence is entailed by new KB if it is entailed by original KB.

Claims: every FOL KB can be propositionalized so as to preserve entailment.

Idea: propositionalize KB and query, apply resolution, return result

Problem: with function symbols, there are infinitely many ground terms, e.g., Father(Father(John))

Theorem: Herbrand (1930). If a sentence $\alpha$ is entailed by an FOL KB, it is entailed by a finite subset of the propositional KB.

Idea: For $n = 0$ to $\infty$ do
- create a propositional KB by instantiating with depth-$n$ terms
- see if $\alpha$ is entailed by this KB

Problem: works if $\alpha$ is not entailed. Instantiations of function symbols make $\alpha$ entailed.

Theorem: Turing (1936), Church (1936), entailment in FOL is semidecidable.

Problems with propositionalization

Propositionalization seems to generate lots of irrelevant sentences.

E.g., from

- $\forall x \ King(x) \land Greedy(x) \Rightarrow Evil(x)$
- $King(John)$
- $\forall y \ Greedy(y)$
- $Brother(Richard, John)$

It seems obvious that $Evil(John)$, but propositionalization produces lots of facts such as $Greedy(Richard)$ that are irrelevant.

With $p$-ary predicates and $n$ constants, there are $p \cdot n^p$ instantiations.

With function symbols, it gets much much worse!
Unification

We can get the inference immediately if we can find a substitution $\theta$ such that $\text{King}(x)$ and $\text{Greedy}(x)$ match $\text{King}(\text{John})$ and $\text{Greedy}(y)$

$\theta = \{x/\text{John}, y/\text{John}\}$ works

$\text{UNIFY}(\alpha, \beta) = \theta$ if $\alpha \theta = \beta \theta$

$p_1 \theta = \{x/\text{John}, y/\text{John}\}$

Standardizing apart eliminates overlap of variables, e.g., $\text{Knows}(z_{17}, OJ)$

Generalized Modus Ponens (GMP)

$p_1', p_2', \ldots, p_n', (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q) \theta$

where $p_i \theta = p_i \theta$ for all $i$

$p_1' \text{ is } \text{King}(\text{John})$

$p_2' \text{ is } \text{Greedy}(y)$

$\theta \text{ is } \{x/\text{John}, y/\text{John}\}$

$q \theta \text{ is } \text{Evil}(\text{John})$

GMP used with KB of definite clauses (exactly one positive literal)

All variables assumed universally quantified

Soundness of GMP

Need to show that

$p_1', \ldots, p_n', (p_1 \wedge \ldots \wedge p_n \Rightarrow q) \theta$

provided that $p_i \theta = p_i \theta$ for all $i$

Lemma: For any definite clause $p$, we have $p \models p \theta$ by UI

1. $(p_1 \wedge \ldots \wedge p_n \Rightarrow q) \models (p_1 \wedge \ldots \wedge p_n \Rightarrow q) \theta = (p_1 \theta \wedge \ldots \wedge p_n \theta \Rightarrow q \theta)$

2. $p_1', \ldots, p_n' \models p_1', \ldots, p_n' \models p_1 \theta \wedge \ldots \wedge p_n \theta$

3. From 1 and 2, $q \theta$ follows by ordinary Modus Ponens

Example knowledge base

The law says that it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American.

Prove that Col. West is a criminal

Example knowledge base contd.

... it is a crime for an American to sell weapons to hostile nations:

$\text{American}(x) \wedge \text{Weapon}(y) \wedge \text{Sells}(x, y, z) \wedge \text{Hostile}(z) \Rightarrow \text{Criminal}(x)$

Nono ... has some missiles

Example knowledge base contd.

... it is a crime for an American to sell weapons to hostile nations:

$\text{American}(x) \wedge \text{Weapon}(y) \wedge \text{Sells}(x, y, z) \wedge \text{Hostile}(z) \Rightarrow \text{Criminal}(x)$

Nono ... has some missiles

Example knowledge base contd.

... it is a crime for an American to sell weapons to hostile nations:

$\text{American}(x) \wedge \text{Weapon}(y) \wedge \text{Sells}(x, y, z) \wedge \text{Hostile}(z) \Rightarrow \text{Criminal}(x)$

Nono ... has some missiles
... it is a crime for an American to sell weapons to hostile nations:
\[ \text{American}(x) \land \text{Weapon}(y) \land \text{Sells}(x, y, z) \land \text{Hostile}(z) \Rightarrow \text{Criminal}(x) \]

\text{Nono} ... has some missiles, i.e., \( \exists x \text{ Owns}(\text{Nono}, x) \land \text{Missile}(x); \)
\[ \text{Owns}(\text{Nono}, M_1) \text{ and } \text{Missile}(M_1) \]
... all of its missiles were sold to it by Colonel West

\[ \forall x \text{ Missile}(x) \land \text{Owns}(\text{Nono}, x) \Rightarrow \text{Sells}(\text{West}, x, \text{Nono}) \]

Missiles are weapons:
\[ \text{Missile}(x) \Rightarrow \text{Weapon}(x) \]

An enemy of America counts as "hostile":
\[ \text{Enemy}(\text{Nono, America}) \]

---

**Example knowledge base contd.**

... it is a crime for an American to sell weapons to hostile nations:
\[ \text{American}(x) \land \text{Weapon}(y) \land \text{Sells}(x, y, z) \land \text{Hostile}(z) \Rightarrow \text{Criminal}(x) \]

\text{Nono} ... has some missiles, i.e., \( \exists x \text{ Owns}(\text{Nono}, x) \land \text{Missile}(x); \)
\[ \text{Owns}(\text{Nono}, M_1) \text{ and } \text{Missile}(M_1) \]
... all of its missiles were sold to it by Colonel West

\[ \forall x \text{ Missile}(x) \land \text{Owns}(\text{Nono}, x) \Rightarrow \text{Sells}(\text{West}, x, \text{Nono}) \]

Missiles are weapons:
\[ \text{Missile}(x) \Rightarrow \text{Weapon}(x) \]

An enemy of America counts as "hostile":

---

**Example knowledge base contd.**

... it is a crime for an American to sell weapons to hostile nations:
\[ \text{American}(x) \land \text{Weapon}(y) \land \text{Sells}(x, y, z) \land \text{Hostile}(z) \Rightarrow \text{Criminal}(x) \]

\text{Nono} ... has some missiles, i.e., \( \exists x \text{ Owns}(\text{Nono}, x) \land \text{Missile}(x); \)
\[ \text{Owns}(\text{Nono}, M_1) \text{ and } \text{Missile}(M_1) \]
... all of its missiles were sold to it by Colonel West

\[ \forall x \text{ Missile}(x) \land \text{Owns}(\text{Nono}, x) \Rightarrow \text{Sells}(\text{West}, x, \text{Nono}) \]

Missiles are weapons:
\[ \text{Missile}(x) \Rightarrow \text{Weapon}(x) \]

An enemy of America counts as "hostile":

---

**Forward chaining algorithm**

function FOL-FC-Ask(KB, p) returns a substitution or false
repeat until new is empty
    new = {} 
    for each sentence r in KB do
        if \( r \) is not a renaming of a sentence already in KB or new then do
            for each \( \theta \) such that \( (p_1 \land \ldots \land p_n) \Rightarrow q \) = STANDARDIZE-APART(p)
                for each \( \phi \) in KB, \( q' \) = SUBST(\( \theta \), q)
                if \( q' \) is not a renaming of a sentence already in KB or new then do
                    add \( q' \) to new
                    q = UNIFY(\( q', \theta \))
                    if \( \phi \) is not fail then return \( \phi \)
            add new to KB
    return false

---

**Forward chaining proof**

\[ \text{American(\text{West})} \quad \text{Missile(M1)} \quad \text{Owns(Nono,M1)} \quad \text{Enemy(Nono,America)} \]
### Properties of forward chaining

Sound and complete for first-order definite clauses
(proof similar to propositional proof)

**Datalog** = first-order definite clauses + no functions (e.g., crime KB)

FC terminates for Datalog in poly iterations: at most \( p \cdot n^k \) literals

May not terminate in general if \( \alpha \) is not entailed

This is unavoidable: entailment with definite clauses is semidecidable

### Backward chaining algorithm

**Function** FOL-BC-Ask\((KB, goals, \theta)\) returns a set of substitutions

**Inputs:**
- \( KB \), a knowledge base
- \( goals \), a list of conjuncts forming a query (\( \theta \) already applied)
- \( \theta \), the current substitution, initially the empty substitution (\( \{} \))

**Local variables:**
- \( answers \), a set of substitutions, initially empty

**If** goals is empty **then return** \( \{ \} \)

\( q' \) ← \( \text{Subst}(\theta, \text{First}(goals)) \)

**For each** sentence \( r \) in \( KB \)

where \( \text{Standardize-Apart}(r) = (p_1 \land \ldots \land p_n \implies q) \)

and \( q' \) ← \( \text{Unify}(q, q') \) succeeds

\( \text{new} \_\text{goals} \) ← \( \{p_1, \ldots, p_n, \text{Rest}(goals)\} \)

\( answers \) ← FOL-BC-Ask\((KB, new\_goals, \text{Compose}(\theta, \theta')) \) \( \cup \) \( answers \)

**Return** \( answers \)
**Backward chaining example**

![Diagram of backward chaining example]

**Properties of backward chaining**

Depth-first recursive proof search: space is linear in size of proof

- Incomplete due to infinite loops
  - fix by checking current goal against every goal on stack
- Inefficient due to repeated subgoals (both success and failure)
  - fix using caching of previous results (extra space!)
- Widely used (without improvements!) for logic programming

**Conversion to CNF**

Everyone who loves all animals is loved by someone:

\[ \forall x \ [ \forall y \ Animal(y) \Rightarrow \ Loves(x,y) ] \Rightarrow [ \exists y \ Loves(y,x) ] \]

1. Eliminate biconditionals and implications

   \[ \forall x \ [ \exists y \ Animal(y) \land \ Loves(x,y) ] \lor [ \exists y \ Loves(y,x) ] \]

2. Move \( \land \) inwards:

   \[ \forall x \ [ \exists y \ Animal(y) \land \ Loves(x,y) ] \lor [ \exists y \ Loves(y,x) ] \]

   \[ \forall x \ [ \exists y \ Animal(y) \land \ Loves(x,y) ] \lor [ \exists y \ Loves(y,x) ] \]

   \[ \forall x \ [ \exists y \ Animal(y) \land \ Loves(x,y) ] \lor [ \exists y \ Loves(y,x) ] \]

3. Standardize variables: each quantifier should use a different one

   \[ \forall x \ [ \exists y \ Animal(y) \land \ Loves(x,y) ] \lor [ \exists z \ Loves(z,x) ] \]

4. Skolemize: a more general form of existential instantiation.

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

   \[ \forall x \ [ Animal(F(x)) \land \ Loves(x,F(x)) ] \lor Loves(G(x),x) \]

5. Drop universal quantifiers:

   \[ Animal(F(x)) \land \ Loves(x,F(x)) ] \lor Loves(G(x),x) \]

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

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

**Resolution: brief summary**

Full first-order version:

\[ \ell_1 \lor \cdots \lor \ell_k \lor m_1 \lor \cdots \lor m_n \]

\[ (\ell_1 \lor \cdots \lor \ell_{k-1} \lor m_1 \lor \cdots \lor m_{n-1} \lor m_n) \theta \]

where \( UNIFY(\ell_i, \neg m_j) = \theta \).

For example,

\[ \neg Rich(x) \lor Unhappy(x) \]

\[ Rich(Ken) \]

with \( \theta = \{ x/Ken \} \)

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

**Conversion to CNF contd.**

3. Standardize variables: each quantifier should use a different one

4. Skolemize: a more general form of existential instantiation.

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

5. Drop universal quantifiers:

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

**Resolution proof: definite clauses**

![Resolution proof diagram]