Inference in FOL

AIMA Ch.9

Outline

- Reducing first-order inference to propositional inference
- Unification
- Generalized Modus Ponens
- Resolution

Inference in FOL

Given $\forall x\ King(x) \land Greedy(x) \Rightarrow Evil(x)$

One can infer

- $King(John) \land Greedy(John) \Rightarrow Evil(John)$
- $King(Richard) \land Greedy(Richard) \Rightarrow Evil(Richard)$
- $King(Father(John)) \land Greedy(Father(John)) \Rightarrow Evil(Father(John))$

Inference in FOL

- **Universal Instantiation** (in a ∀ rule, substitute all symbols)
- **Existential Instantiation** (in a ∃ rule, substitute one symbol, use the rule and discard)
- Skolem Constants is a new variable that represents a new inference

Universal instantiation (UI)

Substitute all symbols

$$\frac{\forall v \ \ \alpha}{\textsc{Subst}(\{v/g\}, \alpha)}$$

for any variable v and ground term g

Inference in FOL

KB contains:

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

Apply UI using *{x/John}* and *{x/Richard}*

- $King(John) \land Greedy(John) \Rightarrow Evil(John)$
- $King(Richard) \land Greedy(Richard) \Rightarrow Evil(Richard)$

And **discard the Universally quantified sentence**. We get KB to be propositions

Existential instantiation (EI)

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

$$\frac{\exists v \ \ \alpha}{\textsc{Subst}(\{v/k\}, \alpha)}$$

E.g., $\exists x \ \ Crown(x) \land OnHead(x, John)$ yields

$$Crown(C_1) \land OnHead(C_1, John)$$

provided C_1 is a new constant symbol, called a Skolem constant

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

Unification

- Finds substitutions that make different logical expressions look identical

We can get the inference immediately if we can find a substitution θ such that $King(x)$ and $Greedy(x)$ match $King(John)$ and $Greedy(y)$

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

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

p	q	θ
$Knows(John, x)$	$Knows(John, Jane)$	
$Knows(John, x)$	$Knows(y, OJ)$	
$Knows(John, x)$	$Knows(y, Mother(y))$	
$Knows(John, x)$	$Knows(x, OJ)$	

Unification

We can get the inference immediately if we can find a substitution θ such that $King(x)$ and $Greedy(x)$ match $King(John)$ and $Greedy(y)$

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

$\textsc{Unify}(\alpha, \beta) = \theta$ if $\alpha\theta = \beta\theta$

p	q	θ
$Knows(John, x)$	$Knows(John, Jane)$	$\{x/Jane\}$
$Knows(John, x)$	$Knows(y, OJ)$	$\{x/OJ, y/John\}$
$Knows(John, x)$	$Knows(y, Mother(y))$	$\{y/John, x/Mother(John)\}$
$Knows(John, x)$	$Knows(x, OJ)$	$fail$

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

Generalized Modus Ponens (GMP)

for atomic sentences $p_i, p_i^{'}$ and q, where there is a substitution θ such that $SUBST(\theta, p_i^{'}) = SUBST(\theta, p_i)$, for all i

$$\frac{p_1^{'}, p_2^{'}, \ldots, p_n^{'}, (p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q)}{SUBST(\theta, q)}$$

$$\begin{array}{ll}
p_1^{'} = King(John) & p_1 = King(x) \\
p_2^{'} = Greedy(y) & p_2 = Greedy(x) \\
\theta = \{x/John, y/John\} & q = Evil(x) \\
SUBST(\theta, q) & .
\end{array}$$

* Variables are assumed to be universally quantified.

Resolution: brief summary

Full first-order version:

$$\frac{\ell_1 \vee \cdots \vee \ell_k, \qquad m_1 \vee \cdots \vee m_n}{(\ell_1 \vee \cdots \vee \ell_{i-1} \vee \ell_{i+1} \vee \cdots \vee \ell_k \vee m_1 \vee \cdots \vee m_{j-1} \vee m_{j+1} \vee \cdots \vee m_n)\theta}$$

where $\text{UNIFY}(\ell_i, \neg m_j) = \theta$.

For example,

$$\frac{\begin{array}{l}\neg Rich(x) \vee Unhappy(x) \\ Rich(Ken)\end{array}}{Unhappy(Ken)}$$

with $\theta = \{x/Ken\}$

Apply resolution steps to $CNF(KB \wedge \neg\alpha)$; complete for FOL

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

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

2. Move $\neg$ inwards: $\neg\forall x, p\ \ \equiv \exists x\ \ \neg p$, $\ \neg\exists x, p\ \ \equiv \forall x\ \ \neg p$:

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

Conversion to CNF contd.

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

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

4. Skolemize: a more general form of existential instantiation.
   Each existential variable is replaced by a Skolem function
   of the enclosing universally quantified variables:

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

5. Drop universal quantifiers:

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

6. Distribute $\land$ over $\lor$:

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

Inference example

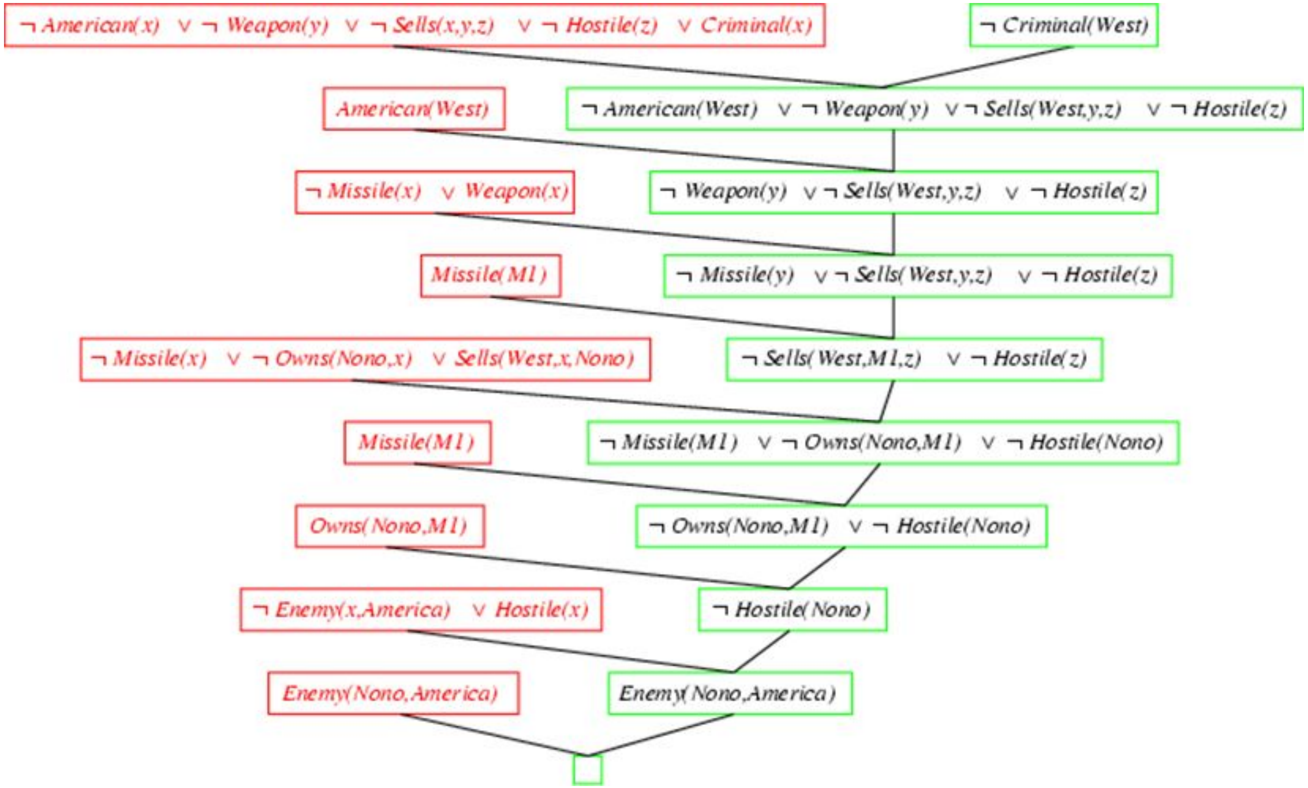
“The Law says that it is a crime for an American to sell weapons to hostile nations. The country Nono, and enemy of America, has some missiles , and all of its missiles were sold to it by Colonel West, who is American”

Prove that Colonel West is a Criminal

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

- R1: American(x) ∧ Weapon(y) ∧ Sells(x, y.z) ∧ Hostile(z) ⇒ Criminal(x)
- R2: Owns(Nono, M1) Nono has some missiles
- R3: Missile(M1)
- R4: Missile(x) ⇒ Weapon(x) # A missile is a weapon
- R5: Missile(x) ∧ Owns(Nono, x) ⇒ Sells(West, x, Nono) #All missiles sold by West
- R6: Enemy(x, America) ⇒ Hostile(x) Enemies of America are hostile
- R7: American(West) West is american
- R8: Enemy(Nono, America)

Resolution proof: definite clauses



FOL in NLTK

http://www.nltk.org/howto/resolution.html

```
>>> from nltk.inference.resolution import *
>>> from nltk.sem import logic
>>> from nltk.sem.logic import *
>>> logic._counter._value = 0
>>> read_expr = logic.Expression.fromstring

>>> p1 = read_expr('all x.(man(x) -> mortal(x))')
>>> p2 = read_expr('man(Socrates)')
>>> c = read_expr('mortal(Socrates)')
>>> ResolutionProverCommand(c, [p1,p2]).prove()
True
>>> print(tp.proof())
[1] {-mortal(Socrates)}     A
[2] {-man(z2), mortal(z2)}  A
[3] {man(Socrates)}         A
[4] {-man(Socrates)}        (1, 2)
[5] {mortal(Socrates)}      (2, 3)
[6] {}
```

Q&A using FOL

```
>>> p1 = read_expr('father_of(art,john)')
>>> p2 = read_expr('father_of(bob,kim)')
>>> p3 = read_expr('all x.all y.(father_of(x,y) -> parent_of(x,y))')
>>> c = read_expr('all x.(parent_of(x,john) -> ANSWER(x))')
>>> logic._counter._value = 0
>>> tp = ResolutionProverCommand(None, [p1,p2,p3,c])
>>> sorted(tp.find_answers())
[<ConstantExpression art>]
>>> print(tp.proof())
[1] {father_of(art,john)}                    A
[2] {father_of(bob,kim)}                     A
[3] {-father_of(z3,z4), parent_of(z3,z4)}  A
[4] {-parent_of(z6,john), ANSWER(z6)}      A
[5] {parent_of(art,john)}                  (1, 3)
[6] {parent_of(bob,kim)}                   (2, 3)
[7] {ANSWER(z6), -father_of(z6,john)}      (3, 4)
[8] {ANSWER(art)}                          (1, 7)
[9] {ANSWER(art)}                          (4, 5)
```

Summary

- UI and EI for inference in FOL
- Unification finds substitution that makes two different logical sentences identical