CS480 – Artificial Intelligence Fall 2015

Topic: Inference in FOL
Chapter: 9
Date: 10/19

Mustafa Bilgic

http://www.cs.iit.edu/~mbilgic

What's the difference?

- Can we use propositional logic inference (resolution, forward chaining, etc) techniques directly?
- No, because FOL has
  - Variables
  - Functions
  - Quantifiers

PROPOSITIONALIZE?

- Can we convert FOL into propositional logic as a pre-processing step? Because if we can, then we can use the same inference techniques…
- Not directly and easily, because
  - Functions can be applied infinitely many times
    - Mother(Mother(Mother(…
  - Variables have to be replaced with all possible assignments from the vocabulary

Godel's Completeness Theorem

- FOL entailment is only semi-decidable
- If a sentence is entailed
  - There is a procedure that will find it
- Else
  - There is no guarantee that the procedure will halt

Algorithms we discussed for PL

1. Model checking
2. Logical equivalence rules
3. Proof-by-contradiction
   - **Resolution**
4. Forward chaining
5. Backward chaining

ALGORITHMS WE WILL DISCUSS FOR FOL

1. Forward chaining
2. **Resolution**

VARIABLES

- To be able to reason with a FOL KB, we need procedures to deal with the variables
- We will discuss two procedures:
  - *Subst*(θ, P)
  - *Unify*(P, Q)

Subst(θ, P)

- θ specifies a substitution for a variable in P
- The result is P with the variable substituted with the specified constant/variable
- Examples
  - $Subst(\{x/\text{John}\}, P(x))$
    - P(John)
  - $Subst(\{x/\text{Mary}\}, P(x) \vee Q(x,y))$
    - $P(\text{Mary}) \vee Q(\text{Mary}, y)$
  - $Subst(\{x/\text{z}\}, P(x) \vee Q(x,y))$
    - $P(z) \vee Q(z,y)$

Unify(P, Q)

- **Unification:** Finding substitutions that make different logical expressions look identical
- *Unify* takes two sentences and returns a substitution if one exists
- $Unify(P, Q) = \theta$ where
  - $Subst(\theta, P) = Subst(\theta, Q)$

UNIFY(P, Q) EXAMPLES

- *Unify*(Knows(John, x), Knows(John, Jane)) =
  - {x/Jane}
- *Unify*(Knows(John, x), Knows(y, Bill)) =
  - {x/Bill, y/John}
- *Unify*(Knows(John, x), Knows(y, Mother(y)) =
  - {y/John, x/Mother(John)}
- *Unify*(Knows(John, x), Knows(x, Elizabeth)) =
  - *fail*

Most General Unifier

- $Unify(\text{Knows}(\text{John}, x), \text{Knows}(y, z)) =$
  - $\{y/\text{John}, x/z\}$
  - $\{y/\text{John}, x/\text{John}, z/\text{John}\}$
  - …
- Most general unifier: $\{y/\text{John}, x/z\}$

Generalized Modes Ponens

- $\forall x$ King(x) $\wedge$ Greedy(x) $\Rightarrow$ Evil(x)
- King(John)
- Greedy(John)
- What can you conclude and why?

Generalized Modes Ponens

- For atomic sentences p_i, p_i', and q, if there is a substitution θ such that $Subst(\theta, p_i') = Subst(\theta, p_i)$ for all i,
- Given
  - $p_1', p_2', \ldots, p_n'$ and
  - $p_1 \wedge p_2 \wedge \ldots \wedge p_n \Rightarrow q$
- Conclude
  - $Subst(\theta, q)$

Generalized Modes Ponens

- $\forall x$ King(x) $\wedge$ Greedy(x) $\Rightarrow$ Evil(x)

- King(John)
- Greedy(John)

- p_1' = King(x), p_1 = King(John),
- p_2' = Greedy(x), p_2 = Greedy(John)
- $\theta = \{x/\text{John}\}$

- Conclude
  - $Subst(\theta, q) = Subst(\{x/\text{John}\}, \text{Evil}(x)) = \text{Evil(John)}$

Forward Chaining

1. Use Generalized Modes Ponens to add new facts
2. Stop when proved or no facts can be added

- The KB has to consist of *definite clauses*
  - Definite clause: disjunction of literals of which exactly one is positive.
  - *How are they related to Horn clauses?*

- Sound and complete for definite clause KBs
- If functions are included
  - semi-decidable

Resolution

- Need to convert into CNF
- Procedure is very similar to the propositional case,
  - Except quantifiers have to be handled

CNF Conversion

1. Eliminate biconditionals ($\Leftrightarrow$)
2. Eliminate implications ($\Rightarrow$)
3. Move $\neg$ inwards
4. Standardize variables
5. Skolemize ← *What's this?*
6. Drop universal qualifiers
7. Distribute $\vee$ over $\wedge$

$\forall x\, \exists y\, \forall z\; P(x, y, z)$

$\forall x\, \forall z\; P(x, F(x), z)$

SKOLEMIZE

- Drop ∃ by replacing the variable with either
  - A new constant, something not in your current domain
  - Or a new function
- Examples
  - $\exists y$ Crown(y) ∧ OnHead(y, John)
    - Crown(A) ∧ OnHead(A, John)
  - $\exists y\ \forall x$ Loves(x,y)
    - $\forall x$ Loves(x, B)
  - $\forall x\ \exists y$ Loves(x, y)
    - $\forall x$ Loves(x, C) ---- *Incorrect! Why?*
    - $\forall x$ Loves(x, F(x))

$\forall x\, \forall y\, \exists z\; P(x, y, z)$

$\forall x\, \forall y\; P(x, y, F(x, y))$

CNF Example

- "Everyone who loves all animals is loved by someone"
- $\forall x\, [\forall y\, \text{Animal}(y) \Rightarrow \text{Loves}(x, y)] \Rightarrow [\exists y\, \text{Loves}(y, x)]$
- **Eliminate implications**
- $\forall x\, [\neg\forall y\, \neg\text{Animal}(y) \vee \text{Loves}(x, y)] \vee [\exists y\, \text{Loves}(y, x)]$
- **Move $\neg$ inwards**
- $\forall x\, [\exists y\, \text{Animal}(y) \wedge \neg\text{Loves}(x, y)] \vee [\exists y\, \text{Loves}(y, x)]$
- **Standardize variables**
- $\forall x\, [\exists y\, \text{Animal}(y) \wedge \neg\text{Loves}(x, y)] \vee [\exists z\, \text{Loves}(z, x)]$

CNF Example

- $\forall x\, [\exists y\, \text{Animal}(y) \wedge \neg\text{Loves}(x, y)] \vee [\exists z\, \text{Loves}(z, x)]$
- **Skolemize:** Eliminate $\exists$ by replacing it with a constant or a function
- y=A, z = B: $\forall x\, [\, \text{Animal}(A) \wedge \neg\text{Loves}(x, A)] \vee [\, \text{Loves}(B, x)]$
  - Incorrect! Why?
- $\forall x\, [\, \text{Animal}(F(x)) \wedge \neg\text{Loves}(x, F(x))] \vee [\, \text{Loves}(G(x), x)]$
  - What are F(x) and G(x)?
- **Drop universal quantifiers**
- $[\, \text{Animal}(F(x)) \wedge \neg\text{Loves}(x, F(x))] \vee [\, \text{Loves}(G(x), x)]$
- **Distribute $\vee$ over $\wedge$**
- $[\, \text{Animal}(F(x)) \vee \text{Loves}(G(x), x)] \wedge [\neg\text{Loves}(x, F(x))] \vee \text{Loves}(G(x), x)]$

Resolution Proof

- Very much the same process we did for propositional logic
- Use unification to substitute variables
- Example on page 348 and 349

Resolution example

1. Everyone who loves all animals is loved by someone.
2. Anyone who kills an animal is loved by no one.
3. Jack loves all animals.
4. Either Jack or Curiosity killed the cat, who is named Tuna.
5. Tuna is a cat.
6. All cats are animals.
7. Did Curiosity kill Tuna?

First, convert these English sentences to FOL using Animal(.), Loves(.,.), Kills(.,.), and Cat(.) predicates. Then, convert it to CNF. Finally, run resolution.

Curiosity CNF (Page 349)

A. $\forall x \; [\forall y \; Animal(y) \Rightarrow Loves(x, y)] \Rightarrow [\exists y \; Loves(y, x)]$

B. $\forall x \; [\exists z \; Animal(z) \wedge Kills(x, z)] \Rightarrow [\forall y \; \neg Loves(y, x)]$

C. $\forall x \; Animal(x) \Rightarrow Loves(Jack, x)$

D. $Kills(Jack, Tuna) \vee Kills(Curiosity, Tuna)$

E. $Cat(Tuna)$

F. $\forall x \; Cat(x) \Rightarrow Animal(x)$

¬G. $\neg Kills(Curiosity, Tuna)$

Now we apply the conversion procedure to convert each sentence to CNF:

A1. $Animal(F(x)) \vee Loves(G(x), x)$

A2. $\neg Loves(x, F(x)) \vee Loves(G(x), x)$

B. $\neg Loves(y, x) \vee \neg Animal(z) \vee \neg Kills(x, z)$

C. $\neg Animal(x) \vee Loves(Jack, x)$

D. $Kills(Jack, Tuna) \vee Kills(Curiosity, Tuna)$

E. $Cat(Tuna)$

F. $\neg Cat(x) \vee Animal(x)$

¬G. $\neg Kills(Curiosity, Tuna)$

CURIOSITY RESOLUTION