Introduction to Artificial Intelligence
COMP 3501 / COMP 4704-4
Lecture 9: Inference in FOL

Prof. Nathan Sturtevant
JGH 318

More FOL examples

- The only two certainties in life are death and taxes.
  - certainty(life) ∧ certainty(taxes) ∧ $\forall_x$ certainty(x) ⇒ (x=life) ∨ (x=taxes)
- The coldest winter I ever spent was a summer in San Francisco.
  - $\exists_y$ summer(y) ∧ in(y, San Francisco) ∧ $\forall_x$ winter(x) ⇒ colder(y, x)
- You can fool some of the people all of the time, and all of the people some of the time, but you cannot fool all of the people all of the time.
- ($\exists_{x,y}$ person(x) ∧ [$\forall_t$ time(t) ⇒ fool(me, x, t)]) ∧
($\forall_x$ person(x) ⇒ [$\exists_t$ time(t) ∧ fool(me, x, t)]) ∧
(¬$\forall_{x,t}$ person(x) ∧ time(t) ⇒ fool(me, x, t))

Using FOL in practice (Knowledge Engineering)

- Identify task and queries
- Assemble basic knowledge
- Define a vocabulary
- Encode the domain
- Encode problem instance(s)
- Query the knowledge
- Debug broken behavior

FOL queries / assertions

- Add sentences to KB using Tell
  - Tell(KB, King(John))
  - Tell(KB, Person(Richard))
  - Tell(KB, $\forall_x$ King(x) ⇒ Person(x))
- Query KB using ASK
  - Ask(KB, Person(John)) → true
  - Ask(KB, $\exists_x$ Person(x)) → true
  - AskVars(KB, Person(x)) → {x/John}, {x/Richard}

Inference in FOL:

- Convert to Prop. Logic
  - Perform all possible substitutions so that you are grounded in propositional logic
- Reason directly in FOL

Reasoning in FOL

- Definite clause:
  - Implication with all positive literals and antecedent is a conjunction of positive literals
  - (or disjunction of all negative literals except one)
- Allow the use of (Generalized) Modus Ponens directly

Inference in FOL

- Will cover one key idea:
  - Reducing FOL to CNF
  - Performing resolution on CNF
- But logic programming & forward chaining all very useful to understand

FOL Reduction to CNF

- Rules for reducing FOL to CNF
  - Eliminate implications, iff
  - Move ¬ inward
  - Standardize variables
  - Skolemize
  - Drop universal quantifiers
  - Distribute ∨ over ∧

Skolemization

- Variables in CNF form are universally quantified
- For existential quantification we need constants
  - $\exists_x$ likes(x, chocolate) → likes(X_1, chocolate)
  - X_1 is a skolem constant
- If we have universal quantifiers, then we need a skolem function which is relative to universal quantifier
  - $\forall_x$ King(x) ∧ [$\exists_y$ Person(y) ∧ loves(x, y)]
  - $\forall_x$ King(x) ∧ [Person(Y_1) ∧ loves(x, Y_1)] [no]
  - $\forall_x$ King(x) ∧ [Person(Y_1(x)) ∧ loves(x, Y_1(x))] [yes]

Unification

- Unification is the process of matching expressions/ variables
  - 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*

Resolution

- Once we have reduced to CNF we can apply resolution as normal
  - Recall:
    - R1: $dog_{fred} \Rightarrow likesbones_{fred}$; R2: dog_{fred}
    - R3: $\neg dog_{fred} \vee likesbones_{fred}$
    - Prove: $likesbones_{fred}$
  - **Can we re-write and prove this in FOL with resolution?**

Another Example

- Figure 9.12 from book