CS 3011: Artificial Intelligence

Knowledge-Based Agents and Logic

Instructors: Dr. Durgesh Singh

CSE Discipline, PDPM IIITDM, Jabalpur -482005

First Order Logic

Limitation of Propositional Logic

P: All dogs are faithful

Q: Tommy is a dog

C: Tommy is faithful

We cannot infer this in propositional logic

- Problem of infinite model
- propositional logic can deal with only a finite number of propositions
- for example, If there are only dogs: Tommy, Jimmy, Laika then I could have written such that T: Tommy is faithful, J: Jimmy is faithful, L : Laika is faithful then all dogs are faithful will be $T \wedge J \wedge L$.

Example

- Propositional logic lacks the expressive power to concisely describe an environment with many objects.
- For example, we were forced to write a separate rule about breezes and pits for each square, such as

$$R_2 : \quad B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1}) .$$
$$R_3 : \quad B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1}) .$$

- In English, on the other hand, it seems easy enough to say, once and for all, "Squares adjacent to pits are breezy.“
- In First-order logic,

$$\forall\, s \;\; Breezy(s) \;\Leftrightarrow\; \exists\, r \;\; Adjacent(r, s) \wedge Pit(r)$$

First-Order Logic

- First-Order logic or Predicate Logic is a generalization of Propositional Logic and allows us to express and infer arguments in infinite models like
  - All man are mortal
  - Some birds cannot fly

Syntax of First-order Logic

$$
\begin{array}{rcl}
\textit{Sentence} & \rightarrow & \textit{AtomicSentence} \mid \textit{ComplexSentence} \\
\textit{AtomicSentence} & \rightarrow & \textit{Predicate} \mid \textit{Predicate}(\textit{Term}, \ldots) \mid \textit{Term} = \textit{Term} \\
\textit{ComplexSentence} & \rightarrow & (\ \textit{Sentence}\ ) \\
& \mid & \neg\ \textit{Sentence} \\
& \mid & \textit{Sentence} \land \textit{Sentence} \\
& \mid & \textit{Sentence} \lor \textit{Sentence} \\
& \mid & \textit{Sentence} \Rightarrow \textit{Sentence} \\
& \mid & \textit{Sentence} \Leftrightarrow \textit{Sentence} \\
& \mid & \textit{Quantifier}\ \textit{Variable}, \ldots\ \textit{Sentence} \\
\\
\textit{Term} & \rightarrow & \textit{Function}(\textit{Term}, \ldots) \\
& \mid & \textit{Constant} \\
& \mid & \textit{Variable} \\
\\
\textit{Quantifier} & \rightarrow & \forall \mid \exists \\
\textit{Constant} & \rightarrow & A \mid X_1 \mid \textit{John} \mid \cdots \\
\textit{Variable} & \rightarrow & a \mid x \mid s \mid \cdots \\
\textit{Predicate} & \rightarrow & \textit{True} \mid \textit{False} \mid \textit{After} \mid \textit{Loves} \mid \textit{Raining} \mid \cdots \\
\textit{Function} & \rightarrow & \textit{Mother} \mid \textit{LeftLeg} \mid \cdots \\
\textsc{Operator Precedence} & : & \neg, =, \land, \lor, \Rightarrow, \Leftrightarrow
\end{array}
$$

First-order logic

- First-order logic (FOL) models the world in terms of
  - **Objects (term),** which are things with individual identities
  - **Properties** of objects that distinguish them from other objects
  - **Relations** that hold among sets of objects
  - **Functions,** which are a subset of relations where there is only one “value” for any given “input”
- Examples:
  - Objects: Students, lectures, companies, cars ...
  - Relations: bigger-than, outside, part-of, has-color, occurs-after, owns, visits, ...
  - Properties: blue, even, large, ...
  - Functions: father-of, best-friend, one-more-than ...

User provides

- **Constant symbols,** which represent individuals in the world
  - Tom
  - 3
  - Green
- **Function symbols,** which map individuals to individuals
  - father-of(Tom) = John
  - color-of(Sky) = Blue
- **Predicate symbols,** which map individuals to truth values
  - greater(5,3); green(Grass) ; color(Grass, Green)

FOL Provides

- **Variable symbols**
  - E.g., x, y, foo
- **Connectives**
  - Same as in PL: not ($\neg$), and ($\wedge$), or ($\vee$), implies ($\rightarrow$), if and only if (biconditional $\leftrightarrow$)
- **Quantifiers**
  - Universal $\forall \mathbf{x}$
  - Existential $\exists \mathbf{x}$

Sentences are built from terms and atoms

- A **term** (denoting a real-world individual) is a constant symbol, a variable symbol, or an n-place function of n terms.

  x and $f(x_1, ..., x_n)$ are terms, where each x_i is a term.

- A **complex sentence** is formed from atomic sentences connected by the logical connectives.

- A **quantified sentence** adds quantifiers $\forall$ and $\exists$

  $(\forall x)P(x,y)$ has x bound as a universally quantified variable, but y is free

Examples

- If x is a man then x is mortal

$$man(x) \rightarrow mortal(x)$$

- If n is a natural number, then n is either even or odd

$$Natural(n) \rightarrow even(n) \lor odd(n)$$

- All dogs are faithful

$$\forall x\ (dog(x) \rightarrow faithful(x))$$

- All birds cannot fly

$$\exists\ x\ (bird\ (x) \land \neg fly\ (x))$$

Quantifiers

- **Universal quantification**
  - $(\forall x)P(x)$ means that P holds for **all** values of x in the domain associated with that variable

    E.g., $\forall x\ (dog(x) \rightarrow faithful(x))$
- **Existential quantification**
  - $(\exists\ x)P(x)$ means that P holds for **some** value of x in the domain associated with that variable

    E.g., $(\exists\ x)\ (bird\ (x) \land \neg fly\ (x))$

Quantifiers

- Universal quantifiers are often used with “implies” to form “rules”:

  $(\forall x)$ (student(x) $\rightarrow$ smart(x)) means “All students are smart”

- Existential quantifiers are usually used with “and” to specify a list of properties about an individual:

  $(\exists x)$ (student(x) $\wedge$ smart(x))means “There is a student who is smart”

Quantifier Scope

- Switching the order of universal quantifiers *does not* change the meaning:
  - $(\forall x)(\forall y)P(x,y) \leftrightarrow (\forall y)(\forall x)\ P(x,y)$
- Similarly, you can switch the order of existential quantifiers:
  - $(\exists x)(\exists y)P(x,y) \leftrightarrow (\exists y)(\exists x)\ P(x,y)$
- Switching the order of universal and existential *does* change meaning:
  - There is a person who likes everyone in the world: $(\exists\ x)(\forall\ y)$ likes(x,y)
  - Everyone in the world is liked by at least one person: $(\forall\ y)(\exists\ x)$ likes(x,y)

Connections between All and Exists

We can relate sentences involving $\forall$ and $\exists$ using De Morgan's laws:

$$(\forall x)\ \neg P(x) \leftrightarrow \neg(\exists x)\ P(x)$$

$$\neg(\forall x)\ P(x) \leftrightarrow (\exists x)\ \neg P(x)$$

$$(\forall x)\ P(x) \leftrightarrow \neg\ (\exists x)\ \neg P(x)$$

$$(\exists x)\ P(x) \leftrightarrow \neg(\forall x)\ \neg P(x)$$

Inference in First Order Logic

Inference Rules in FOL

- **Substitution:** SUBST(θ,α) denote the result of applying the substitution θ to the sentence α.

- **Inference rules for quantifier:**
  - **Universal Elimination**
  - **Existential Elimination/ Skolemization**
  - **Existential Introduction**

Universal Elimination/ Instantiation

- we can infer any sentence P(c) by substituting a ground term c (a constant within domain x) from **∀ x P(x).**

e.g., ∀ x Likes(x, apple)

substituting x by Tom gives

Likes(Tom, apple)

Existential Elimination/ Instantiation

- Replace an existentially quantified variable with a single new constant symbol.
- The constant symbol that does not appear elsewhere in the knowledge base.
  - The new name is called a **Skolem constant**.

e.g., $\exists$ x Likes(x, apple)

substituting x by Person gives

Likes(Person, apple)

Existential introduction

Likes(Tom, apple)

Can be written as

$\exists$x Likes(x, apple)

Unification

- Unification is the process of finding a substitution that makes two atomic sentences identical

  e.g.,

  Unify(Prime(7), Prime(x)) = (x/7)

  Unify(Divides(49,x), Divides(y,7)) = (x/7, y/49)

Example Knowledge Base (EKB)

- If a perfect square is divisible by a prime number, then it is also divisible by square of that prime number.
- Every perfect square is divisible by some prime.
- 36 is a perfect square.
- Does there exist a prime q such that square of q divides 36?

[Use inference rules]

Resolution

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

Example: 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

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 $\lor$:

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

Example

- All people who are graduating are happy
- All happy people are smiling
- Some one is graduating

Conclusion: some one is smiling

[Use Resolution algorithm]