# Knowledge Representation and Inference

IT426: Artificial Intelligence

Second Semester 1444

Information Technology Department

# Propositional Logic (PL)

- *Propositional logic pros:*
  - \+ Propositional logic is **declarative**: Knowledge base and inference are **separated**
  - \+ Propositional logic is **compositional**: meaning of $B_{1,1} \lor P_{1,2}$ is derived from meaning of $B_{1,1}$ and of $P_{1,2}$
  - \+ Meaning in propositional logic is **context-independent** unlike natural language, where meaning depends on context

- *Propositional logic cons:*
  - Propositional logic has **limited expressive power** unlike natural language
  - e.g. cannot say
  - "pits cause breezes in adjacent squares“

    (except by writing one sentence for each square)

**Question** : How can we write one sentence only that can be applied on a group of objects?

# First-order Logic (FOL)

- Is an extension to propositional logic.
- First-order logic (like natural language) assumes the following things in the world:
  - Facts (like propositional logic )
  - Objects: A, B, people, numbers, colors, wars, theories, squares, pits, wumpus, ......
  - Relations: It can be unary relation such as: red, round, is adjacent, or n-any relation such as: the sister of, brother of, has color, comes between evaluates to true or false
  - Function: Father of, best friend, third inning of, end of, ...... gives another object
- Can express the following:
  - Squares neighboring the Wumpus are smelly
  - Squares neighboring a pit are breezy

# Sentences

- Any sentence can be thought of as :
  - Objects
  - Relation
  - Property
  - Function
- “One plus two equals three.”
- “Squares neighboring the Wumpus are smelly.”
- “Evil King John ruled England in 1200.”
- More examples: text book ch8

# FOL Elements

| | |
|---|---|
| **Constant** | 1, 2, A, John, Mumbai, cat,.... |
| **Variables** | x, y, z, a, b,.... |
| **Predicates** | Brother, Father, >,.... |
| **Function** | sqrt, LeftLegOf, .... |
| **Connectives** | $\wedge$, $\vee$, $\neg$, $\Rightarrow$, $\Leftrightarrow$ |
| **Equality** | == |
| **Quantifier** | $\forall$, $\exists$ |

# First order logic (FOL)

- **Examples of things we can say:**
  - All men are mortal:
    - $\forall x \; Man(x) \Rightarrow Mortal(x)$
  - Everybody loves somebody
    - $\forall x \; \exists y \; Loves(x, y)$
  - The meaning of the word “above”
    - $\forall x \; \forall y \; above(x,y) \Leftrightarrow (on(x,y) \lor \exists z \; (on(x,z) \land above(z,y))$

# Syntax Of FOL

- *User defines* these primitives:

1. **Constant symbols** (i.e., the "individuals" in the world)
   e.g., Mary, 3, apple

2. **Predicate/relation symbols** (mapping from individuals to truth values)
   e.g., greater(5,3), green(apple), color(apple, Green)

3. **Function symbols** (mapping individuals to individuals)
   e.g., fatherOf(Mary) = John, colorOf(Sky) = Blue

# Syntax of FOL: Constant Symbols

- **Each constant symbol** names **exactly one object** in a universe of discourse, but:
  - **Not all** objects have symbol names;
  - **Some objects** have **several** symbol names.
- Usually denoted with ***upper-case* first letter.**
  - e.g. Wumpus, Ali.

# Syntax of FOL: Relation (Predicate) Symbols

- **A predicate symbol** is used to represent a ***relation*** in a universe of discourse.
- The sentence: ***Relation(Term1, Term2,…)***
  - is either **TRUE** or **FALSE** depending on whether Relation holds of Term1, Term2,…
- To write "**Malek wrote Muata**" in a universe of discourse of names and written works:
  - **Wrote(Malek, Muata)** → This sentence is true in the intended interpretation.
- Another example:

  \- A proposition in predicate logic: **Taking(IT426, Sarah)**

# Syntax Of FOL: Function Symbols

- Functions refer to objects.
- It gives us a powerful way to refer to objects **without** using a constant symbol to name them.
  - **Father(Ali)** → Refers to the father of Ali
  - **Father(x)** → Refers to the father of variable x

# Syntax Of FOL: Variables

- Used to represent objects or properties of objects **without explicitly naming** the object.
- Usually **lower case.**
- For example:
  - x
  - father
  - square

# Syntax Of FOL: Quantifiers

- Quantifiers: Universal **(∀)** and Existential **(∃)**
- Allow us to **express properties of collections of objects** instead of enumerating objects by name
  - **Universal:** “for all”:
    - ∀ <variables> <sentence>
    - $\forall x\ At(x,KSU) \Rightarrow Smart(x)$
  - **Existential**: “there exists”
    - ∃ <variables> <sentence>
    - $\exists\ x\ At(x, KSU) \wedge Smart(x)$

# Syntax Of FOL: Quantifiers

**(∀)**

- ⇒ is the main connective with **(∀)**
- $\forall x$ At(x,KSU) ⇒ Smart(x)
- “everyone that is at KSU is smart”
- Read as: For all x, For each x ,For every x.

**(∃)**

- ∧ is the main connective with **(∃)**
- $\exists$ x At(x, KSU) ∧ Smart(x)
- “there are students at KSU that are smart”
- Read as: There exists a x, For some x, For at least one x

# Syntax of FOL : Properties of Quantifiers

- $\forall x \forall y$ is the **same** as $\forall y \forall x$
- $\exists x \exists y$ is the **same** as $\exists y \exists x$
- $\exists x \forall y$ is **not the same** as $\forall y \exists x$:
  - $\exists x \forall y$ Loves(x,y)
    **“There is a person who loves everyone in the world”**
  - $\forall y \exists x$ Loves(x,y)
    **“Everyone in the world is loved by at least one person”**
- **Quantifier duality:** each can be expressed using the other
  - $\forall x$ Likes(x,IceCream) $\equiv \neg \exists x \neg$Likes(x,IceCream)
  - $\exists x$ Likes(x,Broccoli) $\equiv \neg \forall x \neg$Likes(x,Broccoli)

# Syntax of FOL: Atomic sentence

- Term is a logical expression that refers to an object
  - constant.
  - Variable
  - function ($term_1$,…,$term_n$). i.e. refer to an object using a function

- Atomic sentence = Predicate and Objects (terms)
  - predicate ($term_1$,…,$term_n$).
  - $term_1 = term_2$
  - E.g: Ginger is a cat: => cat (Ginger)

# Syntax of FOL: Complex Sentence

- Complex sentences are made from atomic sentences using **connectives** and by applying **quantifiers.**
- **Examples:**
  - *Sibling(Ali,Mohamed)* $\Rightarrow$ *Sibling(Mohamed,Ali)*
  - *Greater(1,2)* $\vee$ *Less-or-equal(1,2)*
  - $\forall$ x,y *Sibling(x,y)* $\Rightarrow$ *Sibling(y,x)*

# Syntax of FOL

$$
\begin{aligned}
\textit{Sentence} \rightarrow\ & \textit{Atomic Sentence} \\
& \mid (\textit{Sentence connective Sentence}) \\
& \mid \textit{Quantifier variable,}\ldots \textit{Sentence} \\
& \mid \neg\, \textit{Sentence}
\end{aligned}
$$

$$
\begin{aligned}
&\textit{Atomic Sentence} \rightarrow \textit{Predicate}\ (\textit{Term},\ldots) \mid \textit{Term}=\textit{Term} \\
&\textit{Term} \rightarrow \textit{Function}(\textit{Term},\ldots) \mid \textit{Constant} \mid \textit{variable}
\end{aligned}
$$

$$
\begin{aligned}
&\textit{Connective} \rightarrow \Leftrightarrow \mid \wedge \mid \vee \mid \Rightarrow \\
&\textit{Quantifier} \rightarrow \forall \mid \exists \\
&\textit{Constant} \rightarrow \mathrm{A} \mid \mathrm{X}_1\ldots \\
&\textit{Variable} \rightarrow \mathrm{a} \mid \mathrm{x} \mid \mathrm{s} \mid \ldots \\
&\textit{Predicate} \rightarrow \mathrm{Before} \mid \mathrm{hascolor} \mid \ldots. \\
&\textit{Function} \rightarrow \mathrm{Mother} \mid \mathrm{Leftleg} \mid \ldots
\end{aligned}
$$

# Examples (Text Book Ch8)

- Male and female are disjoint categories:
  - $\forall x\ Male(x) \Leftrightarrow \neg Female(x)$ .
- One's mother is one's female parent:
  - $\forall m,c\ Mother(c)=m \Leftrightarrow Female(m) \wedge Parent(m,c)$.
- Parent and child are inverse relations:
  - $\forall p,c\ Parent(p,c) \Leftrightarrow Child(c,p)$ .

# Recall

- Knowledge base is a set of sentences. Each sentence is expressed in a language called a knowledge representation language.
- A sentence represents some **assertion** about the world.
- Entailment $\alpha \models \beta$ if and only if **in every model in which α is true, β is also true**
- Inference is the process of **deriving** new sentences from old ones **KB** $\vdash_i$ **α**
- Sound inference algorithms derive **only** sentences that are entailed.
- Complete inference algorithms derive **all** sentences that are entailed.
- Model checking Enumerates **all possible models** to check that α is true in all models in which KB is true.
- Inference rules (e.g. modus ponens, resolution rule) are patterns of sound inference that can be **used to find proofs**
- Proof a sequence of **applications of inference rules**.
- Resolution is a sound and complete method of inference
- Forward chaining and backward chaining are natural **reasoning algorithms** for **knowledge bases in Horn form.**
- Horn clause is a disjunction of literals of which at **most one is positive.**

# Inference in FOL

Purpose of inference: KB $\models \alpha$ ?

Inference in FOL can be performed by:

1. Reducing FOL to PL and then apply PL inference

2. Use inference rules of FOL
   - Generalized Modus Ponens
   - Resolution
   - Forward and Backward chaining

# Inference in FOL: From FOL to PL

- First order inference can be done by converting the knowledge base to PL and using propositional inference.
- Two questions:
  - How to convert universal quantifiers ($\forall$)?
    - Replace variable by ground term.
  - How to convert existential quantifiers ($\exists$)?
    - Skolemization.

# Inference in FOL: Substitution

- Given a sentence α and **binding list σ**, the result of applying the **substitution** σ to α is denoted by Subst(σ, α).

Example:

$\sigma = \{x/Ali,\ y/Fatima\} \quad \alpha = Likes(x,y)$

Subst**(**{x/Ali, y/Fatima}**,** Likes(x,y)**)** = Likes(Ali, Fatima)

# Inference in FOL: Universal instantiation (UI)

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

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

- for any variable v and ground term g

  e.g., $\forall x\ King(x) \wedge Greedy(x) \Rightarrow Evil(x)$ yields:

  King(John) ∧ Greedy(John) ⇒ Evil(John)

  King(Richard) ∧ Greedy(Richard) ⇒ Evil(Richard)

  King(Father(John)) ∧ Greedy(Father(John)) ⇒ Evil(Father(John))

- UI can be applied several times to add new sentences

# Inference in FOL: Existential instantiation (EI)

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

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

- E.g., $\exists\, x\ Crown(x) \wedge OnHead(x,John)$ yields:

  $Crown(C1) \wedge OnHead(C1,John)$

  provided $C1$ is a new constant symbol, called a Skolem constant

- EI can be applied once to replace the existential sentence

# Inference in FOL: Reduction to propositional inference

- Suppose the KB contains just the following:
  - $\forall$ x King(x) $\wedge$ Greedy(x) $\Rightarrow$ Evil(x)
  - King(John)
  - Greedy(John)
  - Brother(Richard,John)
- Instantiating the universal sentence in all possible ways, we have:
  - King(John) $\wedge$ Greedy(John) $\Rightarrow$ Evil(John)
  - King(Richard) $\wedge$ Greedy(Richard) $\Rightarrow$ Evil(Richard) → *irrelevant substitution*
  - King(John)
  - Greedy(John)
  - Brother(Richard,John)
- The new KB is *propositionalized*

# Inference in FOL: Reduction to PL

- A ground sentence is entailed by a new KB iff entailed by the original KB.
- 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(Father(John)))*

# Inference in FOL

- **Instead** of translating the knowledge base to PL, we can make the **inference rules work in FOL.**
- For example, given:
  - $\forall$ x King(x) $\wedge$ Greedy(x) $\Rightarrow$ Evil(x)
  - King(John)
  - Greedy(John)
- **Can we prove Evil(John)?**
- The inference that John is evil works like this:
  1. find some x such that x is a king and x is greedy,
  2. and then infer that x is evil.
- Generally:
  1. It is intuitively clear that we can substitute {x/John} and obtain that Evil(John)

# Inference in FOL

- What if we have:
  - $\forall$ x King(x) $\wedge$ Greedy(x) $\Rightarrow$ Evil(x)
  - King(John)
  - $\forall$ y Greedy(y)

- It is intuitively clear that we can substitute {x/John,y/John} and obtain that Evil(John)

# Inference in FOL: Unification

- We can make the inference if we can find a substitution such that King(x) and Greedy(x) match King(John) and Greedy(y) , e.g. {x/John,y/John} works
- This above process is called Unification
- Unification is a process of making two different logical atomic expressions identical by finding a substitution. In other words, it takes two literals as input and makes them identical using substitution.
- It returns fail if the expressions do not match with each other.

# Inference in FOL: Conditions Unification

- Predicate symbol must be same, atoms or expression with different predicate symbol can never be unified.
- Number of arguments in both expressions must be identical.
- Unification will fail if there are two similar variables present in the same expression.

# Inference in FOL: Unification Example

- **Unify(α ,β) = θ** if Subst(θ, α) = Subst(θ, β)

| α | β | Subst |
|---|---|---|
| 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} |

# Inference in FOL

## 1) Generalized Modus Ponens (GMP)

- Suppose that Subst( θ, *pi'*) = Subst( θ, *pi*) for all *i* then:

$$\frac{p1', p2', \ldots , pn', ( p1 \wedge p2 \wedge \ldots \wedge pn \Rightarrow q )}{Subst(\theta, q)}$$

- p1' is King(John) p1 is King(x)
- p2' is Greedy(y) p2 is Greedy(x)
- θ is {x/John,y/John} q is Evil(x)
- Subst(θ, q) is Evil(John)
- All variables assumed universally quantified.

# Inference in FOL

## 2) Resolution

- Full first-order version:

$$\frac{l_1 \vee \cdots \vee l_k, \qquad m_1 \vee \cdots \vee m_n}{\text{Subst}(\theta, l_1 \vee \cdots \vee l_{i-1} \vee l_{i+1} \vee \cdots \vee l_k \vee m_1 \vee \cdots \vee m_{j-1} \vee m_{j+1} \vee \cdots \vee m_n)}$$

where $\theta = \text{Unify}(l_i, \neg m_j)$

$$\frac{\neg Rich(x) \vee Unhappy(x), \qquad Rich(Ken)}{Unhappy(Ken)}$$

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

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

# Inference in FOL: Forward Chaining

For each rule such that a fact unifies with a premise,
if the other premises are known
then add the conclusion to the KB and continue chaining.

- Forward chaining is **data-driven**,
  e.g., inferring conclusions from incoming percepts.

# Inference in FOL: Forward chaining example

- **Rules**

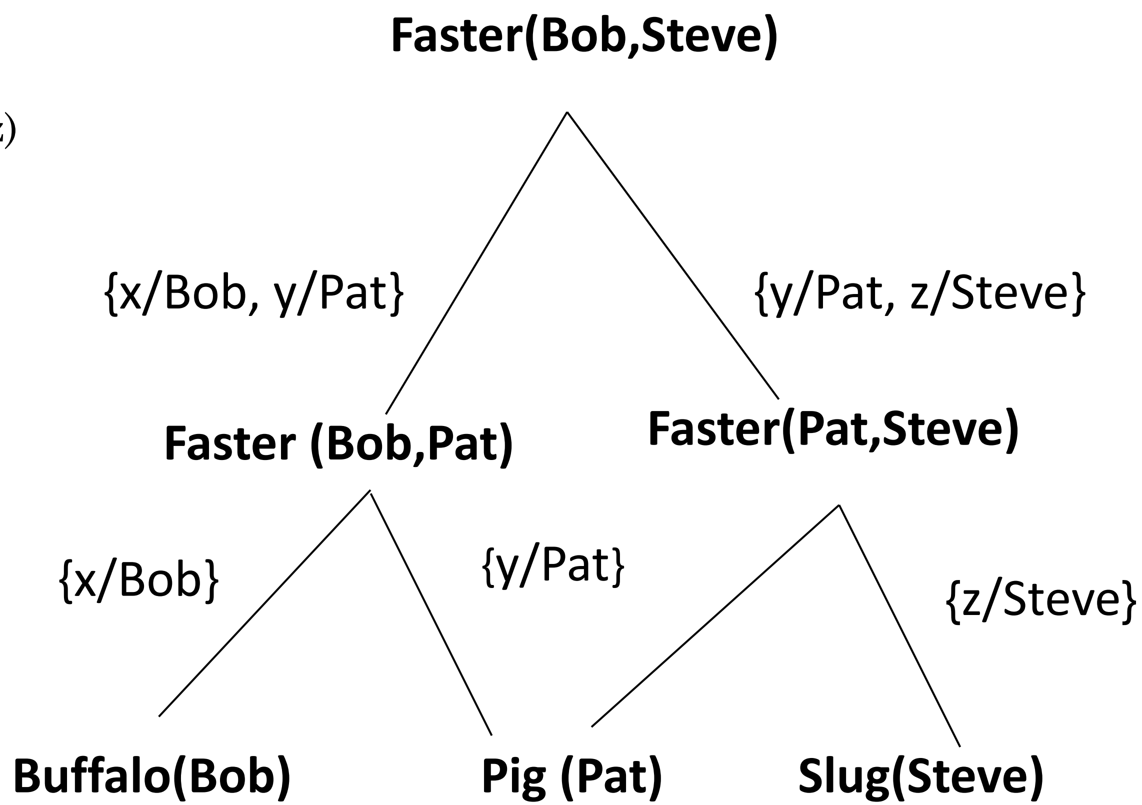
1. Buffalo(x) $\wedge$ Pig(y) $\Rightarrow$ Faster(x, y)
2. Pig(y) $\wedge$ Slug(z) $\Rightarrow$ Faster(y, z)
3. Faster(x, y) $\wedge$ Faster(y, z) $\Rightarrow$ Faster(x, z)

- **Facts**

1. Buffalo(Bob)
2. Pig(Pat)
3. Slug(Steve)

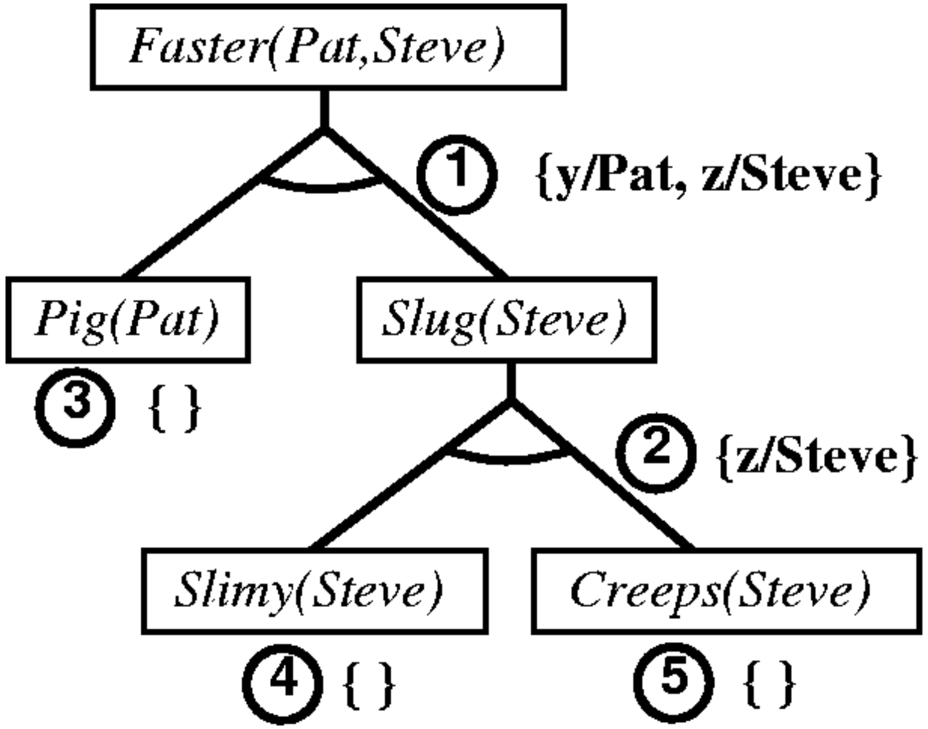
- **New facts**

4. Faster(Bob, Pat)
5. Faster(Pat, Steve)
6. Faster (Bob, Steve)



# Inference in FOL: Backward Chaining

Backward chaining starts with a hypothesis (query) and work backwards, according to the rules in the knowledge base until reaching confirmed findings or facts.




## Example

1. $Pig(y) \land Slug(z) \Rightarrow Faster(y, z)$
2. $Slimy(z) \land Creeps(z) \Rightarrow Slug(z)$
3. Pig(Pat)
4. Slimy(Steve)
5. Creeps(Steve)

# Applications of Knowledge Representation and Reasoning in AI

- Query answering
- Explaining
- Story generation
- Planning
- Diagnosis
- ….

# References


- https://www.javatpoint.com/artificial-intelligence-tutorial
- https://people.cs.pitt.edu/~milos/courses/cs2740/Lectures/class6.pdf