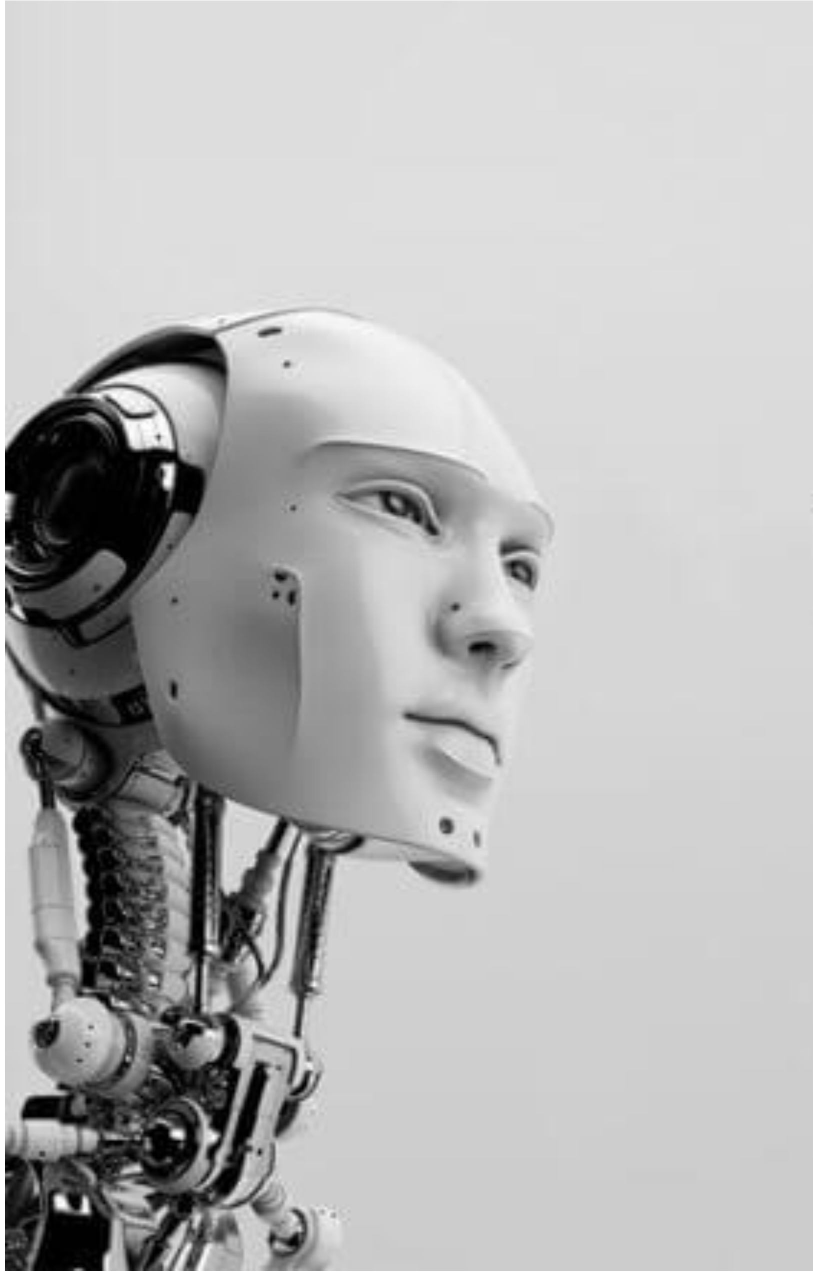

Artificial Intelligence

8 First Order Logic

Russell & Norvig, AI: A Modern Approach, 3rd Ed

Lec Zinia Sultana
CSE Dept, MIST

First Order Logic

Used for representing a complex environments.

- ❖ **Difference with propositional logic:**
  - ✓ **Existence of predicates**
  - ✓ **And Quantifiers**

First Order Logic - Model

- ❑ Object
- ❑ Relation
- ❑ Function

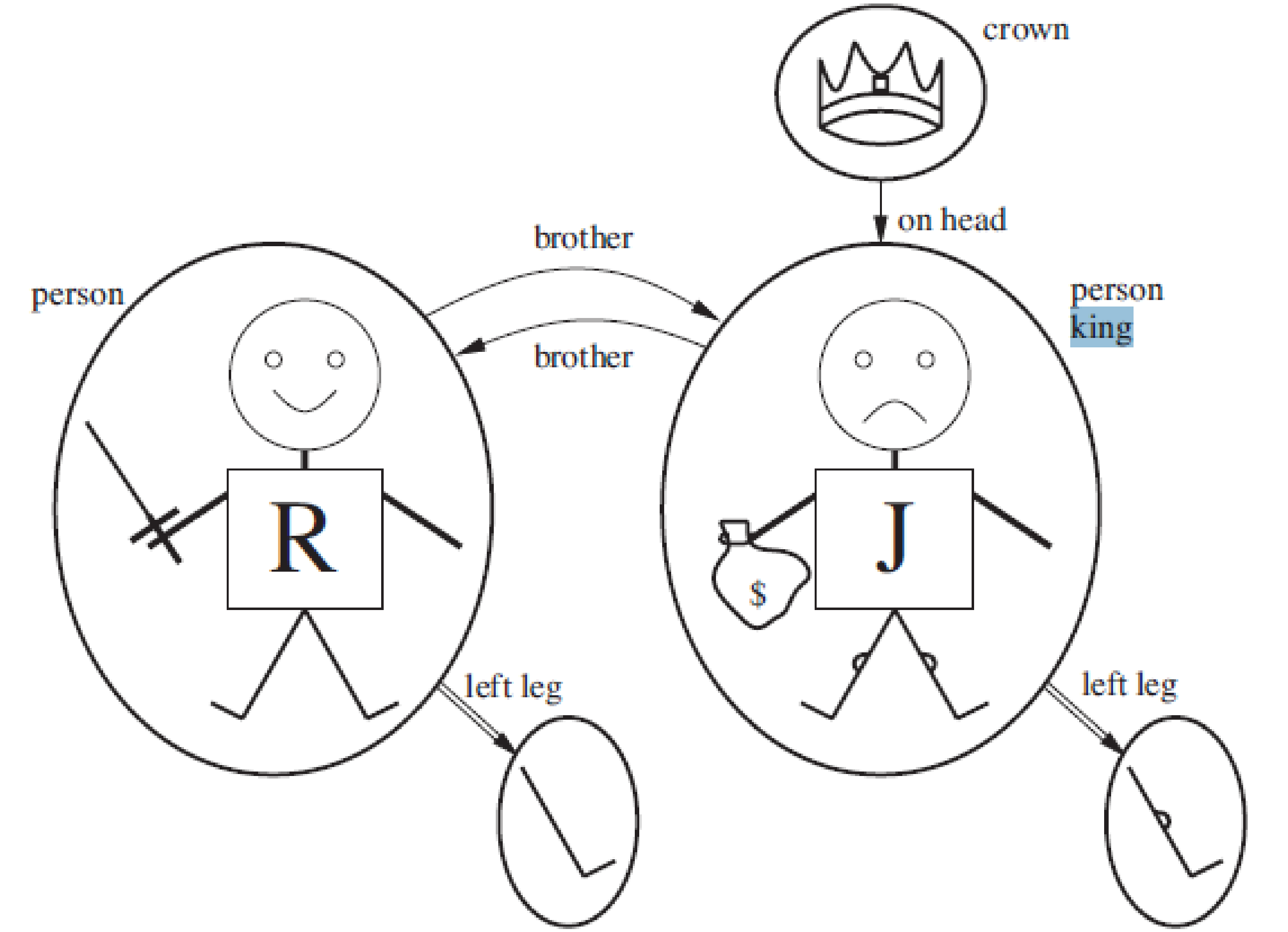

Figure 8.2 A model containing five objects, two binary relations, three unary relations (indicated by labels on the objects), and one unary function, left-leg.

Syntax

- Constant
- Variable – Small Letter
- Predicate – Ret T/F
- Function – Ret any value

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

Syntax

- Two types of quantifiers
  - Universal quantification, $\forall$: for all
  - Existential quantification, $\exists$: for some

First Order Logic – Example 1

Sentences: *Sakib is a cricketer.*

Atomic Sentences: is formed from a predicate symbol optionally followed by a parenthesized list of terms.

1. x is a cricketer ≡ Cricketer(x)

FOL: *Crickter*(Sakib)

First Order Logic – Example 2

Sentences: *Romeo loves Juliet.*

Atomic Sentences: is formed from a predicate symbol optionally followed by a parenthesized list of terms.

1. x loves y $\equiv$ Loves(x, y)

FOL: *Loves*(Romeo, Juliet)

First Order Logic – Example 3

Sentences: *Mary loves everyone.*

Atomic Sentences: is formed from a predicate symbol optionally followed by a parenthesized list of terms.

1. x loves y $\equiv$ Loves(x, y)

FOL: $\forall$x Loves(Mary, x)

First Order Logic – Example 4

Sentences: *There is someone who loves Mary.*

Atomic Sentences: is formed from a predicate symbol optionally followed by a parenthesized list of terms.

1. x loves y $\equiv$ Loves(x, y)

FOL: $\exists$x Loves(x, Mary)

First Order Logic – Example 5

Sentences: *Everyone loves it’s mother.*

Way 1:

Atomic Sentences: is formed from a predicate symbol optionally followed by a parenthesized list of terms.

1. x loves y ≡ Loves (x, y)
2. Return mother of x ≡ Mother (x)

FOL: $\forall_x$ Loves (x, Mother (x))

First Order Logic – Example 5 . . .

Sentences: *Everyone loves it's mother.*

Way 2:

Atomic Sentences: is formed from a predicate symbol optionally followed by a parenthesized list of terms.

1. x's mother is y $\equiv$ Mother (x, y)
2. x loves y $\equiv$ Loves (x, y)

FOL: $\forall_x \; \exists_y$ Mother (x, y) ^ Loves (x, y)

First Order Logic – Example 6

Sentences: *No person likes a professor unless the professor is smart.*

Atomic Sentences:

1. Person(x) ≡ x is a person.
2. Professor (x) ≡ x is a professor.
3. Smart(x) ≡ x is a smart.
4. Likes(x, y) ≡ subject x likes subject y.

FOL: $\forall$ x $\forall$ y Person(x) $\wedge$ Professor(y) $\wedge$ $\neg$Smart(y) $\Rightarrow$ $\neg$likes(x,y)

Example

Sentence	**First Order Logic Representation**
Lucy is a professor	is-prof(lucy)
All professors are people.	$\forall$ x (is-prof(x) $\Rightarrow$ is-person(x))
Fuchs is the dean.	is-dean(fuchs)
All Deans are professors.	$\forall$ x (is-dean(x) $\Rightarrow$ is-prof(x))
Everyone is a friend of someone.	$\forall$ x ($\exists$ y (is-friend-of (y, x)))
Lucy criticized Fuchs.	criticize(lucy,fuchs)

Nested quantifiers

- Consecutive quantifiers of the same type can be written as one quantifier with several variables.
    - Siblinghood is a symmetric relationship
    - $\forall$ x, y Sibling(x, y) $\Leftrightarrow$ Sibling(y, x)

- The order of quantification is therefore very important. It becomes clearer if we insert parentheses.
    - Everybody loves somebody
    - $\forall$ x $\exists$ y Loves(x, y)
    - $\exists$ y $\forall$ x Loves(x, y)

Connections between ∀ and ∃

- The two quantifiers are actually intimately connected with each other, through negation
- ∀ is a conjunction over the universe of objects and ∃ is a disjunction
- They obey De Morgan's rules
  - $\forall x \neg P \equiv \neg \exists x P$
  - $\neg \forall x P \equiv \exists x \neg P$
  - $\forall x P \equiv \neg \exists x \neg P$
  - $\exists x P \equiv \neg \forall x \neg P$
- Everyone likes ice cream : $\forall x\ Likes(x, IceCream)$
- There is no one who does not like ice cream:
  $\neg \exists x\ \neg Likes(x, IceCream)$

Equality

- **Equality symbol** can be used to signify that two terms refer to the same object.
  - Father (John)=Henry

- Richard has at least two brothers.
- $\exists$ x, y Brother (x,Richard) $\wedge$ Brother (y,Richard)
- $\exists$ x, y Brother (x,Richard) $\wedge$ Brother (y,Richard) $\wedge\neg$(x=y)

More Examples – on Kinship

- One's mother is one's female parent:
- $\forall$ m, c Mother (c)=m $\Leftrightarrow$ Female(m) $\wedge$ Parent(m, c) .
- One's husband is one's male spouse:
- $\forall$ w, h Husband(h,w) $\Leftrightarrow$ Male(h) $\wedge$ Spouse(h,w) .
- Male and female are disjoint categories:
- $\forall$ x Male(x) $\Leftrightarrow$ $\neg$Female(x) .
- Parent and child are inverse relations:
- $\forall$ p, c Parent(p, c) $\Leftrightarrow$ Child (c, p) .
- A grandparent is a parent of one's parent:
- $\forall$ g, c Grandparent (g, c) $\Leftrightarrow$ $\exists$p Parent(g, p) $\wedge$ Parent(p, c) .
- A sibling is another child of one's parents:
- $\forall$ x, y Sibling(x, y) $\Leftrightarrow$ x = y $\wedge$ $\exists$p Parent(p, x) $\wedge$ Parent(p, y) .

Try yourself:

1. All professors consider the dean a friend or don’t know him.
2. People only criticize people that are not their friends.
3. Only one student failed in History.
4. No person likes a leader unless the leader is smart
5. Everyone who loves all animals is loved by someone
6. All kids are short.
7. Certain kids own shoes.
8. If someone is a kid and a boy, he likes cars.
9. All kids that own shoes, wear them.
10. All kids have a mother.
11. If a woman is a mother, she has a kid.
12. No kid likes a vegetable if it's overcooked.
13. No mother likes a teacher if he punishes her kid.

The knowledge-engineering process

The general process of knowledge-base construction:

1. Identify the task. – PEAS
2. Assemble the relevant knowledge. – knowledge acquisition
3. Decide on a vocabulary of predicates, functions, and constants.
4. Encode general knowledge about the domain.
5. Encode a description of the specific problem instance.
6. Pose queries to the inference procedure and get answers.
7. Debug the knowledge base.

Thank you ☺