ARTIFICIAL INTELLIGENCE

CSC304
CPE406
SC430

School of Computer Engineering
Nanyang Technological University

Part III – Knowledge and Reasoning

- **6 Agents that Reason Logically**
  - Knowledge-based Agents. – Representations.
  - Propositional Logic. – The Wumpus World.
- **7 First-Order Logic**
  - Syntax and Semantics. – Using First-Order Logic.
  - Logical Agents. – Representing Changes.
  - Deducing Properties of the World.
  - Goal-based Agents.
- **8 Building a Knowledge Base**
  - Knowledge Engineering. – General Ontology.

7 — FIRST-ORDER LOGIC

"In which we introduce a logic that is sufficient for building knowledge-based agents."

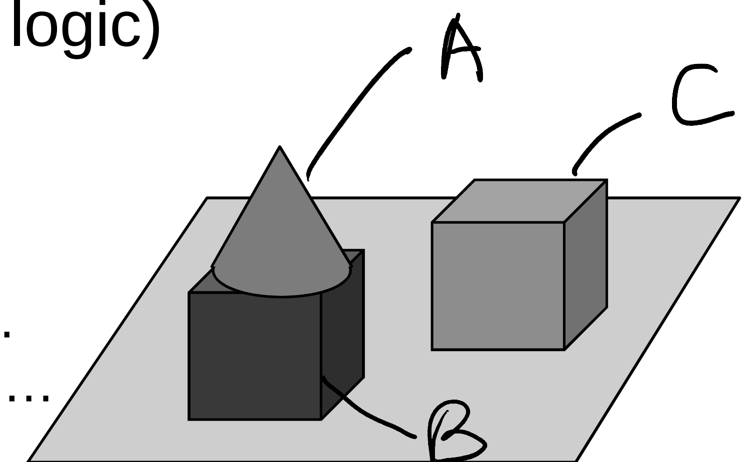
Representing Knowledge

- **<u>Knowledge-based agent</u>**
  – Have representations of the world in which they operate
  – Use those representations to infer what actions to take
- **Ontological commitments**
  – The world as facts (propositional logic)
  – The world as <u>objects</u> (first-order logic) with <u>properties</u> *about* each object, and <u>relations</u> *between* objects
    - e.g. the blocks world:
      Objects: cubes, cylinders, cones, …
      Properties: shape, colour, location, …
      Relations: above, under, next-to, …

A above B
≡ above(A,B)

Color(X)

eg
Color(A)
= green



B next-to C ≡ next-to(B,C)

Why is FOL Important?

- **A very powerful KR scheme**
  - Essential representation of the world
    - Deal with objects, properties, and relations (as Philosophy).
  - Simple, generic representation
    - Does <u>not</u> deal with specialised concepts such as categories, time, and events.
  - Generalisation
    - Allow the making of universal and existential statements
  - Universal language
    - Can express anything that can be programmed.
  - Most studied and best understood
    - More powerful proposals still debated.
    - Less powerful schemes too limited.

Types of statements for expressing general information

- **Facts**
  - Proposition or predicate (relation/properties)
- **Rules**
  - Relationships between objects and predicates
  - General statements of relationships
  - Existential statements of relationships
- **Variables**
  - Proposition, not predicates

Propositional vs. First-Order Logic

general statement new fact

- **Aristotle's syllogism**
  - Socrates is a man. All men are mortal. Therefore Socrates is mortal.

fact 1

Statement	Propositional Logic	First-Order Logic
"Socrates is a man."	SocratesMan, S43	Man(Socrates), P52(S21)
"Plato is a man."	PlatoMan, S157	Man(Plato), P52(S99)
"All men are mortal."	MortalMan S421 Man ⇒ Mortal S9⇒S4	∀x binding Man(x) ⇒ Mortal(x), P52(V1) ⇒ P66(V1)
"Socrates is mortal."	MortalSocrates S957 S43 Λ S421 |– S957 ?!?	Mortal(Socrates) V1←S21, … |– P66(S21)

unificat^n/matching/binding

Syntax and Semantics of FOL

general statement

- **Sentences**

properties/characteristic/relations

  – Built from quantifiers, predicate symbols, and terms

- **Terms**

place holder

  – Represent objects
  – Built from variables, constant and <u>function symbols</u>

defining a constant

- **Constant symbols**
  – Refer to ("name") particular objects of the world
    - The object is specified by the interpretation
      e.g. "John" is a constant , may refer to "John, king of England from 1199 to 1216 and younger brother of Richard Lionheart", or my uncle, or …

Predicate and Function Symbols

Properties → T
→ F

man(x)
← variables

- **Predicate symbols**
  - Refer to particular relations on objects
    - Binary relation specified by the interpretation
      e.g. Brother(KingJohn, RichardLionheart) –> T or F

man(John) → T

man(Jane) → F

  - A *n-ary* relation if defined by a set of *n-tuples*
    - Collection of objects arranged in a fixed order
      e.g. { ⟨KingJohn, RichardLionheart⟩, ⟨KingJohn, Henry⟩, ... }
- **Function symbols**
  - Refer to functional relations on objects
    - Many-to-one relation specified by the interpretation
      e.g. BrotherOf(KingJohn) –> a person, e.g. Richard (*not* T/F)

denote another object →

→ symbol/Skolem Constant

  - Defined by a set of *n+1-tuples*
    - Last element is the function value for the first *n* elements.

Example:

- Leftleg(John) -> function symbol
- John-leftleg -> constant

- Amputated(x) -> a predicate defining the property of x being amputated.
- Amputated(John-leftleg) -> T/F
- Amputated(Leftleg(John)) -> T/F
- <u>What is diabetes?</u>
- Diabetic(x)^Amputated(Leftleg(x)) - variable
- Diabetic(John)^Amputated(Leftleg(John)) – specific (John) compound sentence

Variables and Terms in FOL

UpperCase → constant

Lower case → Variable

- **Variables**
  - Refer to any object of the world
    - e.g. x, person, … as in Brother(KingJohn, person).
  - Can be substituted by a constant symbol
    - e.g. person ← Richard, yielding Brother(KingJohn, Richard).
- **Terms**
  - Logical expressions referring to objects
    - Include constant symbols (“names”) and variables.
    - Make ~~use of function~~ symbols.
      e.g. LeftLegOf(KingJohn) to refer to his leg without naming it

      constant
  - Compositional interpretation
    - e.g. LeftLegOf(), KingJohn –> LeftLegOf(KingJohn).

Sentences in FOL

- **Atomic sentences**
  - State facts, using terms and predicate symbols
    - e.g. Brother(Richard, John).
  - Can have complex terms as arguments
    - e.g. Married($\underbrace{\text{FatherOf(Richard)}}$, $\underbrace{\text{MotherOf(John)}}$).
  - Have a truth value

    both are place holders
    - Depends on both the interpretation and the world.
- **Complex sentences**
  - Combine sentences with connectives
    - e.g. Father(Henry, KingJohn) $\wedge$ Mother(Mary, KingJohn)
  - Connectives identical to propositional logic
    - i.e.: $\wedge$, $\vee$, $\Leftrightarrow$, $\Rightarrow$, $\neg$

Sentence Equivalence

- ***There are many ways to write a logical statement in FOL***
  - Example
    - $A \Rightarrow B$ equivalent to $\neg A \vee B$
      "rule form" *"complementary cases"*

      $\forall x$ $Dog(x) \Rightarrow Mammal(x)$ $\forall x$ $\neg Dog(x) \vee Mammal(x)$
      "dogs are mammals" "either it's not a dog or it's a mammal"
      Whenever it is a dog it is a mammal.
    - $A \wedge B \Rightarrow C$ equivalent to $A \Rightarrow (B \Rightarrow C)$

demorgan

Rewriting Proof: $A \wedge B \Rightarrow C \Leftrightarrow \neg(A \wedge B) \vee C \Leftrightarrow (\neg A \vee \neg B) \vee C$

$\Leftrightarrow$ $\mathbf{\neg A \vee \neg B \vee C}$ $\Leftrightarrow$ $\neg A \vee (\neg B \vee C)$

$\mathbf{\neg P \vee Q \Leftrightarrow P \Rightarrow Q}$

$\Leftrightarrow$ $\neg A \vee (B \Rightarrow C)$ $\Leftrightarrow$ $A \Rightarrow (B \Rightarrow C)$

Sentences in Normal Form

- ***There is only one way to write a logical statement using a Normal Form of FOL***
  - Example $\neg B \Rightarrow \neg A$
    - $A \Rightarrow B$, $A \wedge B \Rightarrow C$ equivalent to $\neg A \vee B$, $\neg A \vee \neg B \vee C$

INF "Implicative Normal Form" "Conjunctive Normal Form"

- ***Rewriting logical sentences allows to determine whether they are equivalent or not*** CNF
  - Example
    - $A \wedge B \Rightarrow C$ and $A \Rightarrow (B \Rightarrow C)$ both have the same CNF: $\neg A \vee \neg B \vee C$

CNF permits unit & binary resolution

- ***Using FOL is the most convenient, but using a Normal Form is the most efficient***

Sentence Verification

- ***Rewriting logical sentences helps to understand their meaning***
  - Example
    - Owns(x,y) $\Rightarrow$ (Dog(y) $\Rightarrow$ AnimalLover(x)) $\quad$ A $\Rightarrow$ (B $\Rightarrow$ C)

      Owns(x,y) Λ Dog(y) $\Rightarrow$ AnimalLover(x) $\quad$ A Λ B $\Rightarrow$ C

      "*A dog owner is an animal lover*"
- ***Rewriting logical sentences helps to verify their meaning is as intended***
  - <u>Example</u>
    - "*Dogs all have the same enemies*"
    - Dog(x) Λ Enemy(z, x) $\Rightarrow$ (Dog(y) $\Rightarrow$ Enemy(z, y)) same as

      Dog(x) Λ Dog(y) Λ Enemy(z, x) $\Rightarrow$ Enemy(z, y)

Universal Quantifier ∀

- **Express properties of collections of objects**
  - Make a statement about *every* objects w/out enumerating
    - e.g. "All kings are mortal

$$\begin{aligned}&King(Henry) \Rightarrow Mortal(Henry)\ \Lambda\\ &King(John) \Rightarrow Mortal(John)\ \Lambda\\ &King(Richard) \Rightarrow Mortal(Richard)\ \Lambda\\ &King(London) \Rightarrow Mortal(London)\ \Lambda\\ &\ldots\end{aligned}$$

instead: $\forall$ x, King(x) $\Rightarrow$ Mortal(x)

- Note: the semantics of the implication says F $\Rightarrow$ F is TRUE,
  thus for those individuals that satisfy the premise King(x)
  the rule asserts the conclusion Mortal(x)
  but for those individuals that do not satisfy the premise
  the rule makes no assertion.

Using the Universal Quantifier

- ***The implication (⇒) is the natural connective to use with the universal quantifier (∀)***
  - Example
    - General form: $\forall x\ P(x) \Rightarrow Q(x)$ e.g. $\forall x\ Dog(x) \Rightarrow Mammal(x)$
      "all dogs are mammals"

      Binds both

    - Use conjunction? $\forall x\ P(x) \wedge Q(x)$ e.g. $\forall x\ Dog(x) \wedge Mammal(x)$

      same as $\forall x\ P(x)$ and $\forall x\ Q(x)$
      e.g. $\forall x\ Dog(x)$ and $\forall x\ Mammal(x)$

      –> yields a very strong statement (too strong! i.e. *incorrect*)

      All mammals are dogs

Existential Quantifier ∃

- **Express properties of some particular objects**
  - Make a statement about *one* object without naming it
    - e.g. "King John has a brother who is king"

      $\exists$ x, Brother(x, KingJohn) Λ King(x)

    instead of

All possibilities instances

Brother(Henry, KingJohn) Λ King(Henry) $\vee$

Brother(KingJohn, KingJohn) Λ King(KingJohn) $\vee$

Brother(Mary, KingJohn) Λ King(Mary) $\vee$

Brother(London, KingJohn) Λ King(London) $\vee$

Brother(Richard, KingJohn) Λ King(Richard) $\vee$

...

Using the Existential Quantifier

- ***The conjunction (Λ) is the natural connective to use with the existential quantifier ($\exists$)***
  - Example
    - General form: $\exists x\ P(x) \wedge Q(x)$ e.g. $\exists x\ Dog(x) \wedge Owns(John, x)$
      "John owns a dog"
    - Use Implication? $\exists x\ P(x) \Rightarrow Q(x)$ e.g. $\exists x\ Dog(x) \Rightarrow Owns(John, x)$

      true for all x such that P(x) is false
      e.g. Dog(<u>Garfield</u>) $\Rightarrow$ Owns(John, Garfield)

    - –> yields a very weak statement (too weak! i.e. *useless*)

If false $\Rightarrow$ no information can be drawn.

end