CZ3005 Artificial Intelligence

Week 10a – First-Order Logic

Yu Han

han.yu@ntu.edu.sg

Nanyang Assistant Professor
School of Computer Science and Engineering
Nanyang Technological University

Recap

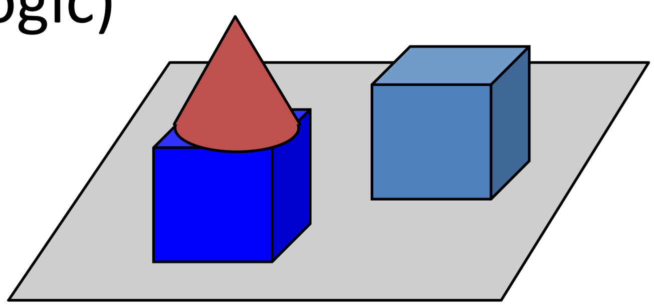
- A **sentence** is either True or False under an interpretation (or a world).
  - When it is True, we say that it is satisfiable (or is a model).
  - When it is False, then we say that it is unsatisfiable.
  - When a sentence is satisfiable under all interpretations (or worlds), then we say that it is a valid sentence or a *tautology*.
- Given that *N* objects are used in a **KB**, there will be 2^N possible interpretations (or worlds).

Recap

- **Soundness** and **Completeness**
  - Refer to an inference procedure
  - We do not say a sentence is sound or complete. An inference procedure is sound if we can derive True sentence from True sentences. That is, it implements entailment.
    - MP implements entailments
  - An inference procedure is complete if we can derive the proof of all entailed sentences (or valid sentences).

Representing Knowledge

- **Knowledge-based agent**
  - 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 objects (first-order logic) with properties *about* each object, and relations *between* objects
    - e.g. the blocks world:
    - Objects: cubes, cylinders, cones, …
      Properties: shape, colour, location, …
      Relations: above, under, next-to, …



First-Order Logic (FOL)

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

Propositional vs. First-Order Logic

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

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 $\Rightarrow$ Mortal S9$\Rightarrow$S4	Man(x) $\Rightarrow$ Mortal(x), P52(V1) $\Rightarrow$ P66(V1)
"Socrates is mortal."	MortalSocrates S957 S43 Λ S421 |– S957 ?!?	Mortal(Socrates) V1$\leftarrow$S21, … |– P66(S21)

Syntax and Semantics of FOL

- **Sentences**
  - Built from quantifiers, predicate symbols, and terms
- **Terms**
  - Represent objects
  - Built from variables, constant and function symbols
- **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 …

Syntax and Semantics of FOL

- **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 $\leftarrow$ 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
  - Compositional interpretation
    - e.g. LeftLegOf(), KingJohn –> LeftLegOf(KingJohn).

Predicate and Function Symbols

- **Predicate symbols**
  - Refer to particular relations on objects
    - Binary relation specified by the interpretation
      e.g. Brother(KingJohn, RichardLionheart) –> T or 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)
  - Defined by a set of *n+1-tuples*
    - Last element is the function value for the first *n* elements.

Functions

- A function of arity *n* takes *n* objects of type $W_1,\ldots,W_n$ as inputs and returns an object of type *W*.
- Example:
  - Plus(3, 4)=7

Object Terms

Functional Term

Predicates

- Predicates are like functions except that their return type is True or False.
- Example:
  - Greater-Than(3, 4)=False

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(FatherOf(Richard), MotherOf(John)).
  - Have a truth value
    - 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"*

      $Dog(x) \Rightarrow Mammal(x)$ $\neg Dog(x) \vee Mammal(x)$

      "dogs are mammals" "either it's not a dog or it's a mammal"

    - $A \wedge B \Rightarrow C$ equivalent to $A \Rightarrow (B \Rightarrow C)$

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

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

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

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

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 $\neg A \vee B$, $\neg A \vee \neg B \vee C$
      "Implicative Normal Form" "Conjunctive Normal Form"
- ***Rewriting logical sentences allows to determine whether they are equivalent or not***
  - Example
    - $A \wedge B \Rightarrow C$ and $A \Rightarrow (B \Rightarrow C)$
      both have the same CNF: $\neg A \vee \neg B \vee C$
- ***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)) A $\Rightarrow$ (B $\Rightarrow$ C)
    - Owns(x,y) Λ Dog(y) $\Rightarrow$ AnimalLover(x) A Λ B $\Rightarrow$ C

      "A dog owner is an animal lover"
- ***Rewriting logical sentences helps to verify their meaning is as intended***
  - Example
    - *"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 $\forall$

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

      $King(Henry) \Rightarrow Mortal(Henry)\ \Lambda$
      $King(John) \Rightarrow Mortal(John)\ \Lambda$
      $King(Richard) \Rightarrow Mortal(Richard)\ \Lambda$
      $King(London) \Rightarrow Mortal(London)\ \Lambda$
      …
  - 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 ($\Rightarrow$) is the natural connective to use with the universal quantifier ($\forall$)**
  - Example
    - General form: $\forall x \; P(x) \Rightarrow Q(x)$
      - e.g. $\forall x \; Dog(x) \Rightarrow Mammal(x)$ "all dogs are mammals"
    - Use conjunction? $\forall x \; P(x) \land Q(x)$
      - e.g. $\forall x \; Dog(x) \land Mammal(x)$
      - same as $\forall x \; P(x)$ and $\forall x \; Q(x)$

        e.g. $\forall x \; Dog(x)$ and $\forall x \; Mammal(x)$

        All dogs are mammals. All mammals are dogs.

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

Existential Quantifier ∃

- **Express properties of some particular objects**
  - Make a statement about <u>one</u> object without naming it
    - e.g., “King John has a brother who is king”
    - ∃ x, Brother(x, KingJohn) Λ King(x)
    - instead of

      Brother(Henry, KingJohn) Λ King(Henry) ∨
      Brother(London, KingJohn) Λ King(London) ∨
      Brother(Richard, KingJohn) Λ King(Richard) ∨
      …

Using the Existential Quantifier

- **The conjunction (Λ) is the natural connective to use with the existential quantifier ($\exists$)**
  - Example
    - General form: $\exists x\ P(x)\ \Lambda\ Q(x)$
      - e.g., $\exists$x Dog (x) Λ 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)
      - Could be true for all x such that P(x) is false
        e.g., Dog(Garfield_the_cat) $\Rightarrow$ Owns(John, Garfield_the_cat)

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

Thank you!