CS 4248
Natural Language Processing

Professor NG Hwee Tou

Department of Computer Science
School of Computing
National University of Singapore
nght@comp.nus.edu.sg

Chapter 17: The Representation of Meaning

- Meaning Representation (Semantic Representation)
  - Formal structures composed from symbols to represent the meaning of sentences
- This chapter focuses on representation of literal meaning of individual sentences

Literal vs. Non-literal Meaning

- John went to the supermarket. He picked up a milk carton on the shelf. He paid at the check-out counter and left.
- What did John buy at the supermarket?

Literal vs. Non-literal Meaning

- John went to the supermarket. He picked up a milk carton on the shelf. He paid at the check-out counter and left.
- What did John buy at the supermarket?
- Inferences needed:
  - Milk was the item paid for
  - Paid for x implies bought x
  - Milk was on the shelf which was in the supermarket
  - He refers to John

Meaning Representation Languages

- Frameworks/languages used to specify the syntax and semantics of meaning representations
- Frequently used meaning representation languages:
  - First-Order Logic (FOL)
  - Semantic Network
  - Conceptual Dependency
  - Frame Representation

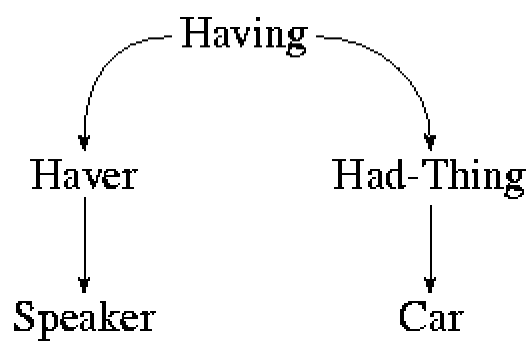
Meaning Representation

Sentence: I have a car.

$$\exists x, y\, Having(x) \wedge Haver(Speaker, x) \wedge HadThing(y, x) \wedge Car(y)$$

FOL

Semantic Network



Frame Representation

Conceptual Dependency

Car
$\Uparrow$ POSS-BY
Speaker

```
Having
 Haver:   Speaker
 HadThing:  Car
```

Desirable Properties of Meaning Representation Language

- Verifiability
- Unambiguous Representations
- Canonical Form
- Inference and Variables
- Expressiveness

Desirable Properties of Meaning Representation Language

- Verifiability
  - Able to determine the truth of the meaning representation of a sentence against the world modeled
  - That is, *match* the meaning representation of a sentence against representations of facts about the world modeled in the system's knowledge base
  - Example:
    - Does Maharani serve vegetarian food?
    - Serves(Maharani, VegetarianFood)

Desirable Properties of Meaning Representation Language

- Unambiguous Representations
  - Semantic ambiguity of input sentences
    - I wanna eat someplace that's close to NUS.
      - » Interpretation #1: eat at some location near NUS
      - » Interpretation #2: eat some location near NUS
  - Goal of semantic analysis is to resolve semantic ambiguity
  - Meaning representation should be free from ambiguity

Desirable Properties of Meaning Representation Language

- Canonical Form
  - Different sentences with the same meaning should be assigned the same meaning representation
  - Examples:
    - Does Maharani have vegetarian dishes?
    - Do they have vegetarian food at Maharani?
    - Are vegetarian dishes served at Maharani?
    - Does Maharani serve vegetarian fare?
  - Different words (dishes, food, fare; have, serve)
  - Different sentence structures (Maharani has vegetarian food, They have vegetarian food at Maharani)
  - Having the same representation for these questions facilitates matching to facts in the knowledge base

Desirable Properties of Meaning Representation Language

- Canonical Form
  - Assign the same meaning to different words
    - Examples: dishes, food, fare
  - Assign the same meaning to different syntactic structures
    - Examples: Active and passive sentences
      - » Maharani serves vegetarian dishes.
      - » Vegetarian dishes are served by Maharani.

Desirable Properties of Meaning Representation Language

- Inference and Variables
  - Inference:
    - Maharani serves vegetarian food.
    - Can vegetarians eat at Maharani?
    - Need the inference rule that vegetarians eat vegetarian food
  - NLP program must be able to draw valid inferences not explicitly stated in the meaning representation of the input sentence or program's knowledge base

Desirable Properties of Meaning Representation Language

- Inference and Variables
  - Variables
    - I’d like to find a restaurant where I can get vegetarian food
    - Serves(x, VegetarianFood)

Desirable Properties of Meaning Representation Language

- Expressiveness
  - Meaning representation language must be able to adequately express and represent the meaning of any natural language sentence

First-Order Logic (FOL)

- A flexible, expressive meaning representation language
- Objects and relations among objects

FOL

$$
\begin{array}{rcl}
\mathit{Formula} & \rightarrow & \mathit{AtomicFormula} \\
& | & \mathit{Formula}\ \mathit{Connective}\ \mathit{Formula} \\
& | & \mathit{Quantifier}\ \mathit{Variable}, \ldots\ \mathit{Formula} \\
& | & \neg\ \mathit{Formula} \\
& | & (\mathit{Formula}) \\
\\
\mathit{AtomicFormula} & \rightarrow & \mathit{Predicate}(\mathit{Term}, \ldots) \\
\\
\mathit{Term} & \rightarrow & \mathit{Function}(\mathit{Term}, \ldots) \\
& | & \mathit{Constant} \\
& | & \mathit{Variable} \\
\\
\mathit{Connective} & \rightarrow & \wedge \mid \vee \mid \Rightarrow \\
\mathit{Quantifier} & \rightarrow & \forall \mid \exists \\
\mathit{Constant} & \rightarrow & A \mid \mathit{VegetarianFood} \mid \mathit{Maharani} \cdots \\
\mathit{Variable} & \rightarrow & x \mid y \mid \cdots \\
\mathit{Predicate} & \rightarrow & \mathit{Serves} \mid \mathit{Near} \mid \cdots \\
\mathit{Function} & \rightarrow & \mathit{LocationOf} \mid \mathit{CuisineOf} \mid \cdots
\end{array}
$$

FOL

- Terms: represent objects
  - Constants:
    - A, B, Maharani, Harry
    - Convention: begin with an uppercase letter
  - Functions:
    - LocationOf(Maharani)
    - The location of Maharani or Maharani’s location
    - Convention: begin with an uppercase letter
  - Variables:
    - x, y
    - Used with quantifiers $\forall$ and $\exists$
    - Convention: lowercase letters

FOL

- Predicates: represent relations
  - A relation that holds among some fixed number of objects
    - Example:
    - Maharani serves vegetarian food.
      - » Serves(Maharani, VegetarianFood)
  - Assert a property of a single object
    - Example:
    - Maharani is a restaurant.
      - » Restaurant(Maharani)
  - Convention: begin with an uppercase letter

FOL

- Logical connectives: $\neg$ $\wedge$ $\vee$ $\Rightarrow$
  - Example:
  - I only have five dollars and I don't have a lot of time.
  - Have(Speaker, FiveDollars) $\wedge$ $\neg$Have(Speaker, LotOfTime)
- P $\Rightarrow$ Q is equivalent to $\neg$P $\vee$ Q

FOL

- Terms and predicates in a FOL formula acquire their meanings by their correspondence to the objects, relations, and properties in the external world being modeled
- A FOL formula can be assigned True or False
- Example:
  - Maharani is near NUS.
  - Near(LocationOf(Maharani), LocationOf(NUS))

FOL

- Model-theoretic semantics
  - A FOL formula is true wrt a model and an interpretation
- Model contains a domain of objects and relations among objects
- Interpretation is a mapping:
  - constant symbol → object
  - predicate symbol → relation (of objects)
  - function symbol → functional relation (of objects)

FOL

Model

Domain D = { a, b, f, g }

Relations:

n = { f }

l = { <a, g>, <b, g> }

Interpretation

constant symbols:

Matthew → a

Franco → b

Med → f

Rio → g

predicate symbols:

Noisy → n

Likes → l

"Franco likes Rio."

FOL formula Likes(Franco, Rio) is true.

"Rio is noisy."

FOL formula Noisy(Rio) is false.

FOL

Truth table:

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$
False	*False*	*True*	*False*	*False*	*True*
False	*True*	*True*	*False*	*True*	*True*
True	*False*	*False*	*False*	*True*	*False*
True	*True*	*False*	*True*	*True*	*True*

FOL

- Variables and quantifiers
- Example 1:
  - Existentially quantified variables
  - Indefinite noun phrases
  - a restaurant that serves vegetarian food near NUS
  - $\exists x$ (Restaurant(x) $\wedge$ Serves(x, VegetarianFood) $\wedge$ Near(LocationOf(x), LocationOf(NUS)))
- Example 2:
  - Universally quantified variables
  - All vegetarian restaurants serve vegetarian food
  - $\forall x$ (VegetarianRestaurant(x) $\Rightarrow$ Serves(x, VegetarianFood))

FOL

- Lambda expression: $\lambda x\ P(x)$
  - $P(x)$: FOL formula
- λ-reduction: apply a lambda expression to a term
  - Replace the λ variable by the term, then remove λx
- Examples:
  - $\lambda x\ P(x)(A)$ gives $P(A)$
  - $\lambda x \lambda y\ Near(x,y)(USA)$ gives $\lambda y\ Near(USA,y)$
  - $\lambda y\ Near(USA,y)(Canada)$ gives $Near(USA, Canada)$

FOL

- Inference
  - Modus ponens (if-then reasoning)

$$\frac{\begin{array}{l}\alpha \\ \alpha \Rightarrow \beta\end{array}}{\beta}$$

- α: antecedent of $\alpha \Rightarrow \beta$
- β: consequent of $\alpha \Rightarrow \beta$

Categories

- Method 1:
  - Create a unary predicate for each category
    - VegetarianRestaurant(Maharani)
  - Problem: Unable to talk about VegetarianRestaurant
    - Not a valid FOL formula:

      MostPopular(Maharani, VegetarianRestaurant)

Categories

- Method 2:
  - Reification: Represent all concepts that we want to make statements about as full-fledged objects
  - ISA(Maharani, VegetarianRestaurant)
  - AKO(VegetarianRestaurant, Restaurant)

Events

- I ate.
- I ate a turkey sandwich.
- I ate a turkey sandwich at my desk.
- I ate at my desk.
- I ate lunch.
- I ate a turkey sandwich for lunch.
- I ate a turkey sandwich for lunch at my desk.

Events

- Method 1:
  - Create as many *different* eating predicates as are needed to handle all of the ways that eat behaves
  - A predicate in FOL has a fixed *arity* (i.e., a fixed number of arguments)

Events

- I ate.
- I ate a turkey sandwich.
- I ate a turkey sandwich at my desk.
- I ate at my desk.
- I ate lunch.
- I ate a turkey sandwich for lunch.
- I ate a turkey sandwich for lunch at my desk.

- Eating1(Speaker)
- Eating2(Speaker, TurkeySandwich)
- Eating3(Speaker, TurkeySandwich, Desk)
- Eating4(Speaker, Desk)
- Eating5(Speaker, Lunch)
- Eating6(Speaker, TurkeySandwich, Lunch)
- Eating7(Speaker, TurkeySandwich, Lunch, Desk)

- Relate them using meaning postulates:
  $\forall w, x, y, z\ \text{Eating7}(w, x, y, z) \Rightarrow \text{Eating6}(w, x, y)$

Events

- Problems:
  - Cumbersome: need too many meaning postulates
  - Difficult to scale up

Events

- Method 2:
  - Use a *single* predicate where as many arguments are included in the definition of the predicate as ever appear with it in an input

Events

$\exists w, x, y$ Eating(Speaker, w, x, y)

$\exists w, x$ Eating(Speaker, TurkeySandwich, w, x)

$\exists w$ Eating(Speaker, TurkeySandwich, w, Desk)

$\exists w, x$ Eating(Speaker, w, x, Desk)

$\exists w, x$ Eating(Speaker, w, Lunch, x)

$\exists w$ Eating(Speaker, TurkeySandwich, Lunch, w)

Eating(Speaker, TurkeySandwich, Lunch, Desk)

Events

- Problems:
  - Make too many commitments
    - Need to commit to all arguments (e.g., every eating event must be associated with a meal, which is not true)
  - Unable to refer to individual events
    - Event is a predicate, not a term

Events

- Method 3:
  – Use reification to elevate events to objects
  – Arguments of an event appear as predicates
  – Do not need to commit to arguments (roles) not mentioned in the input
  – Meaning postulates not needed

Events

- I ate.

  $\exists e\ Eating(e) \wedge Eater(e, Speaker)$

- I ate a turkey sandwich.

  $\exists e\ Eating(e) \wedge Eater(e, Speaker) \wedge Eaten(e, TurkeySandwich)$

- I ate a turkey sandwich for lunch.

  $\exists e\ Eating(e) \wedge Eater(e, Speaker) \wedge Eaten(e, TurkeySandwich) \wedge Meal(e, Lunch)$