EECS 3401 — AI and Logic Prog. — Lecture 2

Adapted from slides of Prof. Yves Lesperance

Vitaliy Batusov
vbatusov@cse.yorku.ca

York University

September 16, 2020

- Required reading: Russell & Norvig, Chapter 8
- Optional reading: same, Chapter 7

Knowledge Representation

- Example: understanding a children's story
- How do we test understanding?

  For one, the subject must be able to answer (correctly) simple questions about the story.

Example: Three Little Pigs

Figure: Pigs build houses using different techniques

Example: Three Little Pigs

Figure: Wolf huffs and puffs

Example: Three Little Pigs

- Why couldn't the wolf blow down the house made of bricks?
- What background knowledge are we drawing on to reach that conclusion?
  - Brick structures are stronger than straw and stick structures
  - Objects such as the wolf have physical limitations. The wolf can only blow so hard.

Knowledge Representation

- Operating in our world requires vast amounts of knowledge
- Also requires reasoning with that knowledge
  - It is doubtful any one of us has studied the blowing ability of wolfs
  - But by knowing the general rules of our world, we can derive this
  - We employ reasoning to make conclusions about the wolf

  - Generally, reasoning effectively compresses knowledge so we don't need to store it as such. Without reasoning, we'd need to store unimaginably many trivial facts.

    Things that can't fit into a teacup: elephants, cars, bricks, shoes, whole coconuts, large dogs, small dogs, etc.

Logical Representations

AI typically employs logical representations of knowledge

- They are mathematically precise, so amenable to analysis (properties, computational complexity of inference, etc.)
- They are formal languages, so can be mechanically manipulated
- They have a formal **syntax** and a formal **semantics**
- They usually have well-developed **proof theories** — formal procedures for reasoning (deriving new statements from the old)
- They are generally **declarative**, easy to extend

Model-Theoretic Semantics

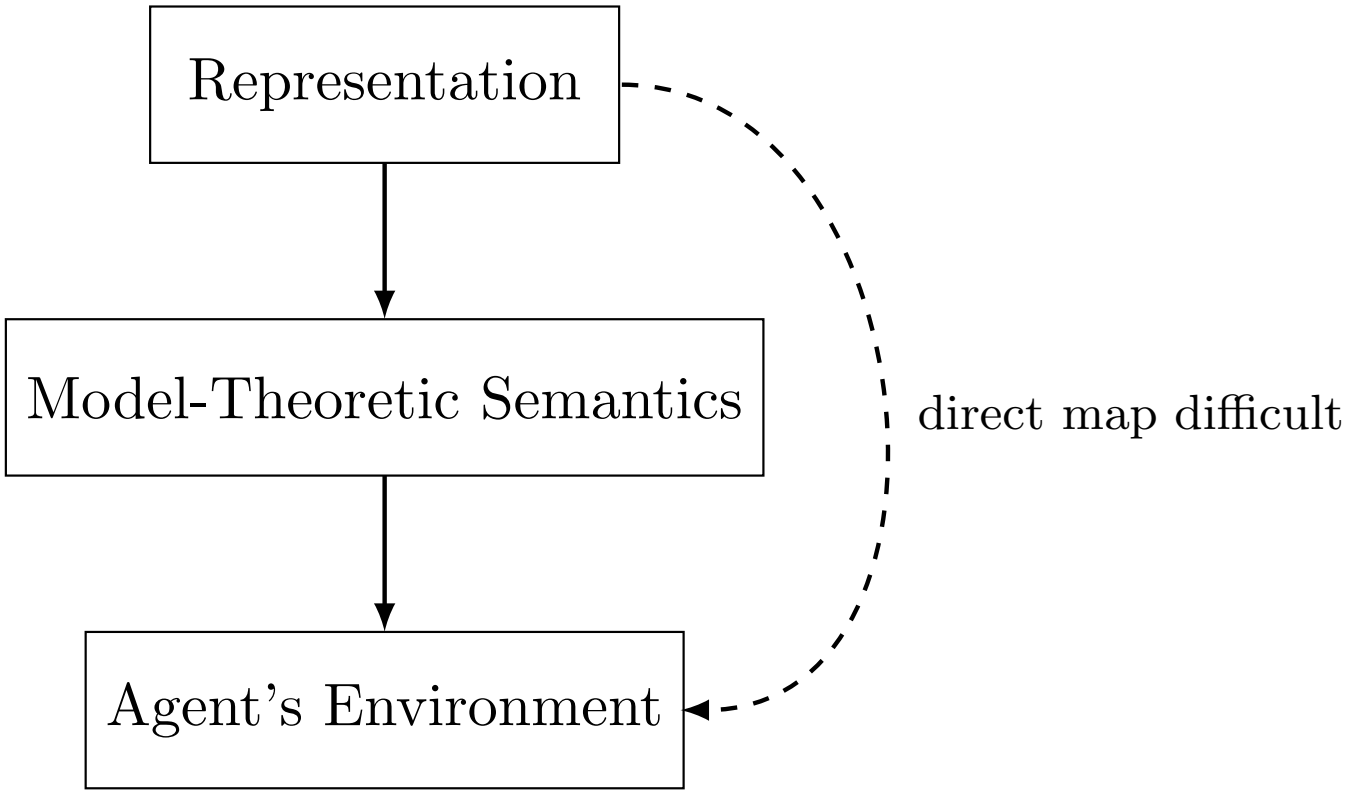
- Suppose our knowledge is represented by some collection of **strings** (**sentences**)
- Generally, what is the **meaning** of a sentence?
  A mapping: $\{\text{set of sentences}\} \mapsto \{\text{features of the world}\}$
- Want to provide an interpretation of every piece of our representation
- Like having an intuitive understanding of what individual statements in a program mean. If you know what the separate instructions mean, can figure out what the whole program does.

Model-Theoretic Semantics

Model-theoretic semantics

- is a formal characterization (in terms of set theory)
- can be used to prove a wide range of properties of the representation
- maps arbitrarily complex sentences of the language (logic) down into intuitive assertions about the real world
- is based on notions that are very close to how we think about the real world. Thus, it provides a bridge from the syntax to an intuitive understanding of what is being asserted

Model-Theoretic Semantics



Semantics: formal details

- A set of **objects**

  Stand for distinct identifiable objects that are important for your application
- Distinguished subsets of objects — **Properties**
- Distinguished sets of tuples of objects — **Relations**
- Distinguished functions mapping tuples of objects to objects — **Functions**

Example

Try viewing the world in the terms of set theory

- Objects:
  students, subjects, assignments, numbers
- Predicates:
  $difficult(subject)$, $cs\_major(student)$
- Relations
  $handed\_in(student, assignment)$
- Functions:
  $grade(student, assignment) \mapsto number$

First-Order Logic (FOL)

Syntax A grammar specifying what are the legal syntactic constructs of the representation

Semantics A formal mapping from syntactic constructs to set-theoretic assertions

FOL Syntax

Symbol = a unique artifact, no matter what it is.
Example: Unicode symbols, digits, emojis, whatever — but needs to be distinguishable from other symbols. Contrary to common usage, and for the purposes of convenience, we consider a string of characters to be a single symbol.

We will need symbols to represent:

- **constants**
- **variables**
- **functions**[1]
- **predicates**[1]

[1] These are associated with an *arity*, the number of arguments

FOL Syntax: building terms

A **term** is either

- a variable
- a constant
- an expression $f(t_1, \ldots, t_k)$ where
  - f is a function symbol
  - k is its arity
  - t_i (for $1 \leq i \leq k$) is a *term*[2]

Think: term = expression that denotes an *object*
Example: if our objects are numbers, then $1 + 2$ is a term that denotes 3

[2] notice induction

FOL Syntax: atomic formulas

An **atom** (or *atomic formula*) is an expression $p(t_1, \ldots, t_k)$ where

- p is a predicate symbol
- k is its arity
- t_i $(1 \leq i \leq k)$ is a term

Note: constants are functions without arguments

Notation: we will use UPPER CASE for variables, lower case for everything else. (This is what Prolog does, let's stay consistent with it)

Semantic Intuition — Terms

Terms denote objects (individuals)

- Constants denote specific objects

  bill, *jane*
- Variables range over individuals

  X — could be *jane* or *bill* or any other object in our universe
- Functions map tuples of objects to other objects

  $father(jane)$, $mother(father(jane))$, $mother(X)$, $distance\_between(X, Y)$

Semantic Intuition — Formulas

Atoms denote facts about the world (can only be **true** or **false**)

- $father\_of(bill, jane)$ — "Bill is the father of Jane"
- $female(jane)$
- $satisfied(client15)$
- $satisfied(C)$
- $desires(client15, rome, week29)$
- $desires(X, Y, Z)$
- $star\_rating(hotel7, 4)$

FOL Syntax — Formulas (I)

- Atoms are formulas ("atomic formulas")
- If ϕ is a formula, then its **negation** $\neg\phi$ is also a formula

  $\neg\phi$ is true whenever ϕ is false
- If $\phi_1, \ldots, \phi_n$ are formulas, then their **conjunction** $\phi_1 \wedge \phi_2 \wedge \ldots \wedge \phi_n$ is also a formula

  $\phi_1 \wedge \phi_2 \wedge \ldots \wedge \phi_n$ is true whenever **every** ϕ_i $(1 \leq i \leq n)$ is true
- If $\phi_1, \ldots, \phi_n$ are formulas, then their **disjunction** $\phi_1 \vee \phi_2 \vee \ldots \vee \phi_n$ is also a formula

  $\phi_1 \vee \phi_2 \vee \ldots \vee \phi_n$ is true whenever **at least one** of ϕ_i $(1 \leq i \leq n)$ is true

FOL Syntax — Formulas (II)

Recall:

$\exists$ Existential quantifier — "There exists..."

$\forall$ Universal quantifier — "For all..."

- If ϕ is a formula, then $\exists X(\phi)$ is also a formula

  Asserts that there is an object such that, if X is bound to it, ϕ becomes true

- If ϕ is a formula, then $\forall X(\phi)$ is also a formula

  Asserts that ϕ is true for every single binding of X

FOL Syntax — Some useful abbreviations

- *Implication* — "if . . . then . . . "

$$\phi_1 \rightarrow \phi_2 \qquad \text{means } \neg\phi_1 \vee \phi_2$$

- *Double (bi-directional) implication* — "if and only if"

$$\phi_1 \leftrightarrow \phi_2 \qquad \text{means } (\phi_1 \rightarrow \phi_2) \wedge (\phi_2 \rightarrow \phi_1)$$

Standard rules for connective precedence apply,
i.e. $\phi_1 \wedge \phi_2 \vee \phi_3$ is $(\phi_1 \wedge \phi_2) \vee \phi_3$

FOL Semantics

- Formulas ban be built up recursively, can be arbitrarily complex
- Syntactically distinct formulas may be logically equivalent

$$\forall X, Y(elephant(X) \wedge teacup(Y) \rightarrow larger\_than(X, Y))$$
$$\forall X, Y(teacup(Y) \wedge elephant(X) \rightarrow larger\_than(X, Y))$$

- The purpose of semantics is to capture this equivalence and, more generally, to make sense of complex formulas

FOL Semantics

- Semantics = a formal mapping from formulas to semantic entities (individuals, sets, relations and functions over individuals)

$$\textit{meaning} : \{\text{formulas}\} \mapsto \{\text{set-theoretic notions}\}$$

- This mapping mirrors the recursive nature of the syntax. Thus, we can give any formula (no matter how complex) a mapping to semantic entities

FOL Semantics — Formal Definition

First, we fix the language. The language $\mathcal{L}$ is defined by its primitive symbols: the sets $\mathcal{F}, \mathcal{P}, \mathcal{V}$

- $\mathcal{F}$ — a set of function symbols (inc. constants)

  Each symbol $f \in \mathcal{F}$ has a particular arity

- $\mathcal{P}$ — a set of predicate and relation symbols

  Each symbol $p \in \mathcal{P}$ has a particular arity

- $\mathcal{V}$ — an infinite set of variables

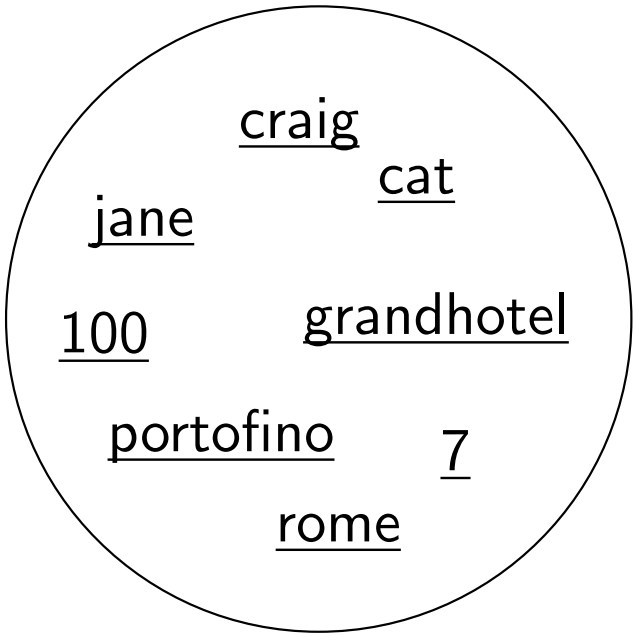
FOL Semantics — Formal Definition

An **interpretation** (structure) is a tuple $\langle \mathcal{D}, \Phi, \Psi, \nu \rangle$, where

- $\mathcal{D}$ is a non-empty set (domain of individuals)
  "universe of discourse"
- $\Phi : \mathcal{F} \mapsto (\mathcal{D}^k \mapsto \mathcal{D})$
  maps every k-ary *function symbol* $f \in \mathcal{F}$ to a k-ary *function* over $\mathcal{D}$
  Careful: f is a symbol, $\Phi(f)$ is the corresponding function over the domain
- $\Psi : \mathcal{P} \mapsto (\mathcal{D}^k \mapsto \{true, false\})$
  maps every k-ary *predicate symbol* $p \in \mathcal{P}$ to an *indicator function* over k-ary tuples of individuals
- $\nu : \mathcal{V} \mapsto \mathcal{D}$ is a variable assignment function. Simply maps every variable to some domain object.

Intuitions

- The domain $\mathcal{D}$ is just a set:



- (Underlined symbols denote domain individuals to avoid confusion. They are not symbols of the language)
- Domains are usually infinite, but let's use finite ones to prime our intuitions

Intuitions for Φ

Recall: $\Phi : \mathcal{F} \mapsto (\mathcal{D}^k \mapsto \mathcal{D})$
Given a k-ary function $f \in \mathcal{F}$ and k individuals, $\Phi(f)$ tells us what $f(d_1, \ldots, d_k)$ is.

- 0-ary functions (constants) are mapped to specific individuals in $\mathcal{D}$
  $\Phi(\mathit{client17}) = \underline{\mathit{craig}}$, $\Phi(\mathit{rome}) = \underline{\mathit{rome}}$
- 1-ary functions are mapped to functions in $\mathcal{D} \mapsto \mathcal{D}$
  $\Phi(\mathit{min\_quality}) = f_{\mathit{min\_quality}}$, $f_{\mathit{min\_quality}}(\underline{\mathit{craig}}) = \underline{3\_\mathit{stars}}$
- 2-ary functions are mapped to functions in $\mathcal{D}^2 \mapsto \mathcal{D}$
  $\Phi(\mathit{distance}) = f_{\mathit{distance}}$, $f_{\mathit{distance}}(\underline{\mathit{toronto}}, \underline{\mathit{sienna}}) = \underline{3256}$
- And so on for n-ary functions

Intuitions for Ψ

Recall: $\Psi : \mathcal{P} \mapsto (\mathcal{D}^k \mapsto \{true, false\})$
Given a k-ary predicate $p \in \mathcal{P}$ and k individuals, $\Psi(p)$ tells us whether the relation denoted by p holds for these particular individuals.

- 0-ary predicates are mapped to true or false
  $\Psi(rainy) = True$, $\Psi(sunny) = False$
- Unary predicates are mapped to indicator functions of subsets of $\mathcal{D}$
  $\Psi(satisfied) = p_{satisfied}$, $p_{satisfied}(\underline{craig}) = True$
- Binary predicates are mapped to indicator functions of subsets of $\mathcal{D}^2$
  $\Psi(location) = p_{location}$, $p_{location}(\underline{grandhotel}, \underline{rome}) = True$,
  $p_{location}(\underline{grandhotel}, \underline{sienna}) = False$
- And so on for n-ary predicates

Intuitions for ν

- Only takes care of quantification. The exact mapping it specifies does not really matter, as we will see later.
- Notation: $\nu[X/d]$ is a *new* variable assignment function, which is exactly just like ν except that it maps the variable X to the individual d. Otherwise, $\nu(Y) = \nu[X/d](Y)$.

FOL Semantics

GIven a language $\mathcal{L}$ and an interpretation $\mathcal{I} = \langle \mathcal{D}, \Phi, \Psi, \nu \rangle$,

- Constant c denotes an individual
  $\mathcal{I}(c) = \Phi(c) \in \mathcal{D}$
- Variable X denotes an individual
  $\mathcal{I}(X) = \nu(X) \in \mathcal{D}$
- A complex term $t = f(t_1, \ldots, t_k)$ denotes an individual
  $\mathcal{I}(t) = \Phi(f)(\mathcal{I}(t_1), \ldots, \mathcal{I}(t_k)) \in \mathcal{D}$
  We recursively find the denotation of each term and then apply the function denoted by f to get the individual.
  Thus, **terms always denote individuals** under an interpretation $\mathcal{I}$

End of lecture

Next time: FOL Semantics continued, examples of interpretations