EECS 3401 — AI and Logic Prog. — Lecture 3

Adapted from slides of Prof. Yves Lesperance

Vitaliy Batusov
vbatusov@cse.yorku.ca

York University

September 21, 2020

- Continued from Lecture 2
- Required reading: Russell & Norvig, Chapter 8
- Optional reading: same, Chapter 7

Recap

- We want to represent knowledge
- We are using First-Order Logic (FOL)
- FOL consists of **Syntax** and **Semantics**
- Syntax — how to form sentences

  a formal grammar
- Semantics — how to give sentences meaning

  map syntactic constructs to set-theoretic constructs

FOL Syntax

We will need **symbols** to represent

- **constants** a, b, cat, $client17$
- **variables** X, Y, $Parent$
- **functions** $weight(\cdot)$, $sum(\cdot, \cdot, \cdot)$, $common\_parent(\cdot, \cdot)$
- **predicates** $heavy(\cdot)$, $siblings(\cdot, \cdot, \cdot)$, $greater\_than(\cdot, \cdot)$

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*[1]

Think: term = expression that denotes an *object*

[1] 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

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 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

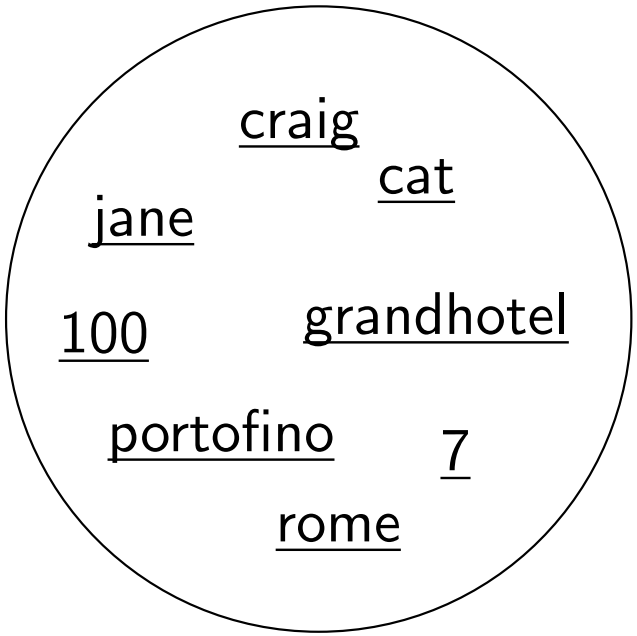
FOL Semantics — Formal Definition

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, $\nu(Y) = \nu[X/d](Y)$.

Symbols, Terms, and Notation

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

- c is a constant **symbol** and also a **term**,
  but $\mathcal{I}(c)$ is an **object** in $\mathcal{D}$
- X is a variable **symbol** and also a **term**,
  but $\mathcal{I}(X)$ is an **object** in $\mathcal{D}$
- $f(\cdot, \cdot)$ is a function **symbol**, $f(c, X)$ is a **term**,
  but $\mathcal{I}(f(c, X))$ is an **object** in $\mathcal{D}$
- $p(\cdot, \cdot)$ is a predicate **symbol**, $p(c, X)$ is an **atom**,
  but $\mathcal{I}(p(c, X))$ is either **True** or **False**

Can also write $\mathcal{I}(c)$ as $c^{\mathcal{I}}$ — looks less cumbersome
So, c is a term, but $c^{\mathcal{I}}$ is the object it refers to according to $\mathcal{I}$

FOL Semantics — Interpreting Terms

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

- Constant c denotes an individual

$$c^{\mathcal{I}} = \Phi(c) \in \mathcal{D}$$

- Variable X denotes an individual

$$X^{\mathcal{I}} = \nu(X) \in \mathcal{D}$$

- A complex term $f(t_1, \ldots, t_k)$ denotes an individual

$$(f(t_1, \ldots, t_k))^{\mathcal{I}} = \Phi(f)(t_1^{\mathcal{I}}, \ldots, t_k^{\mathcal{I}}) \in \mathcal{D}$$

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 $\mathcal{I}$

FOL Semantics — Interpreting Formulas

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

- An atom $p(t_1, \ldots, t_k)$ has the **truth value**

$$(p(t_1, \ldots, t_k))^{\mathcal{I}} = \Psi(p)(t_1^{\mathcal{I}}, \ldots, t_k^{\mathcal{I}}) \quad \in \{True, False\}$$

- A more intuitive way:
  For each k-ary predicate symbol p, $\mathcal{I}$ supplies a set $p^{\mathcal{I}} \subseteq \mathcal{D}^k$, i.e., some set of k-tuples over $\mathcal{D}$. Then, if the tuple of individuals $(t_1^{\mathcal{I}}, \ldots, t_k^{\mathcal{I}})$ is in this set $p^{\mathcal{I}}$, then the atom is true in $\mathcal{I}$.

FOL Semantics — Interpreting Complex Formulas

- $(\neg\phi)^{\mathcal{I}}$ is **True** if and only if $\phi^{\mathcal{I}}$ is **False**
- $(\phi_1 \wedge \ldots \wedge \phi_n)^{\mathcal{I}}$ is **True** if every $\phi_i^{\mathcal{I}}$ $(1 \leq i \leq n)$ is **True**, and **False** otherwise
- $(\phi_1 \vee \ldots \vee \phi_n)^{\mathcal{I}}$ is **True** if at least one of $\phi_i^{\mathcal{I}}$ $(1 \leq i \leq n)$ is **True**. If all are false, then the conjunction is **False**.

FOL Semantics — Formulas with Quantifiers

- $(\exists X(\phi))^{\mathcal{I}}$ is **True** if
  - *there exists* an object $\underline{d} \in \mathcal{D}$
  - such that $\phi^{\mathcal{I}'}$ is **True**,
  - where $\mathcal{I}' = \langle \mathcal{D}, \Phi, \Psi, \nu[X/\underline{d}] \rangle$

$\mathcal{I}'$ is exactly like $\mathcal{I}$ except its variable assignment ν maps the variable X to that special $\underline{d} \in \mathcal{D}$

FOL Semantics — Formulas with Quantifiers

- $(\forall X(\phi))^{\mathcal{I}}$ is **True** if
  - *for every* object $\underline{d} \in \mathcal{D}$,
  - $\phi^{\mathcal{I}'}$ is **True**,
  - where $\mathcal{I}' = \langle \mathcal{D}, \Phi, \Psi, \nu[X/\underline{d}] \rangle$

Example

Consider the statement

$$\forall X(happy(X)) \qquad \text{“everybody is happy”}$$

Is this a true statement in the following interpretation $\mathcal{I}$?

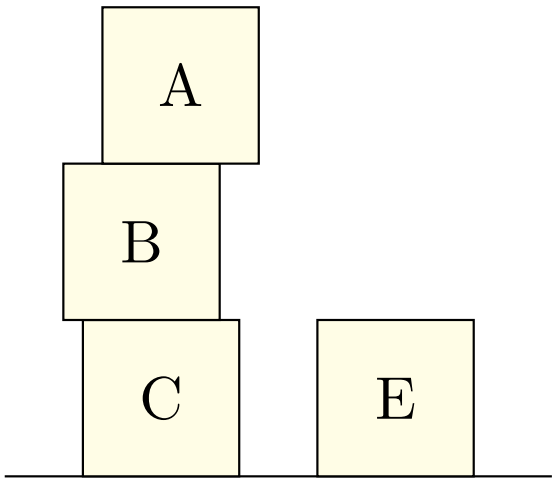
$$\begin{aligned} \mathcal{I}: \quad & \mathcal{D} = \{\underline{bob}, \underline{jack}, \underline{fred}\} \\ & \Psi(happy)(\underline{bob}) = True \\ & \Psi(happy)(\underline{jack}) = True \\ & \Psi(happy)(\underline{fred}) = False \end{aligned}$$

- Test $happy(X)$ under all $\mathcal{I}' = \langle \mathcal{D}, \Phi, \Psi, \nu[X/\underline{d}] \rangle$, $\underline{d} \in \mathcal{D}$

$$\begin{aligned} \text{for } \nu[X/\underline{bob}]: \quad & (happy(X))^{\mathcal{I}'} = \Psi(happy)(\underline{bob}) = True \\ \text{for } \nu[X/\underline{jack}]: \quad & (happy(X))^{\mathcal{I}'} = \Psi(happy)(\underline{jack}) = True \\ \text{for } \nu[X/\underline{fred}]: \quad & (happy(X))^{\mathcal{I}'} = \Psi(happy)(\underline{fred}) = False \end{aligned}$$

Example — Blocks World

Environment



Language

Constants a, b, c, e

Functions none

Predicates
- $on(\cdot, \cdot)$
- $above(\cdot, \cdot)$
- $clear(\cdot)$
- $ontable(\cdot)$

Example — Blocks World

Language

Constants a, b, c, e

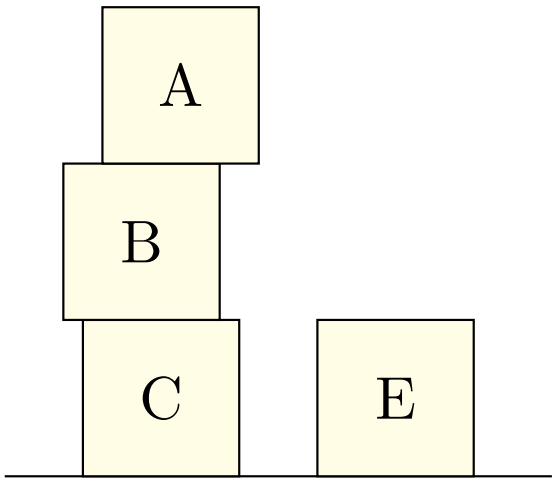
Predicates
- $on(\cdot, \cdot)$
- $above(\cdot, \cdot)$
- $clear(\cdot)$
- $ontable(\cdot)$

A possible interpretation $\mathcal{I}_1$

- $\mathcal{D} = \{\underline{A}, \underline{B}, \underline{C}, \underline{E}\}$
- $\Phi(a) = \underline{A}$, $\Phi(b) = \underline{B}$, $\Phi(c) = \underline{C}$, $\Phi(e) = \underline{E}$
- $\Psi(on) = \{(\underline{A}, \underline{B}), (\underline{B}, \underline{C})\}$, $\Psi(above) = \{(\underline{A}, \underline{B}), (\underline{B}, \underline{C}), (\underline{A}, \underline{C})\}$, $\Psi(clear) = \{\underline{A}, \underline{E}\}$, $\Psi(ontable) = \{\underline{C}, \underline{E}\}$

Example — Blocks World

Environment



A possible interpretation $\mathcal{I}_1$

- $\mathcal{D} = \{\underline{A}, \underline{B}, \underline{C}, \underline{E}\}$
- $\Phi(a) = \underline{A}$, $\Phi(b) = \underline{B}$, $\Phi(c) = \underline{C}$, $\Phi(e) = \underline{E}$
- $\Psi(on) = \{(\underline{A}, \underline{B}), (\underline{B}, \underline{C})\}$, $\Psi(above) = \{(\underline{A}, \underline{B}), (\underline{B}, \underline{C}), (\underline{A}, \underline{C})\}$, $\Psi(clear) = \{\underline{A}, \underline{E}\}$, $\Psi(ontable) = \{\underline{C}, \underline{E}\}$

Example — Blocks World

Interpretation $\mathcal{I}_1$

- $\mathcal{D} = \{\underline{A}, \underline{B}, \underline{C}, \underline{E}\}$
- $\Phi(a) = \underline{A}$, $\Phi(b) = \underline{B}$, $\Phi(c) = \underline{C}$, $\Phi(e) = \underline{E}$
- $\Psi(on) = \{(\underline{A}, \underline{B}), (\underline{B}, \underline{C})\}$, $\Psi(above) = \{(\underline{A}, \underline{B}), (\underline{B}, \underline{C}), (\underline{A}, \underline{C})\}$, $\Psi(clear) = \{\underline{A}, \underline{E}\}$, $\Psi(ontable) = \{\underline{C}, \underline{E}\}$

Let's see what holds in $\mathcal{I}_1$

$\forall X \forall Y (on(X, Y) \rightarrow above(X, Y))$

- check $X = \underline{A}, Y = \underline{B}$ — ok
- check $X = \underline{B}, Y = \underline{C}$ — ok

$\forall X \forall Y (above(X, Y) \rightarrow on(X, Y))$

- check $X = \underline{A}, Y = \underline{B}$ — ok
- check $X = \underline{B}, Y = \underline{C}$ — ok
- check $X = \underline{A}, Y = \underline{C}$

Example — Blocks World

Interpretation $\mathcal{I}_1$

- $\mathcal{D} = \{\underline{A}, \underline{B}, \underline{C}, \underline{E}\}$
- $\Phi(a) = \underline{A}$, $\Phi(b) = \underline{B}$, $\Phi(c) = \underline{C}$, $\Phi(e) = \underline{E}$
- $\Psi(on) = \{(\underline{A}, \underline{B}), (\underline{B}, \underline{C})\}$, $\Psi(above) = \{(\underline{A}, \underline{B}), (\underline{B}, \underline{C}), (\underline{A}, \underline{C})\}$, $\Psi(clear) = \{\underline{A}, \underline{E}\}$, $\Psi(ontable) = \{\underline{C}, \underline{E}\}$

$\forall X \exists Y (clear(X) \vee on(Y, X))$

- check $X = \underline{A}$ — ok
- check $X = \underline{C}$, $Y = \underline{B}$ — ok
- ...

$\exists Y \forall X (clear(X) \vee on(Y, X))$

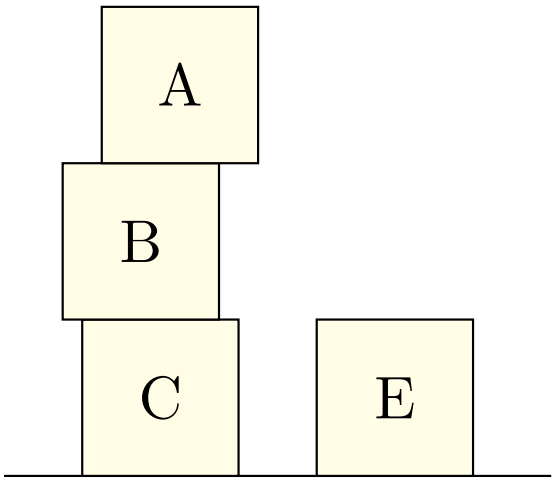
- check $Y = \underline{A}$ – no! $(X = \underline{C})$
- check $Y = \underline{C}$– no! $(X = \underline{B})$
- check $Y = \underline{E}$ – no! $(X = \underline{B})$
- check $Y = \underline{B}$– no! $(X = \underline{B})$

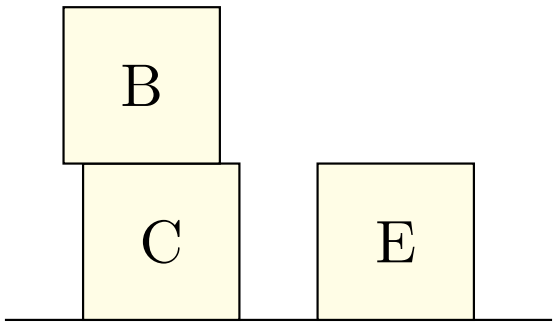
Knowledge Bases and Their Models

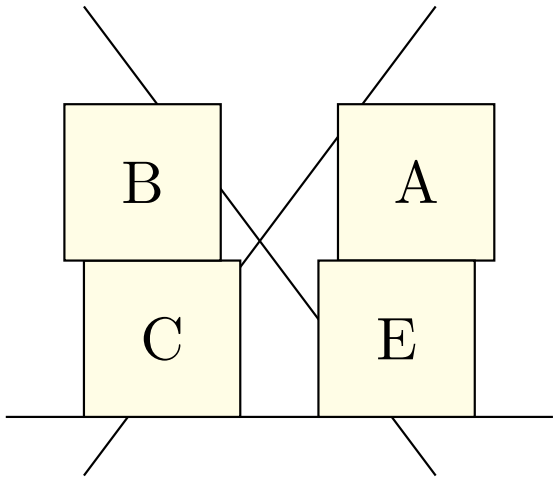
KB:

$$on(b, c)$$
$$clear(e)$$

Various models:







Models

- A formula ϕ may be true or false in a given interpretation $\mathcal{I}$
- If a formula ϕ is true in $\mathcal{I}$, we say that $\mathcal{I}$ is a **model** of ϕ
  Written: $\mathcal{I} \models \phi$ ($\mathcal{I}$ *satisfies* ϕ)
- A **knowledge base** (KB) is a set of formulas (axioms)
- If $\mathcal{I} \models \phi$ for every $\phi \in \text{KB}$, then $\mathcal{I} \models \text{KB}$
  "$\mathcal{I}$ is a **model** of the knowledge base"

The Purpose of Models

- When we create a knowledge base (a set of logical axioms), we intend that the real world (i.e., our set-theoretic abstraction of it) is one of its models
- If it is, then every statement in the KB is true in the real world
- Thus, when reasoning in the KB, we can derive true statements about the real world
- Note: not every thing that is true in reality need be contained in the KB. KB's are typically incomplete in this sense.

Models Support Reasoning

- Suppose some formula ϕ is **not** a part of our KB, but is nevertheless true in every model of the KB, i.e.,

$$\text{for every } \mathcal{I} \text{ such that } \mathcal{I} \models \text{KB}, \text{ we also get } \mathcal{I} \models \phi$$

- We say that ϕ is a **logical consequence** of the KB ("KB **entails** ϕ")
- Since the real world is a model of the KB, then ϕ must be true in the real world
- Thus, entailment is a way of finding new true facts that were not mentioned in the KB

Ponder: if the KB does **not** entail ϕ, does it mean that ϕ is false?

Example of Entailment

- *elephant*(*clyde*)
  The individual denoted by the symbol *clyde* is in the set denoted by the symbol *elephant*
- *teacup*(*cup*)
  cup is a teacup
- A unary predicate p always denotes (is interpreted as) *a set* $p^{\mathcal{I}}$ *of individuals*. Asserting a unary predicate to hold for a term t means that the individual $t^{\mathcal{I}}$ denoted by that term belongs to the set $p^{\mathcal{I}}$.
  Formally, we defined $p^{\mathcal{I}}$ to be an indicator function. Seen in the above sense, the indicator function returns true or false depending on whether the individual is in the set $p^{\mathcal{I}}$.

Example of Entailment

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

- For every pair of individuals: if the first is an elephant and the second is a teacup, then the pair of objects are related to each other by the *larger_than* relation.
- For pairs of individuals who are not elephants and teacups, the formula is immediately true.

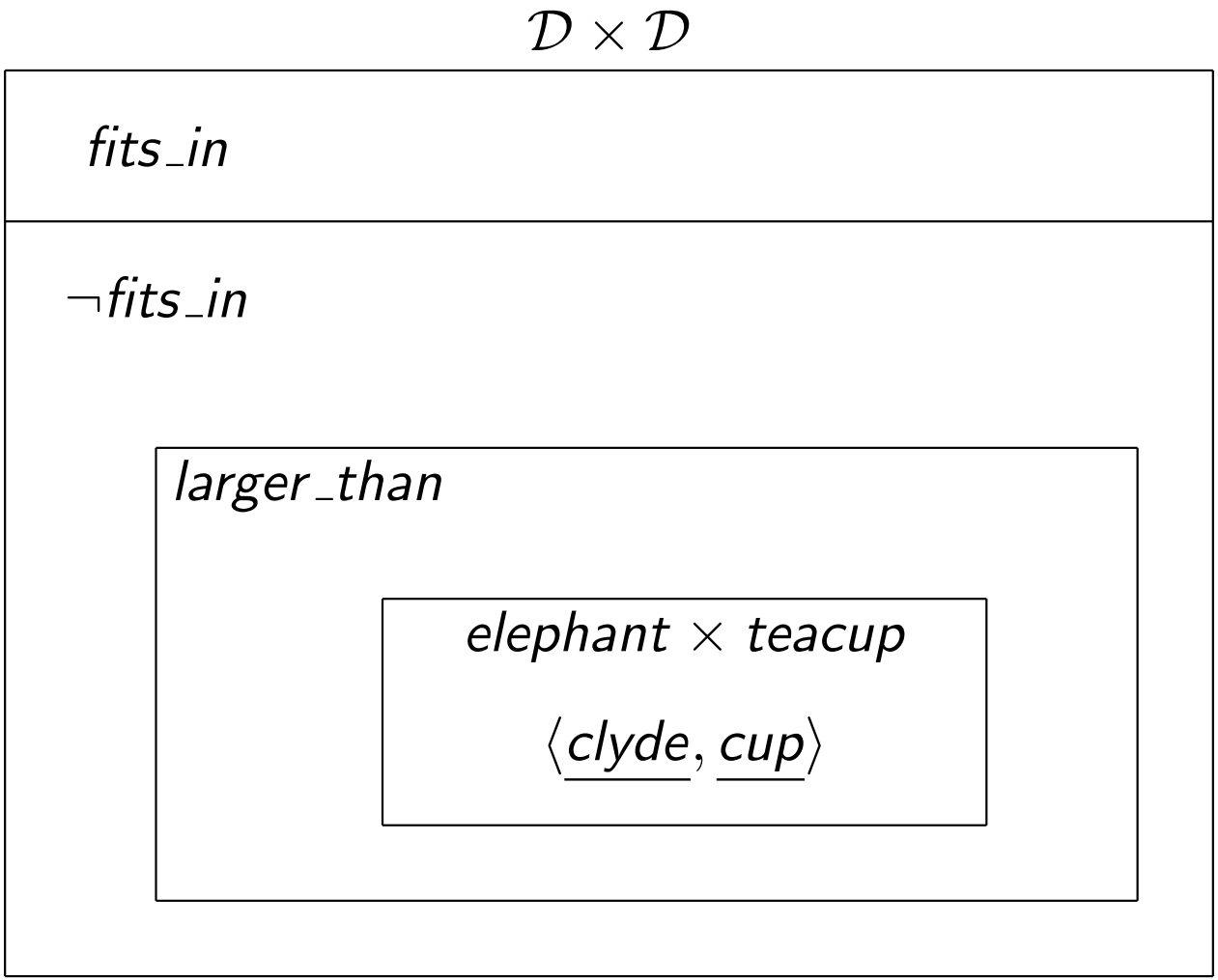
  Recall: $A \rightarrow B$ stands for $\neg A \vee B$

Example of Entailment

$$\forall X \forall Y (\mathit{larger\_than}(X, Y) \rightarrow \neg \mathit{fits\_in}(X, Y))$$

- For every pair of individuals: if X is larger than Y (the pair is in the *larger_than* relation), then we cannot have that X fits inside Y (the pair cannot be in the *fits_in* relation)
- (The intersection of the relations *larger_than* and *fits_in* is the empty set)

Example of Entailment



Example of Entailment

- If an interpretation satisfies the KB, then the set of pairs $elephant \times teacup$ must be a subset of $larger\_than$, which is disjoint from $fits\_in$
- Therefore, $\langle \underline{clyde}, \underline{cup} \rangle$ must be in the complement of the set $fits\_in$
- Therefore, $\neg fits\_in(clyde, cup)$ must be true in every model of the KB
- Therefore,

$$\neg fits\_in(clyde, cup)$$

  is a logical consequence of the above assertions.
- New knowledge!

  Well, not technically new. This statement does not appear explicitly in the KB, but it is contained in (is entailed by) the statements that do

  In the same sense, Euclid's axioms contain the Pythagoras' Theorem

Interpretations and Models

- The more sentences in the KB, the fewer models (satisfying interpretations) there are
- The more axioms you write down, the "closer" you get to the "real world", because each sentence in the KB rules out certain unintended or unwanted models
- This is called **axiomatizing the domain**

Knowledge Representation

- **Knowledge Representation** is the area of AI that focuses on how to represent information about an application domain
- An important aspect is specifying the concepts and relations of interest and how they relate — the domain's *ontology*
- Many large ontologies have been created

  Cyc General-purpose ontology, more than 1 millon terms

  DBpedia represents the content of Wikipedia in a formal language

  SNOMED CT medical terms
- KR is the basis of the Semantic Web, which seeks to make information on the web machine-processable
- Important application is ontology-based data access
- Tools for building ontologies (Protégé)

Knowledge Representation Formalisms

- Many formal languages for representing information
- FOL is quite expressive, but checking entailment in FOL is undecidable in general
- **Description logics** are decidable fragments of FOL for KR and reasoning. Concepts are organized in a hierarchy
- **OWL** (Web Ontology Language) is a family of description logics standardized by the W3C
- Reasoners have been developed for such formalisms (RACER, FACT++)

End of lecture

Next time: Logic programming and Prolog