

# Logic: Datalog

CPSC 322 – Logic 6

Textbook §12.2

# Lecture Overview

- 1 Recap
- 2 Datalog
- 3 Datalog Syntax
- 4 Datalog Semantics

# Bottom-up proof procedure

$KB \vdash g$  if  $g \subseteq C$  at the end of this procedure:

$C := \{\}$ ;

**repeat**

**select** clause " $h \leftarrow b_1 \wedge \dots \wedge b_m$ " in  $KB$  such that  
         $b_i \in C$  for all  $i$ , and  $h \notin C$ ;

$C := C \cup \{h\}$

**until** no more clauses can be selected.

- This procedure is both sound and complete.

# Top-down definite clause interpreter

To solve the query  $?q_1 \wedge \dots \wedge q_k$ :

$ac := \text{“}yes \leftarrow q_1 \wedge \dots \wedge q_k\text{”}$

**repeat**

**select** atom  $a_i$  from the body of  $ac$ ;

**choose** clause  $C$  from  $KB$  with  $a_i$  as head;

replace  $a_i$  in the body of  $ac$  by the body of  $C$

**until**  $ac$  is an answer.

Recall:

- **Don't-care nondeterminism** If one selection doesn't lead to a solution, there is no point trying other alternatives. **select**
- **Don't-know nondeterminism** If one choice doesn't lead to a solution, other choices may. **choose**

# Lecture Overview

- 1 Recap
- 2 Datalog**
- 3 Datalog Syntax
- 4 Datalog Semantics

# Objects and Relations

- It is useful to view the world as consisting of **objects** and **relationships** between these objects.
- Often the propositions we spoke about before can be condensed into a much smaller number of propositions if they are allowed to express relationships between objects and/or functions of objects.
- Thus, reasoning in terms of objects and relationships can be simpler than reasoning in terms of features, as you can express more general knowledge using logical variables.

# Using an RRS

- 1 Begin with a task domain.
- 2 Distinguish those **objects** you want to talk about.
- 3 Determine what **relationships** you want to represent.
- 4 Choose symbols in the computer to denote objects and relations.
- 5 Tell the system knowledge about the domain.
- 6 Ask the system questions.

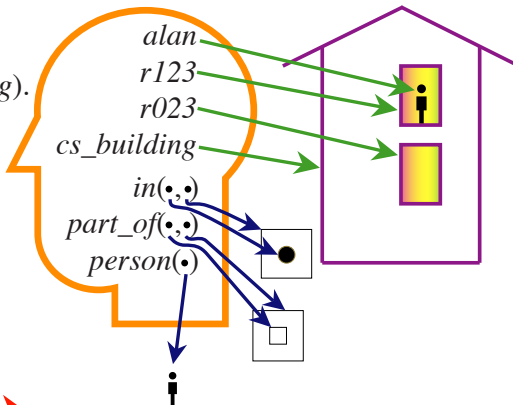
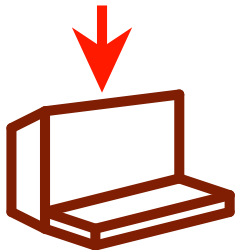
# Example Domain for an RRS

$$\text{in}(\text{alan}, r123).$$

$$\text{part\_of}(r123, \text{cs\_building}).$$

$$\text{in}(X, Y) \leftarrow$$

$$\text{part\_of}(Z, Y) \wedge$$

$$\text{in}(X, Z).$$


$$\text{in}(\text{alan}, \text{cs\_building})$$



# Representational Assumptions of Datalog

- An agent's knowledge can be usefully described in terms of *individuals* and *relations* among individuals.
- An agent's knowledge base consists of *definite* and *positive* statements.
- The environment is *static*.
- There are only a finite number of individuals of interest in the domain. Each individual can be given a unique name.

⇒ Datalog

# Lecture Overview

- 1 Recap
- 2 Datalog
- 3 Datalog Syntax**
- 4 Datalog Semantics

# Syntax of Datalog

## Definition (variable)

A **variable** starts with upper-case letter.

## Definition (constant)

A **constant** starts with lower-case letter or is a sequence of digits.

## Definition (term)

A **term** is either a variable or a constant.

## Definition (predicate symbol)

A **predicate symbol** starts with lower-case letter.

# Syntax of Datalog (cont)

## Definition (atom)

An **atomic symbol** (atom) is of the form  $p$  or  $p(t_1, \dots, t_n)$  where  $p$  is a predicate symbol and  $t_i$  are terms.

## Definition (definite clause)

A **definite clause** is either an atomic symbol (a fact) or of the form:

$$\underbrace{a}_{\text{head}} \leftarrow \underbrace{b_1 \wedge \dots \wedge b_m}_{\text{body}}$$

where  $a$  and  $b_i$  are atomic symbols.

## Definition (knowledge base)

A **knowledge base** is a set of definite clauses.

# Example Knowledge Base

$$\begin{aligned} in(kevin, R) &\leftarrow \\ &\quad teaches(kevin, cs322) \wedge \\ &\quad in(cs322, R). \\ grandfather(william, X) &\leftarrow \\ &\quad father(william, Y) \wedge \\ &\quad parent(Y, X). \\ slithy(foves) &\leftarrow \\ &\quad mimsy \wedge borogroves \wedge \\ &\quad outgrabe(mome, Raths). \end{aligned}$$

# Lecture Overview

- 1 Recap
- 2 Datalog
- 3 Datalog Syntax
- 4 Datalog Semantics**

# Semantics: General Idea

- Recall: a **semantics** specifies the meaning of sentences in the language.
  - ultimately, we want to be able to talk about which sentences are true and which are false
- In propositional logic, all we needed to do in order to come up with an interpretation was to assign truth values to atoms
- For Datalog, an **interpretation** specifies:
  - what objects (individuals) are in the world
  - the correspondence between symbols in the computer and objects & relations in world
    - which constants denote which individuals
    - which predicate symbols denote which relations (and thus, along with the above, which sentences will be true and which will be false)

# Formal Semantics

## Definition (interpretation)

An **interpretation** is a triple  $I = \langle D, \phi, \pi \rangle$ , where

- $D$ , the **domain**, is a nonempty set. Elements of  $D$  are **individuals**.
- $\phi$  is a mapping that assigns to each constant an element of  $D$ . Constant  $c$  **denotes** individual  $\phi(c)$ .
- $\pi$  is a mapping that assigns to each  $n$ -ary predicate symbol a relation: a function from  $D^n$  into  $\{TRUE, FALSE\}$ .



# Example Interpretation

**Constants:** *phone*, *pencil*, *telephone*.

**Predicate Symbol:** *noisy* (unary), *left\_of* (binary).

- $D = \{\langle \text{✂} \rangle, \langle \text{☎} \rangle, \langle \text{✎} \rangle\}$ .
  - These are actually objects in the world, not symbols
- $\phi(\text{phone}) = \langle \text{☎} \rangle$ ,  $\phi(\text{pencil}) = \langle \text{✎} \rangle$ ,  $\phi(\text{telephone}) = \langle \text{☎} \rangle$ .

•  $\pi(\text{noisy})$ : 

$\langle \text{✂} \rangle$	FALSE	$\langle \text{☎} \rangle$	TRUE	$\langle \text{✎} \rangle$	FALSE
----------------------------	-------	----------------------------	------	----------------------------	-------

$\pi(\text{left\_of})$ :

$\langle \text{✂}, \text{✂} \rangle$	FALSE	$\langle \text{✂}, \text{☎} \rangle$	TRUE	$\langle \text{✂}, \text{✎} \rangle$	TRUE
$\langle \text{☎}, \text{✂} \rangle$	FALSE	$\langle \text{☎}, \text{☎} \rangle$	FALSE	$\langle \text{☎}, \text{✎} \rangle$	TRUE
$\langle \text{✎}, \text{✂} \rangle$	FALSE	$\langle \text{✎}, \text{☎} \rangle$	FALSE	$\langle \text{✎}, \text{✎} \rangle$	FALSE

# Important points to note

- The domain  $D$  can contain **real objects**. (e.g., a person, a room, a course).  $D$  can't necessarily be stored in a computer.
- The constants do not have to match up one-to-one with members of the domain. Multiple constants can refer to the **same object**, and some objects can have **no constants** that refer to them.
- $\pi(p)$  specifies whether the relation denoted by the  $n$ -ary predicate symbol  $p$  is true or false for each  $n$ -tuple of individuals.
- If predicate symbol  $p$  has **no arguments**, then  $\pi(p)$  is either *TRUE* or *FALSE*.
  - this was the situation in propositional logic

# Truth in an interpretation

## Definition (truth in an interpretation)

- A constant  $c$  **denotes in  $I$**  the individual  $\phi(c)$ .
  - Ground (variable-free) atom  $p(t_1, \dots, t_n)$  is
    - **true in interpretation  $I$**  if  $\pi(p)(t'_1, \dots, t'_n) = \text{TRUE}$ , where  $t_i$  denotes  $t'_i$  in interpretation  $I$  and
    - **false in interpretation  $I$**  if  $\pi(p)(t'_1, \dots, t'_n) = \text{FALSE}$ .
  - Ground clause  $h \leftarrow b_1 \wedge \dots \wedge b_m$  is
    - **false in interpretation  $I$**  if  $h$  is *false* in  $I$  and each  $b_i$  is *true* in  $I$ , and item **true in interpretation  $I$**  otherwise.
    - A knowledge base,  $KB$ , is **true in interpretation  $I$**  if and only if every clause in  $KB$  is *true* in  $I$ .
- 
- Notice that truth values are only associated with **predicates** (atomic symbols; clauses), not variables and constants!

# Example Truths

In the interpretation given before:

*noisy(phone)*

# Example Truths

In the interpretation given before:

*noisy(phone)*

*noisy(telephone)*

*true*

# Example Truths

In the interpretation given before:

*noisy(phone)*

*true*

*noisy(telephone)*

*true*

*noisy(pencil)*

# Example Truths

In the interpretation given before:

<i>noisy(phone)</i>	<i>true</i>
<i>noisy(telephone)</i>	<i>true</i>
<i>noisy(pencil)</i>	<i>false</i>
<i>left_of(phone, pencil)</i>	

# Example Truths

In the interpretation given before:

<i>noisy(phone)</i>	<i>true</i>
<i>noisy(telephone)</i>	<i>true</i>
<i>noisy(pencil)</i>	<i>false</i>
<i>left_of(phone, pencil)</i>	<i>true</i>
<i>left_of(phone, telephone)</i>	



# Example Truths

In the interpretation given before:

<i>noisy(phone)</i>	<i>true</i>
<i>noisy(telephone)</i>	<i>true</i>
<i>noisy(pencil)</i>	<i>false</i>
<i>left_of(phone, pencil)</i>	<i>true</i>
<i>left_of(phone, telephone)</i>	<i>false</i>
<i>noisy(pencil) ← left_of(phone, telephone)</i>	

# Example Truths

In the interpretation given before:

<i>noisy(phone)</i>	<i>true</i>
<i>noisy(telephone)</i>	<i>true</i>
<i>noisy(pencil)</i>	<i>false</i>
<i>left_of(phone, pencil)</i>	<i>true</i>
<i>left_of(phone, telephone)</i>	<i>false</i>
<i>noisy(pencil) ← left_of(phone, telephone)</i>	<i>true</i>
<i>noisy(pencil) ← left_of(phone, pencil)</i>	

# Example Truths

In the interpretation given before:

<i>noisy(phone)</i>	<i>true</i>
<i>noisy(telephone)</i>	<i>true</i>
<i>noisy(pencil)</i>	<i>false</i>
<i>left_of(phone, pencil)</i>	<i>true</i>
<i>left_of(phone, telephone)</i>	<i>false</i>
<i>noisy(pencil) ← left_of(phone, telephone)</i>	<i>true</i>
<i>noisy(pencil) ← left_of(phone, pencil)</i>	<i>false</i>
<i>noisy(phone) ← noisy(telephone) ∧ noisy(pencil)</i>	

# Example Truths

In the interpretation given before:

<i>noisy(phone)</i>	<i>true</i>
<i>noisy(telephone)</i>	<i>true</i>
<i>noisy(pencil)</i>	<i>false</i>
<i>left_of(phone, pencil)</i>	<i>true</i>
<i>left_of(phone, telephone)</i>	<i>false</i>
<i>noisy(pencil) ← left_of(phone, telephone)</i>	<i>true</i>
<i>noisy(pencil) ← left_of(phone, pencil)</i>	<i>false</i>
<i>noisy(phone) ← noisy(telephone) ∧ noisy(pencil)</i>	<i>true</i>

# Variables

How do we determine the truth value of a clause that includes variables?

## Definition (variable assignment)

A **variable assignment** is a function from variables into the domain.

- Given an interpretation and a variable assignment, each term denotes an individual and each clause is either true or false.
- A clause containing variables is true in an interpretation if it is true **for all** variable assignments.
  - Variables are **universally quantified** in the scope of a clause.

# Models and logical consequences

## Definition (model)

A **model** of a set of clauses is an interpretation in which all the clauses are *true*.

## Definition (logical consequence)

If  $KB$  is a set of clauses and  $g$  is a conjunction of atoms,  $g$  is a **logical consequence** of  $KB$ , written  $KB \models g$ , if  $g$  is *true* in every model of  $KB$ .

- That is,  $KB \models g$  if there is no interpretation in which  $KB$  is *true* and  $g$  is *false*.