

---

## Overview

---

- ⇒ Representation and Reasoning System
- Syntax of Datalog
- Semantics of Datalog
- Adding Variables to Semantics
- Models
- Logical Consequence
- Two Views of Semantics

---

## Previous Class

---

- Introduced a task domain: smart burglar alarm system
- Introduced the symbolic approach
  - Symbols have meaning to the knowledge engineer
  - + Facts about the world
  - Symbols used to build a knowledge base that computer is told about
  - + Rules about the world
- Computer reasons with the facts and rules to make new conclusions

---

## Representation and Reasoning System

---

- A Representation and Reasoning System (RRS) is made up of
  - Formal language (syntax):
    - + Specifies the legal sentences (the range of things that can be said)
  - Semantics:
    - + Specifies the meaning of the symbols (for your domain)
    - + Specifies what is a correct conclusion
  - Reasoning theory or proof procedure:
    - + Specification of how an answer can be produced
    - + Can be nondeterministic
- Implementation of an RRS
  - Reasoning procedure
  - + Resolves nondeterminism of reasoning theory

---

## Different RRS's

---

- Different RRS's
  - With different syntaxes
    - + Actually different connectors: ways to build complex expressions
  - Or with different semantics for connectives
- Different RRS's good for different domains
- The richer the syntax, the more difficult the reasoning procedure
  - ⇒ Choose the simplest RRS possible for your application

---

## Simplifying Assumptions of Initial RRS

---

- 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
  - Only a finite number of individuals of interest in the domain
  - Each individual can be given a unique name
- ⇒ Datalog

---

## Overview

---

- Representation and Reasoning System
- ⇒ Syntax of Datalog
- Semantics of Datalog
  - Adding Variables to Semantics
  - Models
  - Logical Consequence
  - Two Views of Semantics

---

## Syntax of Datalog

---

- **Variable**
  - starts with upper-case letter
- **Constant**
  - starts with lower-case letter or is a sequence of digits (numeral)
- **Predicate symbol**
  - starts with lower-case letter
- **Term**
  - either a variable or a constant
- **Atomic symbol (atom)**
  - of the form  $p$  or  $p(t_1, \dots, t_n)$  where  $p$  is a predicate symbol and  $t_i$  are terms

---

## More Syntax of Datalog

---

- **Definite Clause**
  - either an atomic symbol (a fact) or of the form  
 $a \leftarrow b_1 \wedge \dots \wedge b_m$
- **Query**
  - of the form  $?b_1 \wedge \dots \wedge b_m$
- **Knowledge Base**
  - set of definite clauses

$\Rightarrow$  **Syntax allows us to write sentences about the world**

- Whether sentences are true or not depends on what the symbols mean, which will be specified by the semantics

---

## Example

---

- **Knowledge base**
  - hasdetector(foyer)*
  - hasdetector(diningroom)*
  - ...
  - nodetector(parlour)*
  - ...
  - room(X) ← hasdetector(X)*
  - room(X) ← nodetector(X)*
- *diningroom*, *parlour*, and *foyer* are constants
- *room*, *hasdetector*, and *nodetector* are predicate symbols
- *X* is a variable
- Whether knowledge base is correct depends on semantics

---

## Overview

---

- Representation and Reasoning System
- Syntax of Datalog
  - ⇒ Semantics of Datalog
- Adding Variables to Semantics
- Models
- Logical Consequence
- Two Views of Semantics

---

## Semantics

---

### Semantics concerns two things

- Set of individuals in the domain, and relations between them
    - What individuals and relations you choose depends on what you want to reason about
    - Individuals could even be abstract things like colors, if that is what you want to reason about
  - How constants and predicate symbols in the syntax correspond to the individuals and relations in the domain
- We call this an interpretation:
- A domain, and a mapping from the syntax to the domain

---

## Interpretation

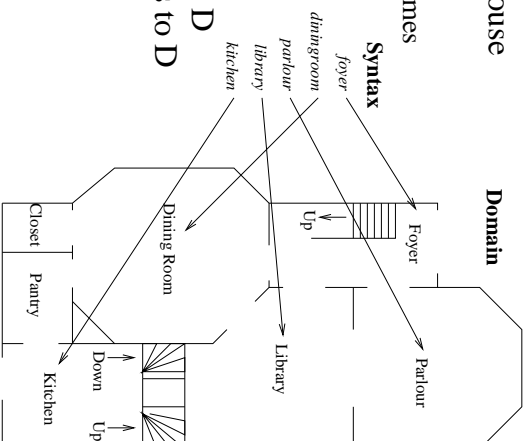
---

An interpretation is a triple  $I = (D, \phi, \pi)$  where

- $D$  the domain, is a nonempty set. Elements of  $D$  are individuals
- $\phi$  maps each constant to an element of  $D$   
Constant  $c$  denotes individual  $\phi(c)$ .
- $\pi$  maps each n-ary predicate symbol to subset of  $D^n$   
- NOTE: it does not map it to a subset of constants<sup>n</sup>  
Common mistake, don't make it on your homework

## Example Interpretation: Burglar Alarm

- $D$  is the set of rooms in the house  
Foyer, Dining Room, Library, ...
- It is the actual rooms, not the names
- $\phi$  maps constants of syntax to objects in the domain  
 $\phi(\text{foyer}) = \text{Foyer}$   
...
- Knowledge Engineer decides  $D$  and mapping of all constants to  $D$



## Example Continued

- Foyer, Library, Dining Room and Kitchen have a detector
- Closet, Pantry, Parlour do not have a detector
- Lets have  $\pi$  map  
*hasdetector* to  $\{\langle \text{Foyer} \rangle, \langle \text{Library} \rangle, \langle \text{Dining Room} \rangle, \langle \text{Kitchen} \rangle\}$   
*nodelector* to  $\{\langle \text{Closet} \rangle, \langle \text{Pantry} \rangle, \langle \text{Parlour} \rangle\}$
- Knowledge Engineer decides on mapping of predicates
  - Must decide on the mapping for all predicates
  - Hence, must do mapping for *room*, even if no facts in  $KB$  about *room*
- This is an example of an *intended interpretation*:
  - The interpretation that the knowledge engineer has in mind when coming up with language and knowledge base

## Second Example

---

- **Example:** (focus on all interpretations, not just intended one)
  - Language with constants  $a$  and  $b$  and 1-ary predicate  $girl(-)$
  - Domain with  $D = \{x, y, z\}$
  - How many different  $\phi$ 's?

	$\phi_i(a)$	$\phi_i(b)$
$\phi_1$		
$\phi_2$		
$\phi_3$		
$\phi_4$		
$\phi_5$		
$\phi_6$		
$\phi_7$		
$\phi_8$		
$\phi_9$		

## Example Continued

---

- **How many  $\pi$ 's?**

	$\pi_i(girl(x))$	$\pi_i(girl(y))$	$\pi_i(girl(z))$
$\pi_1$			
$\pi_2$			
$\pi_3$			
$\pi_4$			
$\pi_5$			
$\pi_6$			
$\pi_7$			
$\pi_8$			

- **How many different interpretations are there altogether**  
(different combinations of  $\phi$  and  $\pi$ )?

## Determining Truth of Ground Atoms

---

- Ground atom has no variables
- $p(t_1, \dots, t_n)$  maps to true if  $(\phi(t_1), \dots, \phi(t_n)) \in \pi(p)$  otherwise to false
- What does *hasdetector*(*foyer*) map to?
  - $\phi(\text{foyer}) = \text{Foyer}$
  - $\pi(\text{hasdetector}) = \{ \langle \text{Foyer} \rangle, \langle \text{Library} \rangle, \langle \text{Dining Room} \rangle, \langle \text{Kitchen} \rangle \}$
  - $\langle \text{Foyer} \rangle \in \{ \langle \text{Foyer} \rangle, \langle \text{Library} \rangle, \langle \text{Dining Room} \rangle, \langle \text{Kitchen} \rangle \}$
  - So it maps to true
- For predicates without arguments  
 $\pi(p)$  is either the set with the empty tuple  $\{ \langle \rangle \}$  or it is empty  $\{ \}$
- Semantics of Ground Atoms comes from interpretation

## Semantics of Connectives

---

- Still need to specify what ‘ $\wedge$ ’ and ‘ $\leftarrow$ ’ mean

$p$	$q$	$p \wedge q$	$p \leftarrow q$
true	true	true	true
true	false	false	true
false	true	false	false
false	false	false	true

- Thus  $h \leftarrow b_1 \wedge \dots \wedge b_m$  is false in interpretation  $I$  if  $h$  is false in  $I$  and each  $b_i$  is true in  $I$
- Semantics of ‘ $\wedge$ ’ and ‘ $\leftarrow$ ’ part of Datalog

---

## Limitations of Datalog

---

*hasdetector(foyer).*  
*nodetector(parlour).*

...

- Both predicates needed
  - Since Datalog does not include an operator that means negation
- No way to ensure that only one of *hasdetector* and *nodetector* is true for any room
  - Up to knowledge engineer to make sure that each room is true of one and only one of *detector* and *nodetector*
  - More expressive formalism would have been nice

---

## Overview

---

- Representation and Reasoning System
- Syntax of Datalog
- Semantics of Datalog
  - ⇒ Adding Variables to Semantics
- Models
- Logical Consequence
- Two Views of Semantics

---

## Semantics & Variables

---

- How do we interpret clauses such as  $room(X) \leftarrow hasdetector(X)$
- Clause is true if it is true for all values of  $X$ 
  - $room(X)$  must be true whenever  $hasdetector(X)$  is true
  - Remember, knowledge engineer had to specify mapping for all predicates, even room
- It really has a universal quantifier
  - For all  $X$   $room(X) \leftarrow hasdetector(X)$
- So, variables have an implicit universal quantifier over the clause

---

## Variable Assignment: Formal Definition

---

- Define a variable assignment  $\rho$ 
  - Maps each variable to some object in the domain
- Together  $\rho$  and  $\phi$  assign each term to some object in the domain
- Together  $\rho$  and interpretation  $I$  map every clause to true or false
  - + Even ungrounded ones
- Now we can say:
  - A clause is true in an interpretation if it is true for all variable assignments

---

## Overview

---

- Representation and Reasoning System
- Syntax of Datalog
- Semantics of Datalog
- Adding Variables to Semantics
- ⇒ Models
- Logical Consequence
- Two Views of Semantics

---

## Sets of Clauses

---

- A set of clauses is true in an interpretation if each clause is true in the interpretation
- Note that we universally quantify for the variables over each clause they used different variables
- In other words, if two clauses use the same variables, it is the same as if
  - $hasdetector(foyer)$
  - $nodetector(parlour)$
  - $room(X) \leftarrow hasdetector(X)$
  - $room(X) \leftarrow nodetector(X)$

## Models

---

- A model of a set of clauses is an interpretation in which all the clauses are true
  - Start with KB and look at what interpretations can be true

- Example  $KB$ :

$p \leftarrow q$ .  
 $q$ .

	$\pi(p)$	$\pi(q)$	$\pi(p \leftarrow q)$	Model of KB?
$I_1$	TRUE	TRUE		
$I_2$	TRUE	FALSE		
$I_3$	FALSE	TRUE		
$I_4$	FALSE	FALSE		

## Example with constants

---

- **Example:** (focus on all interpretations, not just intended one)
  - + Language with constants  $a$  and  $b$  and 1-ary predicate  $girl(-)$
  - + Domain with  $D = \{x, y, z\}$
  - + 9  $\phi^i$ 's and 8  $\pi^i$ 's, so 72 interpretations
- **How many models of  $KB = \{girl(a), girl(b)\}$ ?**  
(Checking each would take too long, so lets break down into subcases)
  - Case 1:  $\phi_i(a) = \phi_i(b)$ 
    - + How many of the 9  $\phi_i$ 's have  $\phi_i(a) = \phi_i(b)$
    - + When  $\phi_i(a) = \phi_i(b) = x$ , which  $\pi^i$ 's make KB true?
    - + So how many models with  $\phi_i(a) = \phi_i(b)$
  - Case 2:  $\phi_i(a) \neq \phi_i(b)$ 
    - + How many of the 9  $\phi^i$ ?
    - + When  $\phi_i(a) = x$  and  $\phi_i(b) = y$ , which  $\pi^i$ 's make the KB true?
    - + So how many models with  $\phi_i(a) \neq \phi_i(b)$ ?

---

## Overview

---

- Representation and Reasoning System
- Syntax of Datalog
- Semantics of Datalog
- Adding Variables to Semantics
- Models
- ⇒ Logical Consequence
- Two Views of Semantics

---

## 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  $KB$ .
  - This tells us that our  $KB$ , by its definition, always forces  $g$  to be true
  - Other terms that mean same thing:
    - $g$  logically follows from  $KB$
    - $KB$  entails  $g$
- That is,  $KB \models g$  if there is no interpretation in which  $KB$  is true and  $g$  is false.
- $KB \not\models g$  if  $g$  is not a logical consequence of  $KB$

## Example Revisited

---

- $KB$ :

$p \leftarrow q$ .

$q$ .

	$\pi(p)$	$\pi(q)$	$\pi(p \leftarrow q)$	model of KB?
$I_1$	TRUE	TRUE		
$I_2$	TRUE	FALSE		
$I_3$	FALSE	TRUE		
$I_4$	FALSE	FALSE		

- Does  $KB \models p$ ?

## Overview

---

- Representation and Reasoning System
  - Syntax of Datalog
  - Semantics of Datalog
  - Adding Variables to Semantics
  - Models
  - Logical Consequence
- $\Rightarrow$  Two Views of Semantics

---

## User's View of Semantics

---

- Choose a task domain: intended interpretation
- Associate constants with individuals you want to name
- For each relation you want to represent, associate a predicate symbol in the language
- Tell the system clauses that are true in the intended interpretation: *axiomatizing the domain*
  - hopefully you tell it enough knowledge about the domain so that it can conclude everything you want it to
- Ask questions about your domain

---

## Computer's view of semantics

---

- Computer given the knowledge base
  - The computer doesn't have access to the intended interpretation
- User asks if a question  $g$ 
  - Computer should answer true if  $KB \models g$ 
    - +  $g$  is true in all models, so is true in user's intended interpretation
  - Otherwise, computer should answer "I don't know"
    - + There is at least one model in which  $g$  is false
    - + Note  $g$  might have been true in user's intended interpretation. In this case, user didn't have enough clauses in the KB to sufficiently narrow down the models
- Aside: computer could answer the question by enumerating over all of the possible interpretations (model checking)
  - But number of interpretations grows quickly!:

## Summary of Semantics

---

- User has intended interpretation  
But just tells the computer a small set of facts that hopefully adequately captures the user's intended interpretation
- Computer answers true if all interpretations that make KB true (models) make the question true
  - Now we have specs for the computer's reasoning algorithm
  - It should answer yes if  $KB \models q$ , other answer don't know