# Chapter 2 & 3:
## A Representation & Reasoning System & Using Definite Knowledge

- Representations & Reasoning Systems (RRS) (2.2)
- Simplifying Assumptions of the Initial RRS (2.3)
- Datalog (2.4)
- Semantics (2.5)
- Questions & Answers (2.6)


D. Poole, A. Mackworth, and R. Goebel, *Computational Intelligence: A Logical Approach*, Oxford University Press, January 1998

## Representations & Reasoning Systems (2.2)

- A Representation and Reasoning System (RRS) is made up of:
  - Formal language: specifies the legal sentences (grammar)
    A knowledge base is a set of sentences in the language
  - Semantics: specifies the meaning of the symbols, sentences
  - Reasoning theory or proof procedure: nondeterministic specification of how an answer can be produced (inference system).

Representations & Reasoning Systems (2.2) (cont.)

## ◆Implementation of an RRS

An implementation of an RRS consists of:

- ◆ Language parser: distinguish legal sentences and maps sentences of the language into data structures (internal form).
- ◆ Reasoning procedure: implementation of reasoning theory + search strategy (solve nondeterminism).

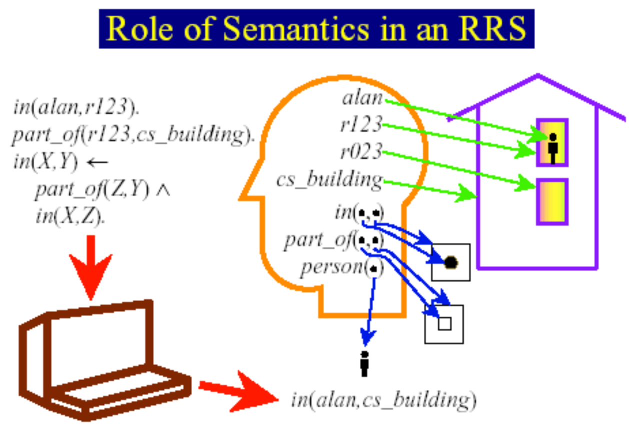
*Note:* the semantics is not reflected in the implementation, (but gives a meaning to the symbols for an external viewer!)

Representations & Reasoning Systems (2.2) (cont.)

## ◆ Using an RRS

1. Begin with a task domain.
2. Distinguish those things you want to talk about (the ontology).
3. Choose symbols in the computer to denote objects and relations.
4. Tell the system knowledge about the domain.
5. Ask the system questions.

Representations & Reasoning Systems (2.2) (cont.)




# Simplifying Assumptions of Initial RRS (2.3)

- An agent's knowledge can be usefully described in terms of individuals and relations among individuals.
- An agent's knowledge base consists of definite (not vague!) 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.

⇒ RRS that makes these assumptions is called "Datalog"

# Datalog (2.4)

- Syntax of Datalog
  - variable starts with upper-case letter (X, Room, B4….)
  - constant starts with lower-case letter or is a sequence of digits (numeral).
  - predicate symbol starts with lower-case letter (alan).
  - term is either a variable or a constant.
  - atomic symbol (atom) is of the form p or $p(t_1, \ldots, t_n)$ where p is a predicate symbol and $t_i$ are terms.

# Datalog (2.4) (cont.)

- Syntax of Datalog (cont.)

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

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

where a and $b_i$ are atomic symbols.

query is of the form $?b_1 \wedge \ldots \wedge b_m$.

knowledge base is a set of definite clauses.

## Datalog (2.4) (cont.)

- Example Knowledge Base

$in(alan, R) \leftarrow$
$\quad teaches(alan, cs322) \wedge$
$\quad in(cs322, R).$
$grandfather(william, X) \leftarrow$
$\quad father(william, Y) \wedge$
$\quad parent(Y, X).$
$slithy(toves) \leftarrow$
$\quad mimsy \wedge borogroves \wedge$
$\quad outgrabe(mome, Raths).$

## Semantics (2.5)

- General Idea
  - A semantics specifies the meaning of sentences in the language.
  - An interpretation specifies:
    - what objects (individuals) are in the world (ontology)
      the correspondence between symbols in the computer and objects & relations in world

      person(alan) = true (in this world !)
      person(r123) = false (in fig. 2.1 p. 26)

      - constants denote individuals
      - predicate symbols denote relations

## Semantics (2.5) (cont.)

- Formal Semantics
  - An interpretation is a triple $I = \langle D, \pi, \phi \rangle$ 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 function from $D^n$ into {TRUE, FALSE}.

## Semantics (2.5) (cont.)

- 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.
  - $\pi(p)$ 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.

## Semantics (2.5) (cont.)

- Truth in an interpretation
  - Each ground term (expression with no variables!) denotes an individual in an interpretation.
  - A constant c denotes in I the individual $\phi$ (c)
  - Ground (variable-free) atom $p(t_1 \ldots t_n)$ is
    - true in interpretation I if $\pi(p)(t'_1, \ldots, t'_n)$ = TRUE, where $t_i$ denotes $t'_1$ in interpretation I and
    - false in interpretation I if $\pi(p)(t'_1, \ldots, t'_n)$ = FALSE.
  - Ground clause $h \leftarrow b_1 \wedge \ldots \wedge b_m$ is false in interpretation I if h is false in I and each bi is true in I, and is true in interpretation I otherwise.

## Semantics (2.5) (cont.)

- Models and logical consequences
  - A knowledge base, KB, is true in interpretation I if and only if every clause in KB is true in I.
  - A model of a set of clauses is an interpretation in which all the clauses are true.
  - If KB is a set of clauses and g is a conjunction of atoms (or just an atom), 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.

## Semantics (2.5) (cont.)

- Simple example

$$KB = \begin{cases} p \leftarrow q. \\ q. \\ r \leftarrow s. \end{cases}$$

| | $\pi(p)$ | $\pi(q)$ | $\pi(r)$ | $\pi(s)$ | |
|---|---|---|---|---|---|
| $I_1$ | *TRUE* | *TRUE* | *TRUE* | *TRUE* | is a model of *KB* |
| $I_2$ | *FALSE* | *FALSE* | *FALSE* | *FALSE* | not a model of *KB* |
| $I_3$ | *TRUE* | *TRUE* | *FALSE* | *FALSE* | is a model of *KB* |
| $I_4$ | *TRUE* | *TRUE* | *TRUE* | *FALSE* | is a model of *KB* |
| $I_5$ | *TRUE* | *TRUE* | *FALSE* | *TRUE* | not a model of *KB* |

$KB \models p, KB \models q, KB \not\models r, KB \not\models s$

## Semantics (2.5) (cont.)

- User's view of Semantics

1. Choose a task domain: intended interpretation.
2. Associate constants with individuals you want to name.
3. For each relation you want to represent, associate a predicate symbol in the language.
4. Tell the system clauses that are true in the intended interpretation: axiomatizing the domain.
5. Ask questions about the intended interpretation.
6. If KB $\models$ g, then g must be true in the intended interpretation.

## Semantics (2.5) (cont.)

- Computer's view of semantics
  - The computer doesn't have access to the intended interpretation.
  - All it knows is the knowledge base.
  - The computer can determine if a formula is a logical consequence of KB.
  - If KB $\models$ g then g must be true in the intended interpretation.
  - If KB $\not\models$ g then there is a model of KB in which g is false. This could be the intended interpretation.

## Questions & Answers (2.6)

- Variables
  - Variables are universally quantified in the scope of a clause.
  - 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

## Questions & Answers (2.6) (cont.)

- Queries and Answers
  - A query is a way to ask if a body is a logical consequence of the knowledge base:

$$?b_1 \wedge \ldots \wedge b_m$$

  - An answer is either
    - an instance of the query that is a logical consequence of the knowledge base KB, or
    - no if no instance is a logical consequence of KB.

## Questions & Answers (2.6) (cont.)

- Examples Queries

$$KB = \begin{cases} in(alan, r123). \\ part\_of(r123, cs\_building). \\ in(X, Y) \leftarrow part\_of(Z, Y) \wedge in(X, Z). \end{cases}$$

| Query | Answer |
|---|---|
| $?part\_of(r123, B).$ | $part\_of(r123, cs\_building)$ |
| $?part\_of(r023, cs\_building).$ | $no$ |
| $?in(alan, r023).$ | $no$ |
| $?in(alan, B).$ | $in(alan, r123)$ |
| | $in(alan, cs\_building)$ |

## Questions & Answers (2.6) (cont.)

- Logical Consequence
  - Atom g is a logical consequence of KB if and only if:
    - g is a fact in KB, or
    - there is a rule

$$g \leftarrow b_1 \wedge \ldots \wedge b_k$$

in KB such that each bi is a logical consequence of KB.

## Questions & Answers (2.6) (cont.)

- Debugging false conclusions
  - To debug answer g that is false in the intended interpretation:
    - If g is a fact in KB, this fact is wrong.
    - Otherwise, suppose g was proved using the rule:

$$g \leftarrow b_1 \wedge \ldots \wedge b_k$$

where each bi is a logical consequence of KB.

- If each $b_i$ is true in the intended interpretation, then this clause is false in the intended interpretation.
- If some bi is false in the intended interpretation, debug $b_i$.

## Questions & Answers (2.6) (cont.)

◆Axiomatizing the Electrical Environment

% $light(L)$ is true if $L$ is a light

$light(l_1)$. $light(l_2)$.

% $down(S)$ is true if switch $S$ is down

$down(s_1)$. $up(s_2)$. $up(s_3)$.

% $ok(D)$ is true if $D$ is not broken

$ok(l_1)$. $ok(l_2)$. $ok(cb_1)$. $ok(cb_2)$.

$?light(l_1)$. $\Longrightarrow$ $yes$

$?light(l_6)$. $\Longrightarrow$ $no$

$?up(X)$. $\Longrightarrow$ $up(s_2), up(s_3)$

## Questions & Answers (2.6) (cont.)

$connected\_to(X, Y)$ is true if component $X$ is connected to $Y$

$connected\_to(w_0, w_1) \leftarrow up(s_2)$.

$connected\_to(w_0, w_2) \leftarrow down(s_2)$.

$connected\_to(w_1, w_3) \leftarrow up(s_1)$.

$connected\_to(w_2, w_3) \leftarrow down(s_1)$.

$connected\_to(w_4, w_3) \leftarrow up(s_3)$.

$connected\_to(p_1, w_3)$.

$?connected\_to(w_0, W)$. $\Longrightarrow$ $W = w_1$

$?connected\_to(w_1, W)$. $\Longrightarrow$ $no$

$?connected\_to(Y, w_3)$. $\Longrightarrow$ $Y = w_2, Y = w_4, Y = p_1$

$?connected\_to(X, W)$. $\Longrightarrow$ $X = w_0, W = w_1, \ldots$

## Questions & Answers (cont.)

% $lit(L)$ is true if the light $L$ is lit

$$lit(L) \leftarrow light(L) \wedge ok(L) \wedge live(L).$$

% $live(C)$ is true if there is power coming into $C$

$$live(Y) \leftarrow connected\_to(Y, Z) \wedge live(Z).$$

$$live(outside).$$

This is a recursive definition of *live*.

## Questions & Answers (cont.)

- Recursion and Mathematical Induction
  - Above(X, Y) ← on (X, Y)
  - Above(X, Y) ← on (X, Z) ∧ above (Z, Y)
  - This can be seen as:
    - Recursive definition of above: prove above in terms of a base case (on) or a simpler instance of itself; or
    - Way to prove above by mathematical induction: the base case is when there are no blocks between X and Y, and if you can prove above when there are n blocks between them, you can prove it when there are n + 1 blocks.

## Questions & Answers (2.6) (cont.)

- Make the following atoms Provable:
  - live($w_5$) given: connected_to($w_5$,outside).
  - live($w_3$)
  - live($w_4$)
  - live($l_2$)