Lecture 14: Rules and Recursion

Dr John Levine

CS103 Machines, Languages and Computation
November 13th 2015

University of Strathclyde Science

Knowledge and Inference

- Using symbolic systems to represent knowledge about the world and make legal inferences:

    1. Tom is a cat.
    2. Jerry is a mouse.
    3. Cats chase mice.
    4. Therefore, Tom chases Jerry.

- Given facts 1, 2 and 3, fact 4 is a logical consequent
- How can we represent facts and rules?
- How can we make legal inferences?

University of Strathclyde Science

Knowledge and Inference

- Using symbolic systems to represent knowledge about the world and make legal inferences:

    1. cat(tom)
    2. mouse(jerry)
    3. cat(X), mouse(Y) → chases(X,Y)
    4. chases(tom,jerry)

- Given facts 1, 2 and 3, fact 4 is a logical consequent
- We can derive fact 4 using just symbolic manipulation based on the *form* of the strings.
- We don’t have to know what the symbols mean!

University of Strathclyde Science

Knowledge and Inference

- Using symbolic systems to represent knowledge about the world and make legal inferences:

    1. foo(dur)
    2. bar(doh)
    3. foo(X), bar(Y) → spong(X,Y)
    4. spong(dur, doh)

- Given facts 1, 2 and 3, fact 4 is a logical consequent
- We can derive fact 4 using just symbolic manipulation based on the *form* of the strings.
- We don't have to know what the symbols mean!

University of Strathclyde Science

Facts and Rules

Here's an example database of facts and rules (taken from Assignment 8 in the third workbook):

cat(tom)
mouse(jerry)
dog(spike)
dog(tyke)
father(spike,tyke)
hits(tom,tyke)

cat(X), mouse(Y) → fights(X,Y)
dog(X), cat(Y) → hates(X,Y)
fights(X,Y) → fights(Y,X)
hits(X,Z), father(Y,Z) → hits(Y,X)

University of Strathclyde Science

Deriving Facts

- To derive a new fact, we apply a rule to some existing facts, setting the variables in the rule consistently
- We write a derivation like this:

  rule to be applied
  existing facts to be used
  variable values
  → *new fact*

- We are then allowed to use the new fact in further derivations.

Example Derivation 1

- Derive fights(tom,jerry)

  cat(X), mouse(Y) → fights(X,Y)
  cat(tom), mouse(jerry)
  [X = tom, Y = jerry]
  → fights(tom,jerry)

University of Strathclyde Science

Example Derivation 2

- Derive fights(jerry,tom)

  cat(X), mouse(Y) → fights(X,Y)
  cat(tom), mouse(jerry)
  [X = tom, Y = jerry]
  → fights(tom,jerry)

  fights(X,Y) → fights(Y,X)
  fights(tom,jerry)
  [X = tom, Y = jerry]
  → fights(jerry,tom)

University of Strathclyde Science

Knowledge Engineering

- The process of turning known facts about the world into facts and rules is called *knowledge engineering*
- A simple approach:
  1. Identify all the objects in the world – Tom, Jerry, Spike, Tyke. These are the things that appear inside the brackets. Give them all names.
  2. Identify the sets these can belong in – the set of all cats, the set of all mice, the set of all small things – these are the predicates, e.g. cat(tom).
  3. Identify relationships between objects – these are the relations, e.g. hates(spike,tom).

University of Strathclyde Science

Exercise

- Translate these sentences into facts and rules, using the knowledge engineering approach suggested:

  Ford is an alien.
  Arthur is a human.
  Marvin is an android.
  Humans are lifeforms.
  Aliens are lifeforms.
  Androids are cleverer than lifeforms.

- Can you derive the fact that Marvin is cleverer than Arthur?

University of Strathclyde Science

Exercise

- Objects: ford, arthur, marvin.
- Sets: alien, human, android, lifeform.
- Relations: is-cleverer-than(X,Y).

alien(ford)
human(arthur)
android(marvin)

human(X) → lifeform(X)
alien(X) → lifeform(X)
android(X), lifeform (Y) → is-cleverer-than(X,Y)

University of Strathclyde Science

Exercise

- Can you derive the fact that Marvin is cleverer than Arthur?

  human(X) → lifeform(X)
  human(arthur)
  [X = arthur]
  → lifeform(arthur)

  android(X), lifeform(Y) → is-cleverer-than(X,Y)
  android(marvin), lifeform(arthur)
  [X = marvin, Y = arthur]
  → is-cleverer-than(marvin, arthur)

University of Strathclyde Science

Recursive Logical Definitions

- Consider part of the Simpsons' family tree:

```
parent(orville,grampa)      parent(homer,bart)
parent(yuma,grampa)         parent(marge,bart)
parent(grampa,homer)        parent(homer,lisa)
parent(mona,homer)          parent(marge,lisa)
parent(jackie,marge)        parent(homer,maggie)
parent(clancy,marge)        parent(marge,maggie)
```

- Who are Bart's ancestors?
- How can we define the ancestor(X,Y) relation?

University of Strathclyde Science

Recursive Logical Definitions

- To find Bart's ancestors we first add his parents to the list of ancestors (Homer and Marge).
- Now we find Homer's ancestors and add those to the list, and then find Marge's ancestors and add those to the list.
- This procedure would go on forever, if it weren't for the fact that our knowledge of the Simpsons' family tree is limited (e.g. we don't know who Orville's parents are).
- We can write our definition in our rules…

Recursive Logical Definitions

parent(X,Y) → ancestor(X,Y).
parent(X,Y), ancestor(Y,Z) → ancestor(X,Z)

- Another way of coding the second rule, also using recursion:

  ancestor(X,Y), ancestor(Y,Z) → ancestor(X,Z)

Using the Rules Forwards

- Derive ancestor(orville, bart)

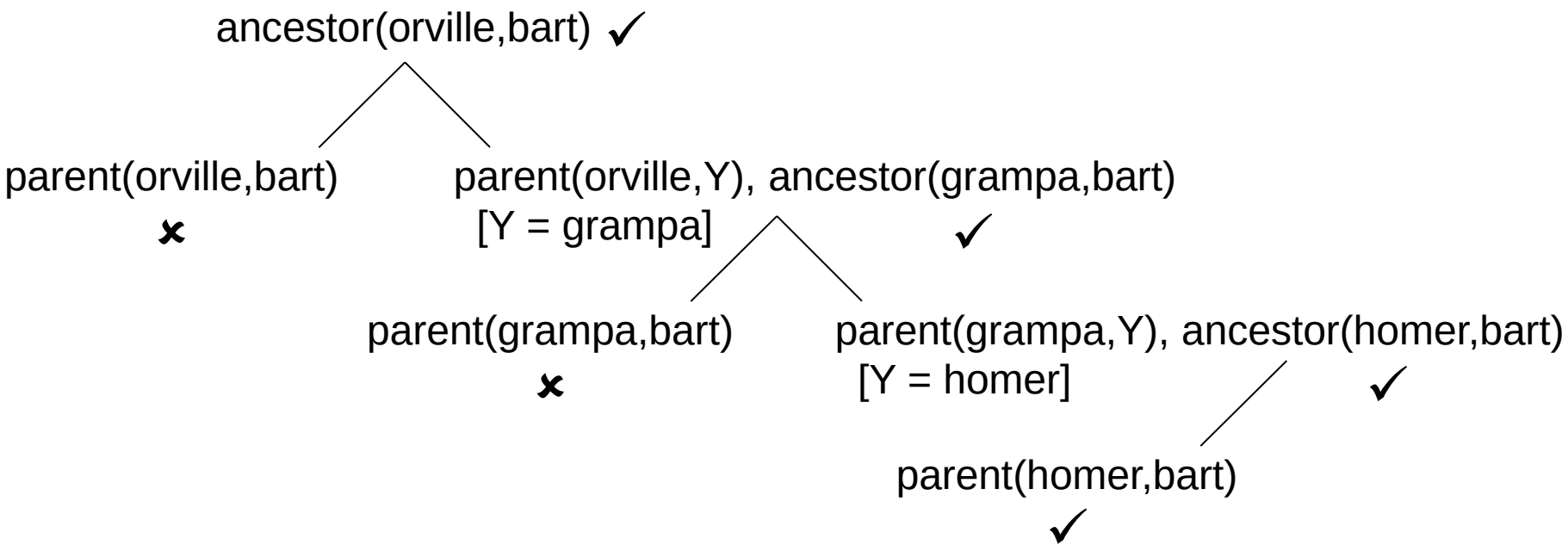
parent(X,Y) → ancestor(X,Y)
parent(homer, bart)
[X = homer, Y = bart]
→ ancestor(homer, bart)

parent(X,Y), ancestor(Y,Z) → ancestor(X,Z)
parent(grampa, homer), ancestor(homer, bart)
[X = grampa, Y = homer, Z = bart]
→ ancestor(grampa, bart)

parent(X,Y), ancestor(Y,Z) → ancestor(X,Z)
parent(orville, grampa), ancestor (grampa, bart)
[X = orville, Y = grampa, Z = bart]
→ ancestor(orville, bart)

Using the Rules Backwards

- We can also start from the fact to be proved and work backwards:



University of Strathclyde Science

Assignment 8

- Assignment 8 gives you practice in writing derivations and doing knowledge engineering with recursive rules
- Please hand your workbooks in to the office by 3pm on Wednesday
- Labs next week: coding algorithms in Python
- Class Test results and feedback on Monday (promise)