

# CS502 Fall 2006

## Assignment 3

### Solutions

#### Question 1

Consider the language with constant symbols  $a$ ,  $b$  and  $c$ , predicate symbols  $p$ ,  $q$ , and  $r$ , and the knowledge base  $KB$  that consists of the clauses:

$$p(X) \leftarrow q(X).$$

$$p(Y) \leftarrow r(Y).$$

$$q(a).$$

$$r(b).$$

Suppose there are two individuals  $fred$  and  $mary$ .

- Give a model of  $KB$  where  $D = \{fred, mary\}$ . You must specify  $\phi$  and  $\pi$ .
- Give an interpretation with the same domain that isn't a model of  $KB$ . You must specify  $\phi$  and  $\pi$ .
- Give three atoms that are logical consequences of the knowledge base.
- Give three atoms that are not logical consequences of the knowledge base.

#### Solutions

- One model is where  $a$ ,  $b$  and  $c$  all denote  $fred$  (i.e.,  $\phi(a) = fred$ ,  $\phi(b) = fred$ , and  $\phi(c) = fred$ ) and all relations are true (i.e.,  $\pi(p)(fred) = true$ ,  $\pi(p)(mary) = true$ ,  $\pi(q)(fred) = true$ ,  $\pi(q)(mary) = true$ ,  $\pi(r)(fred) = true$ ,  $\pi(r)(mary) = true$ ).

Another model is where  $a$  denotes  $fred$  and  $b$  and  $c$  denote  $mary$  (i.e.,  $\phi(a) = fred$ ,  $\phi(b) = mary$ , and  $\phi(c) = mary$ ) and the relations are as follows:  $\pi(p)(fred) = true$ ,  $\pi(p)(mary) = true$ ,  $\pi(q)(fred) = true$ ,  $\pi(q)(mary) = false$ ,  $\pi(r)(fred) = false$ ,  $\pi(r)(mary) = true$ .

Another model is where  $a$ ,  $b$  and  $c$  all denote  $fred$  (i.e.,  $\phi(a) = fred$ ,  $\phi(b) = fred$ , and  $\phi(c) = fred$ ) and the relations are as follows:  $\pi(p)(fred) = true$ ,  $\pi(p)(mary) = false$ ,  $\pi(q)(fred) = true$ ,  $\pi(q)(mary) = false$ ,  $\pi(r)(fred) = true$ ,  $\pi(r)(mary) = false$ .

- We can select an arbitrary clause and falsify it, filling the other atoms arbitrarily. For example, any interpretation with  $a$  denoting  $fred$  (i.e., with  $\phi(a) = fred$ ) and  $\pi(q)(fred) = true$  falsifies the third clause and isn't a model.

Similarly any interpretation with  $\pi(p)(fred) = false$  and  $\pi(p)(fred) = true$  falsifies the first clause, and so isn't a model, independently of the values of  $\phi$  or the other values of  $\pi$ .

So one interpretation that isn't a model is where  $a$  denotes  $fred$  and  $b$  and  $c$  denote  $mary$  (i.e.,  $\phi(a) = fred$ ,  $\phi(b) = mary$ , and  $\phi(c) = mary$ ) and the relations are as follows:  $\pi(p)(fred) = false$ ,  $\pi(p)(mary) = false$ ,  $\pi(q)(fred) = false$ ,  $\pi(q)(mary) = true$ ,  $\pi(r)(fred) = true$ ,  $\pi(r)(mary) = false$ . Every clause in the knowledge base is false in this interpretation.

- Any two of  $q(a)$ ,  $r(b)$ ,  $p(a)$ ,  $p(b)$

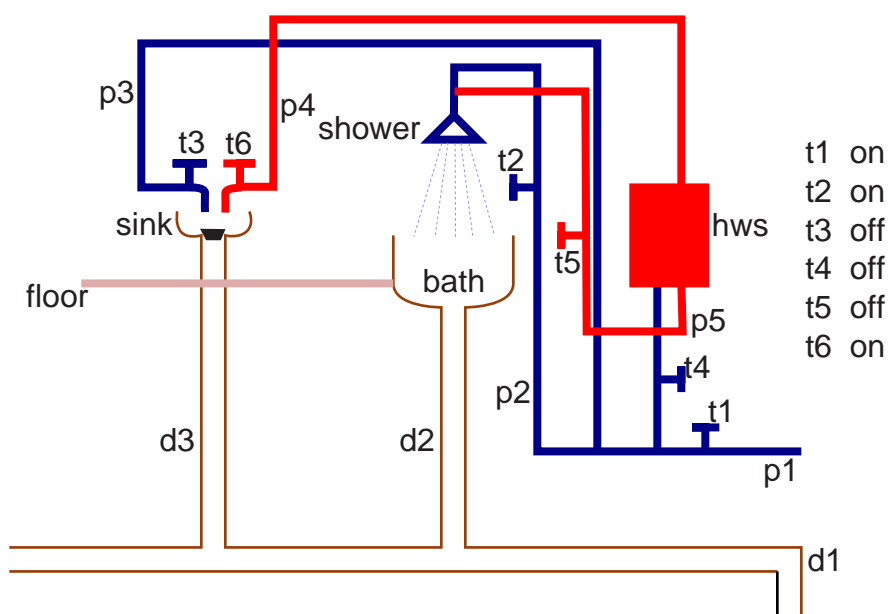


Figure 1: The Plumbing Domain

(d) Any two of  $r(a)$ ,  $q(b)$ ,  $r(c)$ ,  $q(c)$ ,  $p(c)$ .

## Question 2

Consider the domain of house plumbing represented in the diagram of Figure 1.

In this example, constants  $p1$ ,  $p2$  and  $p3$  denote cold water pipes.  $p1$  is the pipe coming in from the main water supply. Constants  $p4$  and  $p5$  represent hot water pipes (coming out from the hot water system). Constants  $t1$ ,  $t2$ ,  $t3$ ,  $t4$  and  $t5$  denote taps and  $d1$ ,  $d2$  and  $d3$  denote drainage pipes. The constants *shower* denotes a shower, *bath* denotes a bath, *sink* denotes a sink, *hws* denotes a hot water system, and *floor* denotes the floor. Figure 1 is intended to give the denotation for the constant symbols.

Suppose we have as predicate symbols:

- *pressurised*, where  $pressurised(P)$  is true if pipe  $P$  has mains pressure in it.  $p1$  is always pressurized. Other pipes are pressurized if they are connected to a pressurized pipe through an open tap.
- *on*, where  $on(T)$  is true if tap  $T$  is on.
- *off*, where  $off(T)$  is true if tap  $T$  is off.
- *wet*, where  $wet(B)$  is true if  $B$  is wet.
- *flow*, where  $flow(P)$  is true if water is flowing through  $P$ .
- *plugged*, where  $plugged(S)$  is true if  $S$  is either a sink or a bath and has the plug in.
- *unplugged*, where  $unplugged(S)$  is true if  $S$  is either a sink or a bath and doesn't have the plug in.

Assume that the taps and plugs have been in the same positions for one hour; you don't need to consider the dynamics of turning on taps and inserting and removing plugs.

The file `plumbingbuggy.pl` (available on the web and in `~cs502/cilog/`) contains a CILog axiomatization for the case where tap  $t1$  is on,  $t2$  is on,  $t3$  is off,  $t4$  is off,  $t5$  is off, and  $t6$  is on. There is a plug in the sink, but not in the bath. (This has been formatted to be unreadable, but you don't need to be able to read the program, or know how it is implemented, to be able to debug it.)

Unfortunately there is a bug in the program. We know this because we can prove that the floor is wet,

even though it isn't wet in the intended interpretation. Using CILog, your goal is to find a clause that is false in the intended interpretation, using just the intended interpretation of the symbols.

You need to hand in a trace of the CILog session, and show clearly which is the buggy rule.

### Solution

```

CILog Version 0.12. Copyright 1998, David Poole.
CILog comes with absolutely no warranty.
All inputs end with a period. Type "help." for help.
cilog: load 'plumbingbuggy.pl'.
CILog theory plumbingbuggy.pl loaded.
cilog: ask wet(floor).
Answer: wet(floor).
Runtime since last report: 0 ms.
[ok,more,how,help]: how.
wet(floor) <-
  1: wet(sink)
  2: plugged(sink)
How? [Number,up,retry,ok,prompt,help]: how 1.
wet(sink) <-
  1: on(t6)
  2: pressurised(p4)
How? [Number,up,retry,ok,prompt,help]: how 2.
pressurised(p4) <-
  1: pressurised(hws)
How? [Number,up,retry,ok,prompt,help]: how 1.
pressurised(hws) <-
  1: on(t2)
  2: pressurised(p2)
How? [Number,up,retry,ok,prompt,help]:

```

The buggy rule is:

```

pressurised(hws) <-
  on(t2)
  pressurised(p2)

```

as the body is true and the head is false in the intended interpretation.

### Question 3

It is often a problem to determine exactly where a document you printed actually got printed. It can depend on which room you selected the output to go to and which printer is actually plugged into the printer port in that room.

Suppose there are three computers, that are named  $c1$ ,  $c2$  and  $c3$ . There are two rooms,  $r101$  and  $r202$ , and three printers  $pr1$ ,  $pr2$  and  $pr3$ . Each computer has a room selected. Each room can have a printer plugged into its port.

Using CILog, we want to axiomatize a domain so that you can answer questions about which room a document was printed and which printer it was printed on.

- (a) Give the intended interpretation of each symbol used.
- (b) Give the general axioms that are true for all configurations of printers.
- (c) Show that your axiomatization works for more than one configuration.

### Solution

- (a) Give the intended interpretation of each symbol used. We assume that the denotation of the constants (i.e.,  $\phi$ ) is given as above, and that we use *doc1*, *doc2*, etc for particular documents. We assume the following meaning for the predicate symbols:
  - *computer\_selects(Comp, Room)* is true if computer *Comp* has selected room *Room* to print on.
  - *plugged\_into(Printer, Room)* is true if *Printer* is plugged into the printer port of *Room*.
  - *printed\_from(Doc, Comp)* is true if document *DOc* is printed from computer *Comp*.
  - *printed\_in\_room(Doc, Room)* is true if the document *Doc* has been printed in *Room*.
  - *on\_printer(Doc, Pr)* is true if document *Doc* has been printed on printer *Pr*.
- (b) Give the general axioms that are true for all configurations of printers.

```
printed_in_room(Doc, Room) <-
  printed_from(Doc, Comp) &
  computer_selects(Comp, Room) .
```

```
on_printer(Doc, Pr) <-
  printed_in_room(Doc, Room) &
  plugged_into(Pr, Room) .
```

- (c) Show that your axiomatization works for more than one configuration.
 

Here is one (sensible) configuration:

```
computer_selects(c1, r101) .
computer_selects(c2, r202) .
computer_selects(c3, r202) .
plugged_into(pr1, r101) .
plugged_into(pr2, r202) .
printed_from(doc1, c1) .
printed_from(doc2, c2) .
printed_from(doc3, c3) .
```

Here is one (not so sensible) configuration:

```
computer_selects(c1, r101) .
computer_selects(c2, r202) .
computer_selects(c3, r202) .
plugged_into(pr1, r101) .
plugged_into(pr2, r101) .
printed_from(doc1, c1) .
printed_from(doc2, c2) .
printed_from(doc3, c3) .
```

To think about: Where does *doc1* get printed? Where does *doc2* get printed (in what room and on what printer)?

## Question 4

Using one of the OWL editors (e.g., Protégé or SWOOP), write an ontology in OWL about buildings, that includes at least the following concepts:

Building, residential building, house, apartment building, condominium building, duplex, townhouse, owner-occupied, rental, multi-family dwelling, commercial building, mixed commercial-residential building.

You can get an OWL implementation from <http://www.cs.ubc.ca/spider/poole/cs502/2006/as3/Building.owl>. This is interesting because it defines a condominium building (CondoBuilding) in terms of the number of units and the ownership: a condominium building is a residential building, with more than two units and is owner-occupied.

## Question 5

### Solution

About 10 minutes for question 1. About 10 minutes for question 2. About 15 minutes for question 3. About 30 minutes for question 4 (once I has loaded Protege and played with it to see how it works). About 1 minute for question 5.

It was probably too easy for a graduate course.

I learned about models and interpretations and about axiomatizing a domain and having an intended interpretation for the symbols and about using an intended interpretation for debugging. I learn that it is not so easy to write an ontology.