CS322 Fall 1997
Assignment 7 (planning)

Due: 1:30pm, Friday 31 October 1997.

In the file `~cs322/cilog/delrob_sitc.pl` (also available on the web). Is an axiomatization of the delivery robot in cilog using the situation calculus. You will need to play with this to do this assignment.

Question 1

Add to the program the ability to paint an object. In particular, add the predicate:

$$colour(Obj, Col, Sit)$$

that is true if object *Obj* has colour *Col* in situation *Sit*.

Assume that the parcel starts off blue. Thus, we have an axiom:

$$colour(parcel, blue, init).$$

Assume there is an action $paint(Obj, Col)$ to paint object *Obj* colour *Col*. For this, lest assume that objects can only be painted red, and they can only be painted when the object and the robot are both at position $o109$. Assume also that colours accumulate on the robot (there is nothing that undoes an object being a color; if you paint the parcel red, it is both red and blue — of course this is unrealistic, but it makes the problem simpler).

Axiomatize the predicate *colour*, and the action *paint* using the situation calculus.

You will not get full marks if you use more that 3 clauses (as well as the clause above defining the colour in the initial situation), or if any of the clauses has more than two atomic symbols in the body. You don't need equality, inequality or negation as failure.

Your output should look something like:

```
cilog: bound 12.
cilog: ask color(parcel,red,S).
Answer:  color(parcel,red, do(paint(parcel,red),
                              do(move(rob,storage,o109),
                                do(pickup(rob,parcel),
                                  do(move(rob,o109,storage),
                                    init))))).
```

Question 2

Note that cilog does depth-bounded search. You will notice that the answer to question 1 was slow, and we needed a depth-bound that was close to the actual depth-bound to make it work.

In this question, I want you to estimate how long it will take to find a solution to the query:

```
ask sitting_at(parcel,lab2,S).
```

(don't bother to try it — it takes way too long to run, but how long, that is the question)

1. Estimate the smallest bound needed to find a plan. Hint: how many steps are needed to solve this problem? How does the number of steps relate to the depth-bound needed? Justify your estimate.

2. Estimate the branching factor of the search tree. To do this you should look at the time for a complete search at level $k+1$ versus a complete search at level k. You should justify your answer both experimentally (by running the program) and theoretically (by considering what is the branching factor). You don't need to run cases with a large runtime to do this.

3. Based on your answers to parts 1 and 2, and the time you found for some run of the program for a small bound, estimate the time for a complete search of the search tree (i.e., to find all answers) at the depth required to find a solution. Justify your solution.

There is a new implementation of cilog (at `~cs322/cilog/cilog` and on the web). that reports runtime.