Dept. of Computer Science & Engineering
York University

Assignment 3

Total marks: 90.

Out: May 7
Due: May 21 at 5pm

Note: Your report for this assignment should be the result of your own individual work. Take care to avoid plagiarism ("copying"). You may discuss the problems with other students, but do not take written notes during these discussions, and do not share your written solutions.

1. [20 points] (Adapted from Nilsson (1971)) Consider the following knowledge:

   > Tony, Mike, and John belong to the Alpine Club. Every member of the Alpine Club who is not a skier is a mountain climber. Mountain climbers do not like rain, and anyone who does not like snow is not a skier. Mike dislikes whatever Tony likes, and likes whatever Tony dislikes. Tony likes rain and snow.

   a) Write sentences in first-order logic that represent this knowledge. Also provide a glossary where you indicate the intended meaning of your predicate, function, and constant symbols in English.

   b) Convert the sentences into clausal form and give the resulting set of clauses.

   c) Use resolution with answer extraction to find the member of the Alpine Club who is a mountain climber but not a skier. State how you represent the query in first-order logic and what clause (with an answer predicate) is added to the theory. Show the complete resolution derivation (in sequence or tree form), clearly indicating which literals/clauses are sesolved and the unifier used.

2. [70 points]

   a) Develop a Prolog implementation of a situation calculus action theory for Shakey's world as described in Exercise 11.13 of the Russell and Norvig textbook. Use the primitive action names that appear in the exercise. For the fluents, use $RobotLoc(location, situation)$, meaning that the robot is at $location$ in situation s, $BoxLoc(box, location, situation)$, meaning that box is at $location$ in situation s, $OnTop(box, situation)$, meaning that the robot is on top of box in situation s, $Up(switch, situation)$, meaning that $switch$ is up in situation s,

and $LightOn(light, situation)$, meaning that $light$ is on in situation s. Also, use the non-fluent predicates $In(location, room)$ and $Controls(switch, light)$, and possibly others to specify the types of entities in the domain, e.g. $IsBox(b)$. Write a precondition axiom for each action and a successor state axiom for each fluent. Also write axioms describing the initial state pictured in Figure 11.17. Assume there is a light in each room (except the corridor), and that it is on if the switch is up. For the initial location of the robot, use a constant such as $locInitRobot$, and similarly for the boxes and switches. There is an example Prolog implementation of a situation calculus action theory for an "elevator control" application domain on the course web site.

b) Suppose that we want to achieve the goal of having Box_2 in $Room_2$. Express this goal as a situation calculus sentence. Also write a ground situation term that represents a plan that achieves this goal when executed in the initial state of Figure 11.17. Use your Prolog implementation of the action theory in a) to confirm that the plan is executable (legal) and achieves the goal.

c) Write a Golog program that represents the plan in b) and show that it can be executed successfully. Use the Golog interpreter on the course web site.

d) Write a Golog procedure $allLightsOn$ that can be executed to turn on all the lights. The procedure should always terminate successfully and should succeed in turning on all the lights as long as a there is a box in some room that can be used by the robot to reach the switch. Run your procedure and show that it can be executed successfully in the initial situation of Figure 11.17 and that all the lights are on in the resulting situation. Your code should define subprocedures as appropriate and not be unnecessarily complex.

(e) Use the Golog iterative deepening planning procedure (in the Golog interpreter file) to generate a plan to achieve the goal of having Box_2 in $Room_2$. For the search procedure to find a solution, you will have to define the forward filtering fluent $Acceptable(a, s)$ appropriately. Try using the following task-specific heuristics: i) until the robot is at the location of Box_2, go is the only acceptable action, and ii) once the robot is at the location of Box_2, $push$ of Box_2 is the only acceptable action.

Document your code appropriately.

To hand in your report for this assignment, submit your code and test results electronically and deliver a printout together with your answers to question 1 in the 3402 drop box in CSE building by the deadline. To submit electronically, use the following Prism lab command:

```
submit 3402 a3 a3.pl a3tests.txt
```

Your Prolog code should work correctly on Prism.