FINAL EXAM: CISC 481/681

NAME:

Dec 11, 2001

The first questions are short answer, and then there's some longer ones covering the later course topics. Always show your work so I can easily assign partial credit. Feel free to continue an answer onto the back of the paper.

1. (9 points; 3 each) How is the open list (list of unexpanded nodes) ordered in each of the following search algorithms (assuming that we next expand the first node on the open list):

   (a) Depth-First Search

   (b) Breadth-First Search

   (c) A* Search

2. (4 points) How does the POP planning algorithm handle interacting goals in planning [Sussman's Anomaly with the blocks is one example of interacting goals]?

3. (4 pts) Given the standard description of a task environment (e.g. accessible vs. non-accessible; static vs dynamic; continuous vs discrete; deterministic vs stochastic), what types of environments are good for the basic POCL/POP (Partial Order Causal Link) planning algorithm we discussed in class?

4. **REPRESENTATIONS IN LOGIC:** (10 points) Suppose we want to represent, using a first-order logic based language like Epilog, certain properties about objects. There are three main approaches:

   (a) Define a PREDICATE for each value of each property. The predicate is TRUE if and only if the object has that property value. For example (O1, O2, etc. are distinct objects):

Triangle(O1)	Circle(O2)
Green(O1)	Green(O2)
Large(O1)	Small(O2)

   (b) Define a RELATIONSHIP for each property that relates the value of the property to the object. For example, the same info as above would become:

Shape(O1, Triangle)	Shape(O2, Circle)
Color(O1, Green)	Color(O2, Green)
Size(O1, Large)	Size(O2, Small)

   (c) Define a relationship called ATTRIBUTE that relates an object, an attribute name, and an attribute value. For example, the same info would now be represented in your knowledge base as:

Attribute(O1, Shape, Triangle)	Attribute(O2, Shape, Circle)
Attribute(O1, Color, Green)	Attribute(O2, Color, Green)
Attribute(O1, Size, Large)	Attribute(O2, Size, Small)

   Question: describe the advantages and disadvantages of these three approaches. Hint: First consider the range of questions/queries that can be answered, from "what shape is O1?" to "In what ways are O1 and O2 similar?". Hint: Then consider what kinds of logical rules can we write, from "Anything that is large is not small" to "All triangles have 3 sides" and "All circles have a diameter, but triangles do not have a diameter"

5. (11 pts)

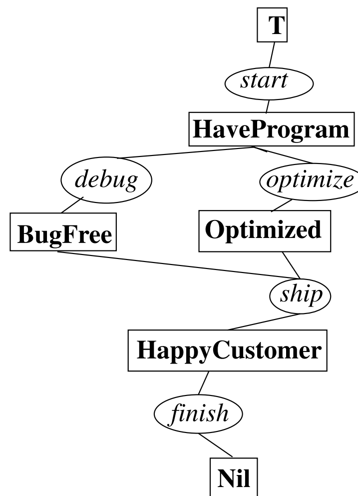
Consider using a partial order planning system for planning the development and release of a software product. Assume that in this domain there exist the following simple operators defined in STRIPS:

Operator	Preconditions	Add List	Delete List
optimize	HaveProgram	Optimized	BugFree
debug	HaveProgram	BugFree	
ship	HaveProgram BugFree Optimized HavePackaging	HappyCustomer	
designpackaging	HaveProgram	HavePackaging	

The planning problem is to get from a state where Haveprogram is true to one in which HappyCustomer is true. As we know, we will also create two "dummy" operators:

Operator	Preconditions	Add List	Delete List
start	T	HaveProgram	
finish	HappyCustomer	NIL	

Suppose we start with the following partial plan, which contains currently no ordering constraints other than those implied by the partial ordering of the plan:



- (6) What are the existing threat arc(s) in this partial plan, if any?

- (5) List the preconditions of any operator in the partial plan that are not yet supported by a causal link (the open list)

6. **PLANNING:** (20 points, 5 each) The famous water jug problem can be stated as follows: you are given 2 water jugs—a 4-gallon jug (call it JUG4) and a 3-gallon jug (call it JUG3). Neither jug has any measuring markers on it. There is a pump that can be used to fill the jugs with water. The goal is to get *exactly* 2 gallons of water into the 4-Gallon jug.

   Your job is to write a set of POP/STRIPS-style planning operators to solve this problem. Include the operators that set up the START and FINISH states. Make doubly sure that your action descriptions are correct and consistent—i.e. they should list all the required preconditions and state explicitly all the conditions that get changed. *DO NOT worry about what the final plan (solution) actually is!!* Here's some useful logical representations you can use:

   - let *cont(j,y)* be a predicate that represents that jug j contains y gallons of liquid. So *cont(JUG4,1)* represents that JUG4 has 1 gallon of water in it.
   - let *capacity(j)* be a function that denotes the total capacity of jug j. So *cont(JUG3, capacity(JUG3))* denotes that JUG3 is full of three gallons of water.
   - In writing preconditions, you may use simple arithmatic operators (e.g. $+$, $-$) and comparisions (e.g., $\geq$). So you might say something like $z + w \geq capacity(JUG3)$.

   Write definitions for the following 4 POP operators:

   (a) Empty(x) — empties the contents of jug x onto the ground.

   (b) Fill(x) — fills jug x completely to its capacity (from the pump)

   (c) Pour-All(x,y) — pours the entire contents of jug x into jug y. Please add a precondition that the contents of jug x must fit into jug y!

   (d) Pour-Till-Full(x,y) — Pours the contents of jug x into jug y just until y is full.

7. **BELIEF NETS:** (20 points total) Orville the robot juggler drops balls quite often when its battery is low. In previous tests, it has been determined that the probability that it will drop a ball when its battery is low is 0.9. Whereas when its battery is not low, the probability that it drops a ball is only 0.01. The battery was recharged not so long ago, and our best guess (before looking at Orville's latest juggling record) is that the odds that the battery is low indicate about a ten percent chance. Another robot observer, with a somewhat unreliable vision system, reports that Orville dropped a ball. The reliability of the observer is given by the following probabilities:

   Pr(observer says Orville drops | Orville does drop) = 0.9

   Pr(observer says Orville drops | Orville doesn't drop) = 0.2

   (a) (8 points) Draw a belief network to represent this information.

   (b) (7 points) Specify the conditional probabilty tables

   (c) (5 points) Calculate the probability that the battery is low given that the observer reports a dropped ball. You don't have to do the actual decimal math, just leave it in the obvious unsimplified form.

   In case you forgot, $\Pr(A \wedge B) = \Pr(A|B)\Pr(B)$, and Normalized Bayes Rule:

$$\Pr(A|B) = \frac{\Pr(B|A)\Pr(A)}{\Pr(B|A)\Pr(A) + \Pr(B|\neg A)\Pr(\neg A)}$$

8. **REINFORCEMENT LEARNING:** (6 points, 3 each) When we studied reinforcement learning in class, we assumed that all the training sequences were finite. Of course there are many real environments without such a clear termination point, and the unlimited accumulation of rewards can lead to problems with infinite utilities.

   To avoid this, a discounting factor γ is often used, where $\gamma < 1$. The utility of a state k steps into the future is discounted (reduced) by multiplying it by γ^k. Thus rewards very very far into the future are effectively 0.

   The Adaptive Dynamic Programming model of utilities says that a state's utility should be updated according to the following equation:

$$U(S_i) = R(S_i) + \sum_{j \in \text{next-states}} M_{ij} U(S_j)$$

   And the Temporal Difference Learning model says that a state's utility should be updated according to *this* equation:

$$U(S_i) = U(S_i) + \alpha(R(S_i) + U(S_j) - U(S_i))$$

   The question: rewrite these two equations so that they incorporate the discounting factor γ. Hint: this is simple and purely mechanical if you understand the equations—don't try to make the problem more difficult.