CS121 – Introduction to Artificial Intelligence – Winter 2011

Homework #7
(Probabilistic Uncertainty and Decision Tree Learning)
Out: 2/28/11 — Due: 3/8/11 (at noon)

How to complete this HW: First copy this file; then type your answers in the file immediately below each question; start each question on a separate page, write your name on every page. Finally, print this file and return it in the drawer marked CS121 of a file cabinet located in the entryway of the Gates building next to office Gates 182 (see red arrow in http://ai.stanford.edu/~latombe/cs121/2011/map.jpg) no later than Tuesday 3/8 at noon.

Your name: ……

Your email address: ……

Note on Honor Code: You must NOT look at previously published solutions of any of these problems in preparing your answers. You may discuss these problems with other students in the class (in fact, you are encouraged to do so) and/or look into other documents (books, web sites), with the exception of published solutions, without taking any written or electronic notes. If you have discussed any of the problems with other students, indicate their name(s) here:
……
Any intentional transgression of these rules will be considered an honor code violation.

General information: Justify your answers, but keep explanations short and to the point. Excessive verbosity will be penalized. If you have any doubt on how to interpret a question, tell us in advance, so that we can help you understand the question, or tell us how you understand it in your returned solution.

Grading:

Problem#	Max. grade	Your grade
I	25	
II	25	
III	25	
IV	25	
Total	100	

Name ……..

I. Deciding under Probabilistic Uncertainty

Consider the robot target-tracking problem studied in the lecture on Deciding under Probabilistic Uncertainty. The task of a robot R is to track a target T in a space with obstacles. The robot and the target are each modeled by a point in a square space W. The obstacles in W are represented by polygons. The robot has a 360° vision sensor that can see arbitrarily far, but it can't see through obstacles.

Time is divided into small units of time. During each unit of time, R either moves Up, Down, Right, or Left by one step of length 1, or does not to move at all. Similarly, during each unit of time, T also moves by one step of length 1 (Up, Down, Right, or Left) or does not move, with each possibility having constant probability 0.2. Both R and T move simultaneously, and the moves of T are not influenced by those of R. Neither R nor T can penetrate an obstacle. You may assume that R and T can possibly be at the same position, in which case R can still see T.

Unlike in the example seen in class, where R had to track T as long as possible, here the task of R is to track T (*i.e.*, to keep it in its field of view) during D consecutive units of time (where D is a positive integer) starting at time 0. If R succeeds in tracking T from time 0 to time D, then R collects a large positive reward +M, otherwise it gets a large negative reward –M. So, if R fails to track T until time D, it does not matter when exactly it lost the target.

In addition, whenever R moves by one unit of length, it consumes 1 unit of energy. When it does not move, it consumes no energy. At time $t = 0$, R has $E < D$ units of energy available in its battery. When R runs out of energy it can no longer move. But its vision and computing systems use a separate battery that can run for a duration greater than D.

At each time $t = 0, 1, 2, ..., D$, the robot senses exactly its current coordinates $[i,j]$ in W. If T is still in its field of view, it also senses exactly the current coordinates $[u,v]$ of T.

1. Assume that at each time $t = 0, 1, 2, ..., D$, the robot represents the current state by $([i,j],[u,v])$, where $[i,j]$ are the current coordinates of R and $[u,v]$ are those of T. Since both R and T move by units of length and W is a square, the resulting state space is finite. Can the robot use the value-iteration method to pre-compute an optimal policy to track the target? Why?

2. Instead of pre-computing an optimal policy, we now assume that at each time $t = 0, 1, 2, ...$, the robot constructs in breadth-first manner a state-action tree rooted at the current state and backs-up utilities to the root of this tree to choose its next move.

   a) In question 1, a state was represented by $([i,j],[u,v])$. What change**S** should be made to this representation to make this construction possible? Explain the reason for each change.

   b) In the new definition, what are the terminal states?

   c) How many immediate successors (states) does a non-terminal state have?

   d) Assume that R can only devote a small amount of time δ to the construction of the state-action tree (δ is much smaller than a unit of time) in order to both choose and execute an action within a unit of time. So, R stops the construction of the state-action tree as soon as the time δ is exceeded and then backup utilities.

Name ……..

i. What should be the reward collected in each state?

ii. How should the robot estimate the utility of each leaf with a non-terminal state? [*The answer is not unique. Tell us which important parameters you would take into account and why*.]

II. MDP and Value Iteration

A robot performs a series of tasks repetitively (the nature of these tasks is irrelevant to this problem). Each task can be performed either FAST or SLOW. In either case, the task is equally well done. But the robot should try to perform tasks as fast as possible.

We model this problem as the following Markov Decision Problem. Performing a task FAST generally brings a reward of + 2, while doing it SLOW only gives a reward of +1. But the robot must also take into account the internal temperature of its actuators, which can be in either one of two states after the completion of each task: OK or HOT. Performing a task SLOW tends to lower the actuator temperature, but doing it FAST tends to raise it. Performing a task FAST when the actuator state is HOT may lead the actuators to overheat, which then requires the robot to stop, cool down for a while, and make repairs, before the actuator state is OK. The reward in this case is −10. The transition probabilities and rewards are defined in the following table:

s	a	s'	$\pi(s,a,s')$	$R(s,a,s')$
OK	SLOW	OK	1.0	+1
OK	FAST	OK	0.5	+2
OK	FAST	HOT	0.5	+2
HOT	SLOW	OK	1.0	+1
HOT	FAST	HOT	0.5	+2
HOT	FAST	OK	0.5	−10

In this table, $\pi(s,a,s')$ denotes the probability of transitioning to state s' by performing a task at speed a from state s and $R(s,a,s')$ is the reward collected during this transition. In the last row of the table, performing a task at speed FAST leads the robot actuators to overheat. After cooling down and repairs, the actuator state s' is OK, but the reward collected during the transition is −10. Note that, unlike in the examples given in class, the rewards are here associated with transitions, not with states, but a reward may depend on the state that is reached.

We ask you to run two cycles of value iteration using a discount factor of 0.8 and fill the entries left in white in the table below. The entry U_i(OK), for i = 1, 2, should be filled with the utility of state OK computed at cycle i. The entry U_1(HOT) should be filled with the utility of state HOT computed at cycle 1. The entries U_0(OK) and U_0(HOT) are set to 0, each. Don't fill U_2(HOT). We also ask you to write down the equations that define the values of U_1(OK), U_1(HOT), U_2(OK).

s	U_0	U_1	U_2
OK	0		
HOT	0		

Name ……..

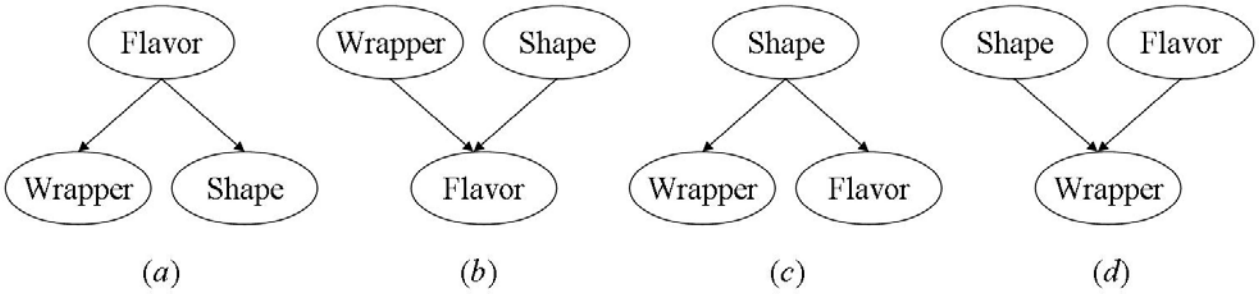
III. Bayesian Net and Probabilistic Reasoning

The Surprise Candy Company makes candy in two flavors: 70% are strawberry flavor and 30% apricot flavor. Each piece of candy is round or square and is wrapped in a red or brown wrapper.

The candies are manufactured on two distinct lines, one for the strawberry candies and the other for the apricot candies. In the strawberry line, 80% of the candies are made round. At the end of this line, 80% of the candies are picked at random (independent of their shapes) and are wrapped in a red wrapper; the other 20% are wrapped in a brown wrapper. Similarly, in the apricot line, 90% of the candies are made square. At the end of this line, 90% of candies are picked at random (independent of their shapes) and are wrapped in a brown wrapper; the other 10% are wrapped in a red wrapper.

This information is presented in a brochure published by the candy company. You decide to buy a candy, but the company requires buyers to pick their candies at random from a large container. Before seeing the candy, you wish to represent the above knowledge in the form of a Bayes net.

Consider the following Bayes nets (the probability tables are not shown in purpose):



1. Is the following statement:
   "Net (*b*) asserts that P((Wrapper=*w*)|(Shape=*s*)) = P(Wrapper=*w*), where *w* = red or brown and *s* = round or square."
   true or false? Why? [*You don't need the probability tables to answer this question.*]

2. From the information in the company brochure, which net can correctly describe the 8 joint probabilities P((Wrapper=*w*)∧(Flavor=*f*)∧(Shape=*s*)), where *w* = red or brown, *f* = strawberry or apricot, and *s* = round or square (assuming suitable probability tables): (*a*), (*b*), (*c*), or (*d*)?

3. Redraw the selected net below and give the correct probability table besides each node.

4. What is the probability that the candy has a red wrapper: 0.8, 0.56, or 0.59? Justify how you get your answer.

5. You now look at the candy and you observe that it is round with a red wrapper. What is the new (posterior) probability that its flavor is strawberry: less that 0.7, between 0.7 and 0.99, or greater than 0.99.

Name ……..

IV. Decision Tree Learning

You have a friend who only does one of four things on every Saturday afternoon: go shopping, watch a movie, play tennis, or just stay in. You have observed your friend's behavior over 11 different weekends. On each of these weekends you have noted the weather (sunny, windy, or rainy), whether her parents visit (visit or no-visit), whether she has drawn cash from an ATM machine (rich or poor), and whether she had an exam during the coming week (exam or no-exam). You have built the following data table:

# ex.	Weather	Parents	Cash	Exam	Decision
1	sunny	visit	rich	yes	cinema
2	sunny	no-visit	rich	no	tennis
3	windy	visit	rich	no	cinema
4	rainy	visit	poor	yes	cinema
5	rainy	no-visit	rich	no	stay-in
6	rainy	visit	poor	no	cinema
7	windy	no-visit	poor	yes	cinema
8	windy	no-visit	rich	yes	shopping
9	windy	visit	rich	no	cinema
10	sunny	no-visit	rich	no	tennis
11	sunny	no-visit	poor	yes	tennis

You now want to build a decision tree to predict the activity of your friend on any future Saturday afternoon from the observed values of Weather, Parents, Cash, and Exam. To do this, you use the greedy Decision Tree Learning algorithm (DTL) described in class. Each path of the decision tree leads to a single value of the goal attribute Decision. DTL selects the attribute to test at each node of the tree by minimizing the number of examples that would be misclassified after that. Note that you will have to adapt slightly DTL to handle attributes like Weather and Decision, since they have more than 2 values (for this reason, in this problem, we use the word "attribute", instead of "predicate").

1. Suppose that you want to build a very simple decision tree that allows you to predict the value of Decision from a single observable attribute (Weather, Parents, Cash, or Exam).
   a. What would be this attribute? If there is a tie among several attributes give each of them.
   b. For each attribute that you have selected at the question 1.a, for each value of this attribute, give the answer that the decision tree would give using the majority rule, as well as the number of misclassified examples.
2. You now want to build a decision tree that allows you to predict your friend's activity from two observable attributes. Thus, this decision tree will have depth 3 (counting the leaves where the value of Decision is given), but may contain more than two nodes testing observable attributes.
   a. From the work you have done to answer question 1, which attribute do you think DTL should choose for the root of the tree? Explain the rationale of your choice.
   b. Assume that you choose to test Weather at the root of the tree (this may, or may not be the correct answer of the previous question). What would be the decision tree of depth 3 built by DTL? If there are multiple solutions, justify your choice.