15-381 Artificial Intelligence: Representation and Problem Solving, Fall 2009: Homework 3

Due: MONDAY, October 12th, beginning of the class

Instructions

- You can work in a group of up to two people. This group does not need to be the same group as for the other homeworks. You only need to turn in one write-up per group, but you must include the andrew ID's of the group members.
- Late homework is due by 3:00 P.M. on the day they are due. Please bring late homeworks to Heather Carney in GHC 7027 (slide it under the door if she is not there; write on the homework the date and time turned in).
- Email all questions to 15381-instructors@cs.cmu.edu

1 Search Review [15 points]

Consider the INFGRID problem, in which a robot must find a path from the start location (x_s, y_s) to the goal location (x_g, y_g) in an infinitely large 2D grid world. Possible moves are one step moves in any of the four directions {North, South, East, West} to another cell, except the robot cannot move into an obstacle cell. We will examine a set of search algorithms we discussed in class, applied to this problem.

1. The Manhattan distance heuristic value of a cell at (x, y) is $h(x, y) = |x - x_g| + |y - y_g|$. Is this an admissible heuristic?
2. For each of the following algorithms, state whether it is (1) complete and (2) optimal for the INFGRID problem. Explain.
   (a) Breadth-first search
   (b) Depth-first search
   (c) Depth-first iterative deepening
   (d) Best-first search with the Manhattan distance heuristic
   (e) A* with the Manhattan distance heuristic

2 Search More [15 points]

This question is about a search algorithm called iterative lengthening search. We havent talked about this algorithm in class - it is not the same thing as iterative deepening search. However, we think you can reason about it based on the uninformed search strategies you have already seen.

Iterative lengthening search is an iterative version of uniform cost search. The main idea of this algorithm is that we use increasing limits on path cost.

- We start searching as though we were performing uniform cost search.

- If we generate a node with a path cost that is greater than the current limit, we immediately discard that node.
- We set the limit for the current iteration to be equal to the lowest path cost of any node that we discarded in the previous iteration.
- We initialize the path cost limit to be equal to the smallest cost between any two nodes in the tree. So, for a uniform cost tree, we initialize the path cost limit to 1.

1. When this algorithm first encounters the goal, will the goal will be on the path with the cheapest cost?
2. Suppose we are searching through a uniform tree with branching factor b, solution depth d, and unit step costs (that is, the cost to move between two nodes is always one). In the worst case, how many iterations will iterative lengthening require?
3. Now suppose that we are using the same tree as in part 2, but our step costs are drawn from the continuous range [0, 1] with a minimum positive cost ϵ. Our path cost limit will be initialized to ϵ. For each of the following statements, state whether it is true or false.
   (a) Iterative lengthening will not find the optimal solution for this tree.
   (b) In the worst case, running iterative lengthening on this tree will require less time than running DFS on this tree.
   (c) In the worst case, iterative deepening search would require less time to run on this tree than iterative lengthening.

3 Probability [5 points]

Let A and B be independent binary random variables with $P(A = 1) = 0.2$ and $P(B = 1) = 0.6$.

1. Let $C = A$ OR B. Compute $P(C = 1)$.
2. Let $D = A$ XOR B. Compute $P(D = 1)$.
3. Show that D and A are not independent.

4 A Constraint Satisfaction Problem [15 points]

Like most things, Minesweeper can be described as a constraint satisfaction problem. Consider the configuration in Figure 1, where each square contains either a number n (touching exactly n bombs), "B" for a bomb, or a variable name (x_i). The problem is to determine whether each of the variable squares contains a bomb, i.e., whether the value of the variable is 1 (or 0 if no bomb).

B	B	x_2	2
x_1	6	B	x_3
3	B	B	3
2	x_5	3	x_4

Figure 1: A Custom sized instance of the Minesweeper game

1. Write down the set of binary constraints for the problem and the constraint graph.

2. Using the variable ordering $(x_1, x_2, x_3, x_4, x_5)$ and numeric value ordering (0 before 1), give the order of assignments made using depth-first search with backtracking (backtrack to the lowest level of search tree possible).[1]

3. Using the variable ordering $(x_1, x_2, x_3, x_4, x_5)$ and numeric value ordering (0 before 1), perform depth-first search with backtracking and forward checking, working out the steps on the provided sheet (Figure 2). Use a new table whenever backtracking (i.e., when you have to delete something).

4. Using the variable ordering $(\mathbf{x_2}, x_3, x_4, x_5, x_1)$ and ***reverse*** numeric value ordering (1 before 0), perform depth-first search with backtracking and forward checking, working out the steps on the provided sheet (Figure 3). Use a new table whenever backtracking (i.e., when you have to delete something).

5. Using the variable ordering $(\mathbf{x_2}, x_3, x_4, x_5, x_1)$ and ***reverse*** numeric value ordering (1 before 0), perform constraint propagation, working out the steps on the provided sheet (Figure 4). Use a new table whenever backtracking (i.e., when you have to delete something).

6. Suppose we want to try the constraint tree algorithm on this graph. We need to assign a value to a variable to eliminate from the graph in order to create a tree and apply the constraint tree algorithm, and continue trying different variables and values (to eliminate) until we get an answer. For such assignment, assume we use numeric value ordering (0 before 1) and give a variable ordering with the shortest running time. How much worse can you do with a different ordering?

[1] Sorry it's painful.

	x_1	x_2	x_3	x_4	x_5
0					
1					

	x_1	x_2	x_3	x_4	x_5
0					
1					

	x_1	x_2	x_3	x_4	x_5
0					
1					

	x_1	x_2	x_3	x_4	x_5
0					
1					

Figure 2: A worksheet for DFS with forward checking

	x_2	x_3	x_4	x_5	x_1
1					
0					

	x_2	x_3	x_4	x_5	x_1
1					
0					

	x_2	x_3	x_4	x_5	x_1
1					
0					

	x_2	x_3	x_4	x_5	x_1
1					
0					

Figure 3: A worksheet for DFS with forward checking

	x_2	x_3	x_4	x_5	x_1
1					
0					

	x_2	x_3	x_4	x_5	x_1
1					
0					

	x_2	x_3	x_4	x_5	x_1
1					
0					

	x_2	x_3	x_4	x_5	x_1
1					
0					

Figure 4: A worksheet for constraint propagation

5 Planning [25 points]

Consider the tower of Hanoi domain with three pegs, namely peg1, peg2, peg3 and with disks of different sizes. A disk is at one peg, represented as (at ?disk ?peg), and can be on top of another disk, represented as (on ?diskA ?diskB). We can assert that a disk is larger than another disk with a predicate (larger ?diskA ?diskB), which is true when ?diskA is larger than ?diskB. The rules of this domain are such that a ?diskA can be moved if it does not have another disk on top of it; and a ?diskA can be moved to an empty peg or to the top of another ?diskB, if (larger ?diskB ?diskA).

1. Using all or some of the predicates: (at ?disk ?peg), (on ?diskA ?diskB), (larger ?diskA ?diskB), (empty ?pegx), (clear ?disk) write actions that capture the rules of the domain in terms of preconditions, and add and delete effects.

   Hint1: A negated precondition (not (on ?diskA ?diskB)) is true if there do not exist instances of the variables ?diskA and ?diskB, such that (on ?diskA ?diskB) is true.

   Hint2: Note that you can choose to use or not use the predicates (empty ?pegx) and (clear ?disk).

2. Consider three disks, D1, D2, and D3, and three pegs, peg1, peg2, peg3, and the following initial state:

   ```
   (larger D3 D2) (larger D3 D1) (larger D2 D1)
   (at D1 peg1) (at D2 peg1) (on D1 D2)
   (at D3 peg2)
   ```

   and the goal state:

   ```
   (at D1 peg3) (at D2 peg3) (at D3 peg3)
   ```

   (a) Using the actions you defined in part 1, show the plan to apply to the initial state to achieve the given goal. Show the actions and the state after each action (as a conjunction of predicates) in the plan.

   (b) Illustrate two concrete choice points that the GPS algorithm would face while solving the given problem above.

6 A Bayes Net [25 points]

For this problem, consider the Bayes net in Figure 5 modeling personal satisfaction of 15-381 students, where { } describes the domain for the value of the node.

1. Write down the expression for the following probability distributions in terms of the conditional distributions implied by the Bayes net (simplify as much as possible).

   (a) The joint distribution $P(G, C, D, P, R, H)$

   (b) $P(H|R, D, C)$

2. From part 1(b), what can you conclude about conditional dependence between **H**appy and **C**ute given **D**ating and g**R**ade?

3. How many parameters must be learned for the Bayes net to complete the conditional probability tables (CPTs)?

4. How many parameters must be learned for the Bayes net to complete the conditional probability tables (CPTs) if we remove the **P**ersonal Hygiene node (and the two outgoing edges) from the graph?

5. Given the CPTs in Figure 6 for the Bayes net above, compute the following probabilities.

   (a) What's the probability that a dating male student with poor personal hygiene and an A in 15-381 is happy, $P(H = True|D = True, G = Male, P = Poor, R = A/B)$?

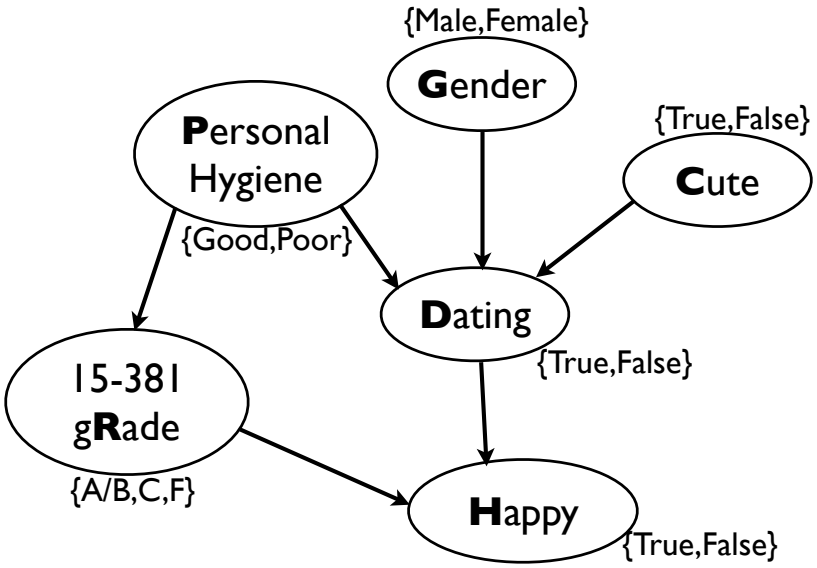

Figure 5: A Bayes net modeling certain characteristics of 15-381 students

(b) How probable is such a student? i.e., what is $P(H = True, D = True, G = Male, P = Poor, R = A/B)$? (Assume infinite precision, i.e., significant digits.)

(c) Give the probability distribution over the cuteness of a male happy dating student with good personal hygiene: $P(C|P = Good, D = True, G = Male, H = True)$. Is he likely to be cute?

P(**P**ersonal Hygiene)

Personal	
Good	0.7
Poor	0.3

P(**G**ender)

Gender	
Male	0.8
Female	0.2

P(**C**ute)

Cute	
TRUE	0.6
FALSE	0.4

P(**D**ating | **C**ute, **G**ender, **P**ersonal Hygiene)

Dating	C=True G=Male P=Good	C=True G=Male P=Poor	C=True G=Fem P=Good	C=True G=Fem P=Poor	C=False G=Male P=Good	C=False G=Male P=Poor	C=False G=Fem P=Good	C=False G=Fem P=Poor
TRUE	0.6	0.3	0.9	0.8	0.3	0.1	0.7	0.6
FALSE	0.4	0.7	0.1	0.2	0.7	0.9	0.3	0.4

P(g**R**ade | **P**ersonal Hygiene)

gRade	P=Good	P=Poor
A/B	0.7	0.6
C	0.25	0.3
F	0.05	0.1

P(**H**appy | **D**ating, g**R**ade)

Happy	D=True R=A/B	D=True R=C	D=True R=F	D=False R=A/B	D=False R=C	D=False R=F
TRUE	0.8	0.7	0.4	0.7	0.6	0.3
FALSE	0.2	0.3	0.6	0.3	0.4	0.7

Figure 6: A set of conditional probability tables for the 15-381 happiness Bayes net.