Midterm Exam

Perfect score: 100.

Problem 1 (*20 points):* Trace the operation of A^* search (the tree version) applied to the problem of getting **to** Bucharest **from** Lugoj using the straight-line distance heuristic. That is, show the sequence of nodes that the algorithm will consider and the f, g, and h score for each node. You don't need to draw the graph, just right down a sequence of $(city, f(city), g(city), h(city))$ in the order in which the nodes are expanded.

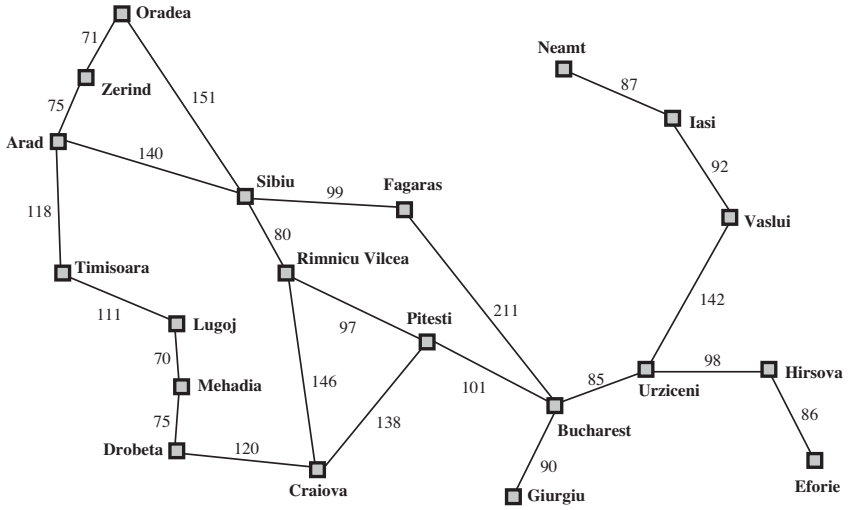


Figure 1: A simplified road map of part of Romania indicating distances between different cities.

Arad	366	**Mehadia**	241
Bucharest	0	**Neamt**	234
Craiova	160	**Oradea**	380
Drobeta	242	**Pitesti**	100
Eforie	161	**Rimnicu Vilcea**	193
Fagaras	176	**Sibiu**	253
Giurgiu	77	**Timisoara**	329
Hirsova	151	**Urziceni**	80
Iasi	226	**Vaslui**	199
Lugoj	244	**Zerind**	374

Figure 2: Straight-line distances to Bucharest

answer:

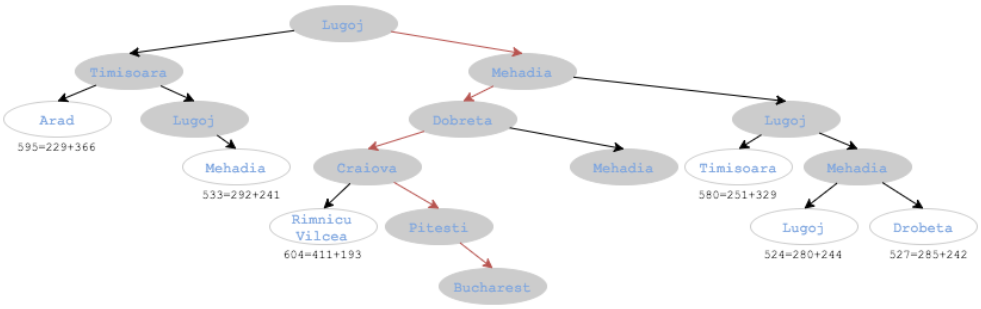


Figure 3: Diagram showing the tree version of A* (after the search has completed) applied to the problem. Expanded nodes shown in gray. List below indicates order of expansion. Subtext shows f-values of nodes in the open list.

format: (city, f(city), g(city), h(city))
(Lugoj,244,0,244)
(Mehadia,311,70,241)
(Lugoj,384,140,244)
(Drobeta,387,145,242)
(Craiova,425,265,160)
(Timisoara,440,111,329)
(Mehadia,451,210,241)
(Mehadia,456,215,241)
(Lugoj,466,222,244)
(Pitesti,503,403,100)
(Bucharest,504,504,0)

Problem 2 (*20 points):* Consider a state space where the start state is number 1 and each state k has two successors: numbers $2k$ and $2k + 1$.

(a) Suppose the goal state is 11. List the order in which states will be visited for breadthfirst search, depth-limited search with limit 3, and iterative deepening search.

answer:
note: answer is also OK if you did not list repeated nodes

(a) **BFS:** 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11

(b) **DLS:** 1, 2, 4, 8, 4, 9, 4, 2, 5, 10, 5, 11

(c) **IDS:** 1, 1, 2, 3, 1, 2, 4, 8, 4, 2, 5, 2, 1, 3, 6, 3, 7, 1, 2, 4, 8, 4, 9, 4, 2, 5, 10, 5, 11

(b) How well would bidirectional search work on this problem? List the order in which states will be visited. What is the branching factor in each direction of the bidirectional search?

answer:
note: slight variations on ordering, e.g. beginning with backward search OK

Bi-directional search would work very well particularly because the branching factor of the backward search is 1.
Branching factor: forward: 2, backward: 1.
State order(forward-searched nodes in bold): **1**, 11, **2**, 5, **3**, 2

Problem 3 (*10 points):* Which of the following statements are correct and which ones are wrong? Provide a short explanation for each answer.

(a) Breadth-first search is a special case of uniform-cost search.

answer: True. BFS is UCS with cost 1 between neighboring nodes.

(b) Depth-first search is a special case of best-first tree search.

answer: True. DFS is best-first with the evaluation function f equal to the negative depth of the tree.

(c) Uniform-cost search is a special case of A^* search.

answer: True. UCS is A^* with h=0.

Problem 4 (*10 points):* Prove that if a heuristic is consistent, it must be admissible. Construct an example of an admissible heuristic that is not consistent. (Hint: you can draw a small graph of 3 nodes and write arbitrary cost and heuristic values so that the heuristic is admissible but not consistent).

answer:

Figure 4: Values inside of nodes indicate h values, g indicates goal state, values on paths indicate costs.

We can prove this by considering a path from state s_0 to the goal state s_g, along with $c(s_i, s_{i+1})$ which is the cost between states s_i and s_{i+1}. Beginning with the consistency definition:

$$h(s_0) \leq h(s_1) + c(s_1, s_2)$$

Summing all such inequalities along the full path to the goal and canceling out common terms yields:

$$h(s_0) \leq h(s_g) + \sum_{i=1}^{g} c(s_{i-1}, s_i)$$

Where the heuristic cost of the goal state is zero:

$$h(s_0) \leq \sum_{i=1}^{g} c(s_{i-1}, s_i)$$

And the summation is equivalent to the value of $h^*(s_0)$, i.e. the actual minimum cost path to the goal:

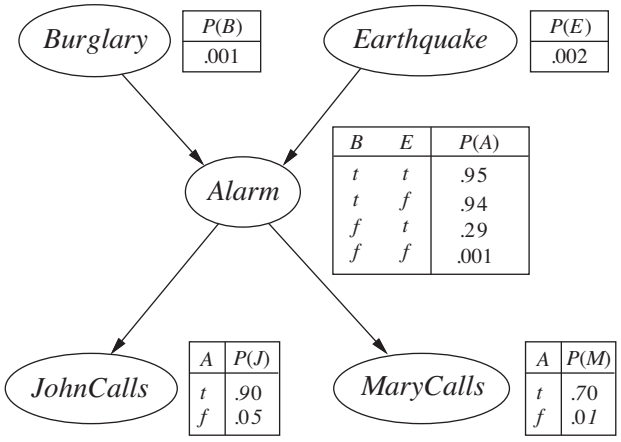
$$h(s_0) \leq h^*(s_0)$$

Problem 5 (*10 points):* In a Constraint Satisfaction Problem (CSP) search, explain why it is a good heuristic to choose the variable that is most constrained but the value that is least constraining.

answer:
Most Constrained Variable: In the search tree, we alternate between choosing variables and choosing values for the variables. At the stage where we choose a variable, we break the search and backtrack if we find one variable that cannot be satisfied. Finding only one such variable is sufficient to say that something was wrong earlier, and we go up in the search to try other assignments. Therefore, we want to fail quickly, which will save us the trouble of trying many variables before finding the one that fails.

Least Constraining Value: Once we choose a variable, we have to try all the possible values before we can say that it failed. Therefore, we will be only losing time with values that fail (since we will still have to check the remaining values). But if we succeed, the search stops and we don't have the try the remaining values.

Problem 6 (*15 points):* Consider the following Bayesian network.
What is the probability of $Burglary = true$ if $JohnCalls = true$?



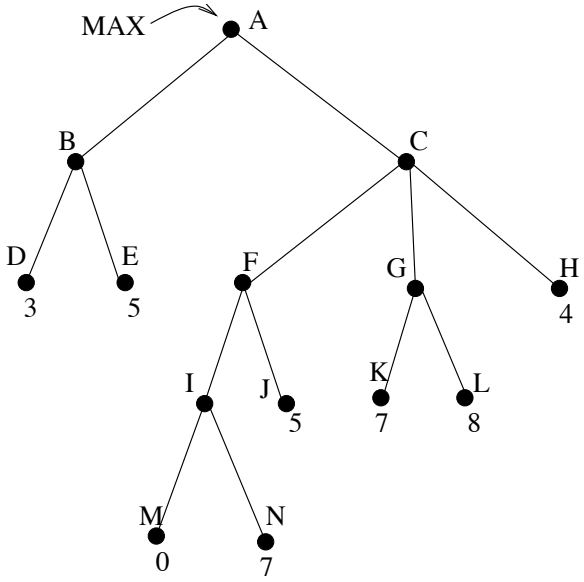
answer:

$$p(B = true|J = true, A, E)$$
$$= \frac{p(B = true)\sum_E p(E)\sum_A p(J = true|A)p(A|B = true, E)}{\sum_B p(B)\sum_E p(E)\sum_A p(J = true|A)p(A|B, E)}$$

$$p(B)p(E)p(J|A)p(A|B,E)$$
$$+p(B)p(E)p(J|\neg A)p(\neg A|B,E)$$
$$+p(B)p(\neg E)p(J|A)p(A|B,\neg E)$$
$$+p(B)p(\neg E)p(J|\neg A)p(\neg A|B,\neg E)$$
$$=\frac{}{p(B)p(E)p(J|A)p(A|B,E) + p(\neg B)p(E)p(J|A)p(A|\neg B,E)}$$
$$+p(B)p(E)p(J|\neg A)p(\neg A|B,E) + p(\neg B)p(E)p(J|\neg A)p(\neg A|\neg B,E)$$
$$+p(B)p(\neg E)p(J|A)p(A|B,\neg E) + p(\neg B)p(\neg E)p(J|A)p(A|\neg B,\neg E)$$
$$+p(B)p(\neg E)p(J|\neg A)p(\neg A|B,\neg E) + p(\neg B)p(\neg E)p(J|\neg A)p(\neg A|\neg B,\neg E)$$

$$(.001) * (.002) * (.90) * (.95)$$
$$+(.001) * (.002) * (.05) * (.05)$$
$$+(.001) * (.998) * (.90) * (.94)$$
$$+(.001) * (.998) * (.05) * (.06)$$
$$=\frac{}{(.001) * (.002) * (.90) * (.95) + (.999) * (.002) * (.90) * (.29)}$$
$$+(.001) * (.002) * (.05) * (.05) + (.999) * (.002) * (.05) * (.71)$$
$$+(.001) * (.998) * (.90) * (.94) + (.999) * (.998) * (.90) * (.001)$$
$$+(.001) * (.998) * (.05) * (.06) + (.999) * (.998) * (.05) * (.999)$$

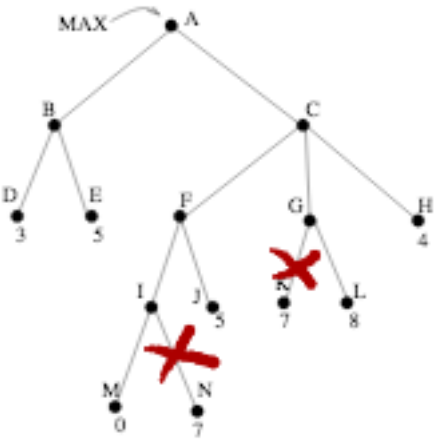
Problem 7 (*15 points):* Consider the following game tree, where the first move is made by the MAX player and the second move is made by the MIN player.



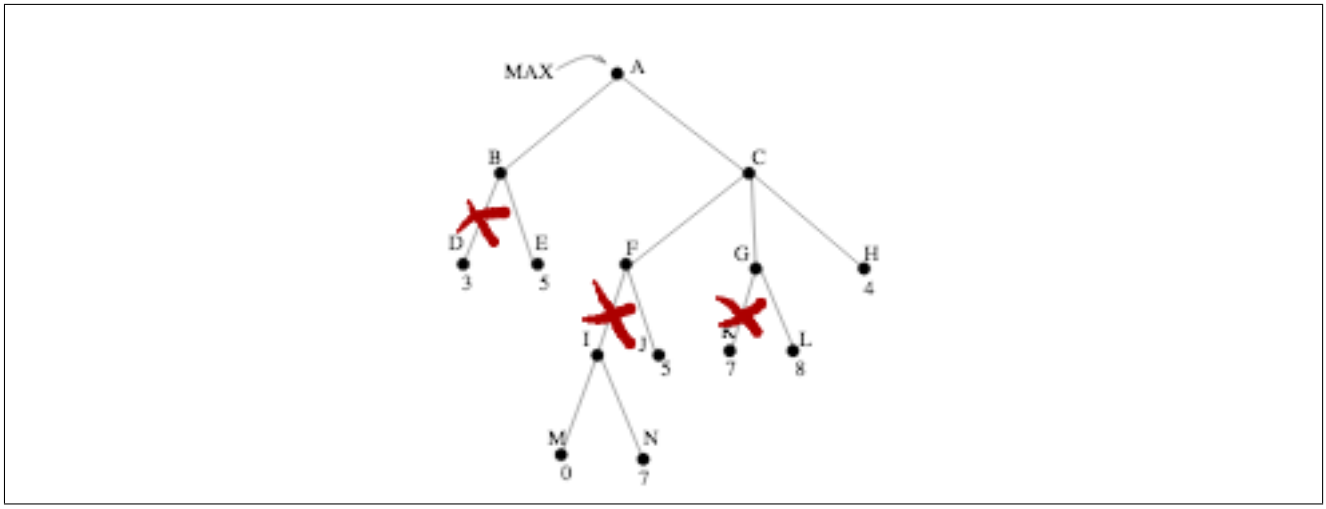
(a) What is the best move for the MAX player using the minimax procedure?

answer:
The best initial move for MAX is to node C (value=4).

(b) Perform a left-to-right (left branch first, then right branch) alpha-beta pruning on the tree. That is, draw only the parts of the tree that are visited and don't draw branches that are cut off (no need to show the alpha or beta values).



(c) Do the same thing as in the previous question, but with a right-to-left ordering of the actions. Discuss why different pruning occurs.


MAX
A
B
C
D
E
F
G
H
3
5
I
J
K
L
4
5
7
8
M
N
0
7