Name: Answer Key

CS 331 Midterm
Spring 2023

You have 90 minutes to complete this midterm. You are only allowed to use your textbook, the pdx files on Canvas, your assignments and solutions to those assignments during this midterm. If you find that you are spending a large amount of time on a difficult question, skip it and return to it when you've finished some of the easier questions.

Section	Marks
Agents	/ 10
Search	/ 10
CSP	/ 10
Games	/ 10
Total	/ 40

Section I: Agents (10 points)

1. Consider the problem of a robot finding an exit from a maze (without a map). You might assume that it can recognize the precise GPS coordinates of its location at any time, For each part below, indicate the choice which best describes the environment for this agent:

a) Fully observable or **Partially observable** [1 point]
[can't see beyond the walls.]

b) Deterministic or **Stochastic** [1 point]
[because real world is noisy. ½ point for choosing deterministic]

c) Episodic or **Sequential** [1 point]
[first step influences the next state.]

d) **Static** or Dynamic **or Semidynamic** [1 point]
[Semidynamic is the best answer due to bounded time. Static is ok if the agent has a lot of time.]

e) Discrete or **Continuous** [1 point]
[Real world states and actions are continuous.]

f) **Single agent** or Multi-agent [1 point]
[Other agents are not relevant to choose the best action.]

g) Known or **Unknown** [1 point]
[because there is no map.]

2. What type of agent would be ideal to solve the above problem? Choose from simple reflex agent, model-based reflex agent, goal-based agent and **utility-based agent**. Would a learning component be useful? Give justifications for your answers. [3 points]

Goal-based cut ½ point. Learning component is useful to memorize where it has been and update the evaluation of states [as in Real-time A* search].

II. Search [10 points]

1. Suppose you want to solve the N-queens problem by formulating it as a standard search problem. Precisely define the states, actions, transition model, initial state and the goal state. [5 points]

   States: Vectors x[.] where x[i] is the column 1..N of the queen in the i'th row. x[i]=0 if there is no queen in the I'th row yet.
   Actions: Place_i [j] to place the current queen [which is in i'th row] in j'th column.
   Transition model:
   Precondition x[i]=0 [no queen yet in i'th row], x[k] <> 0 for i<k [Forces to go sequentially in i]
   Effect x[i]=j
   Initial state: x[.]=[0,0,…0]
   Goal state; for any i <> j (i) x[i] <> x[j] \\ no 2 queens in the same column
   (ii) x[i]+i <> x[j]+j \\ no 2 queens in top-left to bottom-right diagonal
   (iii) x[i]-i <> x[j]-j \\ no 2 queens in bottom-left to top-right diagonal

2. You have unbounded amount of time, but your memory is limited. Which one of the following search algorithms would you choose to solve 8-queens problem? Justify why you chose it and why you didn't choose each of the others.
   a. A* Search Yes/No Reason? [1 point]

      No because A* search is memory intensive.

   b. Depth First Search Yes/No Reason? [1 point]

      Yes because it is memory-efficient.

   c. Iterative-Deepening DFS Yes/No Reason? [1 point]

      No because all solutions are at the same depth. So ID-DFS needlessly repeats search.

   d. Bidirectional Search Yes/No Reason? [1 point]

      No because the goal states are many and not known. Searching backward is not possible.

3. What improvement(s) can you make to the search method you chose above to make it more efficient for the 8-queens problem? [1 point]

We can terminate the recursion as soon as any constraint is violated between already placed queens, because they continue to violate the constraints as more queens are placed. This is the key idea in backtracking search.

III. Constraint Satisfaction [10 points]

1. Precisely state when a pair of variables (X, Y) is arc-consistent? If (X, Y) is arc-consistent, does it mean that (Y, X) is arc-consistent? [2 point]

For all values in the domain of X there should be a value in the domain of Y that satisfies all constraints. No – (Y, X) may not be consistent,

2. Consider a set of variables, {A, B, C, D} over the domains {1, 2, 3, 4} with the constraints {(A < B), (B < C), (C < D)}. Show the domains of each variable after AC3 algorithms processes the variable pairs in this order {(C, D), (B, C), (A, B)} by filling in the table below. [3 points]

3. After the previous step, show the domains of each variable after AC3 processes each variable pair in the set {(B, A), (C, B), (D, C)} in that order. [3 points]

Question	Processing	Domain(A)	Domain(B)	Domain(C)	Domain(D)
	Initialization	{1, 2, 3, 4}	{1, 2, 3, 4}	{1, 2, 3, 4}	{1, 2, 3, 4}
2	(C, D)	{1, 2, 3, 4}	{1, 2, 3, 4}	{1, 2, 3}	{1, 2,3,4}
	(B, C)	{1, 2, 3, 4}	{1, 2}	{1, 2, 3}	{1, 2,3,4}
	(A, B)	{1}	{1, 2}	{1, 2, 3}	{1, 2,3,4}
3	(B, A)	{1}	{2}	{1, 2, 3}	{1,2,3, 4}
	(C, B)	{1}	{2}	{3}	{1,2,3, 4}
	(D, C)	{1}	{2}	{3}	{4}

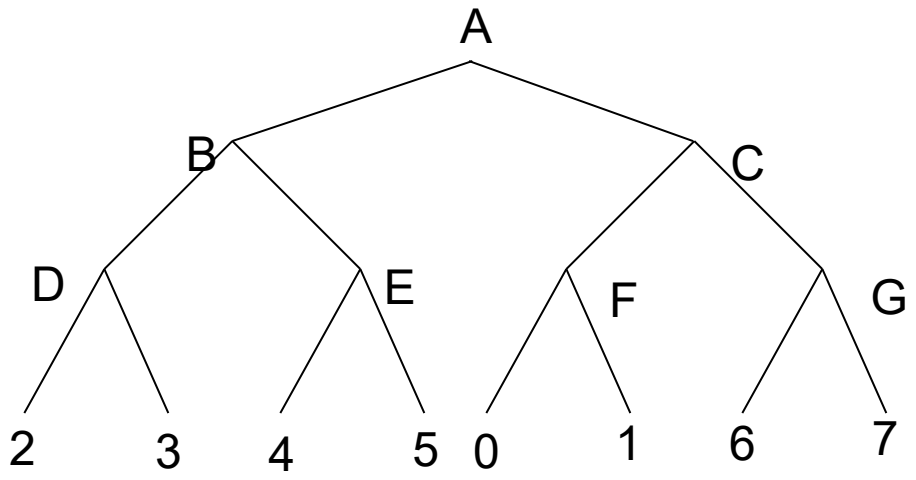
4. Why does AC-3 sometimes adds already processed variable pairs back into the queue to process them again? [2 points]

When the domain of a variable Y is reduced it effects the arc-consistency of other (X,Y) pairs for variables X which may have been already checked. So we need to add such pairs to the queue to recheck reduction in Y's domain would now make some values of X arc-inconsistent. This happens if (B,C) is ordered before (C,D) in question 2 for example. Reducing the domain of C to not include 4 when

processing (C,D), would in turn make the domain of B not include 3. So (B,C) should be processed again even though it is already processed.
Note: Example is not needed to get full credit if the answer is otherwise correct.

IV. Games (10 points)

1. Consider the following Minimax game tree for a 2-player turning-taking game where the root node is Max. Trace the Alpha-beta algorithm on the game tree assuming left-to-right evaluation. Say which internal nodes of the tree are pruned and which leaf nodes are not evaluated. Justify each decision. Show the values returned from each node. Internal nodes are labeled for your convenience. The numbers represent the values of the leaf nodes. They could also be used as labels for convenience since they are all unique. [5 points]



Node	Pruned?	Value Returned	Justification
A	No	3	=Max(B,C)= Max(3,1)
B	No	3	=Min(D,E)=Min(3,5)
C	No	1	=F=1

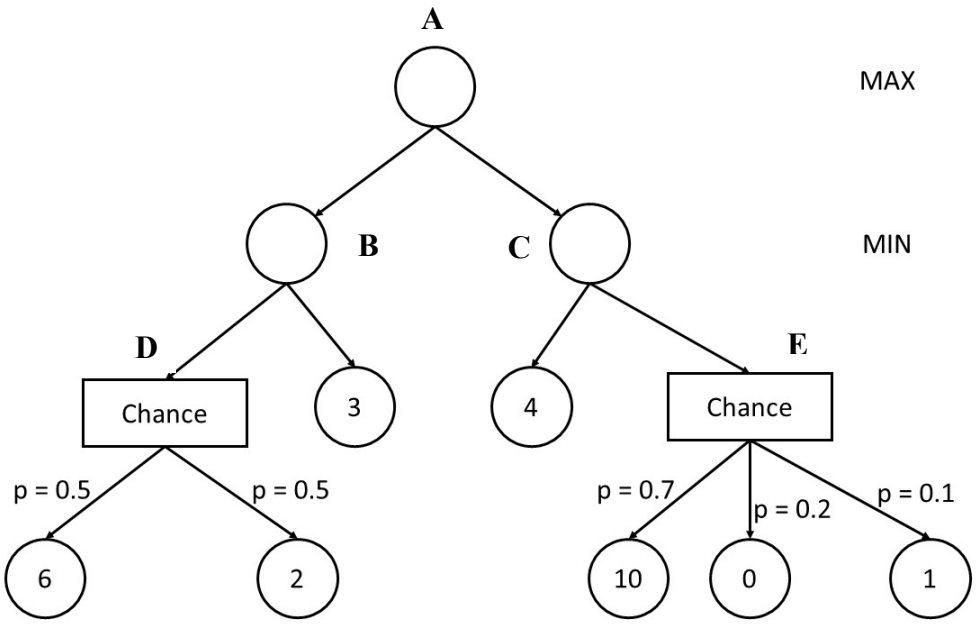
D	No	3	=Max(2,3)
E	No	4	=Max(4)
F	No	1	=Max(0,1)
G	Yes	-	-

Leaf nodes not evaluated: 5, 6 and 7.

Justification: 5 is not evaluated because alpha(E) after evaluating the leaf node 4 is 4 > beta(E)=beta(B)=3.

6 and 7 are not evaluated because G is pruned because beta(C) after evaluating F = 1 < alpha(C) =alpha(A)=3.

2. Below is a portion of a game tree. What are the values of the internal nodes computed by the Expectiminimax? Why? [5 points]



Node	Value	Computed How?
A	4	=Max(B,C)=Max(3,4)
B	3	=Min(D,3)=Min(4,3)
C	4	=Min(4,E)=Min(4,7.1)
D	4	=0.5*6+0.5*2
E	7.1	=0.7*10+0.2*0+0.1*1