Your grades for **midterm1**

Total Score: **79.0**

CSCI561 - Foundations of Artificial Intelligence (20163-CSCI561)

Summary

Class scores distribution

Total (/score/9281219e-2714-4140-9692-f07e5bdc2696)

Q1 (/score/9281219e-2714-4140-9692-f07e5bdc2696/Q1)

Q2 (/score/9281219e-2714-4140-9692-f07e5bdc2696/Q2)

Q3i (/score/9281219e-2714-4140-9692-f07e5bdc2696/Q3i)

Q3ii (/score/9281219e-2714-4140-9692-f07e5bdc2696/Q3ii)

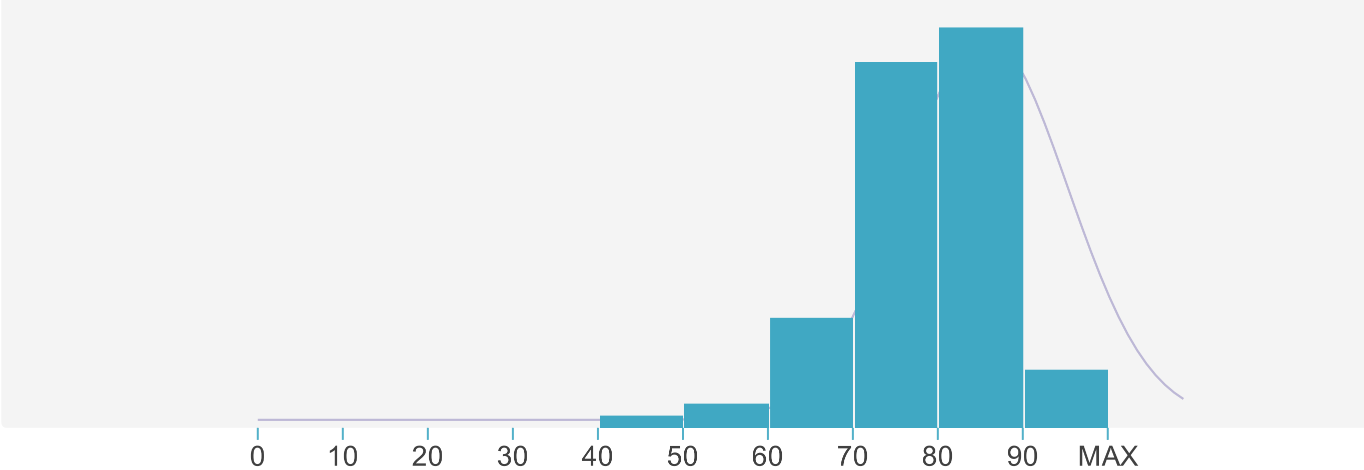
Q4i (/score/9281219e-2714-4140-9692-f07e5bdc2696/Q4i)

Q4ii (/score/9281219e-2714-4140-9692-f07e5bdc2696/Q4ii)

Q5 (/score/9281219e-2714-4140-9692-f07e5bdc2696/Q5)

Q6 (/score/9281219e-2714-4140-9692-f07e5bdc2696/Q6)

Students: 528 Median: 79 Mean: 77.86 Std. Dev: 8.764



Q1 Score: **8.0**

1. [10%] General AI Knowledge

For each of the statements below, fill in the bubble **T** if the statement is **always and unconditionally true**, or fill in the bubble **F** if it is **always false, sometimes false, or just does not make sense**.

	T	F
1	●	F
2	T	●
3	●	F
4	T	●
5	●	F
6	T	●
7	T	●
8	T	●
9	●	F
10	T	●

1- In tree search, we keep no memory of path while we work on a single current state and iteratively improve its "value".

2- A problem is NP if there exists some algorithm that can guess a solution and then verify whether or not the guess is correct in polynomial time.

3- Poker is partially observable.

4- If the cost of applying an operator once is always 1, then DFS is complete.

5- Hill-climbing is an entirely deterministic algorithm.

6- A complete search algorithm will always find a correct solution.

7- DFS has lower asymptotic time complexity than BFS.

8- A rational action is guaranteed to be successful.

9- Learning agents use a "critic" element to evaluate how well they are doing.

10- During search, one usually applies the goal test onto newly expanded children, before queuing-up these children.

Q2 Score: **24.0**

2. [30%] Search

Consider the search space below, where ***S*** is the start state and ***G1, G2 and G3*** satisfy the goal test. Arcs are labeled with the cost of traversing them and the estimated cost to a goal is reported inside nodes.

For each of the following search strategies, indicate which goal state is reached (if any) and list, in order, all the states of the nodes popped off of the OPEN queue, and the cost of the path found by the strategy to reach the goal state from S. When all else is equal, nodes should be removed from OPEN in alphabetical order.

Note how the arcs in the figure are oriented, which means that you can only go from one state to another in the direction of the arrow.

a) [10%] Uniform cost Search

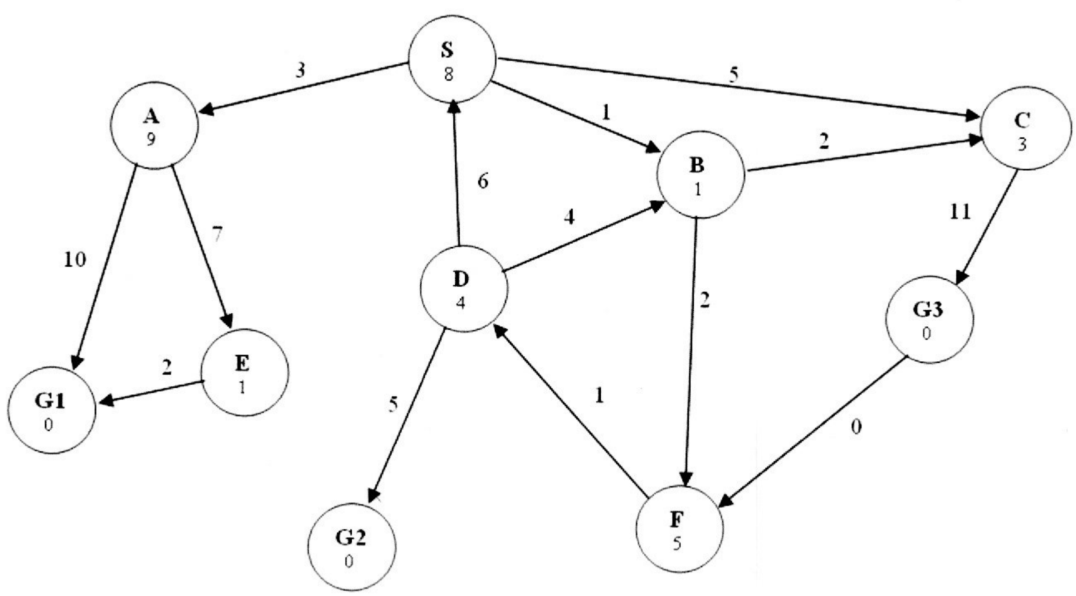
Goal state reached: G2 States popped off OPEN: S, B, A, C, F, D, G2 Path Cost 9

b) [10%] Iterative deepening search

Goal state reached: G1 States popped off OPEN: S, A, B, C, G1 Path Cost 2

c) [10%] A* Search

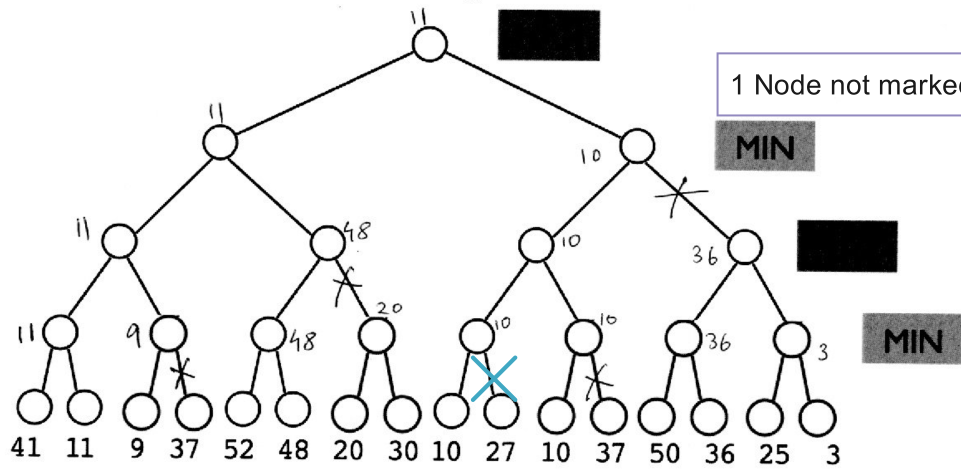
Goal state reached: G2 States popped off OPEN: S, B, C, F, D, G2 Path Cost 9



Q3i Score: **13.0**

3. [20%] Game Playing

Consider the following game tree in which the evaluation function values are shown below each leaf node. Assume that the root node corresponds to the maximizing player. Assume that the search always visits children left-to-right.



1 Node not marked for pruning

(a) [4%] Compute the backed-up values using the minimax algorithm. **Show your answer by writing values next to the appropriate nodes in the above tree.**

4

(b) [4%] Which nodes will not be examined by the alpha-beta pruning algorithm? **Show your answer by crossing out the nodes that will be pruned.**

3

(c) Please answer YES, LIKELY YES, NO or CANNOT SAY to the questions below, and also briefly tell us why. A correct YES, LIKELY YES, NO or CANNOT SAY answer without an explanation is only worth 1% out of 4% for each question below. In general (not just for the above tree), if the search always visits children **right-to-left instead of left-to-right**, please answer these questions:

(i) [3%] Will the minimax value computed at the root be changed?

3

NO, the minimax value won't change since we still have to traverse the entire graph and the leaf node value don't change. If we go from right to left, still the same minimum OR maximum value will be calculated respectively because of no change in leaf nodes.

(ii) [3%] Will the number of nodes pruned by alpha-beta pruning be changed?

3

~~YES~~ LIKELY. YES, The number of nodes pruned depend on the α & β values of each node which inturn depend on the value of leaf nodes. If the value of leaf nodes is important for the no. of α-β cuts. In the above tree, if we traverse from L->R, we get 4 cuts, but if we traverse from R->L we get 0 cuts. There can be case were we get 0 α-β cuts in both traversals depending on the leaf nodes.

Q3ii Score: **2.0**

(iii) [3%] If we now consider the more complicated games where there also are chance nodes in the game, will the minimax value computed at the root be changed when we evaluate right-to-left instead of left-to-right?

YES, since there is an element of chance, minimax value can be different. The right tree might have a higher chance percentage than left one.

0

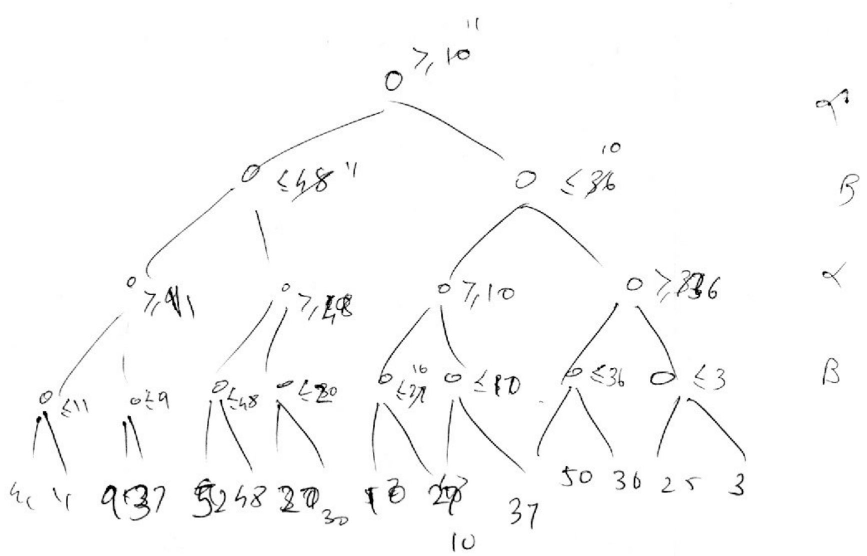
(iv) [3%] Finally, when there are chance nodes, will the number of nodes pruned be changed when we evaluate right-to-left instead of left-to-right?

YES. Because of element of chance and the reason stated in answer (ii)

Likely Yes

2

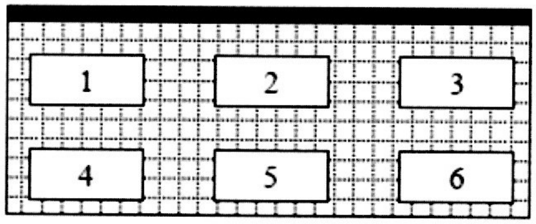
(you may use the space below as scratch paper)



Q4i Score: **5.0**

4. [20%] Constraint Satisfaction

During a make-up examination, an examiner has to arrange six students in a way to make sure that the examination is properly conducted and no one cheats. Two students have makeup exams in Artificial Intelligence (**AI**), one student has a makeup in Database (**DB**), two students have makeups in Mathematics (**M**), and one student in Physics (**P**). As shown in figure below, there exist 6 desks arranged into two rows and three columns.



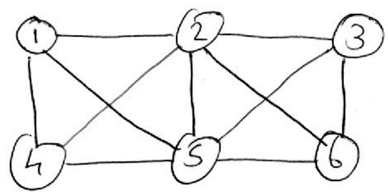
The problem for the examiner is to arrange students so that students having the same makeup exam should not be neighbors (in the same column, in the same row or the same diagonal. For example - neighbors of 5 are 1, 2, 3, 4 and 6, and neighbors of 3 are 2 (same row), 5 (same diagonal) and 6 (same column)).

a) [5%] Propose a formulation of the problem in terms of CSP by specifying variables, domains and constraints. Constraints should be specified formally and precisely, but may be implicit rather than explicit.

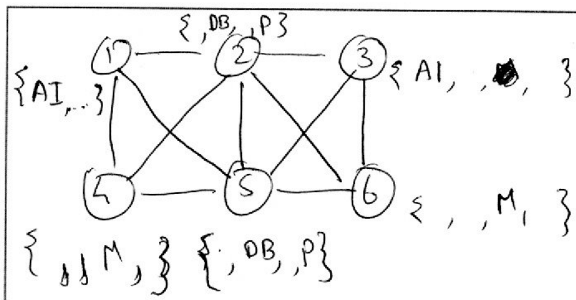
* Variables: the 6 desks arranged in 2 row & 3 columns.
  ie 1, 2, 3, 4, 5
* Domains: For each desk we have to assign a student having exam, {AI, DB, M, P}
* Constraints: ① The neighbors constraints:
  $1 \neq 2, 1 \neq 4, 1 \neq 5, 2 \neq 3, 2 \neq 4, 2 \neq 5, 2 \neq 6,$
  $3 \neq 5, 3 \neq 6, 4 \neq 5, 5 \neq 6,$ ($1 \neq 2$ also means $2 \neq 1$)
  ② Unary constraints: Maximum domains you can use:
  AI=2, DB=1, M=2, P=1 (ie no of makeup exams of each)

Q4ii Score: **14.0**

b) [5%] Draw the constraint graph.



c) [5%] Show the domains of the variables after running AC-3 arc-consistency on your graph from answer (b), after having already enforced any unary constraints.



Assuming AI to assigned to (1) to start

d) [5%] Give one solution to this CSP.

1 AI	2 DB	3 AI
4 M	5 P	6 M

Q5 Score: **5.0**

5. [10%] Game playing theory

Consider A.I. two chess playing programs, called AI1 and AI2, that do **not** use alpha-beta pruning and that look ahead to a depth D.
AI1 and AI2 are identical except that they use different evaluation functions. The evaluation function for AI1 is called eval1(board, player). The evaluation function for AI2 is:

$$\text{eval2(board, player, depth)} = \gamma^{\text{depth}} \times \text{eval1(board, player)}$$

where $\gamma = 0.9$ is a discount factor and depth is the depth of board in the search tree (i.e., the number of moves that board is ahead of the actual current game board). Thus, AI2 uses almost the same evaluation function as AI1, except that AI2 discounts the value of a board by γ^{depth}. The net effect is that the further ahead AI2 looks, the more it reduces the value of boards.

a) [10%] Do AI1 and AI2 play exactly the same, i.e., given any current board, do they always choose the same next move? If your answer is yes, please justify why. If your answer is no, either give a counter example or justify why. Answering yes or no is worth 2%. The justification is worth 8%.

Yes or no: NO

Justification: A chess playing AI program play the next move based on the opponent's current move by examining ~~with~~ which path ~~is the best~~ in the game tree will maximize the output for AI. When AI2 goes deeper and deeper, it reduces the value of each branch by γ (discount factor). The more an AI program looks ahead in the search tree, the more uncertain it is of the outcome of the next move. Thus AI2 will get a more realistic value for each branch and hence will choose the best possible value among them and proceed with the next move. Because of the γ factor, both AI1 & AI2 will most probably play different moves because of a very dense game tree for chess.
Since AI1 doesn't have the γ factor, it may or may not have the same values for evaluation as AI2, thus different results.

Not detailed enough about the terminal boards and uneven depths.

Q6 Score: **8.0**

6. [10%] AI Applications

Based on material from discussion, circle the **best** choice for each question:

(a) [2%] Rationality is:

a. a function that specifies the agent's action in response to every possible percept sequence.
b. a property of agents that choose actions that maximize expected utility, given the percepts to date.
c. a property of agents whose behavior is determined by their own experience rather than solely by their initial programming.
d. All of the above
e. None of the above

(b) [2%] Hill climbing search is best suited to which problem domain?

a. Travelling salesperson problem
b. N-queens problem
c. Sudoku
d. All of the above
e. None of the above

(c) [2%] Which chatbot passed the Turing test in 2014?

a. Mitsuku
b. IBM Watson
c. Ada & Grace
d. All of the above
e. None of the above

(d) [2%] Which statement is NEVER true?

a. Simulated annealing is complete
b. Iterative deepening and Breadth-first searches have the same time complexity
c. Hill climbing search is best used for problem domains with densely packed goals
d. All of the above
e. None of the above

(e) [2%] If h1 and h2 are admissible heuristics, which of the following are also guaranteed to be admissible?

a. h1 ∗ h2
b. max(h1 , h2)
c. (α)h1 + (1−α)h2, for α∈[0,1]
d. All of the above
e. None of the above