KOÇ UNIVERSITY
College of Engineering

COMP 341: INTRODUCTION TO ARTIFICIAL INTELLIGENCE

Midterm 1

Spring 2024, 19/03/2024
Duration: 100 Minutes

Name: Solutions

ID: 00110001 00110000 00110000

- This exam contains 9 pages including this cover page and 5 questions. Check to see if any pages are missing. Put your initials on the top of every page, in case the pages become separated.
- By submitting this exam, you **agree** to fully comply with Koç University Student Code of Conduct, and accept any punishment in case of failure to comply. If you do not submit this exam on time, you will receive 0 credits.
- The exam is **closed book** and **closed notes**. You are **not** allowed to use any additional material including any electronic equipment such as computers and mobile phones.
- You are expected to be able provide clear and concise answers. Gibberish will not receive any credit. Your answers need to be readable by a human, illegible writing is not gradable.
- Read each question carefully and make sure to follow instructions. The written instructions take precedence over an answer that the instructor might give you during the exam, unless the instructor makes a class wide announcement.
- Do not write in the table below.

Question:	1	2	3	4	5	Total
Points:	24	22	17	15	22	100
Score:						

1. (24 points) True or False :

   False ____ The environment of an agent does not affect its rationality.

   True/False ____ Iterative deepening depth first search is optimal if all transition costs are equal. This holds for only the tree-search version which I did not explicitly mention during the class. As a result I will accept both answers.

   False ____ A* Graph Search with admissible heuristics are guaranteed to be optimal for finite graphs.

   False ____ Simulated annealing with $T = \infty$ is equivalent to hill-climbing.

   False ____ Arc consistency can detect all the inconsistent future assignments.

   False ____ Alpha-Beta pruning has better asymptotic space complexity than regular minimax search.

   True ____ When solving constraint satisfaction problems, hill-climbing is not complete (may not be able to find a solution that satisfies the constraints even if one exists).

   True ____ Given two heuristics h_a and h_b, $max(h_a, h_b)$ is dominant over both of them.

2. (22 points) Consider the graph below where **A** is the initial and **G** is the goal state. The directional arcs represent the possible state transitions and cost of each transition is given next to the arcs. The heuristic values are given inside the states with h.

   You are asked to write the visiting/expanding order (order of popping from the frontier, ignore if an already visited state is popped) of the nodes and the resulting solution paths with the graph versions of the given algorithms. The neighbors are added alphabetically to the frontier. The alphabetical order is: A,B,C,D,E,F,G. For algorithms using priority queues, break the ties alphabetically.

   (a) (8 points) Iterative Deepening Depth First Search:
   Expanded Nodes In Order:
   Multiple answers are possible based on how you handle the visited nodes in this search and whether you used recursive or stack based solution

   Recursive: A, A-B-C-D, A-B-C-E-D-G

   Stack Based: A, A-D-C-B, A-D-G

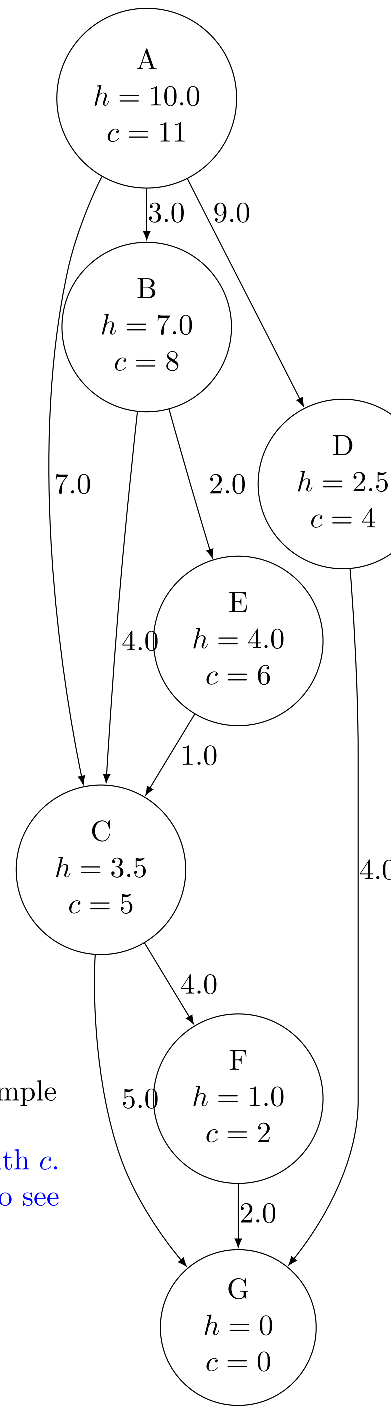
   Resulting Path:
   A-D-G in both cases

   (b) (8 points) A* Search:
   Expanded Nodes In Order:
   A-B-E-C-F-G

   Resulting Path:
   A-B-E-C-G

   (c) (3 points) Is this heuristic admissible? Justify your answer by using the optimal cost and the heuristic value for each state.
   The graph has been updated with the optimal costs to reach to G with c. The updated graph shows that all the heuristic values underestimate the cost to goal thus this heuristic is admissible.

   (d) (3 points) Is this heuristic consistent? Justify your answer by using the optimal cost and the heuristic value for each state. A single example is enough if it is not consistent.
   The graph has been updated with the optimal costs to reach to G with c. The heuristic is not consistent! Just looking at B and E is enough to see this but there are other examples as well.



3. (17 points) You are tasked with scheduling classrooms. There are 5 instructors (given in the set {I1,I2,I3,I4,I5}) and 3 rooms (given in the set {R1,R2,R3}) and you need to assign rooms to instructors. Beware, these instructors are a little bit fussy. The instructors need the rooms in the following times:

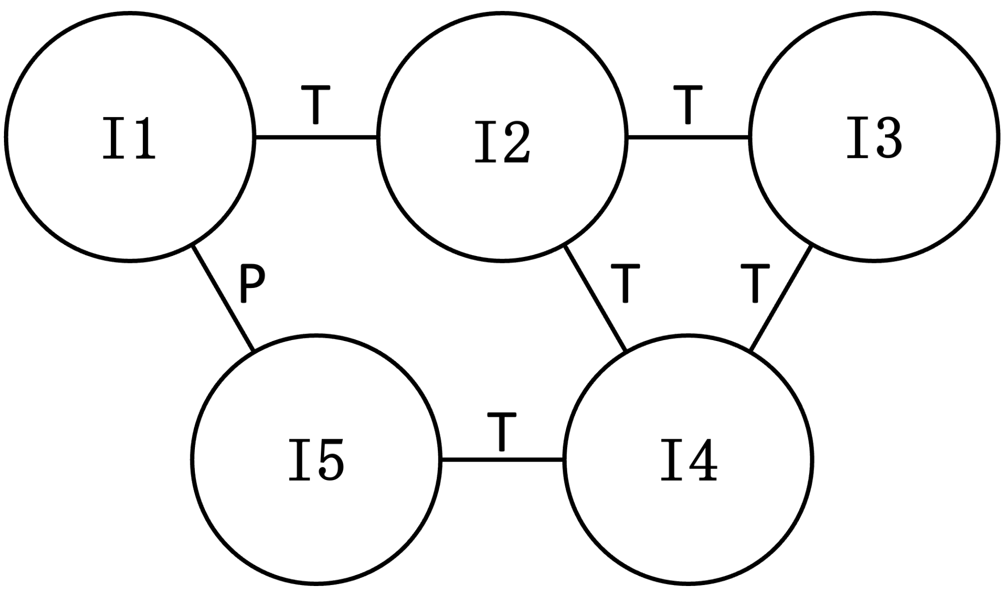
   - I1: 9:00 to 11:00
   - I2: 10:00 to 13:00
   - I3: 12:00 to 14:00
   - I4: 12:00 to 16:00
   - I5: 15:00 to 17:00

   In addition to the times, we have the following constraints:

   - A room can only be assigned to one instructor at a time.
   - Room R3 is too small for instructor I1.
   - Room R3 is too far away from instructor I2 such that he will not use it.
   - Instructor I4 does not like the projector at room R2.
   - Instructor I5 refuses to use the same room as I1, regardless of the time difference.

   Solve this constraint satisfaction problem. The variables are the instructors and the domains are the rooms. Answer the questions below based on the given information. Provide your answers on the next page.

   (a) (5 points) Draw the binary constraint graph of this problem. Write "T" on top of the edges if the constraint is due to time overlaps, "P" if the constraint is due to the instructor preferences and "TP" if it is due to both. Note that unary constraints should not show up!



   Grading: 5 nodes and 6 edges. 0.5 for each edge and 0.25 for correct labels for a total of 4.5 points. Deduct -0.25 for unnecessary edges (no deduction for wrong label). 0.5 point for having all the nodes.

(b) (2 points) Use the unary constraints to restrict the domains of the variables and write the resulting domains in the second row (starting with Step 1) of the table.
Trivial, see the first row of the table.

(c) (10 points) Solve the problem by using:

- The minimum remaining value (MRV) heuristic with degree heuristic as the first and alphabetical ordering as the second tie-breaker to pick the next variable.
- The least constraining value (LCV) heuristic with alphabetical ordering as the ties breaker to pick the next value
- The forward checking (FC) heuristic to filter the domains

Fill the table as you solve the problem. The Var column should have the picked variable and the Val column should have the picked value. Each cell under the variable columns should have the domains of the variables at the start of the step. Put a dash (-) under the variable if you have already made an assignment and an empty set symbol (∅) if the domain ends up being empty. Do not backtrack if you get an empty domain but stop your solution. If a variable ends up having a domain with only a single variable, do not assume that it is assigned automatically. Write the final assignments at the last row (the Final row) but note that Var and Val cells of this row will be empty.

	I1	I2	I3	I4	I5	Var	Val
Step 1	R1,R2	R1,R2	R1,R2,R3	R1,R3	R1,R2,R3	I2	R2
Step 2	R1	-	R1,R3	R1,R3	R1,R2,R3	I1	R1
Step 3	-	-	R1,R3	R1,R3	R2,R3	I4	R1
Step 4	-	-	R3	-	R2,R3	I3	R3
Step 5	-	-	-	-	R2,R3	I5	R2
Final	R1	R2	R3	R1	R2		

See the table and the steps below (2 points per step).

1. I1, I2 and I4 all have a MRV of 2. I2 and I4 have a DH of 3, and I2 comes before I4 alphabetically. For I2, R1 removes 3, R2 removes 2 values from neighbors (I1, I3).
2. I1's MRV is 1. R1 is the only value left. Remove R1 from I5.
3. MRV is tied. DH of I4 is 2, rest is 1 (note that DH is calculated on unassigned neighbors!). R1 and R3 both remove 2, so we pick R1 based on order.
4. MRV of I3 is 1 and only R3 left
5. Only I5 left, picking R2 based on order.

4. (15 points) Now you are asked to solve the same room scheduling problem using hill-descending. The objective is the number of conflicts which you want to minimize. At each step, you are only allowed to change one variable's assignment. The domains of the variables are again {R1, R2, R3}. You are going to fill the table below given the initial assignment. At each step, write the value of the objective function (starting from the Init row) and circle the assignment you are going to change. Stop if you get to a local minimum. The size of the table is not an indication of the solution length. You can extend it if you need to or finish before filling it up. As before, break all ties alphabetically for variables and by ascending order for values. Any step that decreases the objective will receive points, but optimal ones will receive more.

   We are providing the schedule and the constraints below. Copying your constraint graph may also help.

   - I1: 9:00 to 11:00
   - I2: 10:00 to 13:00
   - I3: 12:00 to 14:00
   - I4: 12:00 to 16:00
   - I5: 15:00 to 17:00

   - A room can only be assigned to one instructor at a time.
   - Room R3 is too small for instructor I1.
   - Room R3 is too far away from instructor I2 such that he will not use it.
   - Instructor I4 does not like the projector at room R2.
   - Instructor I5 refuses to use the same room as I1, regardless of the time difference.

   Fill in the table below based on the question. The number of steps given in the table is not an indicative of solution length (can be shorter or longer)

   R3 Version

	I1	I2	I3	I4	I5	Obj.
Init.	R2	R3	R2	R3	R2	3
Step 1	R2	R1	R2	R3	R2	1
Step 2						

   R1 Version

	I1	I2	I3	I4	I5	Obj.
Init.	R2	R3	R2	R3	R2	3
Step 1	R2	R1	R2	R3	R2	1
Step 2	R3	R1	R2	R3	R2	0

   Let's number the constraints:

   1. $I1 \neq R3$ (or $I1 \neq R1$)
   2. $I2 \neq R3$
   3. $I4 \neq R2$
   4. $I1 \neq I2$
   5. $I1 \neq I5$
   6. $I2 \neq I3$
   7. $I2 \neq I4$
   8. $I2 \neq I5$
   9. $I4 \neq I5$

   The "R3" version of the solution:
   Step 1:
   The current assignment is violating constraints 2,5 and 7, thus the objective value is 3.
   I2 is associated with 2 of the violations (2 and 7) so we start checking that variable. Assigning R1 would fix the constraints 2 and 7. Assigning R3 would fix these but lead to violations of 4,6 and 8. Thus we make the assignment I2 = R1

Step 2:
The current assignment is only violating the 5th constraint, resulting in an objective value of 1.
I1 and I5 are associated with this constraint so let's start with these. I1 = R3 would fix 5 but violate 1. I1 = R1 would fix 5 but violate 4. I5 = R1 would fix 5 but violate 8 and I5 = R3 5 but violate 9.
Thus we are stuck at a **local minima**, there are no moves that can decrease the objective value!
Grading (15 for best selection, 12 otherwise):

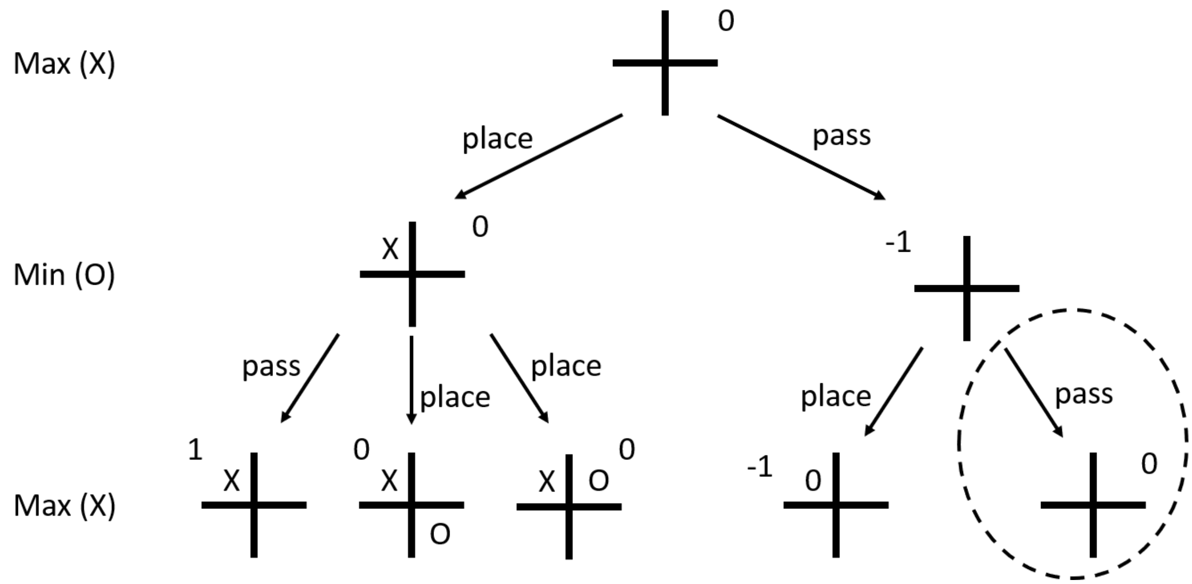
- Step 1: Selecting I2 - R1 6 points. Correct objective 3 points
- Step 2: Figuring out local minima 4 points, correct objective 2 points. Doesn't matter if the table was filled or not for the corresponding row.
- Step 1 (alternative): A selection that decreases the objective 2 points, Correct objective with this selection 3 points
- Step 2: If previous step did not lead to a local minima, 2 points for a selection that decreases the objective, 2 points for correct objective
- Step 3 and beyond: If all the remaining steps look correct, 3 additional points (steps, local minima or end detection etc.)

The "R1" version of the solution:
Step 1:
The current assignment is violating constraints 2,5 and 7, thus the objective value is 3.
I2 is associated with 2 of the violations (2 and 7) so we start checking that variable. Assigning R1 would fix the constraints 2 and 7. Assigning R3 would fix these but lead to violations of 4,6 and 8.
Thus we make the assignment I2 = R1
Step 2:
The current assignment is only violating the 5th constraint, resulting in an objective value of 1.
I1 and I5 are associated with this constraint so let's start with these. I1 = R3 would fix 5. I1 = R1 would fix 5 but violate 1 and 4. I5 = R1 would fix 5 but violate 8 and I5 = R3 5 but violate 9.
Thus we make the assignment I1 = R1.
At this point we can see that the objective is 0 and we have reached our solution.
Grading (15 for best selection, 12 otherwise):

- Step 1: Selecting I2 - R1 4 points. Correct objective 2 points
- Step 2: Selecting I1 - R3 4 points. Correct objective 2 points.
- Final: Detecting objective = 0 and stopping 3 points.
- Step 1 (alternative): A selection that decreases the objective 2 points, Correct objective with this selection 2 points.
- Step 2: If previous step did not lead to a local minima, same as above step. If it did, figuring this out 4 points
- Step 3 and beyond: If all the remaining steps look correct, 4 additional points (steps, local minima or end detection etc.)

5. (22 Points) Consider a game of 2×2 tic-tac-toe (aka X-O-X) where each player has the additional option of *passing* (i.e. not marking any square) and that X goes first. Note that there are only 4 squares in such a game instead of the usual 9. The player who places their mark 2-in-a-row wins.

   (a) (6 points) Draw the full game tree down to depth 2 (1 play for X and 1 play for O). You do not need to show nodes that are rotations or reflections of siblings already shown. (Your tree should have five leaves and 3 levels including the root). You can visualize each game state (node in a tree) as a 2x2 grid to make it easier. The order of actions does not matter for this question



   Compact graph (no reflections etc): 6 points. Missing non-root nodes -1. Non-compact: 4 points max, -1 for any missing type of node.

   (b) (4 points) Suppose the evaluation function is the number of Xs minus the number of Os and that the agent which plays X is the maximizer. Mark the values of all leaves and internal nodes on the map.

   See the tree (on top of the nodes, 0.5 each)

   (c) (4 points) Circle any node that would not be evaluated by alpha-beta during a left-to-right exploration of your tree. If there is nothing to be pruned explicitly say so.

   See the tree (dashed ellipse). If you first used pass from the root node or the right child of root node, this would not have been pruned. So a non-pruned tree can also be a solution.

(d) (4 points) Suppose we wanted to solve the game to find the optimal move (i.e., no depth limit). Explain why alpha-beta with an appropriate move ordering would be much better than minimax. How can you modify the minimax algorithm to work?

Minimax will never terminate due to the *passing* action but alpha-beta pruning would prune the nodes if the neighbors are expanded in a certain order (see the answer to previous part for example). Note that alpha-beta would also never terminate if the passing is always evaluated first. To make these algorithms work, we would need to keep track of previously visited nodes and not evaluate them further, similar to the graph search ideas. This would avoid the infinite loop and let us reach the terminal states to calculate optimal values.

(e) (4 points) Now we change the rules conditioning that the first player to complete 2-in-a-row loses (instead of winning), while keeping the passing action. Describe how we would find the optimal play for this game.

The question was asking for an algorithmic solution which should be self-evident based on the course! If you gave a "human answer" you would be still eligible for partial grade.

For optimal play, we want to search until the terminal states. The problem is with the passing action, we get infinite depth. Moreover, at some point in the game, the optimal action for both players is passing.

Since passing is optimal for both players, minimax and alpha-beta pruning will both lead to infinite depth. We can keep track of previously visited nodes similar to the previous part's answer to avoid this. However, we would not be able to calculate their values! At this point, we can:

- Assume that this case is equivalent to a draw. However, this would not work for games that are winnable by either player the repeated position. For this game though, it would be acceptable.
- Assign an "unkown value token" to these states and assume that this is worse than or equal to winning and better than or equal to losing. The algorithms would need to be changed to incorporate this. However, if the game allows for draws, then we are still in trouble. One can assume that this unknown token always supersedes draw in max and min comparisons, implying that the value of a branch is always unknown even if one of the siblings is a draw. However, this becomes problematic when the current player needs to chose between draw and unknown (i.e. at the root).

I will mostly look at how you used your adversarial search knowledge to argue your points and how correct/logical you were.