KOÇ UNIVERSITY
COLLEGE OF ENGINEERING

COMP 341: INTRODUCTION TO ARTIFICIAL INTELLIGENCE

Midterm 1

FALL 2023, 24/11/2023
DURATION: 120 MINUTES

Name: Solutions

ID: 00110001 00110000 00110000

- This exam contains 10 pages including this cover page and 6 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	6	Total
Points:	16	12	26	16	16	18	104
Score:							

1. (16 points) True or False : 16/16

False Goal based agents care about the goodness of individual states.

F False Breadth First Search should be preferred instead of Depth First Search for systems with limited space.

T True A* Search may not return the same solution with Uniform Cost Search.

T True Genetic algorithms can be used to solve Constraint Satisfaction Problems.

T True Randomness helps to avoid local minima.

F False In constraint satisfaction problems, variables must have the same domain.

T True The value calculated by alpha-beta pruning may not be correct for all the states.

F False Chance nodes can not be used in adversarial games to model the opponent.

2. (12 points) Answer the miscellaneous questions below

(a) (4 points) For the search based solution of CSPs, why are we choosing Depth First Search and not Breadth First Search?

2

We know that the solution is either at a limited depth (finite number of variables assignments) or it does not exist, so DFS is complete for solving CSPs. Related to this (solution being at exactly at a depth or not existing), DFS is faster in finding solutions since it wants to go deep first. On top of these, DFS uses linear space vs exponential space for BFS.)

(b) (4 points) Genetic algorithms may get "stuck" if the population diversity is not enough. Why would this be the case? What mechanism did we learn about that tries to alleviate this?

Why would we get lack of diversity? (This was not the question but the wording is not clear so we will give points to this - 2 points) Initial population may not be diverse, mediocre individuals may be much easier to reach than performant individuals and the two may not be similar, initial populations may not have the genetic makeup to "evolve" performant individuals, performant individuals may be rare/ahrd to find, stochastic mechanisms may not be widespread enough (e.g. more mutations needed than allowed for a single individual) etc.

Why lack of diversity leads top stagnation? (2 points) This is trivial I think. The offspring will not be different than their parents if the population pool is not diverse.

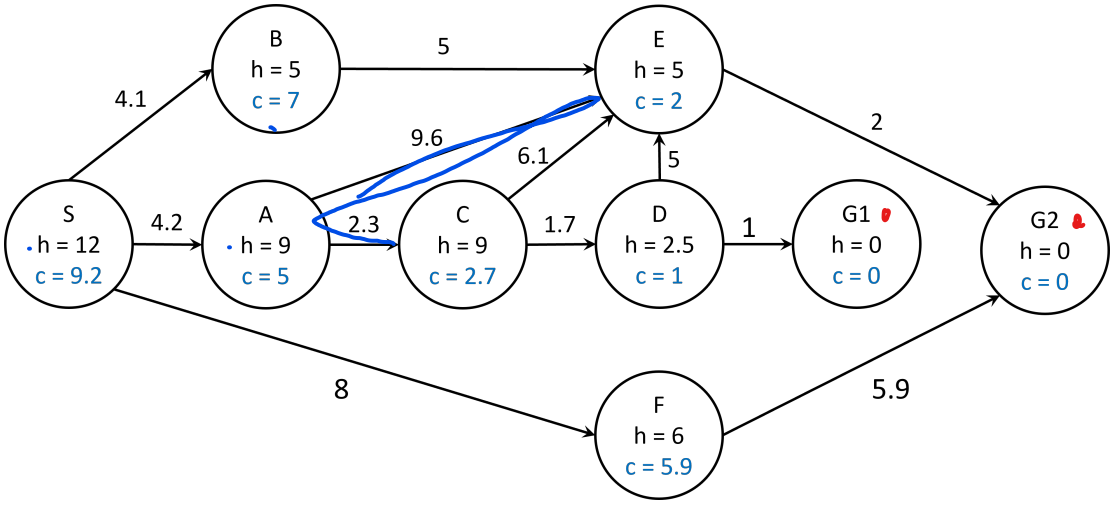
Mechanisms to alleviate this? (2 points, 2 out of 3 is enough) Random/stochastic steps such as soft individual selection (picking proportional to the fitness instead of the best), picking the cross-over location and mutations (These are what we have seen in the class.)

(c) (4 points) Can you come up with an additional idea to overcome this lack of diversity? (This is a bonus question)

This is open-ended. No points for repetition of the mechaisms in the question above. One good idea is enough to get 4 points. Some ideas are (their combination is possible): penalising lack of diversity/rewarding diversity (e.g. modify the fitness function), adaptive mutation rate based on measured diversity, randomly generating new individuals from time to time etc. (Diversity can be measured by clustering with intra- and inter- cluster distance, fitting a probabilitistic model and measuring likelihood etc.)

3. (26 points) You are given the directed and weighted graph below, where **S** is the start state, and **G1** and **G2** are the goal states. The arced arrows represent transitions and directions, and cost of each transition is given next to the arcs. The numbers inside the nodes represent their heuristic value.

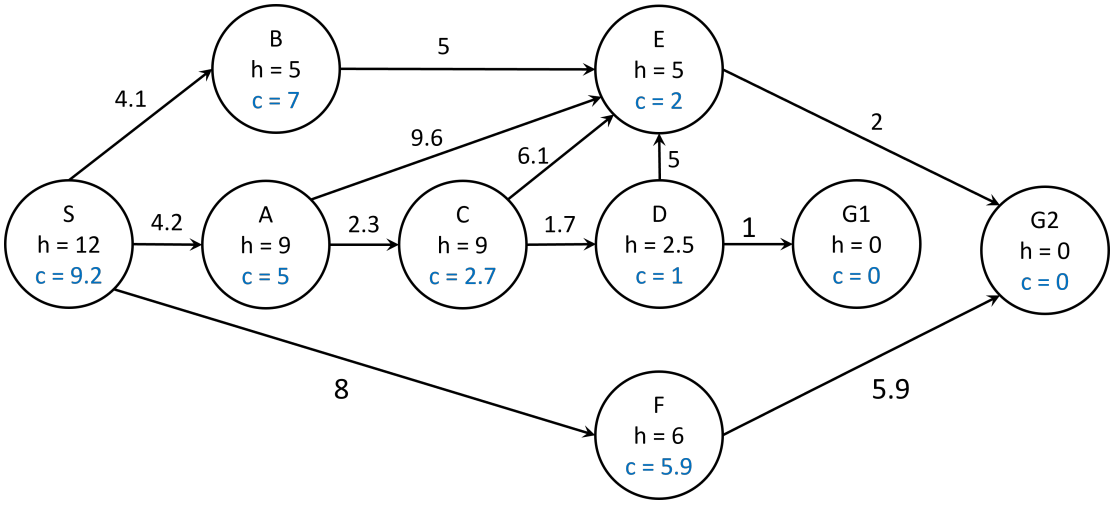
   You are asked to write the visiting/expanding order (order of popping from the frontier) of the nodes and the resulting solution paths with the given algorithms. The neighbors are added alphabetically to the frontier. The alphabetical order is: A,B,C,D,E,F,G1,G2,S. For algorithms using priority queues, break the ties alphabetically and assume that the priority values are updated when an existing node with a higher priority is being pushed.



   (a) (8 points) Iterative Deepening Depth First Search. (You may not need to go down to all the depths, stop early if you reach a goal):
   Popped Node Order:
   Level 0: S
   Stack based:
   Level 1: S-F-B-A
   Level 2: S-F-G2
   Recursive:
   Level 1: Recursive: S-A-B-F
   Level 2: Recursive: S-A-C-E-B-F-G2

   Resulting Path:
   Stack based: S-F-G2
   Recursive: S-F-G2

(repeated to give you an additional figure to work on)

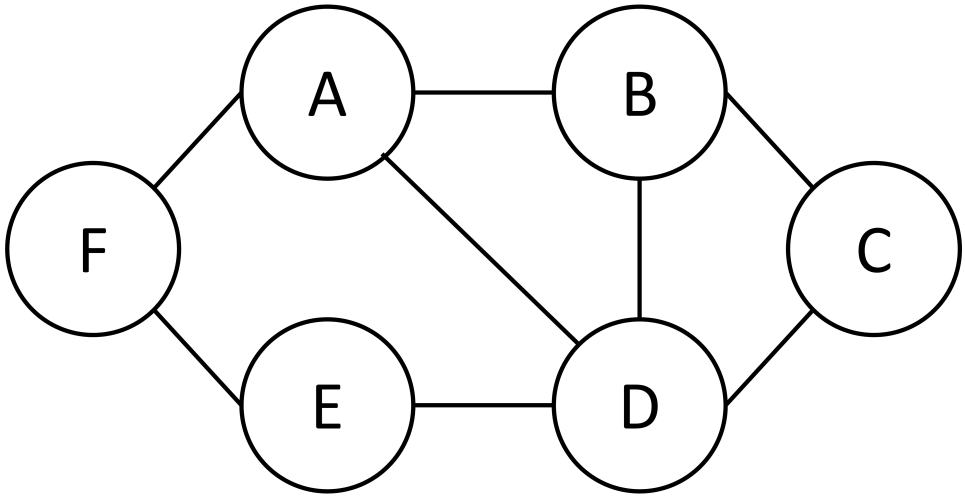


(b) (8 points) Uniform Cost Search:
Popped Node Order:
S-B-A-C-F-D-E-G1
Resulting Path:
S-A-C-D-G1

(c) (8 points) A* Search:
Popped Node Order:
S-B-A-F-G2
Resulting Path:
S-F-G2

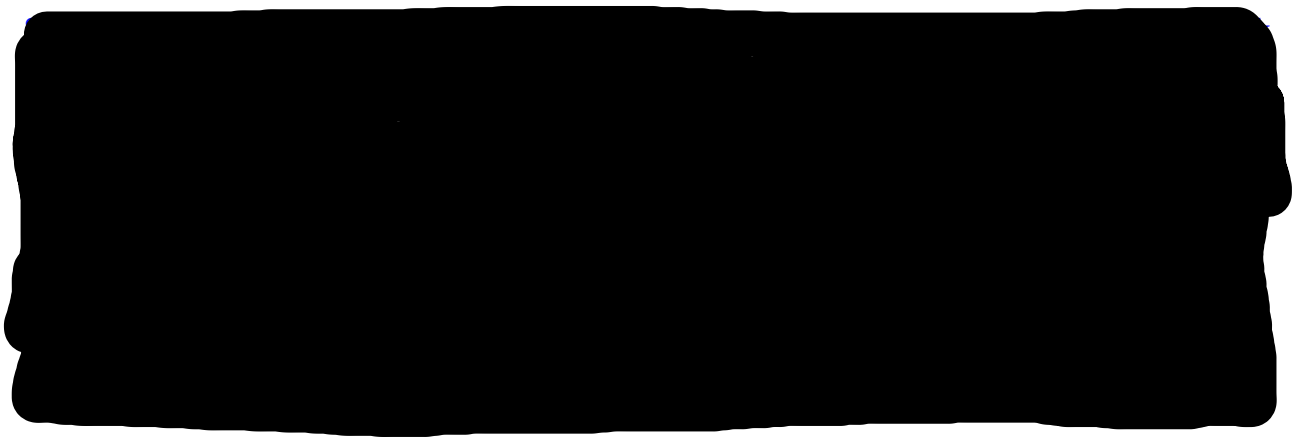
(d) (2 points) Is the given heuristic admissible?
No as clearly seen in the figure that many nodes have higher heuristic value than their costs. Another observation is that the A* solution is not the same as the UCS solution!

4. (16 points) Consider the following constraint graph where the nodes represent variables and arcs represents the difference constraint, i.e, if two variables have an arc between them, they cannot be the same value. The domains of all the variables are initially the same: {1, 2, 3}.

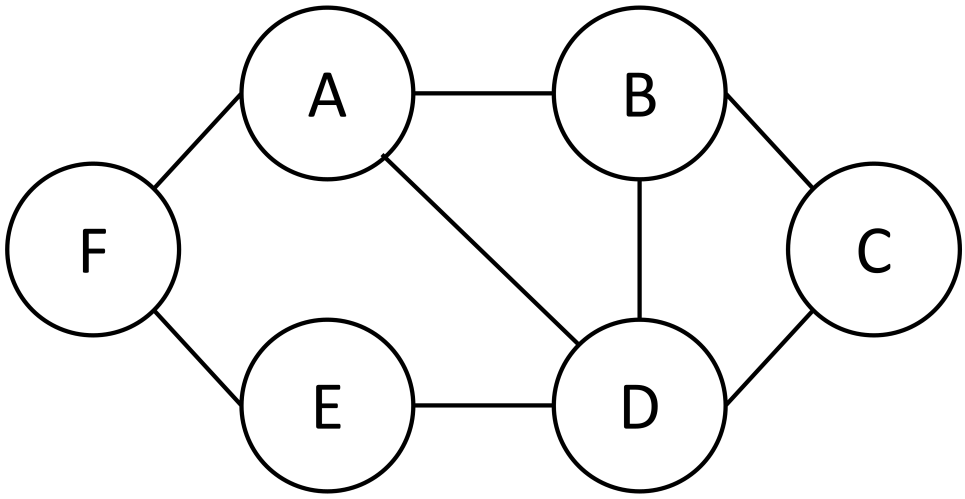


Solve this CSP by using the minimum value remaining (MRV) heuristic, with degree heuristic (DH) as the tie breaker, least constraining value (LCV) heuristic, and forward checking. Fill in table below and provide additonal work in the empty space for partial credit. Under each variable column, write the remaining domains of the variables after each step. Write the variable(s) with the MRV in the MRV column, circling the chosen variable if there is more than 1. Write the chosen value in the LCV column. Lastly when you do FC, cross out the values you eliminate on the corresponding row. Break all remaining ties alphabetically for variables and by ascending order for values (e.g. pick 1 over 2). Put a dash in the remaining cells of the variable you have assigned a value to. Do not backtrack but stop if you get an empty domain. With this condition, you can at most do 6 assignments, hence the table will be enough. Write the final assignments in the last row, leave any cell blank if its domain was empty before assignment.

	A	B	C	D	E	F	MRV	LCV
Init.								
Step 1								
Step 2								
Step 3								
Step 4								
Step 5								
Final								



5. (16 points) Consider the following constraint graph again. This time you are going to solve it 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 {1, 2, 3}.



You are going to fill the table below, given the initial assignment. At each step, write the value of the objective function 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.

	A	B	C	D	E	F	Objective
Init.	1	1	2	1	2	2	
Step 1							
Step 2							
Step 3							
Step 4							
Step 5							
Step 6							

Based on the alphabetical tie breaker, we pick E-3.

Step 2: The current assignment violates B-D thus the objective is 1. We check B and D to see that nothing decreases the objective. Thus we are at a local minima. If we had to chose, we would chose B-2. Filling the table just to show it, you did not need to enter the last row.

Grading:

- Correct variable-value combination selection (e.g. A-3 in the initial step), 3 points per (for a total of 9 points)
- Not correct but still a value that decreases the objective (e.g. D-2 in the initial), 1 points per (for a total of 3 points, more entries are ignored)
- Correct objective 2 points per (for a total of 6, the last step is not needed). Wrong selection but correct objective can still get upto 6 points (max 3 correct objectives, more entries are ignored).
- Local minima stopping 1 point
- Half the points if wrong objective (e.g. number of satisfied constraints)

Alphabetical first solution to the previous questions (8 points)

	A	B	C	D	E	F	Objective
Init.	1	1	2	1	2	2	4
Step 1	3	1	2	1	2	2	2
Step 2	3	1	2	1	3	2	1
Step 3	3	2	2	1	3	2	1
Step 4							
Step 5							
Step 6							

Some of you directly started with the alphabetical order. We will give partial credit to an extent.

Initial Step: The current assignment violates A-D, A-B, B-D and E-F constraints thus the objective is 4. Swapping A to 2 and 3 would decrease the objective to 3 and 2 respectively. We pick A-3.

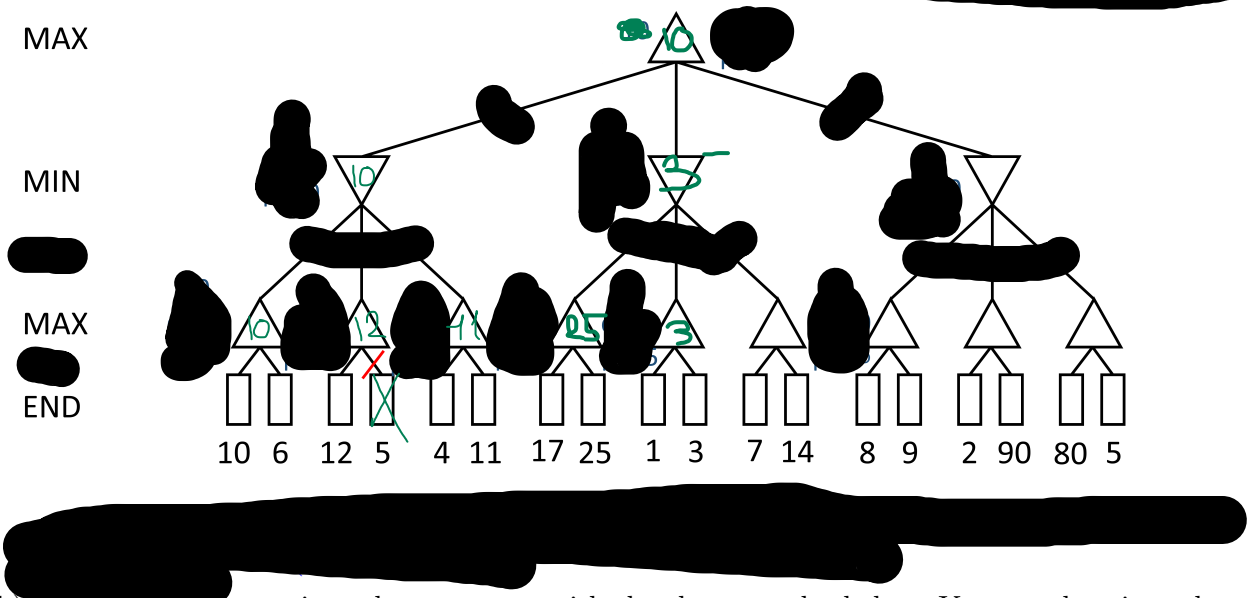
Step 1: The current assignment violates B-D and E-F constraints thus the objective is 2. We check B,D,E and F Swapping B to 2 and 3 would not change the objective. Swapping D to 2 and 3 would not change the objective. Swapping E to 1 would not change the objective and swapping to 3 would decrease the objective to 1. Swapping F to 1 would decrease the objective to 1 and swapping to 3 would not change it.

Based on the alphabetical tie breaker, we pick E-3.

Step 2: The current assignment violates B-D thus the objective is 1. We check B and D to see that nothing decreases the objective. Thus we are at a local minima. If we had to chose, we would chose B-2. Filling the table just to show it, you did not need to enter the last row.

6. (18 points) Adversarial Search

(a) (9 points) Given the game tree below, apply alpha-beta pruning. Assume a depth-first left-to-right expansion. Cross out the pruned paths and write the values of the states, as calculated by the alpha-beta algorithm, next to them



(b) (9 points) You are given the game tree with the chance nodes below. You are also given the knowledge that the **terminal values are guaranteed to be between 0 and 9**. You can use this information to prune this tree. The alpha-beta pruning algorithm needs to be modified to incorporate this by testing both values of the range. Show the connections that can be pruned (cross-out the arcs in the figure above). You can use the empty space for calculations.