NATIONAL UNIVERSITY OF SINGAPORE

SCHOOL OF COMPUTING

MIDTERM ASSESSMENT FOR
Semester 1 AY2020/2021

CS3243: INTRODUCTION TO ARTIFICIAL INTELLIGENCE

September 30, 2020 Time Allowed: 90 Minutes

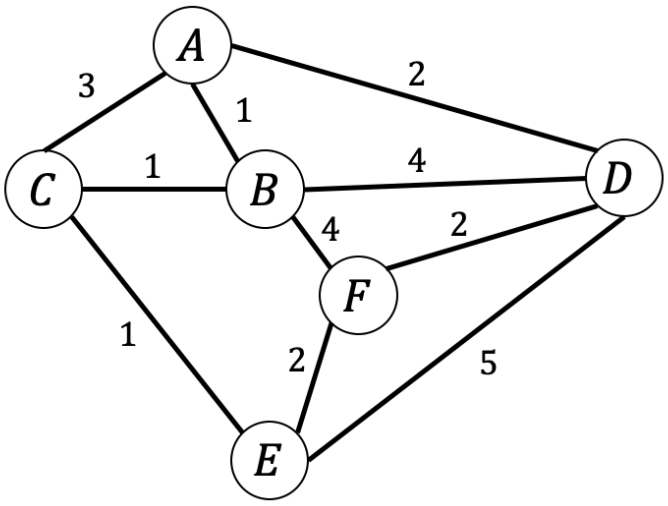
INSTRUCTIONS TO CANDIDATES

1. This assessment contains FIVE (5) questions and each question is worth 20 MARKS. It is set for a total duration of 90 MINUTES. You are to complete all 5 questions.
2. This is an OPEN BOOK assessment, without electronic devices. The only electronic material that you are allowed to access is the file ***midterm-scribes.pdf*** provided by us on LumiNUS. You can access any printed or handwritten material in your physical possession at the start of the exam.
3. You are allowed to use NUS APPROVED CALCULATORS.
4. If something is unclear, solve the question under a reasonable assumption. State your assumption clearly in the answer. If you must seek a clarification, the invigilators will only answers questions with Yes/No/No Comment answers. To ask for clarification to invigilator, please raise your hands on Zoom.
5. In our view, the higher number questions are more difficult but your mileage may vary.
6. You may not communicate with anyone other than invigilators in any way.

STUDENT NUMBER: _____

EXAMINER'S USE ONLY		
Question	Mark	Score
1		
2		
3		
4		
5		
TOTAL		

1. (20 Marks) Suppose we have the following graph with undirected edges. For BFS & DFS - apply alphabetical tie-breaking rule when pushing into the frontier, for UCS, apply alphabetical tie-breaking when there are multiple nodes of same cost in the priority queue.



Start from the node *A*. Suppose the goal node is *F*. Give the sequence of explored (explored means popped from the frontier) nodes by tracing each of the following uninformed search algorithms (using graph search):

BFS: _____

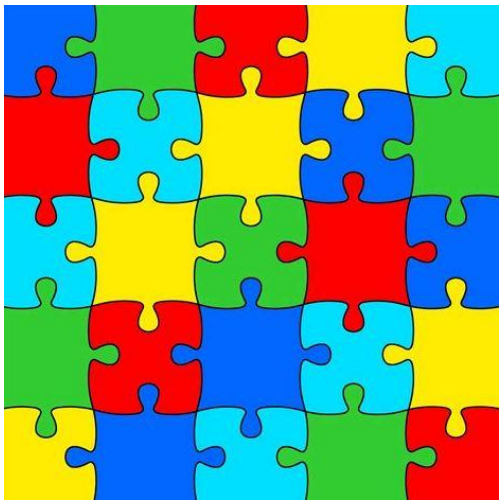
DFS: _____

UCS: _____

You should format your answer in the form X-X-X-..., where X is the alphabet corresponding to the node explored. Make sure your solution has no spaces and you adhere to the format exactly. For example, given the explored sequence X, Y, Z, your answer should be formatted as X-Y-Z

If necessary, you may also include assumptions taken here. However, assumptions that are contradictory to any instructions in the question or that are unreasonable will be invalid.

2. (20 Marks) You are given an n-piece unassembled jigsaw puzzle set (you may assume that each jigsaw piece can be properly connected to either 2, 3 or 4 pieces) that assembles into an (m * k) rectangle (i.e., $n = m * k$). There may be multiple valid final configurations of the puzzle, the picture illustrates an example.



Formulate the above as a search problem. More specifically, define the following:

- **State representation**
- **Initial state**
- **Actions**
- **Transition model**
- **Step cost**
- **Goal test**

If necessary, you may also include assumptions taken here. However, assumptions that are contradictory to any instructions in the question or that are unreasonable will be invalid.

3. (20 Marks) Accordion Solitaire is a card game using one deck of playing cards of n cards. Each card in the deck belongs to exactly one suit: {Spades, Hearts, Clubs, Diamonds} and one rank: {Ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, King}. Do not make assumptions on the card composition of this deck, unless explicitly stated otherwise -- for instance, there may be n = 200 or n = 5000 cards in the deck.

The suites are as follows (ignoring colour):

The objective is to compress the entire deck into one pile.

Gameplay:

- The cards from the entire deck are spread out in a single line (due to screen width constraints, we can also arrange it in a snake fashion as in the figure above, but the idea still remains).
- A pile (initially, one card = one pile), P_1, can be moved on top of another pile, P_2, if P_1 is just before P_2 in the sequence, or three piles before P_2, and, both P_1 and P_2 have the same **suit** or **rank**. Gaps left behind are filled by moving preceding piles to the left (to close the gap). You can choose to make whichever moves are valid (if any) at any point in time.

Another example is as follows. Suppose we have the following state:

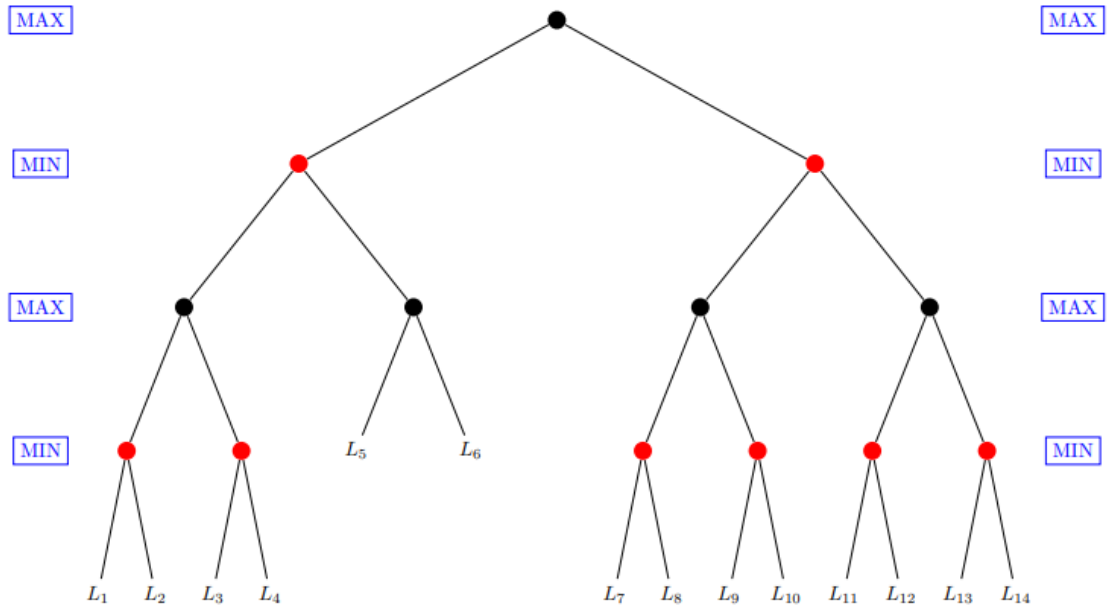
According to this example, either **6 black spades** or **5 red hearts** can be placed over **5 black spades**. These are the only valid moves given the state. The game is won when all cards are compressed into one pile.

(Continued on the next page.)

Design an admissible heuristic for this search problem. Your heuristic may not be h(n) = 0 for all states n, the (abstract) optimal heuristic, or a linear combination/simple function of those two. You may assume that there exists a sequence of moves that leads to winning the game, i.e. there is some path to a goal state. You must prove that your heuristic is admissible.

If necessary, you may also include assumptions taken here. However, assumptions that are contradictory to any instructions in the question or that are unreasonable will be invalid.

4. (20 Marks)



Black nodes correspond to the MAX player, whereas red nodes correspond to the MIN player. Leaf nodes are labelled as L_1, L_2, …, L_{14}.

Assume that we run the alpha-beta algorithm on the above tree from **left to right** (just the way we did in the lecture), what is the maximum number of **leaf nodes** whose evaluation we can skip? (Only fill an integer value between 0 and 14 (inclusive). I.e., if you answer is 0 then you should simply fill 0).

ANSWER: _____

You must justify your answer by

(1) First, assign every leaf node with its utility value belonging to the set {1,2,3,4}. In other words, every leaf node's utility value can be either 1 or 2 or 3 or 4. Now for every value, fill the list of indices of nodes that take the corresponding value.

List of indices of nodes with value 1 _____

List of indices of nodes with value 2 _____

List of indices of nodes with value 3 _____

List of indices of nodes with value 4 _____

Filling instruction: You should only fill the indices of the leaf nodes. For example, if your answer is that the nodes L_1, L_2, L_3, L_4, L_5, L_6 take the utility value 1, then in the blank corresponding to utility value 1, you should fill: 1,2,3,4,5,6.

(Continued on next page.)

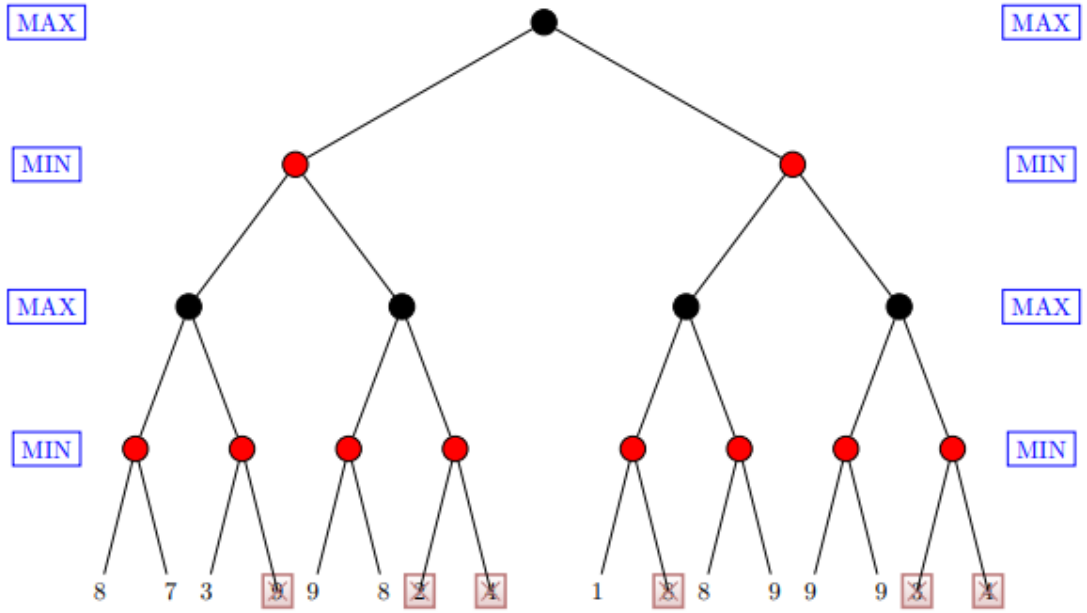
(2) Now, we want to know the leaf nodes that will be evaluated and whose evaluations will be skipped.

List of indices of leaf nodes that will be evaluated: _____

List of indices of leaf nodes whose evaluations will be skipped: _____

Filling instruction: You should only fill the indices of the leaf nodes. For example, if your answer is that the nodes L_1, L_2, L_3, L_4, L_5, L_6 take the utility value 1, then in the blank corresponding to utility value 1, you should fill: 1,2,3,4,5,6.

To avoid any confusion, find below the example from the lecture where every leaf node whose evaluation was skipped is marked with a light red box.



If necessary, you may also include assumptions taken here. However, assumptions that are contradictory to any instructions in the question or that are unreasonable will be invalid.

5. (20 Marks) Assume there exists just one goal node G and one start node E.

Suppose you have the function OPT(s) that returns the optimal path cost from node s to the goal node G.

A path is defined as sequence of nodes, e.g., $\pi = s_0, s_1, s_2, s_3$ and now we say that nodes s_0, s_1, s_2, s_3 lie on π.

We are given a heuristic function h such that:

$$h(s) := \begin{cases} \frac{OPT(s)}{2}, \text{ whenever s lies on an optimal path from E to G} \\ k(s), \text{where } k(s) > OPT(s), \text{ if there is no optimal path } \pi \text{ from E to G such that s lies on } \pi \end{cases}$$

Observe that when s does not lie on any optimal path from E to G, h(s) can take any value larger than OPT(s).

Prove or Disprove the following statement:

"A* algorithm based on graph-search will always find an optimal path from E to G with the above heuristic function h."

You may not assume anything about the graph structure other than the existence of a start node E and goal node G. Also, no further assumptions or restrictions on h are allowed.

In your answer, you can use the following terms introduced in the class without redefining them:

$$\hat{f}(u), \hat{f}_{pop}(u), f(u), g(u), h(u), \hat{g}(u), \hat{g}_{pop}(u) \text{ for node u}$$

For ease of typing, you can use the following shorthand for some of the above terms:

hat-f for $\hat{f}$

hat-g for $\hat{g}$

hat-f-pop for $\hat{f}_{pop}$

hat-g-pop for $\hat{g}_{pop}$

END OF PAPER