The National Higher School of Artificial Intelligence
Introduction to Artificial Intelligence
Final Exam 18/03/2024
Duration: 1 hour 30 minutes

Last Name	
First Name	
Group #	

Exercise 1		**/ 7**
Exercise 2		**/ 6**
Exercise 3		**/ 8**
Exercise 4		**/ 6**
TOTAL		**/27** **/20**

1. **ANSWER ON THESE EXERCISE STATEMENT PAGES. IF YOU DO NOT HAVE ENOUGH, SPACE WRITE ON THE BACK OF THE SAME PAGE.**
2. **Use a PEN NOT a PENCIL everywhere. (A penalty will be applied if you use a pencil.)**
3. **DO NOT** USE ANY RED OR PINK (OR RELATED) COLOUR IN YOUR ANSWERS!
4. **WRITE NEATLY SO AS NOT TO BE PENALISED!**

Exercise 1: **7 marks** **20 minutes**

A. Answer each of the following statements in the table as TRUE or FALSE by putting a X in the corresponding cell. Each correct answer: +0.5; each incorrect answer: -0.25; no answer: 0.

1. Consider a graph search problem where for every action, the cost is at least ε, with $\varepsilon > 0$. Assume the used heuristic is consistent. (0.5 each question)

N°	Statement	TRUE	FALSE
a	Depth-first graph search is guaranteed to return an optimal solution.		*X*
b	Breadth-first graph search is guaranteed to return an optimal solution.		*X*
c	Uniform-cost graph search is guaranteed to return an optimal solution.	X	
d	Greedy graph search is guaranteed to return an optimal solution.		*X*
e	A* graph search is guaranteed to return an optimal solution.	X	
f	A* graph search is guaranteed to expand no more nodes than depth-first graph search.		*X*
g	A* graph search is guaranteed to expand no more nodes than uniform-cost graph search.	X	
h	Iterative deepening search never expands more nodes than breadth-first search before finding the goal.		*F*

2. (0.5 each question) Let $h_1(s)$ be an admissible A* heuristic. Let $h_2(s) = 2\, h_1(s)$. Then:

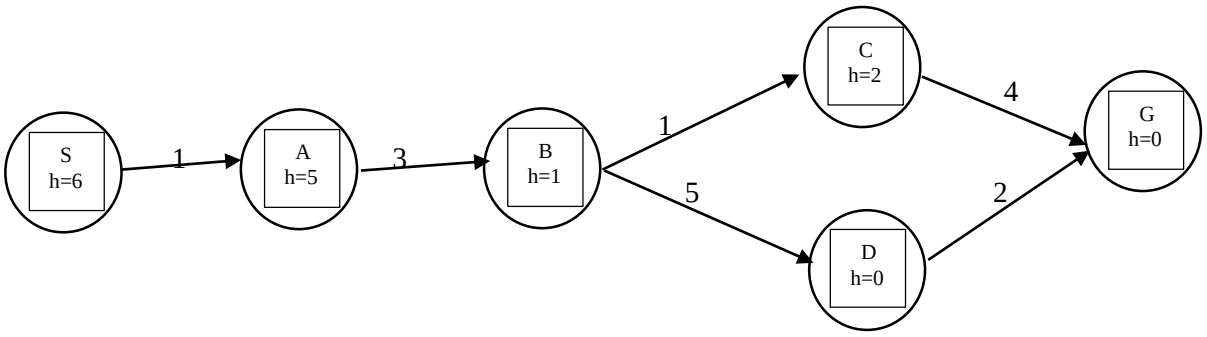
N°	**Statement**	**TRUE**	**FALSE**
a	The solution found by the A* tree search with h_2 is guaranteed to be an optimal solution.		*X*
b	The solution found by the A* tree search with h_2 is guaranteed to have a cost at most twice as much as the optimal path.	*X*	
c	The solution found by the A* graph search with h_2 is guaranteed to be an optimal solution.		*X*

4. Suppose that for a minimisation search problem, two heuristics h_1 and h_2 are admissible and for all nodes n, $h_1(n) < h_2(n)$. Then there do not exist any starting nodes for which A* using h_1 expands fewer nodes than A* using h_2. Give a brief explanation. (0.5 mark)

True: by definition h_2 dominates h_1 and we know from the lectures that "Domination translates directly into efficiency: A* using h_2 will never expand more nodes than A* using h_1".

3. The heuristic values for the graph below are not correct. For which **single state** S, A, B, C, D, or G could you change the heuristic value to make everything admissible and consistent? What interval of values are possible to make this correction? Give a brief explanation. (1 mark)

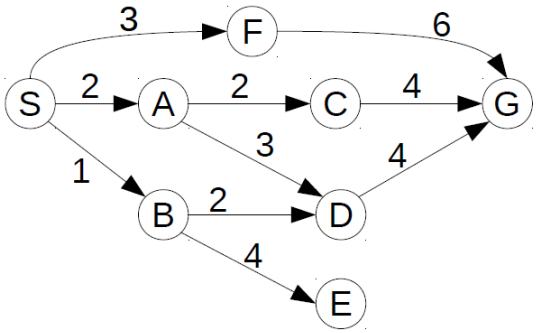
State: **B** Interval: **[2,3]**



Exercise 2: 15 minutes 6 marks

The graph in the figure below shows the state space of a hypothetical search problem. States are denoted by letters, and the cost of each action is indicated on the corresponding edge. Note that the graph is oriented, which means that the actions are not reversible (e.g., it is possible to move from state S to A, but not vice versa). The table next to the state space shows the value of an admissible heuristic function, considering G as the goal state. (It is easy to verify that such a heuristic is admissible.)

state space



heuristic function (goal state: G)

S	A	B	C	D	E	F	G
6	4	5	2	2	8	4	0

Considering S as the initial state, find a solution using each of the following search strategies, drawing the corresponding search tree:

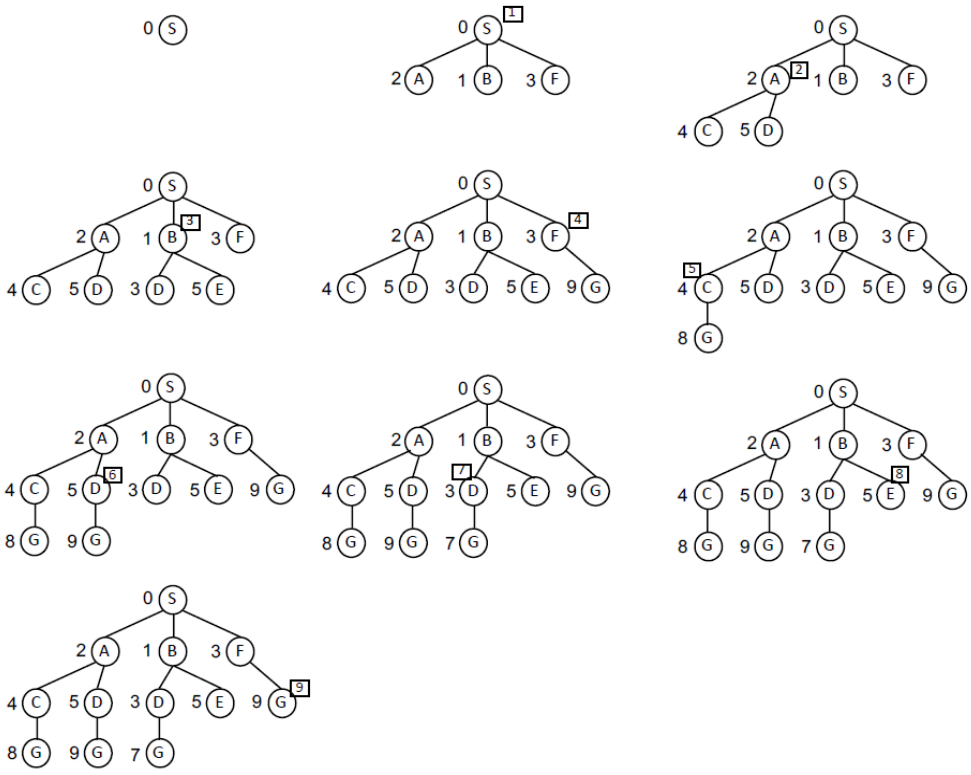
1- Breadth-first search
2- Depth-first search (leftmost node first)
3- A* search with the above heuristic

When drawing the search tree, you should clearly indicate: the order of expansion of each node (e.g., by numbering the expanded nodes according to the order of their expansion); the action corresponding to each edge of the tree; the state, the path cost and the value of the heuristic of each node. In the case of depth-first search, separately draw the search tree after any node is deleted (i.e. whenever backtracking is required).

Solution

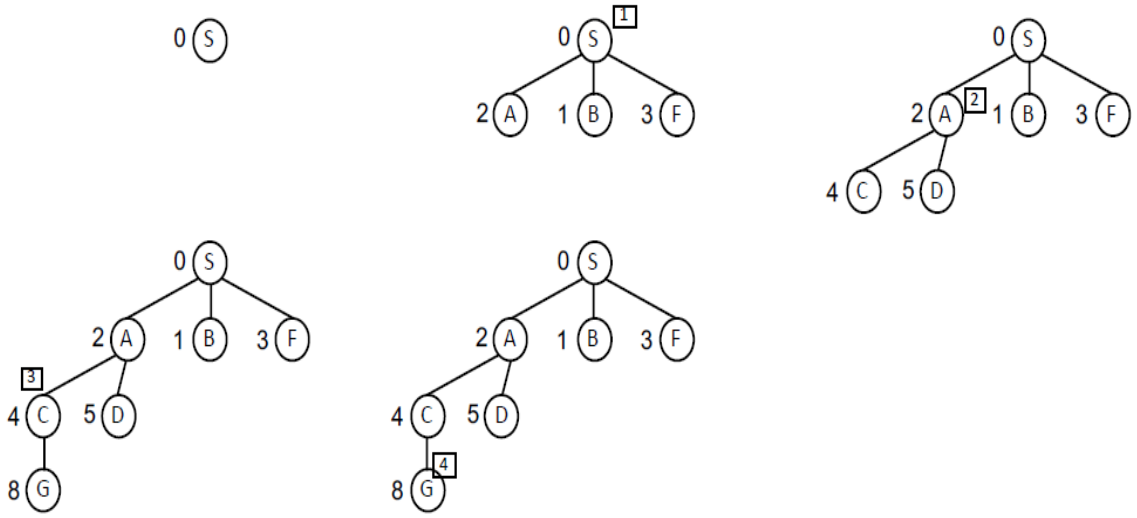
The search trees built by the three strategies are shown below. For breadth- and depth-first search, the number of the left of each node denotes its path cost (although it is not used by these strategies to choose the next leaf node to expand); for A*–search, the same number denotes the sum of the

path cost and of the heuristic function. The order of expansion is denoted by numbers inside squares. For the sake of clarity, the search tree obtained after each expansion step is shown.
Remember that, when several leaf nodes can be chosen for expansion (i.e., when there is more than one leaf node at the lowest depth for breadth-first search, or at the highest depth for depth-first search, or with the lowest value of the sum of the path cost and the heuristic function for A*–search), a random choice should be made among them; in this exercise we assume that the choice is made by considering the alphabetical ordering among the corresponding states (e.g., if the choice is among nodes A, B and F, node A is chosen).
Finally, remember that the goal test is applied when a node is selected for expansion: therefore, a solution is found not when a node containing a goal state is generated (by expanding its parent node), but when it is selected for expansion.
The search tree built by breadth-first search, step by step, is shown in the figure below. Note that different nodes can correspond to the same state (e.g., this happens to states D and G): this means that such a state can be reached from the initial state through different paths in the state space (i.e., different sequences of actions), possibly with different path costs: therefore, such nodes are kept distinct in the search tree. Note also that node E has no successors (no other state can be reached from state E, according to the state space): therefore, when it is selected for expansion (in step 8) no children nodes are added to the tree. Finally, note that the solution found by breadth-first search (S→F→G) is not the optimal one (i.e., the one with minimum path cost): this is due to the fact that breadth-first is not optimal.

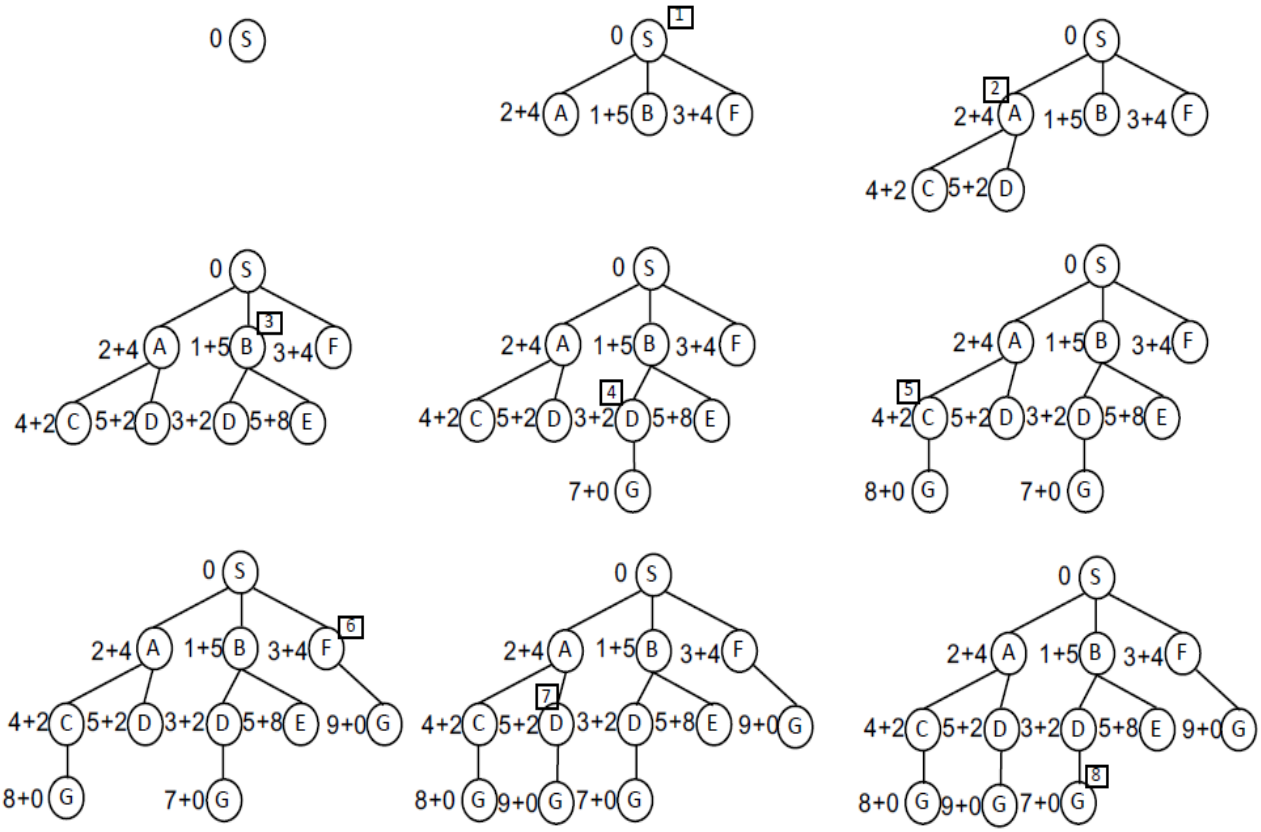


The search tree built by depth-first search is shown below. In this case a solution is found along the path which is explored first (when the node corresponding to state G is selected for expansion at step 4). Therefore, no backtracking step is carried out, and none of the already expanded nodes

is deleted. The solution found by depth-first search is not the minimum-cost one, as for breadth-first, since depth-first search is not optimal, either.



The search tree built by A*–search is shown below. Next to each node the corresponding path cost g and heuristic h are shown as g + h. Remember that, at each step, the leaf node with the minimum value of g+h is chosen for expansion (ties are broken randomly, as for all search strategies). The solution found at step 8 is guaranteed to be the minimum-cost one, since A* is optimal when an admissible heuristic is used.



Exercise 3: **8 marks**

Duration: 25 min max

In our application we need to process strings of letters like 'ABABAECCEC' and apply to them various stepwise transformations, the goal being to accept the string if applying some predefined transformations to it eventually leads to the 1-letter string 'E'. The 1-step transformations (rules) proceed by replacing a pair of letters by an equivalent one according to some logic of the application that you do not need to worry about. These equalities are defined by the following rules:

Rule R_1: AC = E Rule R2: AB = BC Rule R_3: BB = E Rule R_4: Ex = x for any letter x.

For example, the sequence ABBC can be transformed into AEC (using BB = E), and then AC (using Ex = x), and then E (using AC = E). Alternatively, one could apply a transformation to the leftmost pair of letters in 'ABBC', in this case AB, (using AB=BC) and then continue the transformations. We will assume that the logical order of applying the 1-step transformations to the strings of letters is from left to right. This means that for the string 'ABBC', AB=BC would be considered before BB = E. In this exercise, we assume that we want to process the string 'ABABAECCEC'

1- Formulate this problem as a search problem, giving a brief explanation how to calculate the cost from the start to the current position.
2- Suggest a heuristic to estimate the cost from the current position to the goal (two lines max).
3- Generate a tree representation, down to depth 2 of the tree, illustrating the expansion of nodes at each step using
   a. Breadth-First Search (BFS) and
   b. Depth-First Search (DFS).

   Label the expanded nodes with numbers to indicate the order of expansion for each strategy and

the branches with the rule number (R_1, R_2, etc.) that is applied at that point.

4- Do you think this problem would be solved if you continued building the tree as deeply as needed? If yes, give (JUST) a path from the root to the goal. Otherwise, explain what would need to be done to ensure the problem can be solved. Explain your answer in one sentence.

Solution:

1) Problem formulation:

Initial state = Any given string composed of the letters 'A', 'B', 'C', 'E'.

0.25 mark

Actions: 4 actions, each consisting in the application of one of the 4 rules:

Rule R_1: AC = E ; Rule R2: AB = BC ; Rule R_3: BB = E ; Rule R_4: Ex = x for any letter x.

0.25 mark

Goal Test: is the state (string) the same as 'E' **0.25 mark**

Transition model = {S'/

- Result(S,R1,S') for any state S and S' is such that 'AC' is the leftmost substring to which any of the 4 transformation rules can be applied and 'AC' is replaced in S by 'E' to produce S';
- Result(S,R2,S') for any state S and S' is such that 'AB' is the leftmost substring to which any of the 4 transformation rules can be applied and 'AB' is replaced in S by 'BC' to produce S'
- Result(S,R3,S') for any state S and S' is such that 'BB' is the leftmost substring to which any of the 4 transformation rules can be applied and 'BB' is replaced in S by 'E' to produce S';
- Result(S,R4,S') for any state S and S' is such that 'Ex', x being any of the letters 'A', 'B', 'C', 'E' is the leftmost substring to which any of the 4 transformation rules can be applied and 'Ex' is replaced in S by 'x' to produce S'}

1 mark if a reasonable representation of the model actions is given

Path cost function: number of transformations (i.e. applications of any of the 4 rules) from initial state to the current state. **0.25 mark**

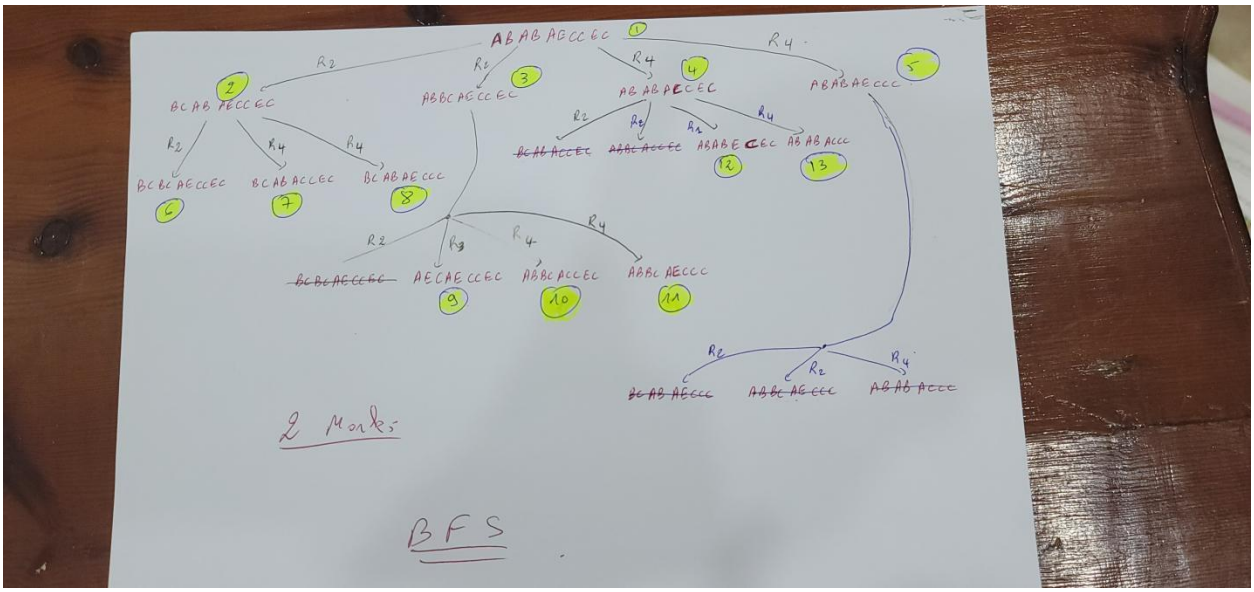
2) **Heuristic function**: **1 mark**

a. the number of pairs of letters to which no rule is applicable + (the number of letters in the current state – 1) OR (the following, acceptable but not as precise)

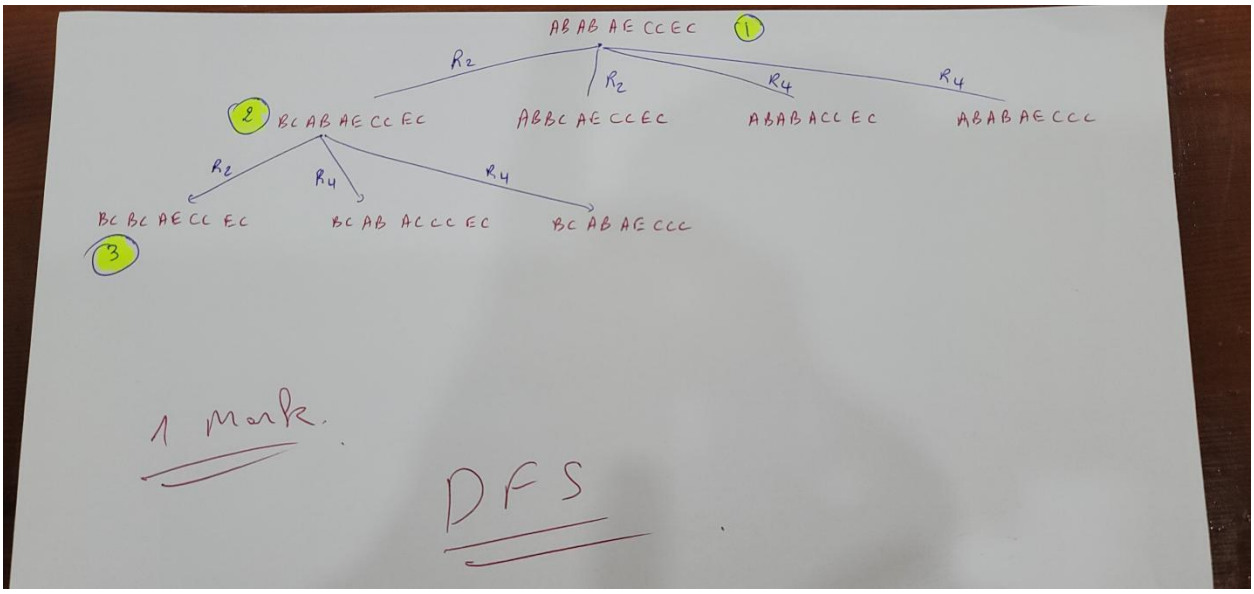
b. the number of mismatched characters between the current state and the goal state + (the number of letters in the current state – 1).

3)

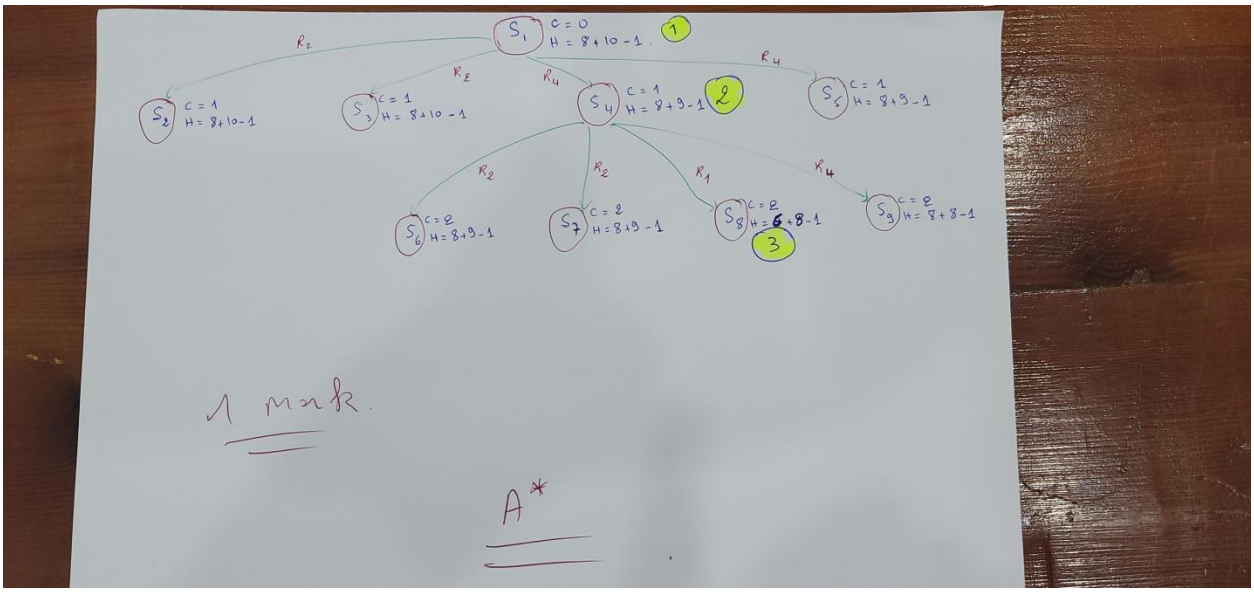
a) **BFS** **2 marks** if the below tree is developed.



b) **DFS** **1 marks** if the below tree is developed.



b) **A*** **1 mark**



4) **1 mark**

Two new rules should be added to avoid failure getting stuck when expanding nodes precisely because of 'BS' and 'CC' not being transformable. So, R_5: 'BC': 'E' and R_6: 'CC': 'E'

Exercise 4 (30 minutes and on 5 points see the correction for details): **6 marks**

Two monkeys, Momo and Coco, are trapped in a 2D square jungle with n rows and n columns (see Figure 1 below). Momo starts at (1, 1) and Coco starts at (n,1). The goal is that one of them reaches the exit which is at square (n, n). On each step (or iteration) the two monkeys move **simultaneously** by one square. Each one of them can move either up, down, left, or right (for example the first step can be (Momo **Up**, Coco **Left**)). However, they can't occupy the same square.

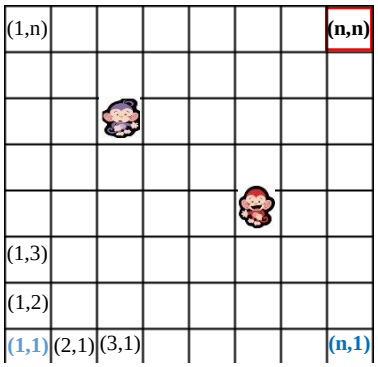


Figure 1: A Random State in the 2D Grid of the Monkey Problem

1. With a brief explanation, give an upper bound on the total size of the state space defined by your formulation as a function of *n*.
2. Calculate the branching factor for this problem. Explain briefly.
3. What is the maximum number of nodes expanded by breadth-first tree search at depth k as a function of k? Explain briefly.
4. What is the maximum number of nodes expanded by breadth-first tree search for the whole search tree at depth k? Explain briefly.
5. Suppose that, for each monkey i at position (x_i, y_i), the heuristic h_i gives the number of moves (steps) required to get to the goal location (n, n) as follows: $\mathbf{h_i\ (x_i, y_i)=|n- x_i| + |n- y_i|}$.

Say if each of the following heuristics is admissible and provide a brief explanation for your answer:

a. h_1+h_2 steps
b. max (h_1, h_2) steps
c. min (h_1, h_2) steps

Solution:

1. Give an upper bound on the total size of the state space defined by your formulation as a function of *n (1 point).*
We only need to track the positions of Momo and Coco (2 variables). Each can be in any of the n x n squares, resulting in a state space of n^2 * n^2 = n^4.

2. Calculate the branching factor (1 point).
The branching factor is 4*4 = 16 (up, down, left, right for each monkey).

3. What is the maximum number of nodes expanded by breadth-first tree search at depth k as a function of k (1 point)?
at depth 0 ➔ 1 == (16)^0
at depth 1 ➔ (16)^1
at depth 2 ➔ (16)^2
at depth 3 ➔ (16)^3
….
at depth k ➔ (16)^k

4. What is the maximum number of nodes expanded by breadth-first tree search for the whole search tree at depth k (1.5 points)?
The sum of the k+1 terms (from depth 0 to depth k) which are (16)^0, (16)^1, (16)^2, (16)^k form a geometric series with first term a=1 and a common ratio r = 16:

$$S = a * \frac{r^d - 1}{r - 1} = 1 * \frac{16^{k+1} - 1}{16 - 1} = \frac{16^{k+1} - 1}{15}$$

5. Suppose that for each monkey i at position (x_i, y_i) the heuristic h_i gives the number of moves (steps) required to get to the goal location (n, n) as follow: $\mathbf{h_i = |n - x_i| + |n - y_i|}$.
Assess the admissibility of each heuristic below and provide brief explanations:

a. h1+h2 steps (0.5 point for correct answer and explanation)
b. max (h1, h2) steps (0.5 point for correct answer and explanation)
c. min (h1, h2) steps (0.5 point for correct answer and explanation)

➔Here one thing that needs to be noticed is that each step in our problem formulation is a **simultaneous move** by both monkeys (meaning in a single step both monkeys will move by one square). Thus, the optimal number of steps required to reach the goal target is **exactly n steps** which is **min (h1, h2)**. So only c is admissible. The other two are an overestimate.