CSPB 3202 - Truong - Artificial Intelligence

Started on	Saturday, 22 June 2024, 7:00 PM
State	Finished
Completed on	Saturday, 22 June 2024, 8:19 PM
Time taken	1 hour 18 mins
Grade	Not yet graded

Question **1**

Correct

Mark 5.00 out of 5.00

First consider general search problems, which of the following are true?

Select one or more:

- [x] a. A heuristic that always evaluates to h(s) = 1 for non-goal search nodes s is always admissible. ✖
- [ ] b. A* graph search is guaranteed to expand no more nodes than DFS.
- [x] c. The max of two admissible heuristics is always admissible. ✔
- [ ] d. A* tree search is optimal with any heuristic function.
- [ ] e. A* tree search is complete with any heuristic function
- [x] f. Code that implements A* tree search can be used to run UCS ✔

Your answer is correct.

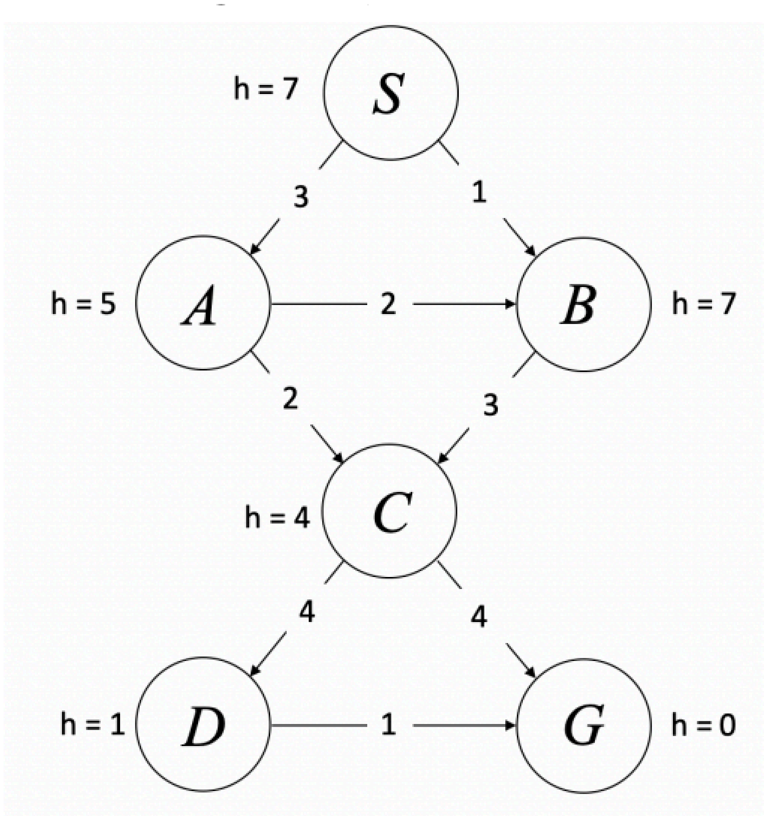
The correct answers are: Code that implements A* tree search can be used to run UCS, The max of two admissible heuristics is always admissible.

Question **2**

Partially correct

Mark 2.50 out of 5.00

We will investigate various search algorithms for the following graph. Edges are labeled with their costs, and heuristic values h for states are labeled next to the states. S is the start state, and G is the goal state. In all search algorithms, assume ties are broken in alphabetical order.



Given the above heuristics, what is the order that the states are going to be expanded in, assuming we run A* graph search with the heuristic values provided.

S	1	✔
A	2	✔
B	Not expanded	✖
C	3	✖
D	Not expanded	✔
G	4	✖

Your answer is partially correct.

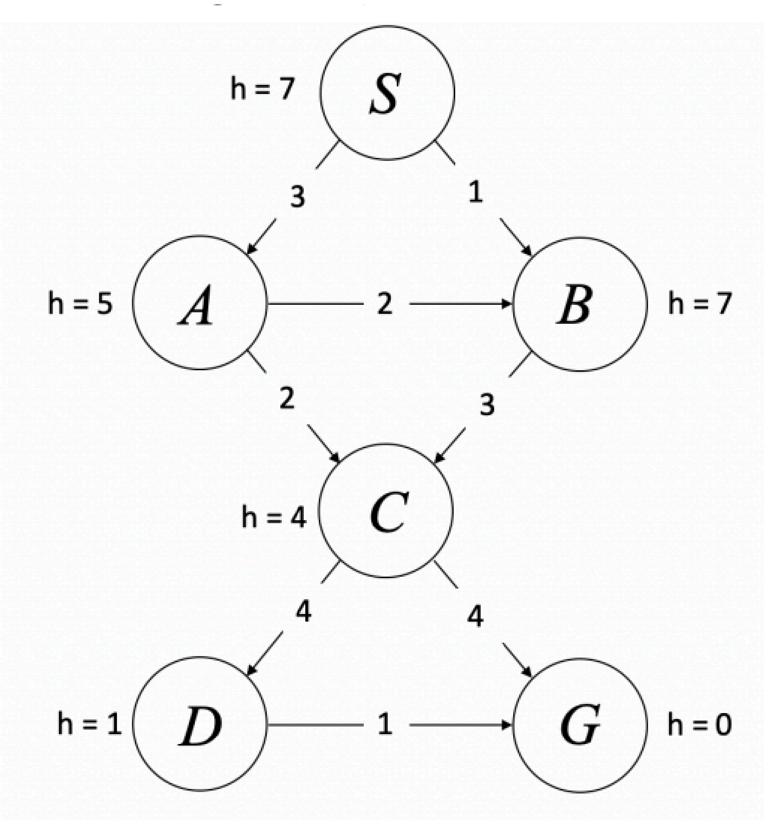
You have correctly selected 3.
The correct answer is: S → 1, A → 2, B → 3, C → 4, D → Not expanded, G → 5

Question **3**

Partially correct

Mark 2.00 out of 5.00

We will investigate various search algorithms for the following graph. Edges are labeled with their costs, and heuristic values h for states are labeled next to the states. S is the start state, and G is the goal state. In all search algorithms, assume ties are broken in alphabetical order.



Assuming we run A* graph search with the heuristic values provided, what path is returned?

Select one:

- S → A → B → C → D → G
- S → A → C → G (selected) — Tie is broken alphabetical order (as mentioned in the question text), so the state A is visited before B, however, the state B is also visited later, and from A* graph search, SBCG is optimal. Remember that when the heuristic is both admissible and consistent, the A* graph search returns the optimal path.
- S → A → C → D → G
- S → B → C → G
- S → A → B → C → G
- None of the above

Your answer is partially correct.

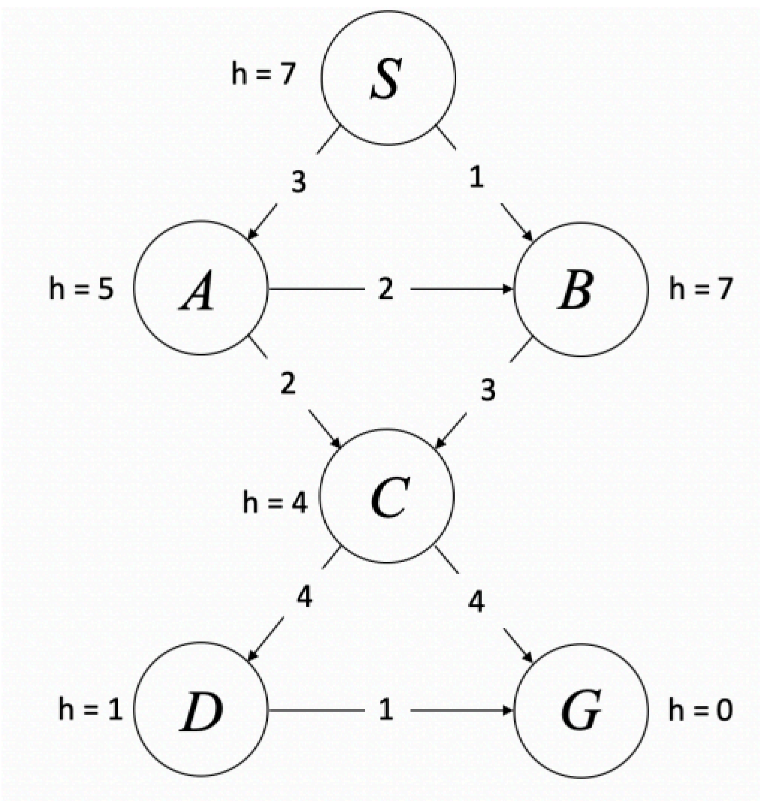
The correct answer is: S → B → C → G

Question **4**

Correct

Mark 10.00 out of 10.00

We will investigate various search algorithms for the following graph. Edges are labeled with their costs, and heuristic values h for states are labeled next to the states. S is the start state, and G is the goal state. In all search algorithms, assume ties are broken in alphabetical order.



Given the above heuristics, what is the order that the states are going to be expanded in, assuming we run greedy graph search with the heuristic values provided.

S	1
A	2
B	Not expanded
C	3
D	Not expanded
G	4

Your answer is correct.

The correct answer is: S → 1, A → 2, B → Not expanded, C → 3, D → Not expanded, G → 4

Question **5**

Correct

Mark 5.00 out of 5.00

Given two admissible heuristics h_A and h_B, which of the following are guaranteed to also be admissible heuristics? (Select all that apply)

- [ ] $h_A + h_B$
- [x] $\frac{1}{2} h_A$ ✔
- [x] $\frac{1}{2} h_B$ ✔
- [x] $\frac{1}{2}(h_A + h_B)$ ✔
- [ ] $h_A \times h_B$
- [x] max(h_A,h_B) ✔
- [x] min(h_A,h_B) ✔

Your answer is correct.

Question 6

Partially correct

Mark 2.50 out of 5.00

Consider performing tree search for some search graph. Let depth(n) be the depth of search node n and cost(n) be the total cost from the start state to node n. Let Gd be a goal node with minimum depth, and Gc be a goal node with minimum total cost.

For iterative deepening (where we repeatedly run DFS and increase the maximum depth allowed by 1), mark all conditions that are guaranteed to be true for every node n that could be expanded during the search, or mark "None of the above" if none of the conditions are guaranteed.

- [ ] cost(n) ≤ cost(Gc)
- [ ] cost(n) ≤ cost(Gd)
- [ ] depth(n) ≤ depth(Gc)
- [x] depth(n) ≤ depth(Gd) ✔
- [ ] None of above

Your answer is partially correct.

1 of your answers is correct.

When running iterative deepening we will explore all nodes of depth k, before we explore any nodes of depth k + 1. As a result will never explore any nodes that have depth greater than Gd because we would stop exploring once we reached Gd. This also means we would never explore any nodes with depth greater than Gc because Gc has to have depth greater than or equal to the minimum depth goal, Gd

Question **7**

Partially correct

Mark 5.00 out of 10.00

We have five planes: A, B, C, D, and E and two runways: international and domestic. We would like to schedule a time slot and runway for each aircraft to either land or take off. We have four time slots: {1, 2, 3, 4} for each runway, during which we can schedule a landing or take off of a plane. We must find an assignment that meets the following constraints:

• Plane B has lost an engine and must land in time slot 1. B=1 ✔

• Plane D can only arrive at the airport to land during or after time slot 3. D>=3 ✔

• Plane A is running low on fuel but can last until at most time slot 2. A<=2 ✔

• Plane D must land before plane C takes off, because some passengers must transfer from D to C. D<C ✔

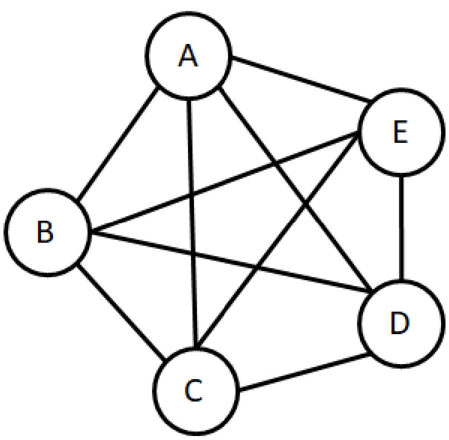
• No two aircrafts can reserve the same time slot for the same runway. A!=B!=C!=D!=E ✔

(a). Fill the boxes above with constraint expression. For example, plane X lands before Y will be expressed as X<Y (do not put any space, otherwide the grading will not work.), where Y can be letter if it's airplane or a number if it's a time slot. Allowed (in)equality symbols are: <, >, <=, >=, !=, =. Another example that X=Y, Z!=X is X=Y!=Z.

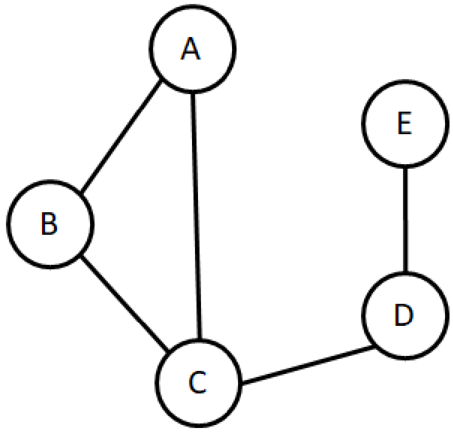
(b). Now, we add the following two constraints:
• Planes A, B, and C cater to international flights and can only use the international runway.
• Planes D and E cater to domestic flights and can only use the domestic runway.

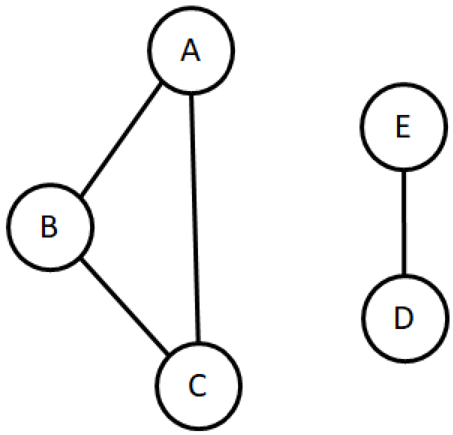
With the addition of the two constraints above, we completely reformulate the CSP. You are given the variables and domains of the new formulation. Complete the constraint graph for this problem given the original constraints and the two added ones.

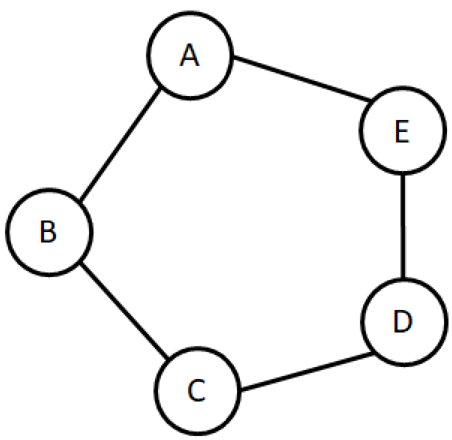
☐

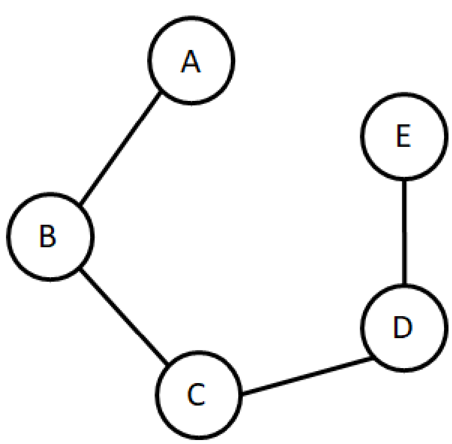


☑


A
B
C
D
E


A
B
C
D
E


A
B
C
D
E


A
B
C
D
E

(c) What are the domains of the variables after enforcing arc-consistency? Begin by enforcing unary constraints. (Check the box which values are no longer in the domain.)

A ☐1 ☑2 ✖ ☐3 ☐4

B ☑1 ✖ ☐2 ☐3 ☐4

C ☐1 ☐2 ☐3 ☑4 ✖

D ☐1 ☐2 ☑3 ✖ ☐4

E ☑1 ✖ ☑2 ✖ ☐3 ☑4 ✖

Your answer is partially correct.

6 of your answers are correct.

Question **8**

Correct

Mark 5.00 out of 5.00

What is the maximum number of times a backtracking search algorithm might have to backtrack in a general CSP, if it is running arc consistency and applying the MRV and LCV heuristics?

Select one:

- a. 0
- b. $O(1)$
- c. $O(nd^2)$
- d. $O(n^2d^3)$
- e. $O(d^n)$ ✔
- f. ∞

Your answer is correct.

$O(d^n)$. The MRV and LCV heuristics are often helpful to guide the search, but are not guaranteed to reduce backtracking in the worst case.

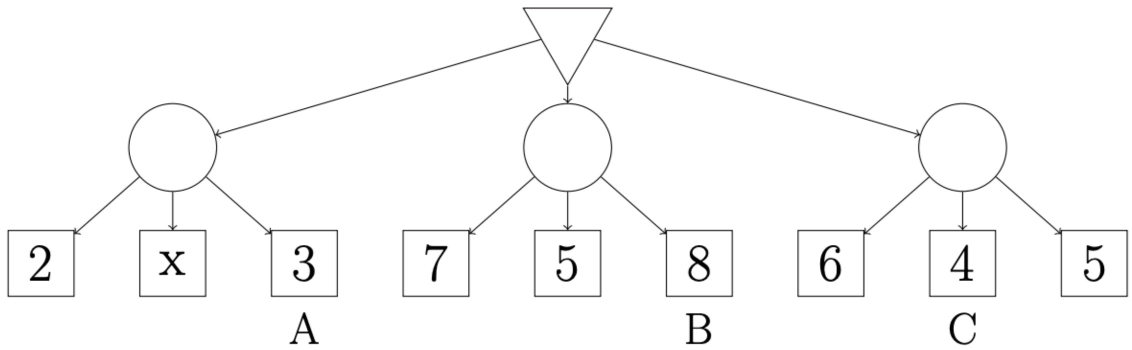
The correct answer is: $O(d^n)$

Question 9

Partially correct

Mark 2.00 out of 6.00

For each of the leaves labeled A, B, and C in the tree below, determine which values of x will cause the leaf to be pruned, given the information that all values are non-negative and all nodes have 3 children. Assume we do not prune on equality.



Check and fill the right answer. For the inequality item, choose "none" if no value of X will cause pruning, or "any" if the node will be pruned for all values of X. If you choose "none" or "any" for the inequality, fill 0 for the numeric box.

For A to be pruned: X needs to be ☐smaller ☐larger ☑none ✔ ☐any than

0

✔

For B to be pruned: X needs to be ☐smaller ☐larger ☑none ✖ ☐any than

0

✖

For C to be pruned: X needs to be ☐smaller ☐larger ☑none ✖ ☐any than

0

✖

Your answer is partially correct.

2 of your answers are correct.

To prune B, the value of the first expectation node must be less than the value of the second even it B were 0

To prune B, the value of the first expectation node must be less than the value of the second even it B were 0

(2+x+3)/3<(7+5)/3 ---> x<7

To prune C, the value of the first expectation node must have value less than the value of the third if both of the last two leaves had value 0

To prune C, the value of the first expectation node must have value less than the value of the third if both of the last two leaves had value 0

(2+x+3)/3<6/3 ---> x<1

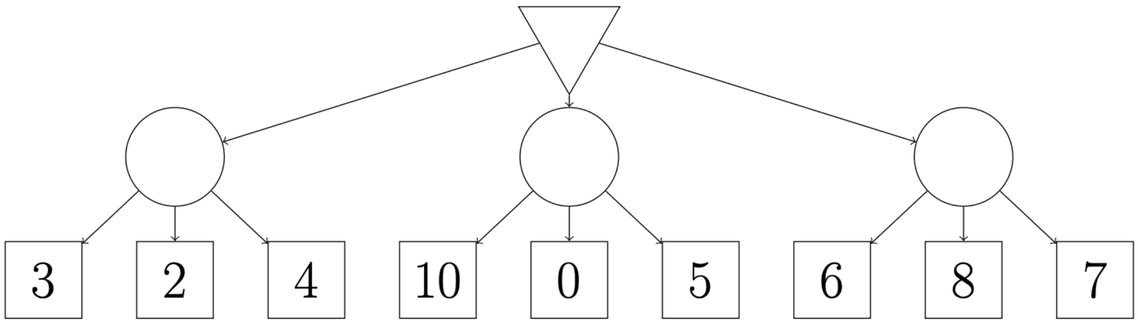
Question **10**

Correct

Mark 10.00 out of 10.00

In this problem we model a game with a minimizing player and a random player. We call this combination “expectimin” to contrast it with expectimax with a maximizing and random player. Assume all children of expectation nodes have equal probability and sibling nodes are visited left to right for all parts of this question.

Fill out the “expectimin” tree below.



value of the triangle: 3

value of the middle circle: 5

value of the right circle: 7

value of the left circle: 3

Your answer is correct.

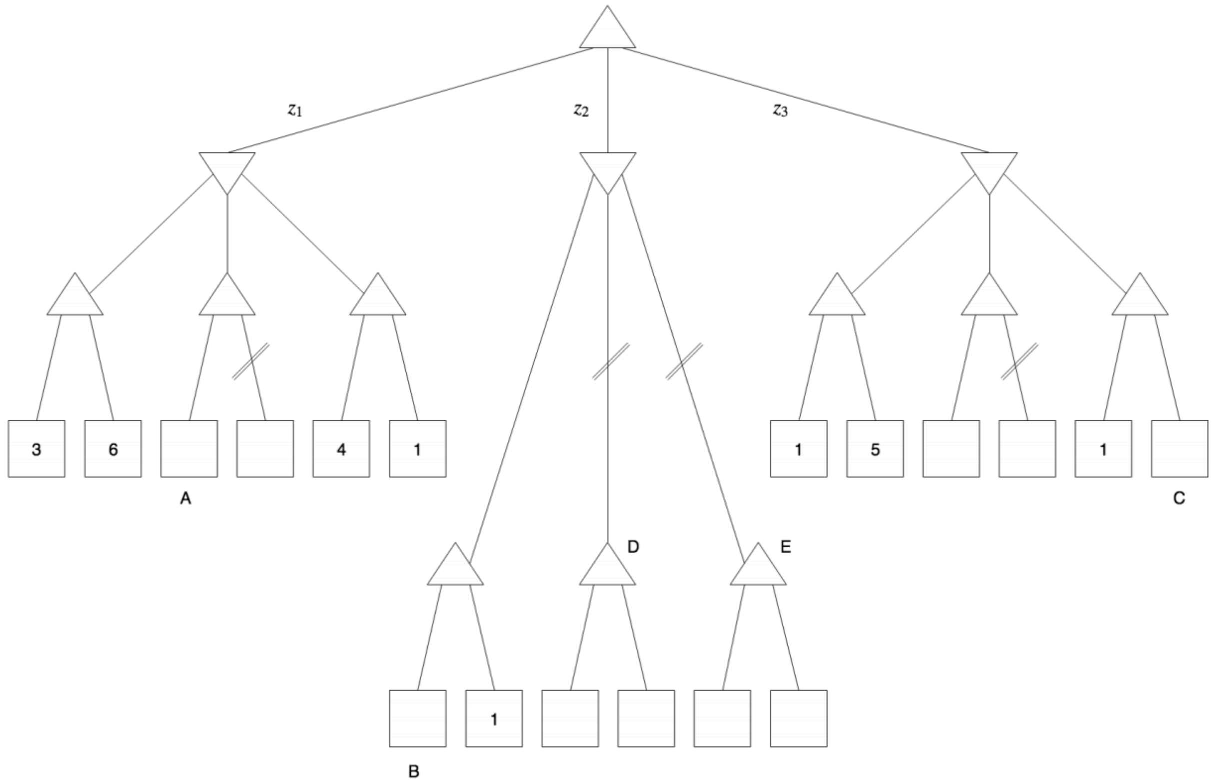
The correct answer is: value of the triangle: → 3, value of the middle circle: → 5, value of the right circle: → 7, value of the left circle: → 3

Question **11**

Complete

Marked out of 6.00

Consider the minimax tree below. whose leaves represent payoffs for the maximizer. The crossed out edges show the edges that are pruned when doing naive alpha-beta pruning visiting children nodes from left to right. Assume that we prune on equalities (as in, we prune the rest of the children if the current child is ≤ α (if the parent is a minimizer) or ≥ β (if the parent is a maximizer)).



Please give the range of A and B.(Write ∞ and −∞ if there is no upper or lower bound, respectively.)

We know that in the leftmost subtree, the left maximizer is going to pick 6. For the middle maximizer, in order for the right child of this maximizer to be pruned and for the children of the rightmost maximizer to not be pruned, we constitute that node **A** must be greater than the left node of the right most maximizer. In mathematical terms, this means node **A** must be:

$$A = [5, \infty]$$

The leftmost minimizer is going to pick three from its available options. So, $z_1 = 3$.

In the middle subtree, the two rightmost branches are pruned. Since parents of the children are maximizers, we once again are constituting that node **B** is greater than or equal to the remaining children of this subtree. Taking into account that we prune on equalities, if the subsequent value of z_2 were to be the same as z_1, then all of its children would be pruned. To avoid this, we need node **B** to be essentially at least one less than that of z_1. So, for node **B** this means it must be:

$$B = [-\infty, 2]$$

6 <=A <= inf

-inf <= B <=4

Question **12**

Correct

Mark 3.00 out of 3.00

Alpha-beta pruning will prune the same branches after a monotonically increasing function is applied to the value of the leaves.

Select one:

- (selected) True ✔
- False

correct

The correct answer is 'True'.

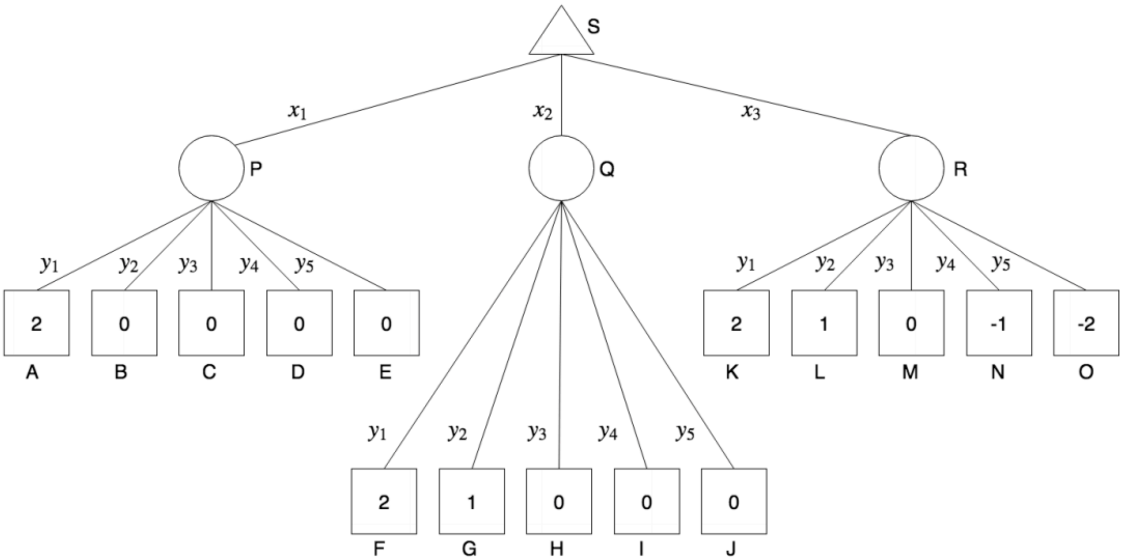
Question **13**

Incorrect

Mark 0.00 out of 5.00

Alyssa P. Hacker and Ben Bitdiddle are bidding in an auction for a bike. Alyssa will either bid x1, x2, or x3 for the bike. She knows that Ben will bid y1, y2, y3, y4, or y5, but she does not know which. All bids are nonnegative.

Alyssa wants to maximize her payoff given by the expectimax tree below. The leaf nodes show Alyssa's payoff. The nodes are labeled by letters, and the edges are labeled by the bid values xi and yi . The maximization node S represents Alyssa, and the branches below it represent each of her bids: x1, x2, x3. The chance nodes P, Q, R represent Ben, and the branches below them represent each of his bids: y1, y2, y3, y4, y5.



Alyssa does expectimax search by visiting child nodes from left to right. Ordinarily expectimax trees cannot be pruned without some additional information about the tree. Suppose, however, that Alyssa knows that the leaf nodes are ordered such that payoffs are non-increasing from left to right (the leaf nodes of the above diagram is an example of this ordering). Recall that if node X is a child of a maximizer node, a child of node X may be pruned if we know that the value of node X will never be > some threshold (in other words, it is ≤ that threshold). Given this information, if it is possible to prune any branches from the tree, mark them below. Otherwise, mark "None of the above."

Select one or more:

- [ ] A
- [ ] B
- [ ] C
- [ ] D
- [ ] E
- [ ] F
- [ ] G
- [ ] H
- [ ] I
- [ ] J
- [ ] K

- [ ] L
- [ ] M
- [ ] N
- [ ] O
- [x] None of the above ✖

Your answer is incorrect.

The correct answers are: N, O