Question #: 1

[Question Group: Search Formulation]
Spaceship Transport
Captain Picard is leading a mission to transport n groups of people from different home planets [g_1,...,g_n] using m spaceships [s_1,...,s_m]. Here, g_i denotes the number of people in the i-th group, and s_j denotes the capacity of the j-th spaceship. Each group of people cannot be broken apart and must be in the same ship. Picard needs to assign each group to a ship and ensure that all groups are assigned to spaceships. At one time, Picard can only allow one group to enter or exit one spaceship. Since the spaceships will depart tomorrow, Picard must come up with an efficient plan, i.e., a plan with the smallest number of actions.

Suppose that you use the following state representation:
- $[z_1,...,z_{m+1}]$ where z_i is the set of groups that are assigned to the spaceship i for $1 \leq i \leq m$. z_{m+1} is the set of unassigned groups.

Consider the following invariants, initial states, goal tests, and actions:
Invariants:
- I_1: The number of people in each spaceship must not exceed its capacity.

I_1: $\forall_{i \in [1,...,m]} \sum_{j \in z\_i} g_j \leq s_i$, note: z_i =z_i

- I_2: The union of all groups in all spaceships and unassigned groups must be the same as $\{1, ..., n\}$.

I_2: $\cup_{i \in [1,...,m+1]} z_i = \{1, ..., n\}$, where ∪ is union operator, e.g., $\{1,2\} \cup \{2,3\} = \{1,2,3\}$

- I_3: Total number of people in all spaceships must be the same as the total capacity of all spaceships

I_3: $\sum_{x \in w} g_x = \sum_{i \in [1,...,m]} s_i$, where $w = \bigoplus_{i \in [1,...,m]} z_i$ and $\bigoplus$ is concatenation operator, e.g., $[1,2] \bigoplus [2,3] = [1,2,2,3]$

Initial States:
- S_1: All spaceships are empty and all groups are unassigned

S_1: $\forall_{i \in [1,...,m]} z_i = []$, $z_{m+1} = [1,...,n]$

- S_2: Randomly assign each group to a spaceship or let it unassigned

S_2: randomly assign $[1,...,n]$ to $z_1, \ldots, z_{m+1}$ without replacement

Goal test:
- G_1: The union of all groups in all spaceships must be the same as $\{1, ..., n\}$.

G_1: $\cup_{i \in [1,...,m]} z_i = \{1, ..., n\}$

- G_2: No unassigned groups

G_2: $z_{m+1} = []$

Actions:
- A_1: Move each $x \in z_{m+1}$ to each z_i where $i \leq m$
- A_2: Move each $x \in z_i$ to z_{m+1} for each $i \leq m$

- A_3: Swap each $x \in z_i$ with each $y \in z_j$ for any (random) i,j

Which of the above invariant, initial state, goal test, and action tuple(s) is/are reasonable?

A. $I_1\&I_2 - S_1 - G_1 - A_1\&A_2\&A_3$
✓B. $I_1\&I_2 - S_1 - G_1\&G_2 - A_1$
C. $I_1\&I_2 - S_2 - G_1\&G_2 - A_3$
D. $I_1\&I_3 - S_1 - G_1\&G_2 - A_1\&A_2\&A_3$
E. $I_1\&I_3 - S_2 - G_1 - A_1\&A_2$
F. $I_1\&I_3 - S_2 - G_1\&G_2 - A_1\&A_2\&A_3$
G. $I_1\&I_2\&I_3 - S_1 - G_2 - A_2\&A_3$
H. $I_1\&I_2\&I_3 - S_1 - G_1\&G_2 - A_1\&A_2$
I. $I_1\&I_2\&I_3 - S_2 - G_1\&G_2 - A_1\&A_2\&A_3$
J. None of the above

Notes:

- Any answers that contain A_3 is incorrect because swapping action is not possible based on the description i.e., “Picard can only allow one group to enter or exit one spaceship”.
- Any answers that contain I_3 is incorrect because it is not necessary that the number of people in all spaceships equals the combined capacity of all spaceships.

Item Weight: 4.0

Question #: 2

[Question Group: Uninformed Search]
Spaceship Transport
Captain Picard is leading a mission to transport n groups of people from different home planets $[g_1,...,g_n]$ using m spaceships $[s_1,...,s_m]$. Here, g_i denotes the number of people in the i-th group, and s_j denotes the capacity of the j-th spaceship. Each group of people cannot be broken apart and must be in the same ship. Picard needs to assign each group to a ship and ensure that all groups are assigned to spaceships. At one time, Picard can only allow one group to enter or exit one spaceship. Since the spaceships will depart tomorrow, Picard must come up with an efficient plan, i.e., a plan with the smallest number of actions.
Suppose that we formulate the above problem as a search problem as follows.
State representation: $[x_1,...,x_n]$ where $x_i \in [0,...,m]$ represents the assignment of group i to a spaceship among the m spaceships, 0 means not assigned.
Invariant:

- $x_i \in [0,...,m]$
- Total number of people in each spaceship is less than the capacity

Initial state: [0,...,0], all zero (unassigned)
Goal test: $[x_1,...,x_n]$ where $x_i \in [1,...,m]$ (all assigned)
Actions: Set each x_i to 0, 1, ..., or m. i.e., n*(m+1) actions

---

Is it true that the search formulation results in a state space where a state can be **visited multiple times**? Note: we do not care about the search algorithms in this question since we are asking about state space, not search tree/graph.

✓A. True
B. False

Notes:
Based on the action definition, x_i can be set to any value in [0, …, m]. Thus, we can revisit any state multiple times.

Item Weight: 1.0

Question #: 3

[Question Group: Uninformed Search]

Is it true that the search formulation results in a state space with **many goal states?**

✓A. True
B. False

Notes:
It is possible that the solution is not unique. For instance, suppose that we have 2 spaceships with the same capacity of 10 and two group of people (10 people each). Then, the goal states are [1,2] and [2,1].

Item Weight: 1.0

Question #: 4

[Question Group: Uninformed Search]

Which of the following tree search algorithm(s) can we employ such that the search tree is **finite** (if there is a solution)?

✓A. Breadth-first search (BFS)
B. Depth-first search (DFS)
✓C. Uniform-cost search (UCS) with constant cost
✓D. Depth-limited search with BFS
✓E. Depth-limited search with DFS
✓F. Iterative deepening search (IDS)
G. None of the above

Notes:

- B is incorrect because the action is reversible and DFS can get stuck in an infinite loop by going back and forth between states, e.g., set x_i to 0, then 1, then 0, etc.
- UCS with constant cost is the same as BFS.
- Deph-limited search will always terminate and thus is finite
- BFS and IDS behave in a similar way
- BFS, IDS, and UCS will explore a finite number of states if there is a solution since the branching factor is finite

Item Weight: 2.0

Question #: 5

[Question Group: Uninformed Search]

Which of the following tree search algorithm(s) can we employ such that the search **always terminates** (including if there is no solution)?

A. Breadth-first search (BFS)
B. Depth-first search (DFS)
C. Uniform-cost search (UCS) with constant cost
✓D. Depth-limited search with BFS
✓E. Depth-limited search with DFS
F. Iterative deepening search (IDS)
G. None of the above

Notes:

- DFS may get stuck in an infinite loop as explained previously.
- BFS, UCS, and IDS may not terminate if there is no solution, i.e., they will keep on searching and revisiting previously visited states.
- Depth-limited search will always terminate since the depth is limited and the branching factor is finite.

Update:

- IDS definition in the lecture is ambiguous. It can be interpreted as running DLS from 0 depth up to an infinite depth or from 0 depth until a finite depth. To clarify, the usual implementation of IDS uses no max depth limit, so it runs DLS from 0 depth to an infinite depth.
- Given the above, we'll allow both interpretations:
  1. Under the assumption that IDS has a finite depth: IDS and DLSes (D, E and F)
  2. Otherwise: DLSes (D and E)

Item Weight: 2.0

Question #: 6

[Question Group: Uninformed Search]

Which of the following tree search algorithm(s) can we employ such that the search **always finds an answer** (valid sequence of moves) if a solution exists?

✓A. Breadth-first search (BFS)
B. Depth-first search (DFS)
✓C. Uniform-cost search (UCS) with constant cost
D. Depth-limited search with BFS
E. Depth-limited search with DFS
✓F. Iterative deepening search (IDS)
G. None of the above

Notes:

- DFS may get stuck in an infinite loop as explained previously.
- Depth-limited search may not find an answer if the depth is not set correctly.

Update:

See Q5 explanation about IDS and we'll allow the following interpretations:

1. Assume correct max depth: DLSs, BFS, UCS, IDS (A, C, D, E and F)
2. Assume: max depth can be wrong + IDS has finite depth limit: BFS, UCS (A and C)

3. Otherwise: BFS, UCS, IDS (A, C and F)

.

Item Weight: 2.0

Question #: 7

[Question Group: Uninformed Search]

Which of the following **tree search** algorithm(s) is/are the **best** for the problem? Best means the algorithm(s) should be complete, optimal, efficient, and aware if there is no solution.

A. Breadth-first search (BFS)
B. Depth-first search (DFS)
C. Uniform-cost search (UCS) with constant cost
D. Depth-limited search with BFS
E. Depth-limited search with DFS
F. Iterative deepening search (IDS)
✓G. None of the above

Notes:
None of the algorithms satisfy the definition of “Best” given in the question.
Update:
Using the state representation given in the question, we can actually bound the depth of the search. Since we didn't explicitly state the assumption regarding whether DLS uses the correct depth or any depth, we'll allow both interpretations.

1. Assume: correct max depth: DLS with DFS (E)
2. Assume: max depth can be wrong: None of the above (G)

Item Weight: 2.0

Question #: 8

[Question Group: Uninformed Search]

Which of the following **graph search** algorithm(s) is/are the **best** for the problem? Best means the algorithm(s) should be complete, optimal, efficient, and aware if there is no solution.

✓A. Breadth-first search (BFS)
B. Depth-first search (DFS)
✓C. Uniform-cost search (UCS) with constant cost
D. Depth-limited search with BFS
E. Depth-limited search with DFS
✓F. Iterative deepening search (IDS)
G. None of the above

Notes:
The three algorithms above satisfy the definition. Efficient here could refer to time efficiency and space efficiency. Since there is a trade-off between the two, BFS and UCS with constant cost and IDS are correct, the former two have better time-complexity (no overhead) while the latter has better space complexity.
Update:
See explanation about DLS in Q7
In the question, we didn't specify explicitly regarding the consideration of space and time efficiency. Thus, we will allow the following:

1. Assume: correct max depth: DLS with DFS (E)
2. Assume: max depth can be wrong + IDS has depth limit: BFS, UCS (A and C)
3. Assume: max depth can be wrong + IDS no depth limit + efficiency considering big-O notation: IDS (F)
4. Assume: max depth can be wrong + IDS no depth limit + efficiency counting the overhead: IDS, BFS, or UCS (A, C and F)

Item Weight: 2.0

Question #: 9

[Question Group: Local Search]
Spaceship Transport: Variant #1
Captain Picard is leading a mission to transport *n* groups of people from different home planets $[g_1,...,g_n]$ using spaceships. Here, g_i denotes the number of people in the *i*-th group, and each spaceship can hold *k* people. Each group of people cannot be broken apart and must be in the same ship. Picard needs to assign each group to a ship and ensure that all groups are assigned to spaceships. There is no limit to the number of spaceships. In this case, Picard wants to ensure the use of the minimum number of spaceships.
Suppose that we formulate the above problem as a local search problem and use the following state representation: $[a_1,...,a_n]$ where $a_i \geq 0$ represents the assignment of group *i* to the a_i-th spaceship, a_i =0 means group *i* is not assigned yet.

Consider the following initial states and successor functions:
Initial states:

- S_1 - $[0,\ldots,0]$

- S_2 - a_i randomly sampled from $[1,\ldots,n]$

Successor functions:

- F_1 - Select a random group i, then set $a_i = 0$, $a_i = 1$, ..., or, $a_i = n$ (there will be $n+1$ neighbors)

- F_2 - Select a random group i where $a_i = 0$, then set $a_i = 1$, $a_i = 2$, ..., or, $a_i = n$ (there will be n neighbors)

- F_3 - Swap a_i with a_j for any i,j

- F_4 - Select a random group i where $a_i = 0$, then set $a_i = 1$, $a_i = 2$, ..., or, $a_i = n$ (there will be n neighbors) +swap a_i with a_j for any i,j, where $a_i > 0$ and $a_j > 0$

The successor function ensures that the number of people in the ai-th spaceship is less than *k*
Which of the above initial state and successor function pair(s) is/are reasonable?

✓A. $S_1 - F_1$
B. $S_1 - F_2$
C. $S_1 - F_3$
✓D. $S_1 - F_4$
✓E. $S_2 - F_1$
F. $S_2 - F_2$
✓G. $S_2 - F_3$
✓H. $S_2 - F_4$
I. None of the above

Notes:

- B is sampling a state through a greedy sequential random sampling
- C is simply swapping 0s
- F is random sampling one state

Update:

We didn't define what "+" means, but this one doesn't matter. We didn't properly define what reasonable actually means. Thus, we will allow two different interpretations of reasonability:

1. Reasonable: not just random sample a state: S1-F1, S1-F4, S2-F1, S2-F3, S2-F4 (A, D, E, G and H)
2. Reasonable: can explore all states: S1-F1 & S2-F1 (A and E)

Item Weight: 3.0

Question #: 10

[Question Group: Local Search]

Let

- *s* be the current state,
- *set*(x) returns a set of unique items in the list x,
- *len*(x) returns the number of items in a set/list x,
- *count*(x,i) returns the number of items in x that has the value i,
- *indices*(x,i) returns the indices of items in x that has the value i,
- $f_1(s) = n - count(s,0)$,
- $f_2(s) = len(set(s))$,
- $f_3(s,i) = \sum_{j \in indices(s,i)} g_j$. i.e., the total number of people in spaceship i if s is the state and i the index of the spaceship

Suppose that we are doing hill-climbing where we want to maximize an evaluation function. Regardless of the initial state and the successor function, which of the following evaluation function(s) is/are reasonable (i.e., maximizing it/them lead(s) us closer to the objective)?

A. $f(s) = f_1(s)$
B. $f(s) = 1/f_2(s)$
C. $f(s) = \min_i f_3(s,i)$
D. $f(s) = f_1(s) + \sum_i g_i * f_2(s)$
E. $f(s) = f_1(s) + \min_i f_3(s,i)$
F. $f(s) = \sum_i g_i / f_2(s) + \min_i f_3(s,i)$
✓ G. $f(s) = f_1(s) + \sum_i f_3(s,i) / f_2(s)$
H. None of the above

Notes:

- Considering only f_1 is incorrect because it could lead to a trivial solution where each group is assigned to one spaceship.
- Considering only $1/f_2$ is incorrect because it could lead to a trivial solution where no group is

assigned, giving a maximum value.

- $\min_i f_3(s,i)$ is always 0 since i is unbounded (i.e., there is no limit in the number of spaceships).
- f_2 is incorrect because it could lead to maximizing the number of spaceships.

Item Weight: 3.0

Question #: 11

[Question Group: Local Search]

Using the above state representation, which of the following uninformed search algorithm(s) can we use to find the optimal solution assuming that it exists?

✓A. Breadth-first search (BFS)
B. Depth-first search (DFS)
✓C. Uniform-cost search (UCS) with constant cost
D. DLS with BFS
E. DLS with DFS
✓F. Iterative deepening search (IDS)
G. None of the above

Update:
Here, only the state representation is fixed, and we can use any transition function. Let $[a_1, a_2, ..., a_n]$ be any state.
One possible transition function is: Pick any nonempty subset of indices such that $a_i = 0$ for every i in the subset and the sum of group sizes in the subset does not exceed k, and change a_i in the selected subset to $\max(a_1, a_2, ..., a_n) + 1$.

Let's suppose n = 4, $[g_1, g_2, g_3, g_4] = [1, 2, 3, 4]$, and k = 6.
Initial state: $[a_1, a_2, a_3, a_4] = [0, 0, 0, 0]$. The set of indices i such that $a_i = 0$ is {1, 2, 3, 4}.

The non-empty subsets of indices of {1, 2, 3, 4} such that the sum of the corresponding g_i for all i in the subset is no more than k = 6 are {1}, {2}, {3}, {4}, {1, 2}, {1, 3}, {1, 4}, {2, 3}, {2, 4}, {1, 2, 3}. So we can select any of the subsets and set a_i in the selected subset as 1 ($\max(a_1, a_2, a_3, a_4) + 1 = 1$). Therefore, the corresponding neighbors are (in the same order): [1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1], [1, 1, 0, 0], [1, 0, 1, 0], [1, 0, 0, 1], [0, 1, 1, 0], [0, 1, 0, 1], [1, 1, 1, 0]. This is considered as one step, and notice that exactly one spaceship is used.

Now, let's suppose we have selected subset {1, 3} and are currently at the state [1, 0, 1, 0]. The set of indices i such that $a_i = 0$ is {2, 4}. The non-empty subsets of indices of {2, 4} such that the sum of the corresponding g_i for all i in the subset is no more than k = 6 are {2}, {4}, {2, 4}. So we can select any of the subsets and set a_i in the selected subset as 2 (max(a_1, a_2, a_3, a_4) + 1 = 2). Therefore, the corresponding neighbors are (in the same order): [1, 2, 1, 0], [1, 0, 1, 2], [1, 2, 1, 2]. Notice that exactly two spaceships are used.

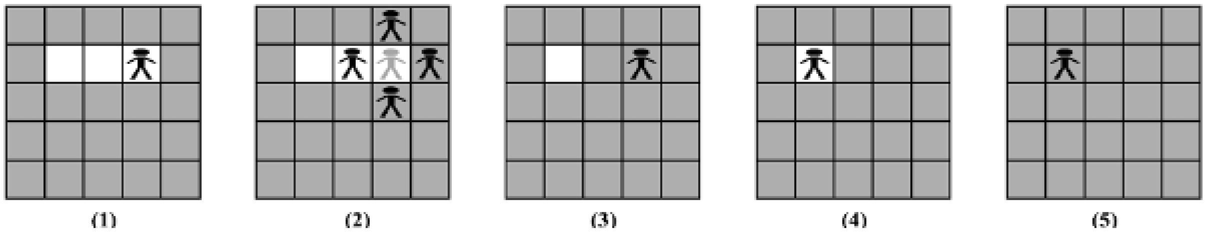
Basically, we consider all possible group assignments starting from spaceship 1, 2, ... until all groups are assigned. In this case, we can utilize BFS, UCS, or IDS (A, C, and F). The original answer G is not correct.

Reference:
https://coursemology.org/courses/2714/forums/general-discussion/topics/official-midterm-queries

Item Weight: 2

Question #: 12

[Question Group: Heuristics: Light]
Light
The first floor of the DBZ Bank building contains *n* x *n* rooms, each equipped with a light. Some of these lights are currently on. Your task is to **turn off all the lights** on the first floor. You are denoted as *p*, and your current position is represented by coordinates (x_p, y_p). The position of each room *r* can be represented by coordinates (x_r, y_r), where $r \in [0,...,n^2-1]$ is the index of the room. If you are in room *r* located at coordinates (x_r, y_r), then $x_p = x_r$ and $y_p = y_r$. Importantly, you must remain within the *n* x *n* rooms at all times. In each step, you are **only allowed** to take the following actions:
1.Move one room to the left/right/up/down with a cost of 1.
2.Stay in the current room and turn off the light in the room if it is on with a cost of 1.
3.Stay in the current room and turn on the light in the room if it is off with a cost of 1.
Please note that due to the special arrangement on the first floor, **turning off the light in room *r* will also turn off the light in the directly adjacent room to the left if it is currently on**, and it will remain off if it is currently off. This only applies when the position of the directly adjacent room to the left is within the *n* x *n* grid. Turning on the light in room *r* will not affect the lights in other rooms.
For instance, consider Figure (1), where your current position is (1, 3), and the positions of rooms with lights on are {(1, 1), (1, 2), (1, 3)}. You can now choose to move one room to the left/right/up/down, as shown in Figure (2). Alternatively, you can stay in the current room and turn off the light, as it is currently on. In this case, both lights in rooms (1, 2) and (1, 3) will be turned off due to the special arrangement on the first floor. To turn off all the lights, you can then move to room (1, 1) and turn off the light, as illustrated in Figures (4) and (5) respectively.



After discussing this problem with your cousin, Ben Bitdiddle, he proposed some heuristics to be used with A^* search. Your task is to evaluate each heuristic.
Hint: The Manhattan Distance is defined as $MD(a,b) = |x_a - x_b| + |y_a - y_b|$, where x_a and y_a represent the row and column index of a, and the same applies to b.

---

h_A: The number of rooms with lights on.
Is h_A admissible for this Light problem?

A. Yes, it is admissible.
✓B. No, it is not admissible.

Notes:
Consider a situation where your current position is (0, 2), and the rooms with lights on are (0, 1) and (0, 2). In this case, h_A = 2, which exceeds the real cost, h* = 1 (turn off the light in room (0, 2)). Thus, h_A is not admissible.

Item Weight: 2.0

Question #: 13

[Question Group: Heuristics: Light]

h_A: The number of rooms with lights on.
Is h_A consistent for this Light problem?

A. Yes, it is consistent.
✓B. No, it is not consistent.

Notes:

Consider a situation where your current position is (0, 2), and the rooms with lights on are (0, 1) and (0, 2). In this case, the next state (n') can be the goal state if we choose to turn off the light in room (0, 2). $h_A(n) = 2 > c(n, a, n') + h_A(n') = 1 + 0$. Thus, h_A is not consistent.

Item Weight: 2.0

Question #: 14

[Question Group: Heuristics: Light]

h_B: Divide the sum of all Manhattan distances between rooms with light on and your current position by 2.

$h_B = (\sum_{r \in R} MD(r,p))/2$.

Here, R contains the indices of all the rooms with lights on.

Is h_B admissible for this Light problem?

A. Yes, it is admissible.

✓B. No, it is not admissible.

Notes:

Consider a situation where your current position is (3, 3), and the rooms with lights on are {(1, 0), (1, 1), (2, 1), (2, 2)}. In this case, h_B = (5+4+3+2)/2=7, which exceeds the real cost, h* = 6 (1. Move from (3, 3) to (3, 2); 2. Move from (3, 2) to (2, 2); 3. Turn off the light in room (2, 2); 4. Move from (2, 2) to (2, 1); 5. Move from (2, 1) to (1, 1); 6. Turn off the light in room (1, 1)). Thus, h_B is not admissible.

Item Weight: 2.0

Question #: 15

[Question Group: Heuristics: Light]

h_B: Divide the sum of all Manhattan distances between rooms with light on and your current position by 2.

$h_B = (\sum_{r \in R} MD(r,p))/2$.

Here, R contains the indices of all the rooms with lights on.

Is h_B consistent for this Light problem?

A. Yes, it is consistent.

✓B. No, it is not consistent.

Notes:

Consider a situation where your current position is (3, 3), and the rooms with lights on are {(1, 0), (1, 1), (2, 1), (2, 2)}. In this case, h_B(n) = (5+4+3+2)/2=7. If you choose to move from (3, 3) to (3, 2) with a cost of 1, then h_B(n') = (4+3+2+1)/2=5. h_B(n) = 7 > c(n, a, n') + h_B(n') = 1 + 5. Thus, h_B is not consistent.

Item Weight: 2.0

Question #: 16

[Question Group: Heuristics: Light]

h_C: The maximum Manhattan distance between rooms with lights on and your current position.

$h_C = \max_{r \in R} MD(r,p)$.

Here, *R* contains the indices of all the rooms with lights on.

Is h_C admissible for this Light problem?

✓A. Yes, it is admissible.

B. No, it is not admissible.

Notes:

To turn off all the lights, we must turn off the light in the farthest room, r_f. There are two cases for turning off the light in the farthest room:

In case 1, if the light in the directly adjacent room to the right of r_f is off. You can move to room r_f and choose to turn off the light. The actual cost to turn off the light in the farthest room is h_C + 1.

In case 2, if the light in the directly adjacent room to the right of r_f is also on, you can move to that adjacent room and choose to turn off its light. The actual cost to turn off the light in the farthest room then becomes (h_C – 1) + 1 = h_C.

Thus, the actual cost to turn off the light in the farthest room is at least h_C, and h_C is admissible.

Item Weight: 2.0

Question #: 17

[Question Group: Heuristics: Light]

h_C: The maximum Manhattan distance between rooms with lights on and your current position.

$h_C = \max_{r \in R} MD(r,p)$.

Here, R contains the indices of all the rooms with lights on.
Is h_C consistent for this Light problem?

✓A. Yes, it is consistent.
B. No, it is not consistent.

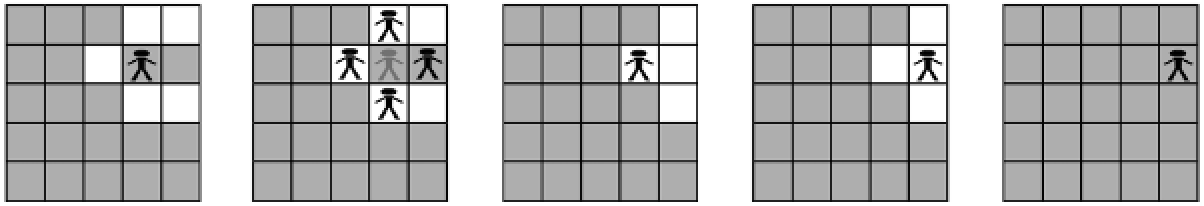
Notes:
Considering the available actions in the Light problem, h_C can be reduced by at most 1 in each step, i.e., h_C (n) - h_C (n') ≤ 1 = c(n, a, n'). Therefore, h_C (n) ≤ h_C (n') + c(n, a, n'), and h_C is consistent.

Item Weight: 2.0

Question #: 18

[Question Group: Heuristics: Light Variant #1]
Light Variant #1
The second floor of the DBZ Bank building contains n x n rooms, each equipped with a light. Some of these lights are currently on. Your task is to **turn off all the lights** on the second floor. You are denoted as p, and your current position is represented by coordinates (x_p, y_p). The position of each room r can be represented by coordinates (x_r, y_r), where $r \in [0,...,n^2-1]$ is the index of the room. If you are in room r located at coordinates (x_r, y_r), then $x_p = x_r$ and $y_p = y_r$. Importantly, you must remain within the n x n rooms at all times. In each step, you are **only allowed** to take the following actions:
1.Move one room to the left/right/up/down with a cost of 1.
2.Stay in the current room and turn off the light in the room if it is on with a cost of 1.
3.Stay in the current room and turn on the light in the room if it is off with a cost of 1.
Please note that due to a special arrangement on the second floor, **changing the status of the light in room *r* will toggle the status of lights in neighboring rooms**. Specifically, if you are now in room r, and you switch the light in room r from off to on or from on to off, the lights in rooms at positions ($x_r - 1$, y_r), (x_r, $y_r - 1$), ($x_r + 1$, y_r), and (x_r, $y_r + 1$) will be toggled if their positions are within the n x n grid.
As shown in Figure (1), you are currently located at (1, 3), and there are five rooms with lights on: {(0, 3), (0, 4), (1, 2), (2, 3), (2, 4)}. You are allowed to move one room to the left/right/up/down, as illustrated in Figure (2). Alternatively, you can choose to stay in the current room and turn on the light, as it is currently off. If you choose to turn on the light in room (1, 3), then as depicted in Figure (3), the lights in room (1, 4) will also be turned on, while the lights in rooms (1, 2), (0, 3), and (2, 3) will be turned off. To turn off all the

lights, you can opt to move to (1, 4) (Figure (4)) and turn off the lights in room (1, 4). Consequently, the lights in rooms (1, 3), (0, 4), and (2, 4) will also be turned off, as shown in Figure (5).



(1) (2) (3) (4) (5)

After discussing this problem with your cousin, Ben Bitdiddle, he proposed some heuristics to be used with A^* search. Your task is to evaluate each heuristic.
Hint: The Manhattan Distance is defined as $MD(a,b) = |x_a - x_b| + |y_a - y_b|$, where x_a and y_a represent the row and column index of a, and the same applies to b.

h_D: The minimum Manhattan distance between rooms with lights on and your current position.

$h_D = \min_{r \in R} MD(r,p)$.
Here, R contains the indices of all the rooms with lights on.
Is h_D admissible for this Light Variant #1 problem?

✓A. Yes, it is admissible.
B. No, it is not admissible.

Notes:
To turn off all the lights, we need to turn off the light in the farthest room, r_f. To do so, we need at least move to one of its neighboring rooms and change the status of the light in that room. Then the cost is

$(\max_{r \in R} MD(r,p) - 1) + 1 = \max_{r \in R} MD(r,p) >= h_D$. Thus, h_D is admissible.

Item Weight: 2.0

Question #: 19

[Question Group: Heuristics: Light Variant #1]

h_D: The minimum Manhattan distance between rooms with lights on and your current position.

$h_D = \min_{r \in R} MD(r,p)$.
Here, R contains the indices of all the rooms with lights on.
Is h_D consistent for this Light Variant #1 problem?

A. Yes, it is consistent.

✓B. No, it is not consistent.

Notes:

Consider a situation where your current position is (3, 3), and the rooms with lights on is (0, 0). In this case, h_D(n) = 6. If you choose to turn on the light in room (3, 3) with a cost of 1, then h_D(n') = 0. h_D(n) = 6 > c(n, a, n') + h_D(n') = 1 + 0. Thus, h_D is not consistent.

Item Weight: 2.0

Question #: 20

[Question Group: Heuristics: Light Variant #1]

h_E: Divide the number of rooms with light on by 5.
Is h_E admissible for this Light Variant #1 problem?

✓A. Yes, it is admissible.

B. No, it is not admissible.

Notes:

Referring to Q21, we can prove that h_E is consistent. Since consistency implies admissibility, h_E is admissible.

Item Weight: 2.0

Question #: 21

[Question Group: Heuristics: Light Variant #1]

h_E: Divide the number of rooms with light on by 5.
Is h_E consistent for this Light Variant #1 problem?

✓A. Yes, it is consistent.

B. No, it is not consistent.

Notes:

Considering the available actions in the Light Variant #1 problem, the maximum number of lights can be turned off in one step is 5. So h_E can be reduced by at most 1 in each step, i.e., h_E (n) – h_E (n') ≤ 1 = c(n, a, n'). Therefore, h_E (n) ≤ h_E (n') + c(n, a, n'), and h_E is consistent.

Item Weight: 2.0

Question #: 22

[Question Group: Heuristics: Light Variant #1]

h_F: The average of the maximum and minimum Manhattan distances between rooms with light on and your current position.

$h_F = (\max_{r \in R} MD(r,p) + \min_{r \in R} MD(r,p))/2$.

Here, R contains the indices of all the rooms with lights on.

Is h_F admissible for this Light Variant #1 problem?

✓A. Yes, it is admissible.

B. No, it is not admissible.

Notes:

To turn off all the lights, we need to turn off the light in the farthest room, r_f. To do so, we need at least move to one of its neighboring rooms and change the status of the light in that room. Then the cost is

$(\max_{r \in R} MD(r,p) - 1) + 1 = \max_{r \in R} MD(r,p) >= h_F$. Thus, h_F is admissible.

Item Weight: 2.0

Question #: 23

[Question Group: Heuristics: Light Variant #1]

h_F: The average of the maximum and minimum Manhattan distances between rooms with light on and your current position.

$h_F = (\max_{r \in R} MD(r,p) + \min_{r \in R} MD(r,p))/2$.

Here, R contains the indices of all the rooms with lights on.

Is h_F consistent for this Light Variant #1 problem?

A. Yes, it is consistent.

✓B. No, it is not consistent.

Notes:

Consider a situation where your current position is (3, 3), and the rooms with lights on is (0, 0) and (0, 1). In this case, h_F(n) = (6+5)/2=5.5. If you choose to turn on the light in room (3, 3) with a cost of 1, then h_F(n') = (6+0)/2=3. h_F(n) = 5.5 > c(n, a, n') + h_D(n') = 1 + 3. Thus, h_F is not consistent.

Item Weight: 2.0

Question #: 24

[Question Group: Heuristics: Light Variant #1]

If you are required to use A^* graph search for this Light Variant #1 problem, which heuristic(s) should you pick among h_D, h_E and h_F?

1.h_D: The minimum Manhattan distance between rooms with lights on and your current position.

$h_D = \min_{r \in R} MD(r,p)$. Here, R contains the indices of all the rooms with lights on.

2.h_E: Divide the number of rooms with light on by 5.

3.h_F: The average of the maximum and minimum Manhattan distances between rooms with light on and your current position.

$h_F = (\max_{r \in R} MD(r,p) + \min_{r \in R} MD(r,p))/2$. Here, R contains the indices of all the rooms with lights on.

A. h_D

✓B. h_E

C. h_F

D. None of the above

Notes:

If h(n) is consistent, A* using graph search is optimal. h_D and h_F are not consistent, so we need to select h_E.

Item Weight: 2.0

Question #: 25

[Question Group: Decision Tree]
As DBZ Bank launches its recruitment drive, you, a machine learning specialist within its ranks, are tasked with the pivotal role of shortlisting candidates for interviews. Drawing from your expertise, you've chosen to kickstart the project by constructing a decision tree based on the bank's historical data presented in table below.

	Education	Experience	Reference	Salary Expectation	Outcome
0	Bachelor	1-3 years	bad	high	reject
1	Bachelor	1-3 years	good	high	offer
2	Bachelor	< 1 year	bad	low	offer
3	Bachelor	< 1 year	good	high	reject
4	Bachelor	> 3 years	good	high	reject
5	Master	< 1 year	good	high	offer
6	Master	> 3 years	good	high	offer
7	Ph.D	1-3 years	bad	low	offer
8	Ph.D	1-3 years	good	low	offer
9	Ph.D	> 3 years	good	low	offer

What is the entropy of the **Outcomes (offer/reject)** in the table, rounded to 2 decimal places? 0.88

Notes:

$$
\begin{aligned}
Entropy &= I(\frac{3}{10}, \frac{7}{10}) \\
&= -\frac{3}{10}\log_2(\frac{3}{10}) - \frac{7}{10}\log_2(\frac{7}{10}) \\
&= 0.88
\end{aligned}
$$

Update:
Answers with rounding errors are accepted.

Item Weight: 2.0

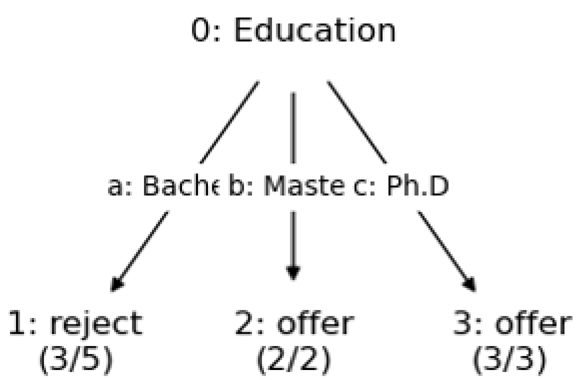
Question #: 26

[Question Group: Decision Tree]

You decide to start by creating a **one-level** decision tree with only one split, using information gain. What is **root node** for your **one-level** decision tree?

✓A. Education
B. Experience
C. Reference
D. Salary Expectation

Notes:



Item Weight: 2.0

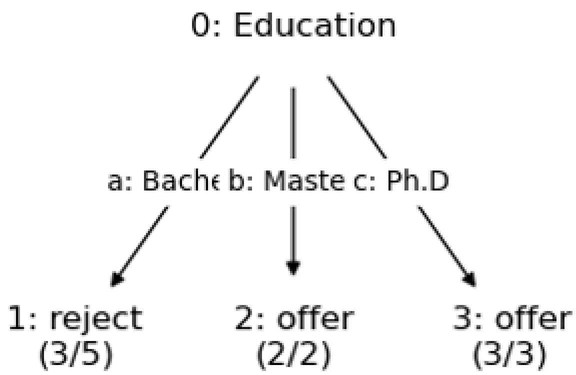
Question #: 27

[Question Group: Decision Tree]

Based on your **one-level** decision tree created according to information gain. What is the **Outcome** for the following applicant?
Applicant Information:
Education: **Bachelor**
Experience: **<1 year**
Reference: **good**
Salary Expectation: **low**

A. offer
✓B. reject
C. offer/reject

Notes:



Item Weight: 1.0

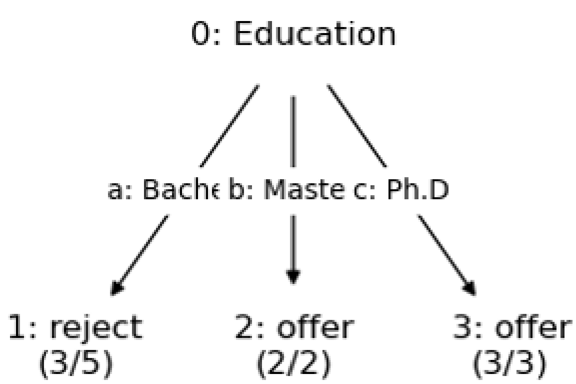
Question #: 28

[Question Group: Decision Tree]

Based on your **one-level** decision tree created according to information gain. What is the **Outcome** for the following applicant?
Applicant Information:
Education: **Master**
Experience: **>3 years**
Reference: **bad**
Salary Expectation: **high**

✓A. offer
B. reject
C. offer/reject

Notes:



Item Weight: 1.0

Question #: 29

[Question Group: Full Decision Tree]
As DBZ Bank launches its recruitment drive, you, a machine learning specialist within its ranks, are tasked with the pivotal role of shortlisting candidates for interviews. Drawing from your expertise, you've chosen to kickstart the project by constructing a decision tree based on the bank's historical data presented in table below.

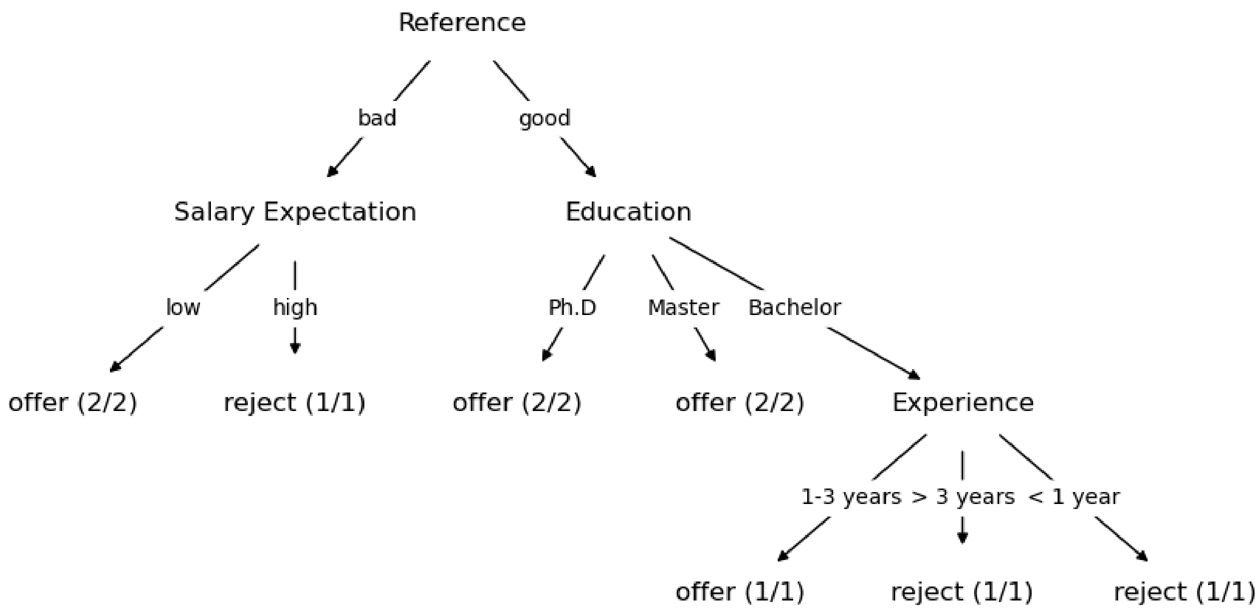
	Education	Experience	Reference	Salary Expectation	Outcome
0	Bachelor	1-3 years	bad	high	reject
1	Bachelor	1-3 years	good	high	offer
2	Bachelor	< 1 year	bad	low	offer
3	Bachelor	< 1 year	good	high	reject
4	Bachelor	> 3 years	good	high	reject
5	Master	< 1 year	good	high	offer
6	Master	> 3 years	good	high	offer
7	Ph.D	1-3 years	bad	low	offer
8	Ph.D	1-3 years	good	low	offer
9	Ph.D	> 3 years	good	low	offer

Suppose that you pick "**Reference**" as the root of your decision tree. Use the data in the table and information gain to create the full decision tree. In case of a tie, the priority order for constructing the tree is Education >Experience >Reference >Salary Expectation.

What is the **Outcome** for the following applicant according to your full decision tree?
Applicant Information:
Education: **Bachelor**
Experience: **<1 year**
Reference: **good**
Salary Expectation: **low**

A. offer
✓B. reject
C. offer/reject

Notes:



Update:

Q29 is a duplicate of Q31 and is voided.

Item Weight: 1.0

Question #: 30

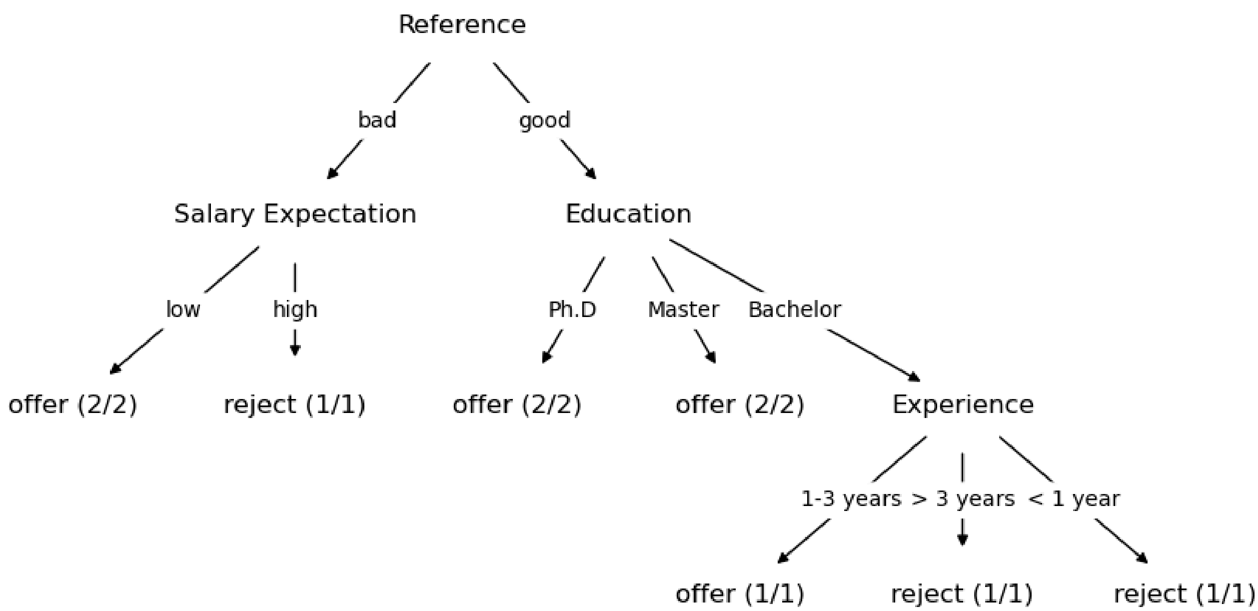
[Question Group: Full Decision Tree]

What is the **Outcome** for the following applicant according to your full decision tree?
Applicant Information:
Education: **Master**
Experience: **1-3 years**
Reference: **good**
Salary Expectation: **high**

✓A. offer

B. reject
C. offer/reject

Notes:



Item Weight: 1.0

Question #: 31

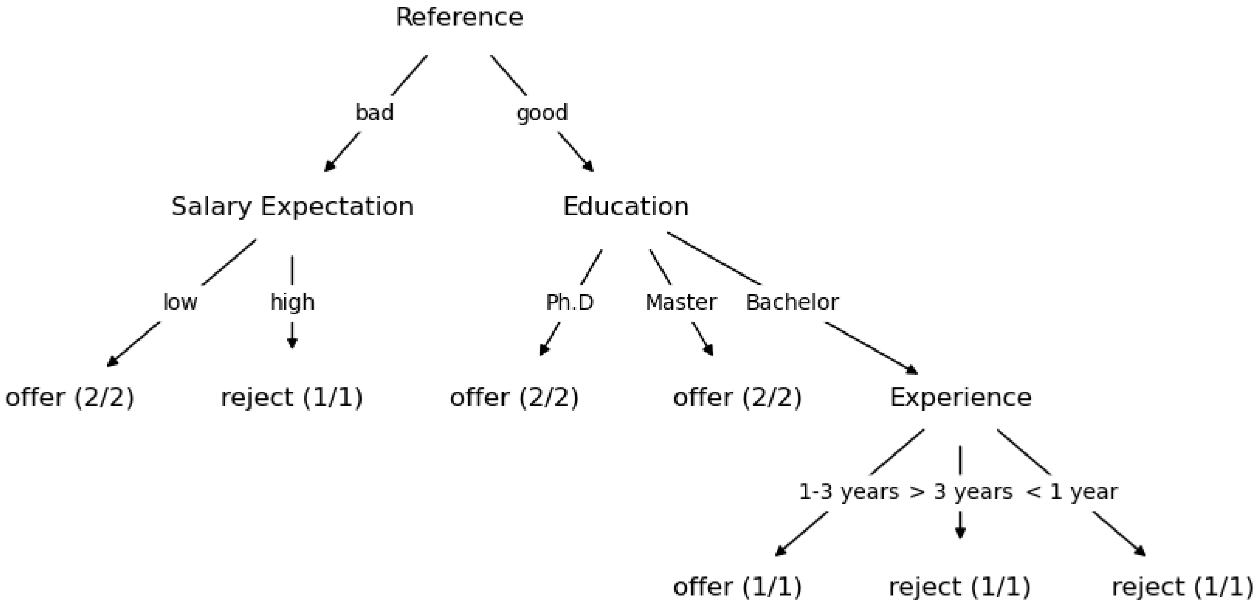
[Question Group: Full Decision Tree]

What is the **Outcome** for the following applicant according to your full decision tree?
Applicant Information:
Education: **Bachelor**
Experience: **<1 year**
Reference: **good**
Salary Expectation: **low**

A. offer
✓B. reject

C. offer/reject

Notes:



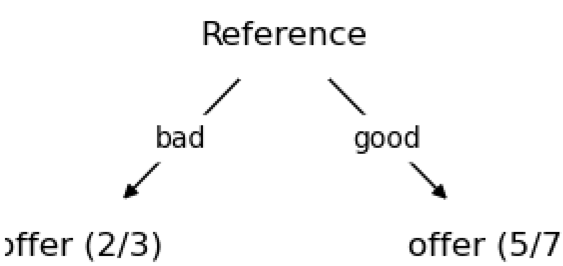
Item Weight: 1.0

Question #: 32

[Question Group: Full Decision Tree]

If you aim to prune the full decision tree, ensure that each leaf node contains at least 3 training data points.
How many leaf nodes in your pruned decision tree? __2 leaf nodes

Notes:



Update:
Simplifying decision tree following the lecture slides: okay. So 1 leaf node is also accepted.

Item Weight: 2.0

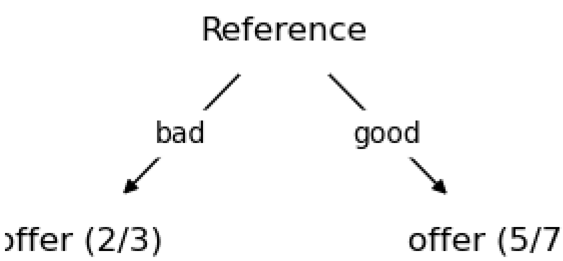
Question #: 33

[Question Group: Full Decision Tree]

If you aim to prune the full decision tree, ensure that each leaf node contains at least 3 training data points. According to your pruned decision tree, what is the **Outcome** for the following applicant?
Applicant Information:
Education: **Master**
Experience: **<1 year**
Reference: **bad**
Salary Expectation: **low**

- ✓A. offer
- B. reject
- C. offer/reject

Notes:

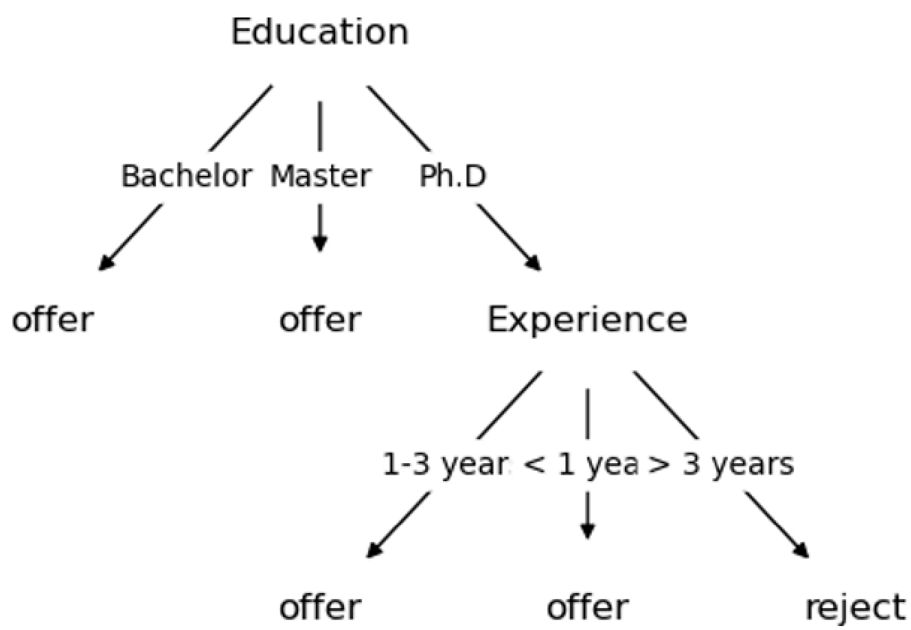


Item Weight: 1.0

Question #: 34

[Question Group: Decision Tree Performance Measurement]

One of your colleagues also design his own decision tree to help with recruitement, as shown below.



	Education	Experience	Reference	Salary Expectation	Outcome
0	Bachelor	1-3 years	bad	high	reject
1	Bachelor	1-3 years	good	high	offer
2	Bachelor	< 1 year	bad	low	offer
3	Bachelor	< 1 year	good	high	reject
4	Bachelor	> 3 years	good	high	reject
5	Master	< 1 year	good	high	offer
6	Master	> 3 years	good	high	offer
7	Ph.D	1-3 years	bad	low	offer
8	Ph.D	1-3 years	good	low	offer
9	Ph.D	> 3 years	good	low	offer

Using the data from the table, please figure out the number of **True Positives (TP)** of his decision tree. Use "offer" as a positive label and "reject" as a negative label.

The number of **True Positives (TP)** is __6__.

Item Weight: 0.5

Question #: 35

[Question Group: Decision Tree Performance Measurement]

Using the data from the table, please figure out the number of **False Positives (FP)** of his decision tree. Use "offer" as a positive label and "reject" as a negative label.
The number of **False Positives (FP)** is __3__.

Item Weight: 0.5

Question #: 36

[Question Group: Decision Tree Performance Measurement]

Using the data from the table, please figure out the number of **True Negatives(TN)** of his decision tree. Use "offer" as a positive label and "reject" as a negative label.
The number of **True Negatives(TN)** is __0__.

Item Weight: 0.5

Question #: 37

[Question Group: Decision Tree Performance Measurement]

Using the data from the table, please figure out the number of **False Negatives (FN)** of his decision tree. Use "offer" as a positive label and "reject" as a negative label.
The number of **False Negatives (FN)** is __1__.

Item Weight: 0.5

Question #: 38

[Question Group: Decision Tree Performance Measurement]

Using the data from the table, please figure out the **Precision** of his decision tree, rounded to 2 decimal places. Use "offer" as a positive label and "reject" as a negative label.
Precision is <u>0.67</u>.

Item Weight: 1.0

Question #: 39

[Question Group: Decision Tree Performance Measurement]

Using the data from the table, please figure out the **Recall** of his decision tree, rounded to 2 decimal places. Use "offer" as a positive label and "reject" as a negative label.
Recall is <u>0.86</u>.

Item Weight: 1.0

Question #: 40

[Question Group: Decision Tree Performance Measurement]
Using the data from the table, please figure out the **F1 Score** of his decision tree, rounded to 2 decimal places. Use "offer" as a positive label and "reject" as a negative label.
F1 Score is <u>0.75</u>.

Item Weight: 1.0

Question #: 41

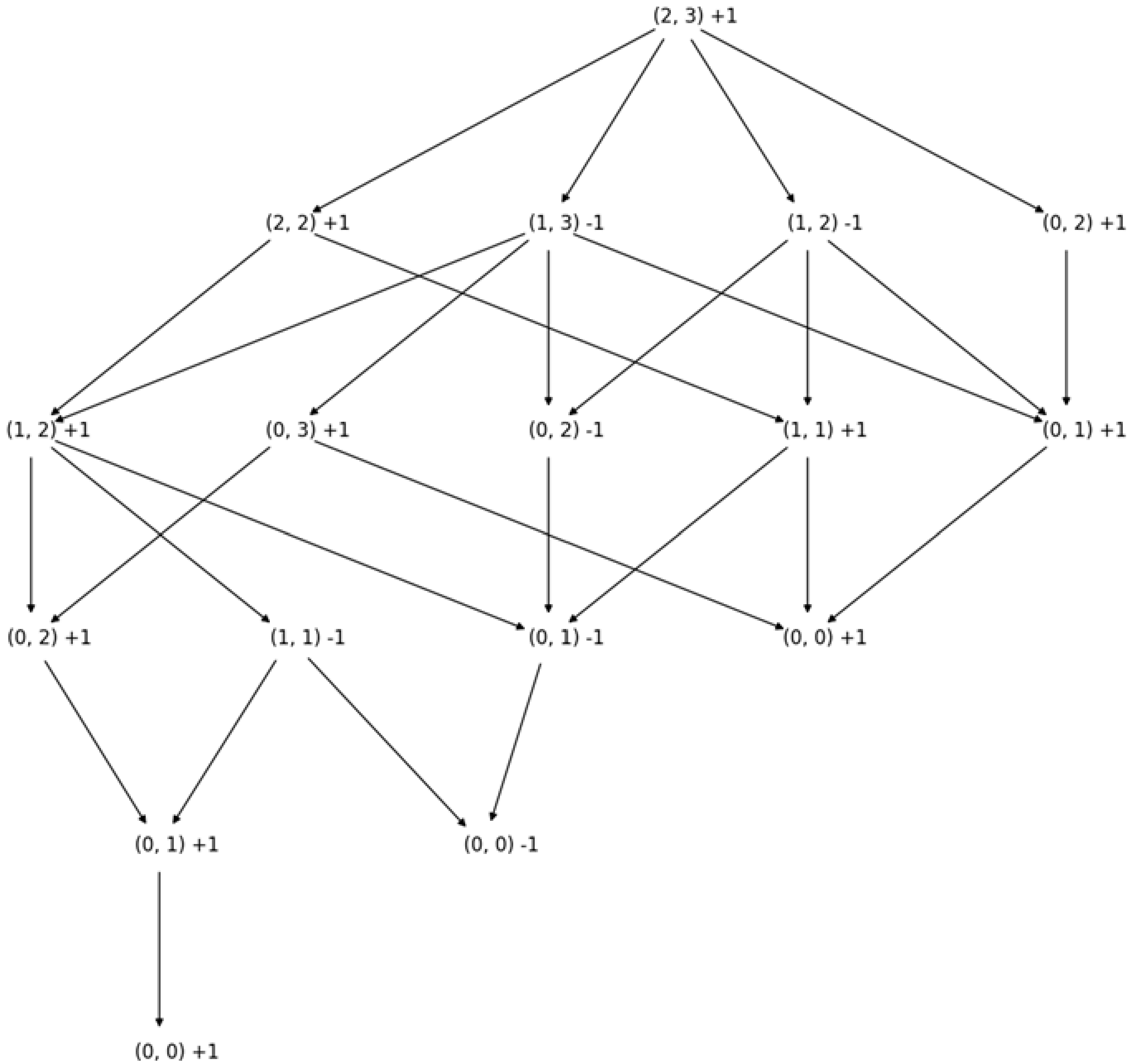
[Question Group: Game Tree &Minimax]
Consider a two-player game featuring **two piles** of sticks: one with **2 sticks** and the other with **3 sticks**. Players alternate turns, selecting to remove either **1 or 3 sticks** from <u>one or both piles</u>. If sticks are

removed from both piles, the number of sticks taken from each must be equal. The game concludes when a player takes the last stick or sticks, thus winning the game.
For example, at the beginning, the first player can opt to remove 1 stick from the first or second pile, 1 stick from each pile, or 3 sticks from the second pile. Notably, removing 3 sticks from each pile or the first pile isn't feasible since there are only 2 sticks in the first pile. If the first player removes 1 stick from each pile, the first pile now holds 1 stick, while the second pile holds 2 sticks. The second player then has the choice to remove either 1 stick from the first or second pile, or 1 stick from each pile. The game proceeds in this fashion until one player has no move left. At this juncture, the opposing player wins the game.
Note: In this game, if the max player wins, the utility is +1. If the max player loses, the utility is -1. A draw results in a utility of 0.
Suppose we represent the game state with a **monotonically increasing** sequence of integers. For instance, if there are two piles of sticks with the first pile containing 2 sticks and the second pile containing 1 stick, it will be represented as (1,2). Similarly, if there are two piles of sticks with the first pile containing 1 stick and the second pile containing 2 sticks, it will also be represented as (1,2). Now, suppose that we employ the Minimax algorithm to solve the game and constructing the complete game tree.

Which of the following state(s) is/are the children of state (0, 0)? Select all that is/are true.

A. (0, 0)
B. (0, 1)
C. (0, 2)
D. (0, 3)
E. (1, 1)
F. (1, 2)
G. (1, 3)
H. (2, 2)
I. (2, 3)
✓J. None of the above

Notes:



Item Weight: 0.5

Question #: 42

[Question Group: Game Tree &Minimax]

Which of the following state(s) is/are the children of state (0, 1)? Select all that is/are true.

✓A. (0, 0)
B. (0, 1)
C. (0, 2)
D. (0, 3)
E. (1, 1)
F. (1, 2)
G. (1, 3)
H. (2, 2)
I. (2, 3)
J. None of the above

Item Weight: 0.5

Question #: 43

[Question Group: Game Tree &Minimax]

Which of the following state(s) is/are the children of state (0, 2)? Select all that is/are true.

A. (0, 0)
✓B. (0, 1)
C. (0, 2)
D. (0, 3)
E. (1, 1)
F. (1, 2)
G. (1, 3)
H. (2, 2)
I. (2, 3)
J. None of the above

Item Weight: 0.5

Question #: 44

[Question Group: Game Tree &Minimax]

Which of the following state(s) is/are the children of state (0, 3)? Select all that is/are true.

✓A. (0, 0)
B. (0, 1)
✓C. (0, 2)
D. (0, 3)
E. (1, 1)
F. (1, 2)
G. (1, 3)
H. (2, 2)
I. (2, 3)
J. None of the above

Item Weight: 0.5

Question #: 45

[Question Group: Game Tree &Minimax]

Which of the following state(s) is/are the children of state (1, 1)? Select all that is/are true.

✓A. (0, 0)
✓B. (0, 1)
C. (0, 2)
D. (0, 3)
E. (1, 1)
F. (1, 2)
G. (1, 3)
H. (2, 2)
I. (2, 3)
J. None of the above

Item Weight: 0.5

Question #: 46

[Question Group: Game Tree &Minimax]

Which of the following state(s) is/are the children of state (1, 2)? Select all that is/are true.

A. (0, 0)
✓B. (0, 1)
✓C. (0, 2)
D. (0, 3)
✓E. (1, 1)
F. (1, 2)
G. (1, 3)
H. (2, 2)
I. (2, 3)
J. None of the above

Item Weight: 0.5

Question #: 47

[Question Group: Game Tree &Minimax]

Which of the following state(s) is/are the children of state (1, 3)? Select all that is/are true.

A. (0, 0)
✓B. (0, 1)
✓C. (0, 2)
✓D. (0, 3)
E. (1, 1)
✓F. (1, 2)
G. (1, 3)
H. (2, 2)
I. (2, 3)
J. None of the above

Item Weight: 0.5

Question #: 48

[Question Group: Game Tree &Minimax]

Which of the following state(s) is/are the children of state (2, 2)? Select all that is/are true.

A. (0, 0)
B. (0, 1)
C. (0, 2)
D. (0, 3)
✓E. (1, 1)
✓F. (1, 2)
G. (1, 3)
H. (2, 2)
I. (2, 3)
J. None of the above

Item Weight: 0.5

Question #: 49

[Question Group: Game Tree &Minimax]

Which of the following state(s) is/are the children of state (2, 3)? Select all that is/are true.

A. (0, 0)
B. (0, 1)
✓C. (0, 2)
D. (0, 3)
E. (1, 1)
✓F. (1, 2)
✓G. (1, 3)
✓H. (2, 2)
I. (2, 3)
J. None of the above

Item Weight: 0.5

Question #: 50

[Question Group: Game Tree &Minimax]

What is the value of state (1, 2) in depth 1?

✓A. -1
B. 0
C. 1
D. None of the above

Item Weight: 0.5

Question #: 51

[Question Group: Game Tree &Minimax]

What is the value of state (2, 3) in depth 0?

A. -1
B. 0
✓C. 1
D. None of the above

Item Weight: 0.5

Question #: 52

[Question Group: Game Tree &Minimax]

What is the value of state (0, 0) in depth 3?

A. -1
B. 0
✓C. 1
D. None of the above

Item Weight: 0.5

Question #: 53

[Question Group: Game Tree &Minimax]

What is the value of state (0, 2) in depth 1?

A. -1
B. 0
✓C. 1
D. None of the above

Item Weight: 0.5

Question #: 54

[Question Group: Game Tree &Minimax]

What is the value of state (0, 0) in depth 5?

A. -1
B. 0
✓C. 1
D. None of the above

Item Weight: 0.5

Question #: 55

[Question Group: Game Tree &Minimax]

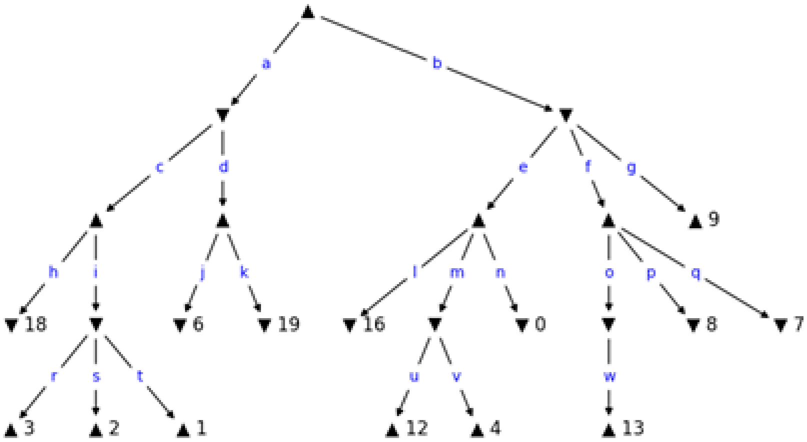
Which player do we expect to win?

✓A. First player
B. Second player
C. Draw
D. It depends

Item Weight: 0.5

Question #: 56

[Question Group: Alpha-beta Pruning]
Consider the following minimax tree:



What is the root value?

A. 0
B. 1
C. 2
D. 3
E. 4
F. 6
G. 7
H. 8
I. 9
J. 12
K. 13
L. 16
✓M. 18
N. 19

Notes:
See next question.

Item Weight: 1.0

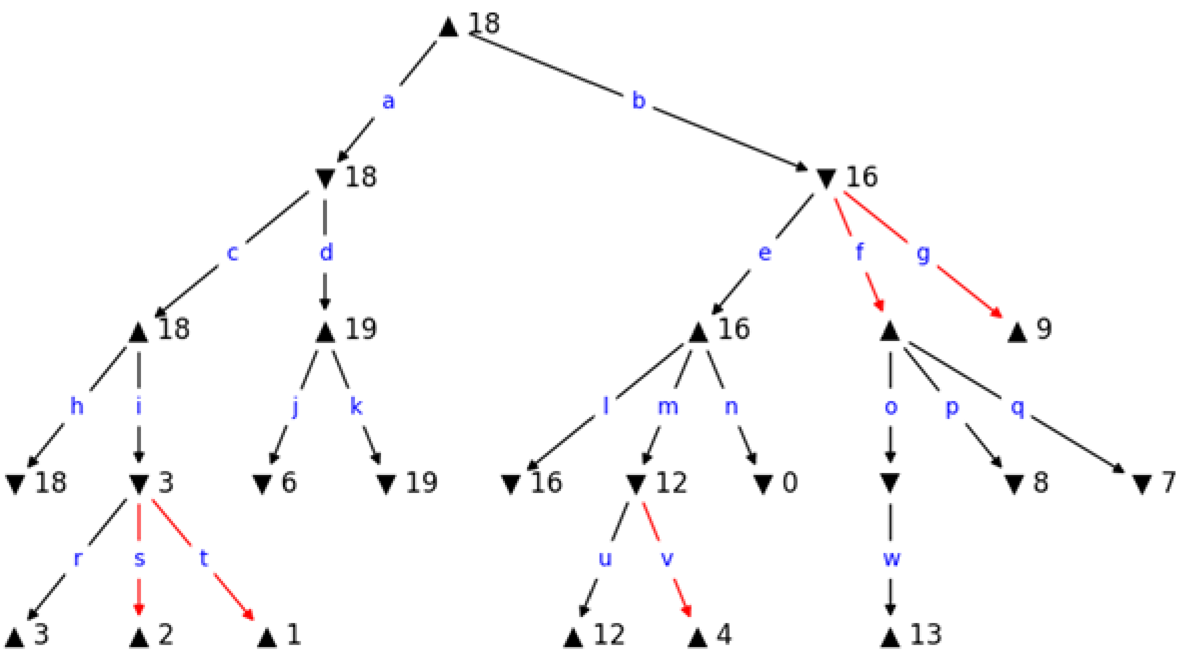
Question #: 57

[Question Group: Alpha-beta Pruning]

Suppose we traverse this tree with DFS from **left-to-right**. Select all the link(s) that would be pruned by alpha-beta pruning algorithm. Select only the links that are directly pruned and not those that are indirectly pruned because they are in a subtree of a pruned link.
Which of the following link(s) is/are pruned? Select all that is/are true.

A. a
B. b
C. c
D. d
E. e
✓F. f
✓G. g
H. h
I. i
J. j
K. k
L. l
M. m
N. n
O. o
P. p
Q. q
R. r
✓S. s
✓T. t
U. u
✓V. v
W. w
X. None of the above

Notes:



Item Weight: 4.0

Question #: 58

[Question Group: Alpha-beta Pruning]

Suppose we traverse this tree with DFS from **right-to-left**. Select all the link(s) that would be pruned by alpha-beta pruning algorithm. Select only the links that are directly pruned and not those that are indirectly pruned because they are in a subtree of a pruned link.
Which of the following link(s) is/are pruned? Select all that is/are true.

A. a
B. b
C. c
D. d
E. e
F. f
G. g
H. h
I. i
J. j
K. k
L. l
M. m
N. n

O. o

P. p

Q. q

✓R. r

✓S. s

T. t

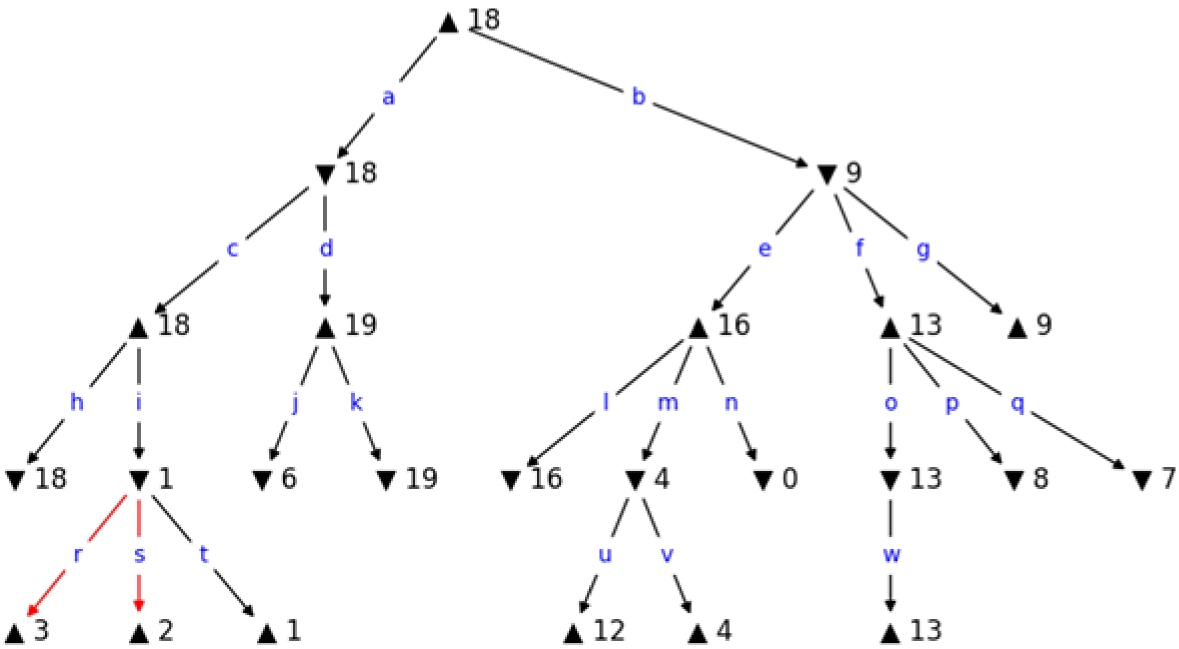
U. u

V. v

W. w

X. None of the above

Notes:



Item Weight: 4.0