15-281 Artificial Intelligence HW2

INSTRUCTIONS

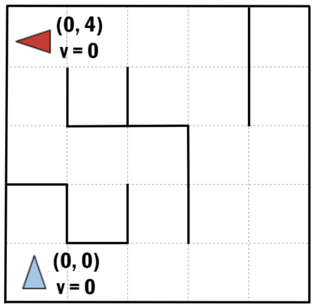
- **Due: Tuesday, 31 January 2023 at 10:00 PM EDT.** Remember that you may use up to 1 slip day for the Written Homework making the last day to submit **Wednesday, 1 February 2023 at 10:00 PM EDT**.
- **Format:** Write your answers in the `yoursolution.tex` file and compile a pdf (preferred) or you can type directly on the blank pdf. Make sure that your answers are within the dedicated regions for each question/part. If you do not follow this format, we may deduct points. We will NOT accept handwritten solutions of any kind.
- **Images:** To insert pictures, we recommend drawing it on PowerPoint or Google Drawings, saving it as an image and including it in your latex source.
- **How to submit:** Submit a pdf with your answers on Gradescope. Log in and click on our class 15-281 and click on the submission titled HW2 and upload your pdf containing your answers.
- **Policy:** See the course website for homework policies and Academic Integrity.

Name	
Andrew ID	
Hours to complete?	

For staff use only

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Total
/22	/15	/24	/17	/8	/11	/3	/100

Q1. [22 pts] Search and Heuristics



Imagine a car-like agent wishes to exit a maze like the one shown above. The agent is directional and at all times faces some direction $d \in (N, S, E, W)$. With a single action, the agent can *either* move forward at an adjustable velocity v or turn.

The moving actions are *faster*, *maintain* and *slower*. For these actions, the agent then moves a number of squares equal to its **new** adjusted velocity. Let v denote the agent's current velocity and let v' denote the agent's new adjusted velocity.

- *Faster*: $v' = v + 1$
- *Slower*: $v' = v$ - 1
- *Maintain*: $v' = v$

The turning actions are *left* and *right*, which change the agent's direction by 90 degrees. Turning is only permitted when the velocity is zero (and leaves it at zero).

- *Left*: change the agent's direction by 90 degrees counterclockwise
- *Right*: change the agent's direction by 90 degrees clockwise

For example, if the agent is currently on (0, 0) facing north with velocity 0 (as pictured) and wants to get to (2, 0) facing east with velocity 0, the sequence of actions will be: *right, faster, maintain, slower*.

Illegal actions include

- Any action that would result in a collision with a wall
- Any action that would reduce v below 0 or above a maximum speed V_{max}
- Maintaining a velocity of 0

The agent's goal is to find a plan which parks it (stationary) in the goal direction on the exit square using as few actions (time steps) as possible. Note that the cost of a path is defined by the number of actions the agent takes.

(a) [2 pts] Suppose the agent wants to take the leftmost path (i.e., the one that passes through (1,2)) from the start (0,0) facing north to the goal (0,4) facing west. Write down the sequence of actions for it to take.

Actions:

(b) [2 pts] If the grid is M by N, what is the size of the state space? You should assume that all configurations are reachable from the start state.

State Space Size:

(c) [4 pts] What is the maximum branching factor of this problem? Draw an example state (x, y, orientation, velocity, grid/walls) that has this branching factor, and list the set of available actions. For example, in the above picture, if the agent was in (0, 0) facing North with a velocity of 0, the branching factor would be 3. The agent could turn left, right, or go faster.

You may assume that illegal actions are simply not returned by the problem model and therefore not counted in the branching factor. You do not necessarily have to use the example grid above. Make sure to label your drawing.

Maximum Branching Factor:

Maximum Branching Example State and Available Actions:

(d) [4 pts] Is the Manhattan distance from the agent's location to the exit's location admissible?

If not, draw an example state (x, y, orientation, velocity, grid/walls) where this heuristic overestimates at that state, and specify: 1) the heuristic value at that state and 2) the actual cost from that state to the goal.

You may assume that illegal actions are simply not returned by the problem model. You do not necessarily have to use the example grid above. Make sure to label your drawing, including the goal state.

○ Yes ○ No

Example State, Heuristic Value, Actual Cost:

(e) [4 pts] Is the following heuristic admissible? *Manhattan distance* / V_{max}.

If not, draw an example state (x, y, orientation, velocity, grid/walls) where this heuristic overestimates at that state, and specify: 1) the heuristic value at that state and 2) the actual cost from that state to the goal.

You may assume that illegal actions are simply not returned by the problem model. You do not necessarily have to use the example grid above. Make sure to label your drawing, including the goal state.

◯ Yes ◯ No

Example State, Heuristic Value, Actual Cost:

(f) [2 pts] If we used an inadmissible heuristic in A* Tree search, could it change the completeness of the search? Assume the graph is finite and the heuristic is non-negative.

◯ Yes ◯ No

(g) [2 pts] If we used an inadmissible heuristic in A* Tree search, could it change the optimality of the search? Assume the graph is finite and the heuristic is non-negative.

◯ Yes ◯ No

(h) [2 pts] What is a general advantage that an inadmissible heuristic might have over an admissible one. Select all that apply.

☐ An inadmissible heuristic may be easier to compute, leading to a faster state heuristic computation time.

☐ An inadmissible heuristic can be a closer estimate to the actual cost (even if it's an overestimate) than an admissible heuristic, thus exploring fewer nodes.

☐ An inadmissible heuristic will still find optimal paths when the actual costs are non-negative.

☐ An inadmissible heuristic may be used to completely block off searching part of a graph in a search algorithm.

Q2. [15 pts] Search Nodes

Consider the tree search (i.e. no explored set) of an arbitrary search problem with max branching factor b. Each search node n has a backward (cumulative) cost of $g(n)$, an admissible heuristic of $h(n)$, and a depth of $d(n)$. Let n_c be a minimum-cost goal node, and let n_s be a shallowest goal node.

For each of the following, give an expression that characterizes the set of nodes that are explored before the search terminates. For instance, if we asked for the set of nodes with positive heuristic value, you could say: for all n, such that $h(n) \geq 0$. Don't worry about ties (so you won't need to worry about $>$ versus $\geq$). If there are no nodes for which the expression is true, you must write "none."

(a) [5 pts] Give an inequality in terms of the functions g, h, and d, as well as the nodes n_c and n_s defined above to describe the nodes n that are explored in a **breadth-first search** before terminating.

Inequality: All n, such that:

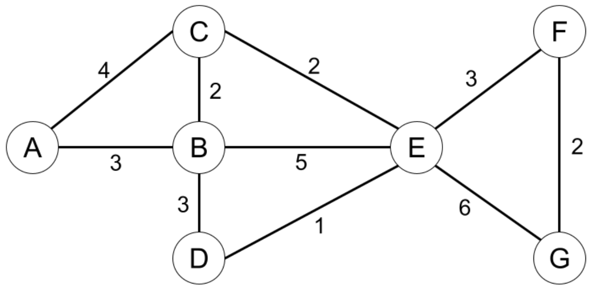
(b) [5 pts] Give an inequality in terms of the functions g, h, and d, as well as the nodes n_c and n_s defined above to describe the nodes n that are explored in a **uniform cost search** before terminating.

Inequality: All n, such that:

(c) [5 pts] Now for this question, assume the heuristic h is consistent. Give an inequality in terms of the functions g, h, and d, as well as the nodes n_c and n_s defined above to describe the nodes n that are explored in an **A* search** before terminating.

Inequality: All n, such that:

Q3. [24 pts] Searching a Graph



Node	h_1	h_2
A	12	11
B	6	7
C	9	6
D	3	4
E	3	5
F	2	1
G	0	0

Consider the graph shown above. A is the start state and G is the goal state. The costs for each edge are shown on the graph. Each edge can be traversed in both directions (each edge can be traversed in either direction). Please refer to the search algorithms **exactly as presented on the lecture slides** as the ordering of the actions matters.

(a) [15 pts] For each of the following **graph search** strategies, mark with an X which (if any) of the listed paths it could return. Note that for some search strategies the specific path returned might depend on tie-breaking behavior. In any such cases, make sure to mark **all** paths that could be returned under some tie-breaking scheme. If a graph search strategy returns a path not listed, **write out the correct path** in the *Other* column.

Algorithm	A-C-E-G	A-C-E-F-G	A-B-D-E-F-G	Other
UCS	(i)	(ii)	(iii)	(iv)
Greedy with heuristic h_1	(v)	(vi)	(vii)	(viii)
Greedy with heuristic h_2	(ix)	(x)	(xi)	(xii)
A* with heuristic h_1	(xiii)	(xiv)	(xv)	(xvi)
A* with heuristic h_2	(xvii)	(xviii)	(xix)	(xx)

(b) [1 pt] What is the cost of the optimal path for uniform cost search from A to G?

Answer:

(c) [4 pts] Is h_1 admissible? Is it consistent?

Admissible: ◯ Yes ◯ No

Consistent: ◯ Yes ◯ No

(d) [4 pts] Is h_2 admissible? Is it consistent?

Admissible: ◯ Yes ◯ No

Consistent: ◯ Yes ◯ No

Q4. [17 pts] Search: Multiple Choice and Short Answer Questions

(a) [12 pts] Consider the following true/false questions with each question worth 2 points. For the following search problems, assume every action has a cost of at least ϵ, with $\epsilon > 0$. Assume any heuristics used are consistent.

Depth-first tree-search on a finite graph is guaranteed to be complete.
◯ **True** ◯ **False**

Breadth-first tree-search on a finite graph is guaranteed to be complete.
◯ **True** ◯ **False**

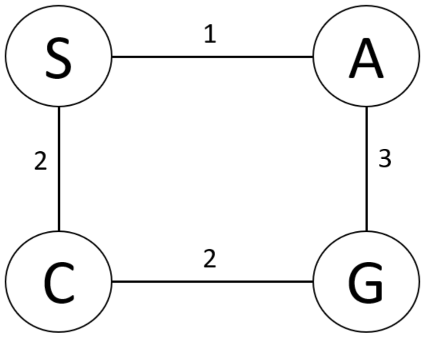
Iterative deepening tree-search on a finite graph is guaranteed to be complete.
◯ **True** ◯ **False**

For all graphs without cycles, graph-search contains a larger frontier than tree-search.
◯ **True** ◯ **False**

Iterative deepening graph-search has the time complexity of BFS and the space complexity of DFS.
◯ **True** ◯ **False**

If h_1(s) is a consistent heuristic and h_2(s) is a consistent heuristic, then min(h_1(s), h_2(s)) must be consistent.
◯ **True** ◯ **False**

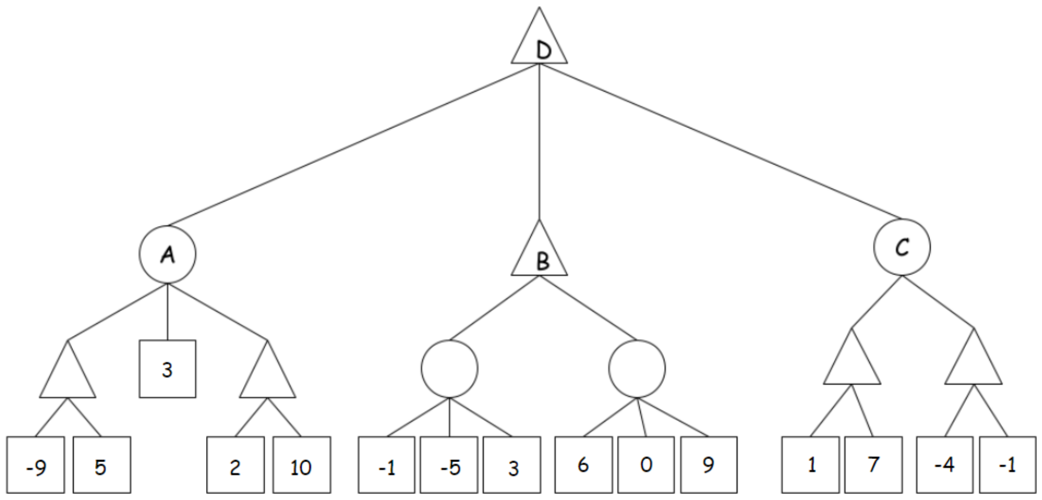
(b) [5 pts] Consider the state space graph shown below. S is the start state and G is the goal state. The costs for each edge are shown on the graph. For the following table below, fill in potential heuristic values such that the heuristic is admissible but not consistent.



Heuristic Function	
State	$h(s)$
S	
A	
C	
G	0

Q5. [8 pts] Expectimax Search

We have replaced the minimization nodes with expectation nodes (circles). Use the algorithm presented in class and the search tree to answer the following questions. Assume equal weight.



(a) [2 pts] What is the value of node A?

Answer:

(b) [2 pts] What is the value of node B?

Answer:

(c) [2 pts] What is the value of node C?

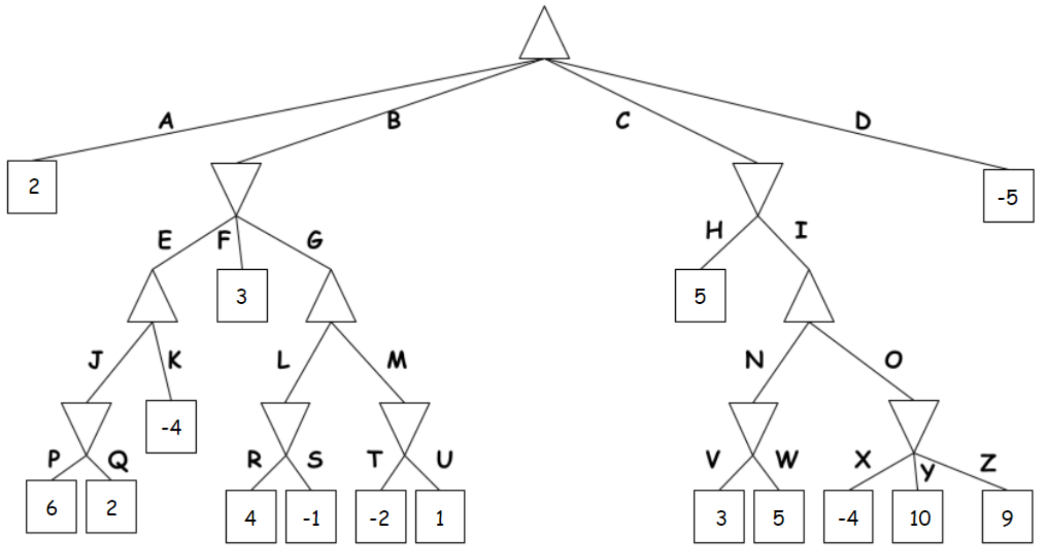
Answer:

(d) [2 pts] What is the value of node D?

Answer:

Q6. [11 pts] Minimax and Alpha-Beta Pruning

Note: Triangle nodes pointing up refer to max nodes, and triangle nodes pointing down refer to min nodes.



(a) [2 pts] What are the minimax values with alpha-beta pruning for the minimizer nodes at level 2 (assuming the root node is at level 1)?

(i) [1 pt]

Node B:

(ii) [1 pt]

Node C:

(b) [1 pt] What is the minimax value of the root?

Answer:

(c) [4 pts] Assuming children are visited in left-to-right order, which branches (letters) would be pruned by alpha-beta pruning?

NOTE: select the top most branch that can be pruned. For example, if branch B is pruned, then we know E, F, and G are pruned as well. However, in this case, only select B as your answer.

Answer:

(d) [4 pts] Which of the below orderings of node B's children would result in the most pruning (select all that apply)? Assume all other nodes are visited in left-to-right order, except for children of node B. For example, if you believe right-to-left order would result in the most pruning, you should select G,F,E

☐ E,F,G ☐ E,G,F ☐ F,E,G ☐ F,G,E ☐ G,E,F ☐ G,F,E

Q7. [3 pts] Ethics

(a) [3 pts] Please review this article discussed in recitation: *This Navigation App Keeps You Out of High-crime Areas in Real Time*,
`https://www.androidauthority.com/redzone-navigation-app-avoids-high-crime-areas-686894/`.

(i) [1 pt] Let's assume this navigation app uses A* search with an admissible and consistent heuristic to find and return paths. Suppose you are working on this app, and your boss proposes a new heuristic. There is some crime rate metric that ranges from 0 to 5, where 5 is high crime. Your boss proposes that you add this crime rate metric to the application's current heuristic, such that **the heuristic is still admissible and consistent.**

Will the path **necessarily** change with the new heuristic? If so, will the new path be optimal? If not, under what conditions would the path change?

Answer:

(ii) [1 pt] If for some reason you choose not to adjust the heuristic, what is one way you could modify the graph structure so that your implementation could return paths that avoid high crime areas? Give your answer in one sentence.

Answer:

(iii) [1 pt] In what ways could people and stores residing in high crime neighborhoods be impacted by changes in navigation algorithms, particularly focusing on the economic and/or societal impact? Please provide 2-3 sentences.

Answer: