CS 221, Fall 2020
Quiz 3

You have 1 hour to complete the quiz. You are allowed to consult course notes and books but no communication or general internet access is allowed. Good luck!

Stanford University Honor Code

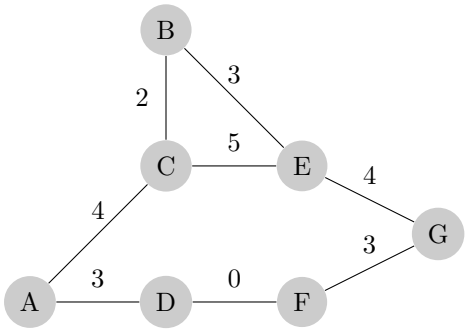
The Honor Code is the University's statement on academic integrity written by students in 1921. It articulates University expectations of students and faculty in establishing and maintaining the highest standards in academic work:

- The Honor Code is an undertaking of the students, individually and collectively:
  - that they will not give or receive aid in examinations; that they will not give or receive unpermitted aid in class work, in the preparation of reports, or in any other work that is to be used by the instructor as the basis of grading;
  - that they will do their share and take an active part in seeing to it that others as well as themselves uphold the spirit and letter of the Honor Code.
- The faculty on its part manifests its confidence in the honor of its students by refraining from proctoring examinations and from taking unusual and unreasonable precautions to prevent the forms of dishonesty mentioned above. The faculty will also avoid, as far as practicable, academic procedures that create temptations to violate the Honor Code.
- While the faculty alone has the right and obligation to set academic requirements, the students and faculty will work together to establish optimal conditions for honorable academic work.

Question	**Score**
1	/ 8
2	/ 4
Total score:	**/ 22**

Multiple Choice

1. **[2 points]** Consider finding the minimum cost path from A to G in the following graph. The edge costs are indicated in the graph.



[1 point] Can we use the dynamic programming graph search algorithm presented in lecture to find the minimum cost path from A to G?

(a) Yes

(b) No

Answer: No, because the graph is not acyclic.

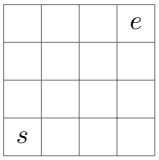
[1 point] Can we use uniform cost search to find the minimum cost path from A to G?

(a) Yes

(b) No

Answer: Yes, because all the edge costs are non-negative.

2. **[1 point]** Suppose we start at s in the following grid world:



We can take actions $a \in \{(0,1),(0,-1),(1,-1),(1,0),(1,1),(-1,-1),(-1,0),(-1,1)\}$, that correspond to moving in each possible direction (e.g. $a = (0,1)$ corresponds to moving vertically up one square).

Suppose our goal is to get to the upper right corner, i.e. the square marked e at position $(3,3)$, with as low a cost as possible. Our state is defined as $(\text{CurrentPosition}, \text{PastCost})$. As an example, the start state s is just $((0,0),0)$, which we can index as $s[0] = (0,0), s[1] = 0$.

The costs are defined as:

$$\text{Cost(s,a)} = \begin{cases} 1 & \text{if ManhattanDistance}(s[0], \text{Succ}(s,a)[0]) = 1 \\ 3 & \text{if ManhattanDistance}(s[0], \text{Succ}(s,a)[0]) = 2 \end{cases}$$

and the successor function is defined as:

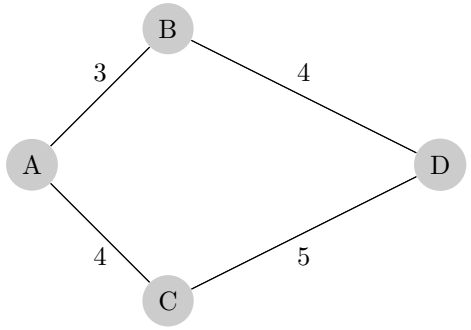
$$\text{Succ}(s,a) = (s[0] + a, s[1] + \text{Cost}(s,a))$$

Suppose we take action $a_1 = (1,1)$ followed by action $a_2 = (-1,0)$. What is our state now?

Recall that the state is of the form $(\text{CurrentPosition}, \text{PastCost})$, where CurrentPosition is a tuple and PastCost an integer. For example, we write the start state as $((0,0),0)$. Write your answer with no spacing in between characters.

Answer: ((0,1),4).

3. **[2 points]** Consider the following graph. We **start at node A**, and our **goal is to get to node D**. The edge costs are indicated in the figure.



We want to use A^* to solve this search problem. Given the set of nodes $\mathcal{V}$ and heuristic function $h(s) : \mathcal{V} \to \mathbb{R}$ defined in the following table:

s	$h(s)$
A	6
B	2
C	2
D	0

[1 point] Is $h(s)$ a consistent heuristic?

(a) Yes

(b) No

Answer: No, $h(s)$ is not a consistent heuristic. Consider the path A-B-D. There, we have Cost'(A,B) = Cost(A, B) + $h(B) - h(A) = 3 + 2 - 6 = -1 < 0$.

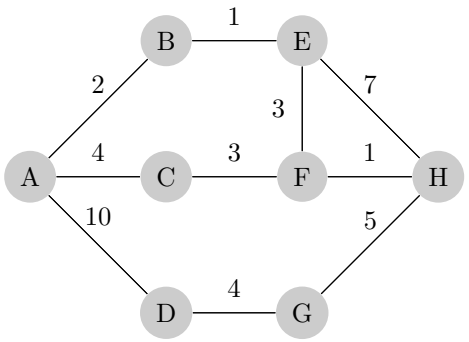
[1 point] Is $h(s)$ an admissible heuristic?

(a) Yes

(b) No

Answer: Yes, $h(s)$ is an admissible heuristic, because for every node s it underestimates FutureCost(s).

4. **[2 points]** Consider the following graph. We **start at node A**, and our **goal is to get to node H**. The edge costs are indicated in the figure.



We want to use UCS to solve this search problem, and start by adding A to the frontier. Step through the algorithm until the goal H has been visited. Keep track of what the frontier set and explored set contain.

[1 point] In what order are the nodes added to the explored setg? Give your answer in the form $A, \text{Node}_2, \text{Node}_3, ..., H$ **with no spacing in between nodes**. Note that you should include A at the start and H at the end.

Answer: The answer is A, B, E, C, F, H.

[1 point] What does the frontier look like immediately **before we remove H from it**? Write your answer in the form $\text{Node}_1 p_1, \text{Node}_2 p_2, ...,$ where $p_1 \leq p_2 \leq p_3 ...$ are the priorities of the respective nodes, in ascending order (i.e. smallest priority furthest to the left). For example, if the frontier contains node X with priority 5, and node Y with priority 3, you would write this as $Y3, X5$ (with no spacing in between nodes, as in the previous question).

Answer: The answer is $H7, D10$.

5. **[1 point]** Suppose h_a and h_b are consistent heuristics for some search problem we are trying to solve. A friend of yours proposes to use h_c instead, but you're not quite sure if you can use it. The values of the three heuristics evaluated at all states of the search problem are tabulated below.

s	$h_a(s)$	$h_b(s)$	$h_c(s)$
A	5	4	5
B	4	5	5
C	3	3	3
D	2	1	2
E	0	0	0

Is $h_c(s)$ a consistent heuristic?

(a) Yes

(b) No

(c) We cannot tell without seeing the graph.

Answer: Yes, $h_c(s)$ is a consistent heuristic, since it is the maximum of two consistent heuristics.

Short Answer

6. **[4 points]** You are a freshman at Stanford and struggle to arrive on time to your 9AM seminar in the Gates building because it's easy to get lost on the way. You decide to **find the fastest path** to **bike** from your dorm to Gates. You first attempt to use uniform cost search, but find it computationally intractable (exploring and mapping all routes by bike would take far more time than you have) and instead switch to using A* search. You need a heuristic that guarantees you find the fastest path. Throughout this question we assume that you do not have access to an oracle (e.g. Google maps does not provide the true fastest route or accurate time estimates along any specified route), but that you can use a python program, a stopwatch, an accurate GPS, a speedometer, and a compass.

   [1 point] First you relax the search problem. Select all valid relaxations:

   (a) Finding the fastest path to Gates when walking at a continuous speed (one fifth biking speed).

   (b) Finding the fastest path to Gates with half the streets closed (randomly selected).

   (c) Finding the fastest path to Gates with all other campus buildings removed (you can bike in a straight line to Gates at a constant speed).

   (d) Finding the fastest path to Gates when riding an electric bike that increases your speed by 1.5x.

   Answer: (c), (d). In order for a relaxation to be valid, we require that all edge costs be $\leq$ what they were in the original problem. In other words, $\text{Cost}_{\text{rel}}(s, a) \leq \text{Cost}(s, a)$ for all state-action pairs, which is only true for (c) and (d).

   [1 point] Second, you define a heuristic in terms of one of the relaxations. Which relaxation gives the best design trade-offs?

   (a) Finding the fastest path to Gates when walking at a continuous speed (one fifth biking speed).

   (b) Finding the fastest path to Gates with half the streets closed (randomly selected).

   (c) Finding the fastest path to Gates with all other campus buildings removed (you can bike in a straight line to Gates at a constant speed).

   (d) Finding the fastest path to Gates when riding an electric bike that increases your speed by 1.5x.

   Answer: (c). (a) and (b) do not lead to consistent heuristics, so they are out of the question. (d) leads to a consistent heuristic, but it is as hard as the original problem (biking a fixed distance takes two thirds the original time), making it computationally intractable.

   [1 point] Justify your answer to the previous question. Below, select the reason why the relaxation you chose in the previous question gives the best design trade-offs among the relaxations listed.

   (a) computational efficiency of computing the heuristic

   (b) informativeness of the heuristic

   (c) consistency of the heuristic

   (d) admissibility of the heuristic

   Answer: (a). Relaxations (a) and (b) do not yield consistent heuristics, so we need only concern ourselves with (d). Relaxation (d), as mentioned in the answer to the previous question, is as hard as the original problem, so it is not a useful heuristic. Relaxation (c) can be computed very efficiently using the distance to Gates (computed using a GPS) and your constant biking speed (computed using the GPS and a stopwatch or your speedometer).

[1 point] Let's say you selected a relaxation that results in a consistent heuristic $h_1(s)$. Your friend tells you that they have another consistent heuristic $h_2(s)$ that is different than the one you're using. They state that by combining your heuristic with theirs, specifically by taking the maximum of the two, you can get a new heuristic $h_3(s) = \max\{h_1(s), h_2(s)\}$. Your friend claims that if you use this heuristic you are guaranteed to explore **no more states** than you would using only $h_1(s)$. Is your friend correct?

(a) Yes

(b) No

Answer: Yes. Since h_3 is the max of two consistent heuristics, it is guaranteed to be *at least as informed* as either one of them separately. Therefore, in the worst case scenario, it will do as well as just h_1, but it has potential to do better if the information provided by h_2 improves its estimates of the FutureCost.