CS 3600 Midterm (section A)

Due Oct 25, at 11:59pm Eastern

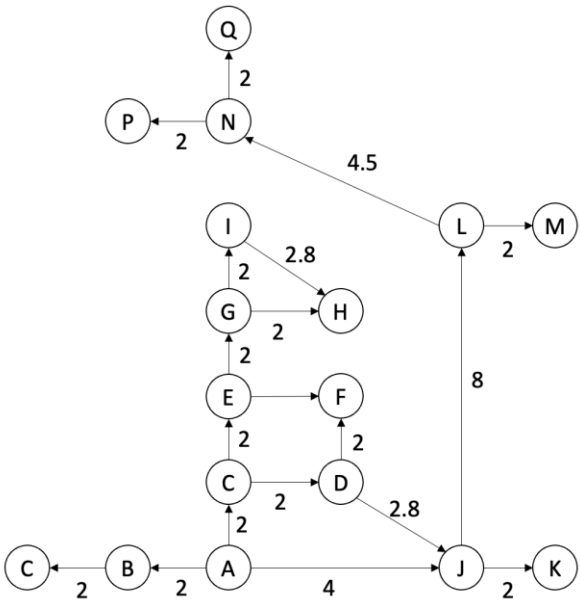
We expect this exam to take ~1.5 hours, however you may use your time in any way you see fit. The exam is open notes, open book, and open lecture videos. The course collaboration policy and late day policy are in effect.

You may edit the file to insert your answers or print, write by hand, and scan back in. You may add extra white space between questions if you need it.

Submit your final version as a PDF to Gradescope.

List any collaborations here: _____

Question 1. Consider the following layout of your secret underground lair built into the side of a mountain. The circles are rooms and the edges are corridors. The numbers are the distance in decameters. You have cleverly built a robot to deliver your meals. Unfortunately, because of the geology of the mountain all the corridors are sloped and the robot can only traverse the corridors going downhill (minions must carry the robot back uphill when it has finished making its delivery). The direction of the arcs in the graph indicate the downhill direction. The robot always starts in room A (the kitchen).



1.a (1 pt.) Compute the average number of successors: _____

1.b (1 pt.) Compute the average number of predecessors. The predecessors are found by reversing the directionality of the arcs: _____

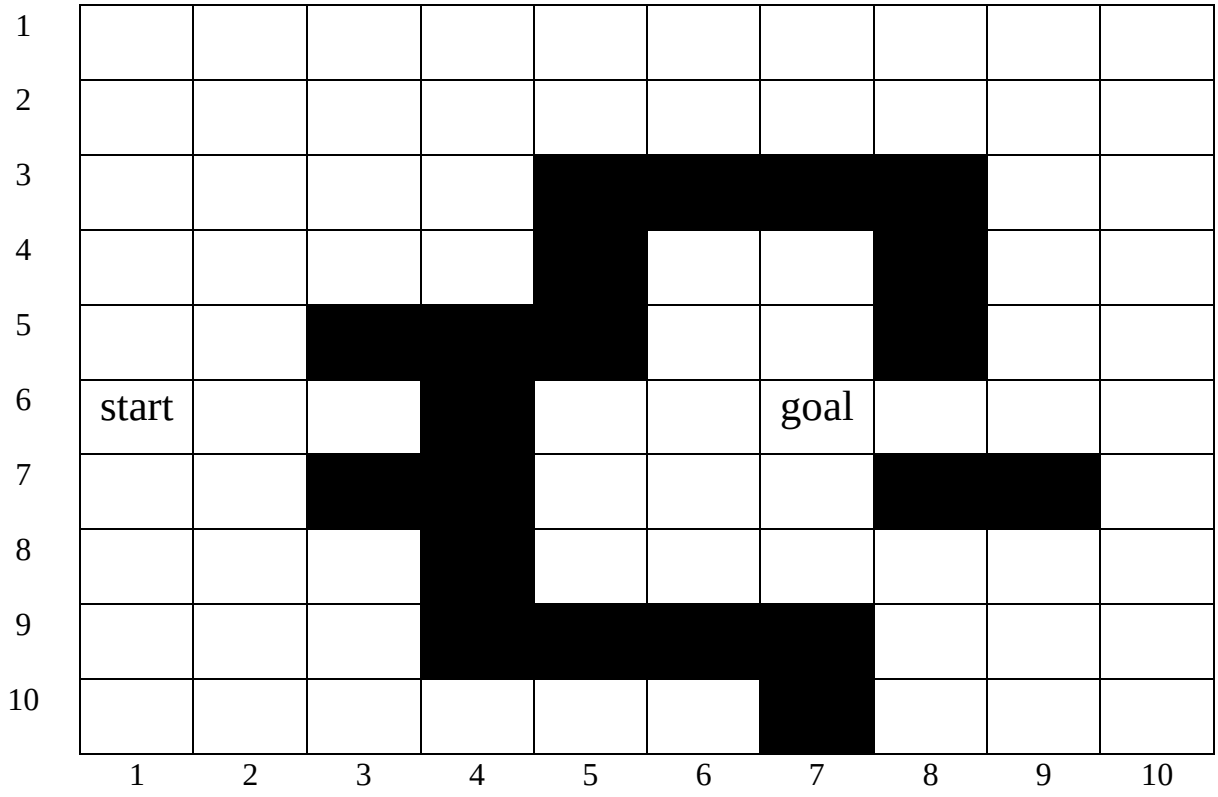
1.c (2 pts.) Suppose you are in your science lab in room P when you get hungry. The agent could run the breadth-first search *forward* from the kitchen to the delivery destination, or run breadth-first search *backward* by reversing the directionality of the arcs and starting from the destination and working back to the kitchen.

Should the search be conducted forward or backward? _____

Explain why:

1.d (1 pts.) Is it always more efficient to run the search in the direction you specified in 1.c regardless of what the robot's destination is? Explain why.

Question 2. Consider the following grid world. The agent can move up, down, left, and right. There is a cost of 1 for every action. The agent starts in the designated cell and must reach the cell marked "goal". The goal is a terminal state and also has a reward value of 100.



2.a. (1 pt.) If the transition function is deterministic, one would use A* with a heuristic h(state) = manhattan_distance(state, goal). What is the length of the optimal path from the start to the goal?

_____

2.b. (1 pt.) How many states must be visited by A* using the above heuristic? Hint: the maximum f-value of any state that can be part of the solution path is 18.

_____

2.c. (2 pts.) Suppose we didn't have the transition function (and thus didn't know if it was deterministic or not) and thus used Q-learning instead. We refer to the fact that there is only a single state, the "goal", that provides a reward as a "sparse" reward function. With an empty Q-table and with most states returning zero reward, it can take quite a while for the agent to find the goal randomly and to start propagating q-values to neighboring states. If we had a "dense" reward function, then every state would provide a reward and the agent would learn much faster.

A perfect dense reward function would be one in which each state gave out reward proportional to its true utility (i.e. imagine we had a transition function and used value iteration until convergence then copied the utility values into the reward function).

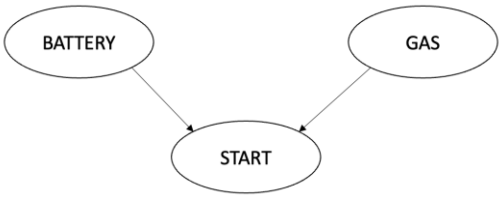
Given that we cannot know the true state utilities ahead of time, perhaps we can instead compute in O(n) time (where n is the number of states) a dense reward function that approximates the true utility values for each state. That is, we would need to create a reward function by performing a O(1) operation on each state. Describe in a few sentences how to compute such a reward function.

Hint: think about how admissible heuristics are created.

2.d. (1 pt.) Suppose the q-learning agent is using an epsilon-greedy exploration strategy using the dense rewards you computed above. Is it a problem if the reward values for states might be misleading? For example, the state at (x=3, y=6) might end up with a reward that is higher than the rewards given at (x=3, y=5) or (x=2, y=5) or (x=2, y=6). That is, will the agent get stuck and never learn a policy that reaches the goal? Why or why not?

Question 3: Consider the following Bayesian network that models components of a car.

- BATTERY: The car's battery is working (true or false)
- GAS: The car's gas tank has gas (true or false)
- START: The engine will start (true or false).



The engine will not start when the car is out of gas or if the battery is not working. If the car has gas and the battery is working there is an 80% chance that the car will start. History suggests that the battery works 70% of the time and that the car has gas 60% of the time.

3.a. (1 pt.) The Bayesian Net tells us that BATTERY and GAS must be independent. Use the definition of independence to fill out the following joint probability table:

	GAS = TRUE	GAS = FALSE
BATTERY = TRUE		
BATTERY = FALSE		

3.b. (2 pts.) Suppose you know that the car doesn't start. Use the definition of independence to prove or show that BATTERY and GAS are dependent on each other when the car is observed to not start. Recall that dependence means that when one variable changes its value, then the distribution over the values of the other variable changes. Show your work to receive any partial credit.

3.c. (2 pt.) Given your answer to 3.b and the fact that the car doesn't start, explain how you could use a voltmeter on the battery to predict whether the gas tank is empty. Draw a new Bayesian Network that illustrates how the test works. [Hint: your network should now have 4 nodes, and the new node should be an emission model]