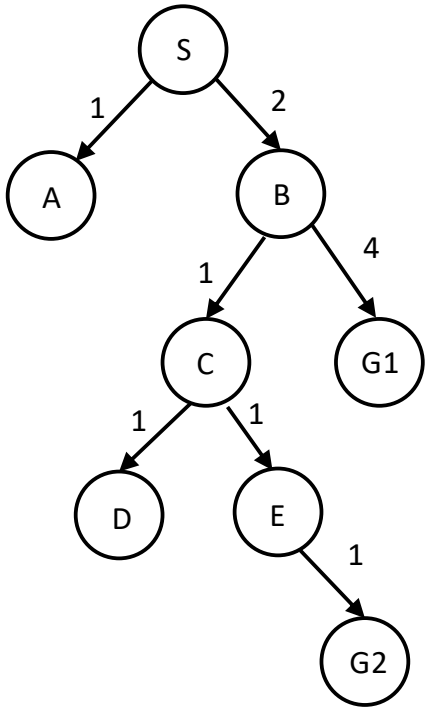
COMPSCI 370 **Instructor:** Ronald Parr

Homework 1 **Due:** Monday, February 6, 2023

The first five questions refer to the state space depicted by the following tree, which has costs shown next to the arcs. Two states, G1 and G2, satisfy the goal test.



For the questions below, we will ask you to show the frontier of the search tree just before search pops a node off the queue. Assume that nodes are popped in left to right order, so the node about to be popped is the leftmost in the list of nodes. Keep showing the state of the queue each time a node is about to be popped off until the search terminates.

1 BFS (5 points)

Show the queue for BFS, assuming the goal test is applied when a node is first generated. When two siblings are pushed onto the queue, assume the leftmost sibling is closer to the top. For example, the first two lines of your answer should look like this:

- S
- A, B

Note that given the assumptions in this question, goal nodes will not appear in the queue.

2 BFS II (5 points)

Show the queue for BFS, assuming the goal test is applied when a node is popped. When two siblings are pushed onto the queue, assume the leftmost sibling is closer to the top.

3 DFS (5 points)

Show the queue for DFS, assuming the goal test is applied when a node is first generated. When two siblings are pushed onto the queue, assume the leftmost sibling is closer to the top. Note that given the assumptions in this question, goal nodes will not appear in the queue.

4 DFS II (5 points)

Show the queue for DFS, assuming the goal test is applied when a node is popped. When two siblings are pushed onto the queue, assume the leftmost sibling is closer to the top.

5 UCS (5 points)

Show the queue for UCS. Recall that UCS always applies the goal test when a node is popped to ensure optimality. When two siblings are pushed onto the queue, assume the leftmost sibling is closer to the top. Since UCS uses a priority queue, list the elements of the queue as ordered pairs, in the form of (state,priority). For example, the first two lines of your solution should look like this:

- (s,0)
- (a,1), (b,2)

6 Almost Admissible Heuristics (10 points)

Suppose you have a heuristic h, which is almost but not quite admissible - it can overestimate the cost by at most ϵ. The solution found if you run A^* with this heuristic is no longer guaranteed to be optimal, but you can prove that, in the worst case, it will return a solution that is at most ϵ more expensive than optimal. Provide a proof for this claim.

7 Public Transportation (20 points)

Suppose you are formulating a search problem for public transportation instead of driving - for example, when you click the public transportation icon on Google Maps. In this case, the user's goals would typically be to minimize travel time or cost. Minimizing travel cost is pretty straightforward because the cost for using various forms of public transportation between different pick-up and drop-off points is well known.

Suppose the goal is to minimize travel *time*. We will consider two cases where the techniques we've developed so far in class can be helpful. To keep things simple, we'll assume that the start and goal are always known public transportation stop.

1. Assume that although public transportation has known and accurate travel times between stops, it does not have a clearly advertised schedule. For example, for a particular bus line, you might know all the travel times between stops, but only that a bus arrives every X minutes. a) How would you incorporate this information into the cost function so that the cost for any path is the expected total travel time? (5 points) b) What reasonable, non-trivial admissible heuristic be for this problem? Here, non-trivial means that it's not always a constant but also that it's significantly less expensive to compute than actually solving the problem exactly. Note that there's probably just one reasonable way to design the cost function, but many possible answers for the heuristic. (5 points)

2. Assume that public transportation runs on a fixed, clearly advertised schedule and is never late. In some ways, this makes things simpler, but it's now possible to compute the exact travel time. a) What is a good representation for the state? (This requires a bit more thought than the previous question.) (5 points) b) What is a good choice for a non-trivial, admissible heuristic. Here, non-trivial means that it's not always a constant but also that it's less expensive to compute than actually solving the problem exactly. Note that there's probably just one good way to represent the state, but there are many possible good answers for the heuristic. (5 points)

You may be wondering about the in-between case where there is a known schedule but a probability distribution over arrival times and travel times. This case requires more sophisticated techniques than basic search.

8 Consistency (10 points)

Assume h_1 and h_2 are both consistent heuristics. Is the average, $h_3(S) = \frac{h_1(S)+h_2(S)}{2}$ always consistent? Prove this or disprove it by example.