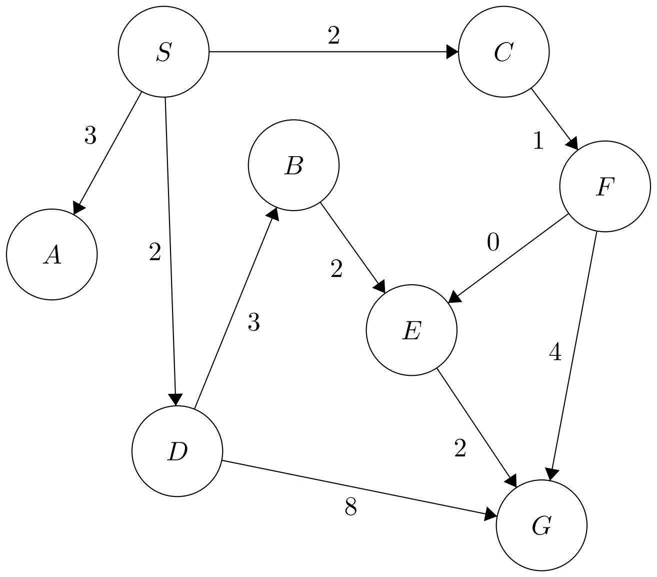
Ethics Warmup

Take a look at this description of RedZone, a navigation app that allows you to avoid high crime regions: https://www.androidauthority.com/redzone-navigation-app-avoids-high-crime-areas-686894/

1. If this app became commonly used, what impact could it have on businesses in the "red zone"?

2. Yes or No - would you want this application to roll out to your hometown?

1 Search Algorithms



Using each of the following graph search algorithms presented in lecture, write out the order in which nodes are added to the explored set, with start state S and goal state G. Break ties in alphabetical order by *the last state in the path.* Additionally, what is the path returned by each algorithm? What is the total cost of each path?

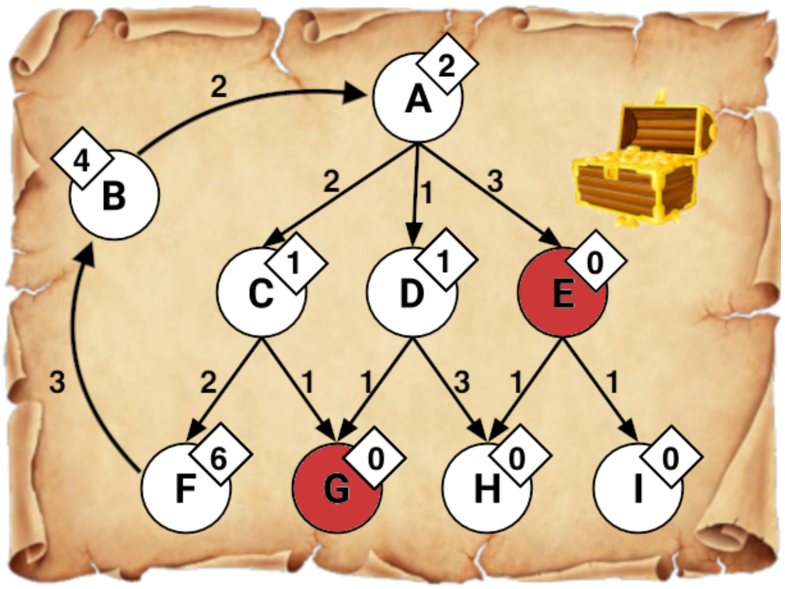
(a) Breadth-first

(b) Depth-first

(c) Iterative deepening

2 Treasure Hunting

We are lost at sea, trying to find a hidden treasure. We have a treasure map that tells us which paths we can take and approximately how far away the treasure is but it's not very accurate. We do know that the map will never overestimate the distance to the treasure. There is treasure on two different islands (but we only need to reach one of them).



A is the the island where we are currently and the shaded (red) states are the locations of the treasure. Arrows encode possible actions from each island, and numbers by the arrows represent action costs. Note that arrows are directed; for example, $A \rightarrow C$ is a valid action, but $C \rightarrow A$ is not. Numbers shown in diamonds are heuristic values that estimate the optimal (minimal) cost to get from that island to any treasure.

Run each of the following search algorithms with graph search and write down the nodes that are added to the explored set during the course of the search, as well as the final path returned and the corresponding cost of the final path, if applicable. When popping off of the frontier, assume that ties are broken alphabetically.

(a) **Depth-First Search**

Explored set:

Path returned:

(b) **Breadth-First Search**

Explored set:

Path returned:

(c) **Uniform-Cost Search**

Explored set:

Path returned and cost:

(d) **Greedy Search**

Explored set:

Path returned and cost:

(e) **A* Search**

Explored set:

Path returned and cost:

3 True/False Section

For each of the following questions, answer true or false and provide a brief explanation (or counterexample, if applicable).

(a) Depth-first search always expands at least as many nodes as A^* search with an admissible heuristic.

(b) Assume that for a single move, a rook can move any number of squares on a chessboard in a straight line, either vertically or horizontally, but cannot jump over other pieces. Manhattan distance is an admissible heuristic for the smallest number of moves to move the rook from square A to square B.

(c) Euclidean distance is an admissible heuristic for Pacman path-planning problems.

(d) The sum of several admissible heuristics is still an admissible heuristic.

(e) Admissibility of a heuristic for A^* search implies consistency as well.

(f) A^* with graph search is always optimal with an admissible heuristic.

For (g) and (h), consider an adversarial game tree where the root node is a maximizer, and the minimax value of the game (i.e., the value of the root node after running minimax search on the game tree) is V_M. Now, also consider an otherwise identical tree where every minimizer node is replaced with a chance node (with an arbitrary but known probability distribution). The expectimax value of the modified game tree is V_E.

(g) V_M is guaranteed to be less than or equal to V_E.

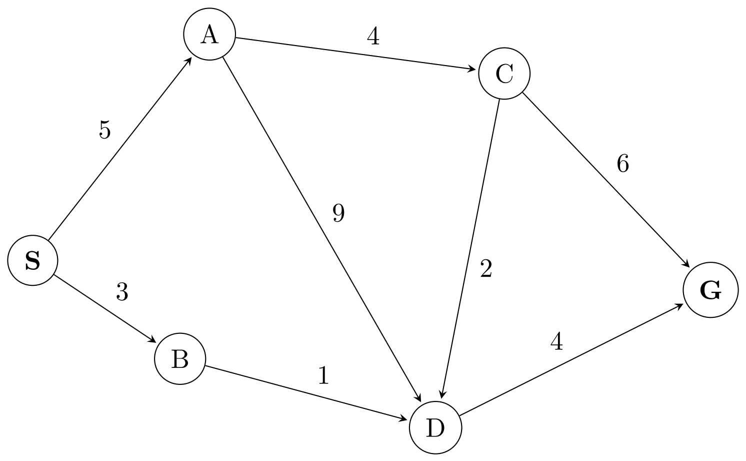
(h) Using the optimal minimax policy in the game corresponding to the modified (chance) game tree is guaranteed to result in a payoff of at least V_E.

4 Designing & Understanding Heuristics

Today, we will be taking a closer look at how the performance of A^* is affected by the heuristics it uses. To do this, we'll be using the graph below. You may have noticed that no heuristic values have been provided (*Recall:* What is A^* without heuristic values?). This is because we'll be working together to come up with heuristics ourselves!

In groups, design both an admissible heuristic and a consistent heuristic for the graph below by annotating each node with a heuristic value. (Note: you do NOT need to find a closed form way to represent the heuristic function.)

When you have completed your heuristics, work together to answer the questions below.



(a) Write down the path found by running A^* using your heuristic on the graph above.

(b) Work with your group to come up with a heuristic that's admissible but not consistent.

(c) (Bonus) Explain why a consistent heuristic must also be admissible. You may assume that the heuristic value at a goal node is always 0.