CSCI-404 Artificial Intelligence
Spring 2020, Homework Set 2

Due 02/13/2020, Thursday, 11:00AM

Question 1 Uninformed Search (20 points)

Consider the search tree shown in Figure 1. The number next to each edge is the cost to perform the action when the agent moves along that edge. Using graph search, *i.e.*, revisiting the same state is not allowed, please list the order in which nodes will be visited using:

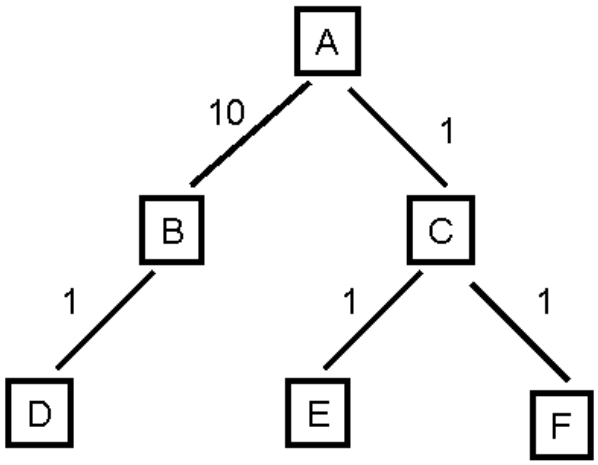


Fig. 1: An example search tree.

- breadth-first search (implemented by FIFO). C->B->F->E->D
- depth-first search (implemented by LIFO). C->E->F->B->D
- iterative deepening search. A->B->C->D->E->F
- uniform cost search (implemented by a priority queue). A>C->E->F->B->D

There may exist more than one solutions for each of the four search strategies due to randomness and uninformedness. Thus there could exist more than one correct answers for each question. You are not required to list all the correct answers. Instead, only one correct answer is required to get full credits.

Question 2 Search on Social Network Graphs (40 points)

A social network graph (SNG) is a graph where each vertex is a person and each edge represents an acquaintance. In other words, an SNG is a graph showing who knows who. For example, in the graph shown on Figure 2, George knows Mary and John, Mary knows Christine, Peter and George, John knows Christine, Helen and George, Christine knows Mary and John, Helen knows John, Peter knows Mary. The degrees of separation measure how closely connected two people are in the graph. For example, John has 0 degrees of separation from himself, 1 degree of separation from Christine, 2 degrees of separation from Mary, and 3 degrees of separation from Peter.

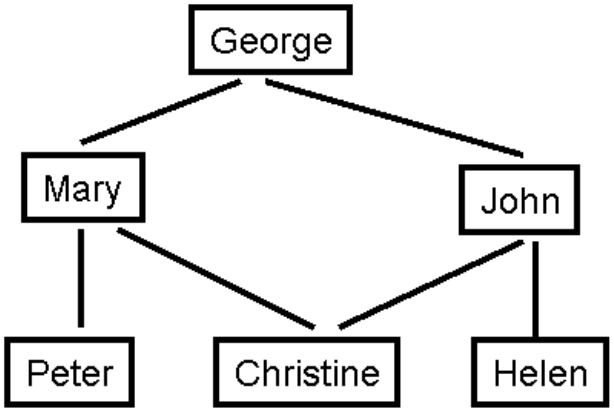


Fig. 2: A social network graph (SNG).

1. (10 points) From among breadth-first search, depth-first search, iterative deepening search, and uniform cost search, if tree search is being used, *i.e.*, there is no tracking on visited states, which one(s) guarantee finding the correct number of degrees of separation between any two people in the graph? BFS, DFS, and iterative deepening.
2. (10 points) For the SNG shown in Figure 2, draw the first three levels of the search tree, with John as the starting point (the first level of the tree is the root). Is there a one-to-one correspondence between nodes in the search tree and vertices in the SNG? Why, or why not? In your answer here, you should assume that the search algorithm does not try to avoid revisiting the same state. There is not, because the same state is visited twice on George and Chrstine who both know Mary
3. (10 points) Draw an SNG containing exactly 5 people, where at least two people have 4 degrees of separation between them.
4. (10 points) Draw an SNG containing exactly 5 people, where all people have 1 degree of separation between them.

Question 3 Admissible heuristics (10 points)

Suppose that we are given a road map of the United States, *i.e.*, we are given a list of cities and roads of the country, such that each road directly connects two cities. Additionally, we are given the distance from every city on the map to Chicago. Consider the following heuristic (for possible use with A* search): for each city A,

$$h(A) = \text{the distance from } A \text{ to Chicago} + \text{the distance from Chicago to the goal}.$$

Is this heuristic admissible? Justify your answer.

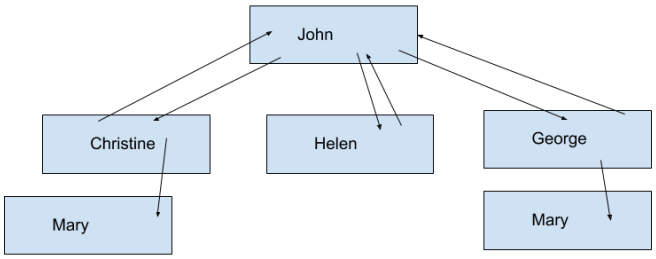
No, there are many cases in which you do not need to travel to Chicago in order to get the shortest distance. Admissible heurisitic requires us not overestimating the cost of reaching the goal, unlike this model.

Question 4 Informed Search (30 points)

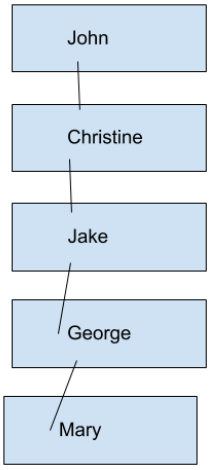
Figures 3 and Figure 4 show maps where all the towns are on a grid. Each town T has coordinates (T_i, T_j), where T_i and T_j are non-negative integers. We use the term "*Euclidean distance*" for the straight-line distance between two towns, and the term "*driving distance*" for the length of the shortest driving route connecting two towns. The only roads that exist connect towns that have Euclidean (straight-line) distance 1 from each other (however, there may be towns with Euclidean distance 1 from each other that are *NOT* directly connected by a road, for example in Figure 4).

Question 2 Drawings:

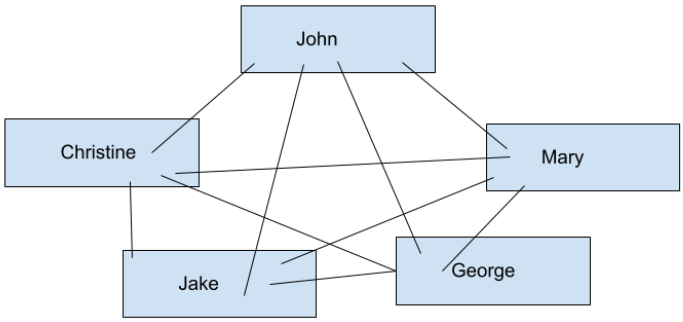
Part 2:



Part 3:



Part 4:



Consider greedy search, where the node to be expanded is always the one with the shortest Euclidean distance to the destination. Also consider A* search, where $h(n)$ is the Euclidean distance from n to the destination. In A* search, the next node is picked not based on $h(n)$ but based on $f(n) = g(n) + h(n)$. For each of the maps showing on Figures 3 and Figure 4, which of the following statements is true?

- Greedy search always performs better than or the same as A*.
- Greedy search always performs worse than or the same as A*.
- Greedy search performs sometimes better, sometimes worse, and sometimes the same as A*, depending on the start and end states.

Please justify your answer. For the purposes of this question, the performance of a search algorithm is simply measured by the number of nodes visited by that algorithm. Please provide answers for Figure 3 and Figure 4 separately.

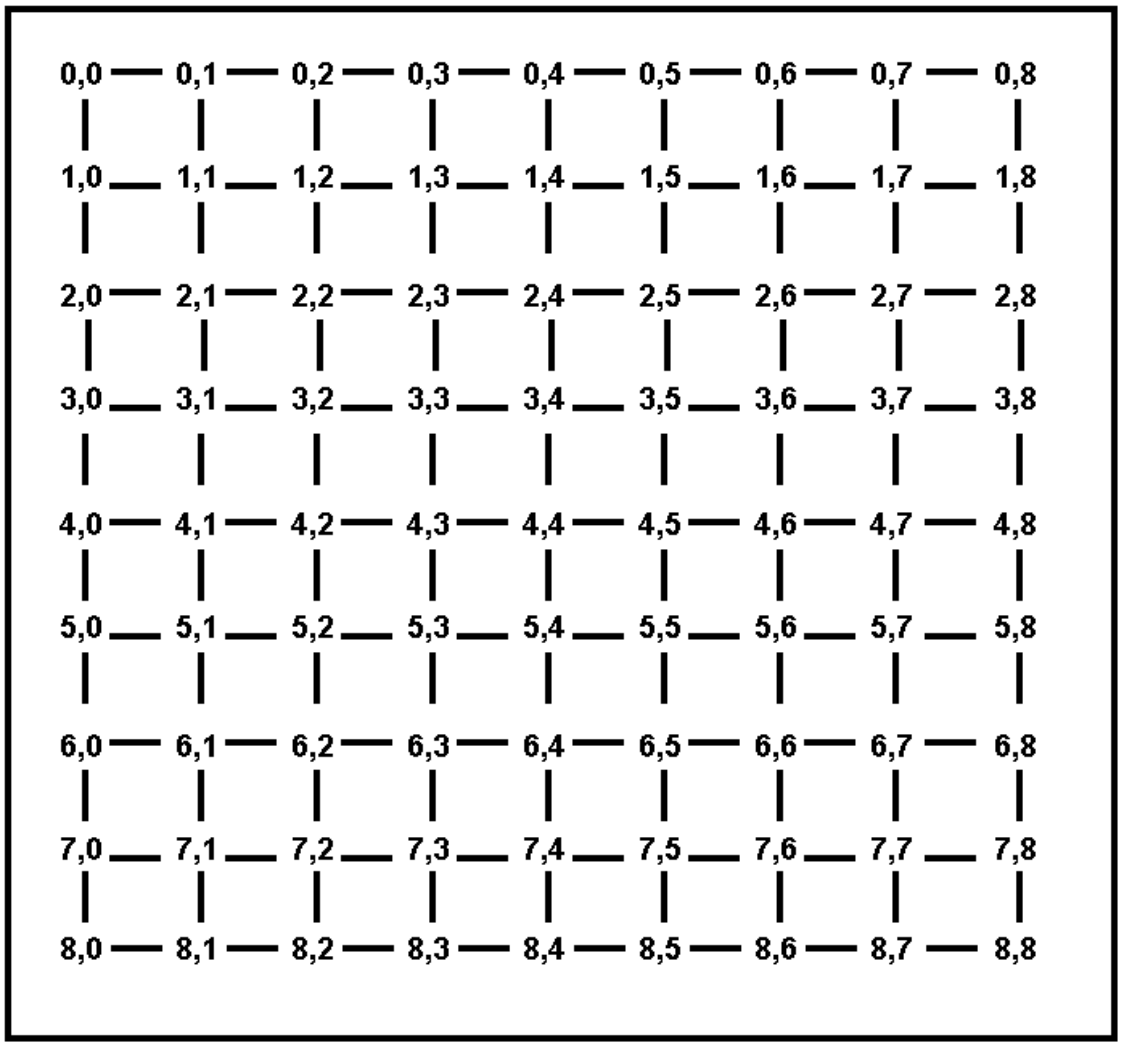


Fig. 3: A map of cities on a fully connected grid. Every city is simply named by its coordinates.

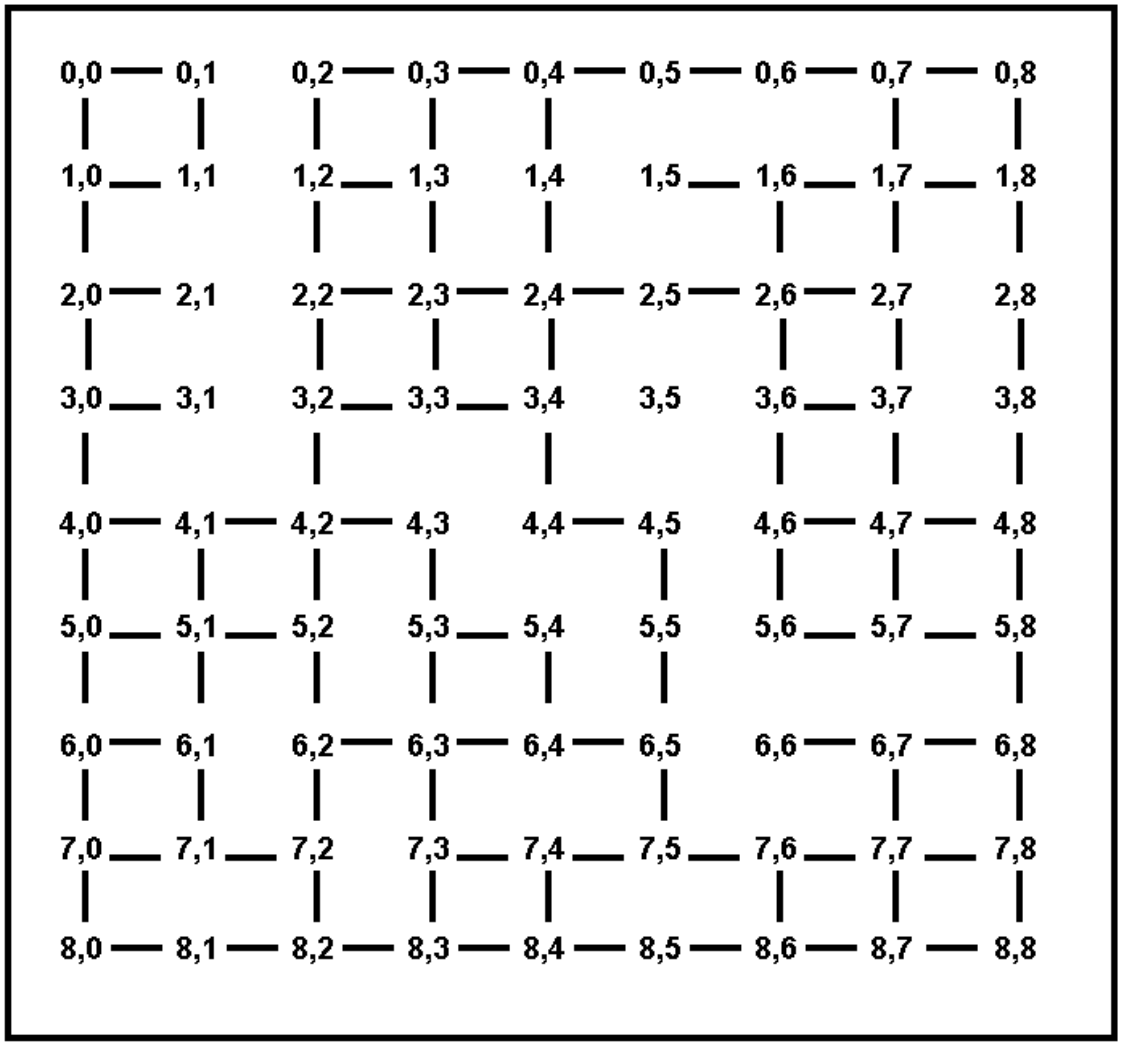


Fig. 4: A map of cities on a partially connected grid. Every city is simply named by its coordinates.

Question 4 Answers:

Consider greedy search, where the node to be expanded is always the one with the shortest Euclidean distance to the destination. Also consider A* search, where h(n) is the Euclidean distance from n to the destination. In A* search, the next node is picked not based on h(n) but based on f(n) = g(n) + h(n). For each of the maps showing on Figures 3 and Figure 4, which of the following statements is true?

Please justify your answer. For the purposes of this question, the performance of a search algorithm is simply measured by the number of nodes visited by that algorithm. Please provide answer for Figure 3 and Figure 4 separately.

Figure 3:

-Greedy search always performs better than or the same as A*.
True
Greedy and A* will do the same thing in this situation.
They will both expand towards the node that is the closest Euclidean distance away.

-Greedy search always performs worse than or the same as A*.
True
They will perform the same, because they visit the same nodes.

-Greedy search performs sometimes better, sometimes worse, and sometimes the same as A*, depending on the start and end stated
False, see other responses.

Figure 4:

-Greedy search always performs better than or the same as A*.
True, because A* visits more nodes and Greedy search is sometimes better in visited node performance when greedy fails to find a solution by getting stuck in a loop.

-Greedy search always performs worse than or the same as A*.
False, there is a case where greedy search gets caught in a loop and visits fewer nodes compared to A* which would visit more nodes by exiting said loop. An example of this is when searching from node: (6,8) to node: (6,4). Greedy search would go from (6,6) to (6,7) repeatedly and A* would break the loop.

-Greedy search performs sometimes better, sometimes worse, and sometimes the same as A*, depending on the start and end stated
False, greedy search will never visit fewer nodes