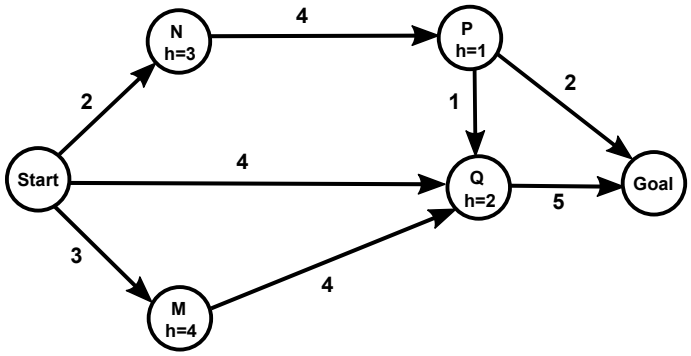
Homework 2

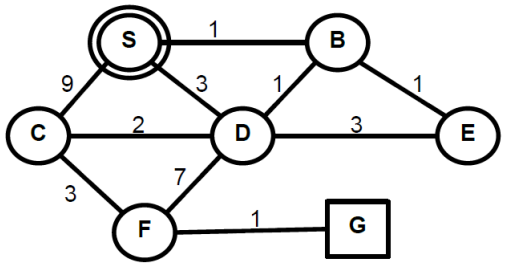
Due on Wednesday, October 5, 11:59 pm

NOTE: You should work individually for the following homework assignment. Please upload a pdf and a python file (.py) with your answers to the MOODLE.

1. **[25 points]** Compute the order in which states of the above graph are expanded and the returned path for each of these graph search algorithms: i) Depth-first search; ii) Breadth-first search; iii) Uniform cost search; iv) Greedy best first search using the heuristic h shown on the graph; and v) A* search using the same heuristic. Is the heuristic h(n) admissible? Explain why or why not.



2. **[10 points]** You are asked to compare different heuristics and to determine which, if any, dominate each other. You are executing search algorithms through this graph (i.e., you do not remember previously visited nodes). The start node is S, and the goal node is G. Actual step costs are shown next to each link. Heuristics are given in the following table. h* (n) is the optimal or true heuristic, and h1, h2, h3 are various other heuristics.



Heuristic Table

State	h1(n)	h2(n)	h3(n)	h*(n)
S	5	5	5	8
B	4	4	5	7
C	5	3	2	4
D	3	1	4	6
E	4	3	4	8
F	1	0	1	1
G	0	0	0	0

(a) **[5 points]** Which heuristic functions are admissible among h1, h2, h3?

(b) **[5 points]** Which heuristic functions are consistent among h1, h2, h3?

3. **[15 points]** You are designing a menu for a special event. There are several choices, each represented as a variable: (A)ppetizer, (B)everage, main (C)ourse, and (D)essert. The domains of the variables are as follows:

   A: (v)eggies, (e)scargot

   B: (w)ater, (s)oda, (m)ilk

   C: (f)ish, (b)eef, (p)asta

   D: (a)pple pie, (i)ce cream, (ch)eese

   Because all of your guests get the same menu, it must obey the following dietary constraints:

   (i) Vegetarian options: The appetizer must be veggies or the main course must be pasta or fish (or both).

   (ii) Total budget: If you serve the escargot, you cannot afford any beverage other than water.

   (iii) Calcium requirement: You must serve at least one of milk, ice cream, or cheese.

   (a) **[4 points]** Draw the constraint graph over the variables A, B, C, and D.

   (b) **[4 points]** Imagine we first assign A=e. Cross out eliminated values to show the domains of the variables after forward checking.

   (c) **[4 points]** Again imagine we first assign A=e. Cross out eliminated values to show the domains of the variables after arc consistency has been enforced.

   (d) **[3 points]** Give a solution for this CSP or state that none exists.

4. **[5 points]** In your own words, explain why it is a good heuristic in a CSP search to choose the variable that is the **most** constrained but the value that is **least** constraining.

5. **[10 points] Programming Assignment Part I:** This assignment is a continuation of the assignment of the previous homework that requires you to install https://github.com/aimacode/aima-python.

   (a) Run the file

   `search.py`

   in the interactive mode as

   `python3 -i search.py`

   (b) Run the following lines of code in the interactive command line:

   ```
   >>> compare_graph_searchers()
   ```

   Take a screenshot of this. Explain the output of this code.

   (c) Run the following lines:

   ```
   >>> puzzle = EightPuzzle((2, 4, 3, 1, 5, 6, 7, 8, 0))
   >>> puzzle.check_solvability((2, 4, 3, 1, 5, 6, 7, 8, 0))
   >>> astar_search(puzzle).solution()
   ```

Take a screenshot of the output and explain it. How is the puzzle solved? What heuristic is used? How can you change the heuristic in the code?

6. **[35 points] Programming Assignment Part II:** In this part of the programming assignment, your task is to implement search algorithms and use them in solving maze problems. More concretely, you will program an agent to find a path through the maze and reach the exit. You can take advantage of some Python implementations such as: https://www.redblobgames.com/pathfinding/a-star/implementation.html, but make sure you cite all your sources as comments in your code.

   Your program will solve the problem of finding the shortest path given an initial start state and one goal state. The maze layout will be provided as a simple text file, in which '%' means obstacles, 'S' is the starting position, and 'G' the goal location. See the sample mazes files uploaded with the homework for an illustration.

   The agent can move in one of four directions: North, West, South, and East.

   You should implement the state representation, transition model, and goal test necessary to solve the problem. Then, you should implement the following search algorithms that were covered in class and are in the textbook:

   (a) Depth-first search
   (b) Greedy best-first search
   (c) A* search

   Your program should run using Python 3. Your code can only import extra modules, but only if they are part of the standard python library. You should upload your source code (**.py file only**) along with your homework solution.

   For this part of the assignment, you will use the Euclidean distance from the current position to the goal as the heuristic function for A* search.

   For each maze, your solution should include an output (this can be shown in the command line), the solution cost, and the number of nodes expanded in your search.

   Your program should run as follows:

   ```
   python3 search.py --method astar maze.txt
   ```

   where astar is the name of the method, which can be one of depth, greedy, or astar, and maze.txt is the input file. We will supply three mazes that you can use to test your program, but you can test other mazes.

   Please describe the algorithms and data structures that you used for the implementation of all three search strategies. Answer the following questions in your solution:

   - What is a state?
   - What is a node?
   - Are they the same or different in your implementations?
   - What is the frontier?
   - Do you maintain an explored states list?
   - How are repeated states detected and managed?