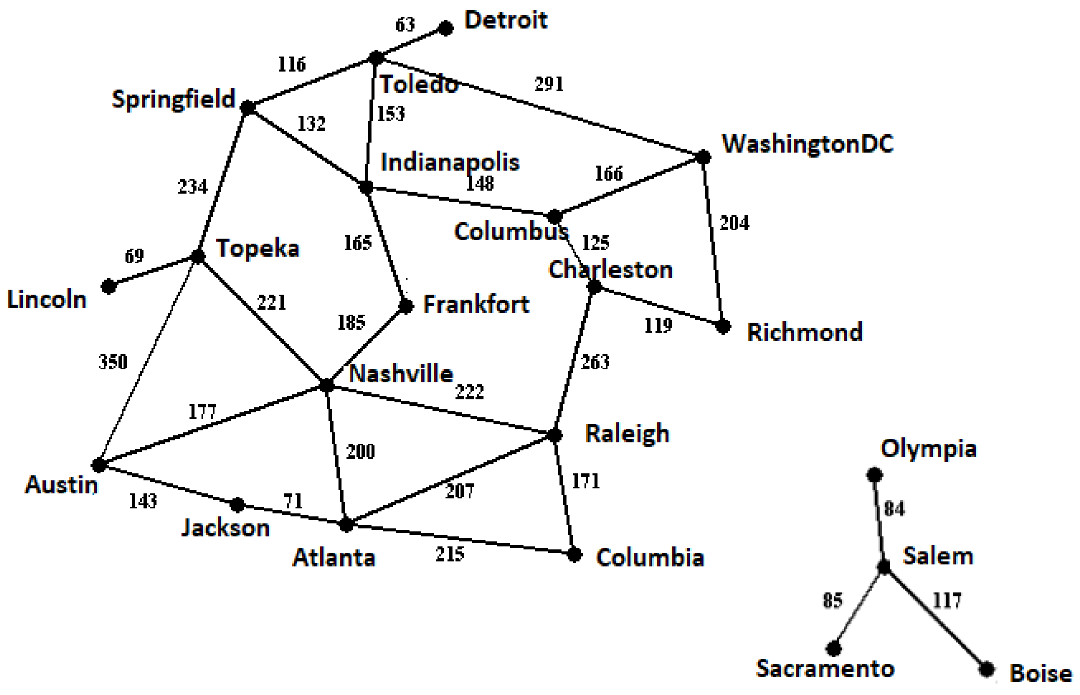
EECS 3401 – Project1 [20 Points]

Due by 24th October 11:59pm EST

USA Road Map



For this project, you must implement four search algorithms (UCS, BFS, DFS & A-Star Search) using Python, that can find a path between any two locations on the USA map. The pseudocodes for these search algorithms can be found on this link https://github.com/aimacode/aima-pseudocode.

Your program (python file) will be named find_path.py, and will take the command line arguments as follows:

find_path.py ucs/bfs/dfs/astar input_file.txt point_of_origin point_of_destination heuristic_file.txt

In the above command line the flag ucs, bfs, dfs and astar is used to select uniformed and informed search strategies.

File input_file.txt is the text file that contains input information that describes the path between different cities of the world. In the input_file1.txt file the map which is shown in the above image has been described. Each line of the input_file1.txt has three items location1(point of origin), location2(point of destination), and edge cost (distance in miles). The file concludes the input in input_file1.txt by the word "END".

Usage of command line: find_path.py ucs input_file1.txt Columbia WashingtonDC

The input_file1.txt is just a single test case on which you can test your code. The project will be executed on different test cases while grading to check if the implementation is correct. Your project should be able to execute as shown in the usage of command line.

The implementation should output distance between point of origin and point of destination, and a list of all locations on the path.

Sample Input & Output:

Input-1:

find_path.py ucs input_file1.txt Columbia WashingtonDC

Output-1:

distance: 725 mi

path:

Columbia to Raleigh: 171 mi

Raleigh to Charleston: 263 mi

Charleston to Columbus: 125 mi

Columbus to WashingtonDC: 166 mi

Input-2:

find_path.py ucs input_file1.txt Boise Nashville

Output-2:

distance: infinity

path:
none

When we want to make use of astar search as in sample input-2, we have an extra argument heuristic_Frankfort.txt. This file contains the heuristic values for every location with respect to given point of destination as different points of destinations need different heuristic values. So, the heuristic_Frankfort.txt file contains a heuristic value for every location assuming Frankfort location as the point of destination. Your code implementation should make use of this file to reduce the number of nodes it ends up expanding.

Usage of command line: find_path.py astar input_file1.txt Columbia Frankfort heuristic_Frankfort.txt

Make sure your code is well commented.

Submission Instructions:

- Your submission should be a zipped file, having the name Firstname_Lastname_studentid_Project1.zip