Mimir Search Daniel Engbert

Home CMSC 471 Sp17 Project 1: Graph Searches

Project 1: Graph Searches Active Open Workspace Submit Files

Project 1: Graph Searches

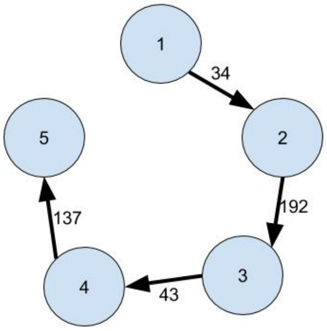
For this assignment you will be implementing three different graph searches in Python 3. The searches you are required to implement are Breadth First Search (BFS), Depth First Search (DFS), and Uniform Cost Search). Each of these searches are to be **implemented for a directed graph.** The input graph will be given to you in a file that describes the edges and weights in the graph, you are free to use any data structure that you see fit. You may not use any built in python modules that implement these graph searches for you.

Graph Input

The directed graph will be given to you as a file that looks something like this:

```
1 2 34
2 3 192
3 4 43
4 5 137
```

The first two numbers on a line represent an edge connecting those two vertices in this graph. The first number is that edge's source, and the second number is that edge's destination. The third number on a line represents the weight of that edge. This edge is directed from the source to the destination. **All three numbers are guaranteed to be integers.** The graph described above look like this:



Running Your Code

Your file name must be `Search.py`. It will be given several command line arguments when tested:

```
python3 Search.py input_file start_node end_node search_type
```

`input_file` is a path to the edge list storing the graph.
`start_node` and `end_node` are the integer names of the start and end node of the search.
`search_type`: used to determine what search to use in a single run ('BFS' or 'DFS' or 'UCS')

Your output should just be a list of the path that was found. For the example above the command:

```
python Search.py example.txt 1 3 BFS
```

Would return:

```
[1, 2, 3]
```

Assumptions:

- You will not have to deal with a multi graph. No pair of nodes will have more than one edge between them.
- If you explore the entire graph and still cannot find a path, simply print an empty list.
- The graphs WILL have cycles, so you'll have to keep track of every node you've visited.
- There will be no bad inputs. All files and command line args used to test are formatted correctly.
- BFS, DFS: nodes should be added to in numerical order to your expansion data structure.

Assigned on 2/06/2017 4:41:00 AM | Due on 2/20/2017 11:59:00 PM 5000 Submissions Remaining

Looks like you don't have any submissions yet

Click submit to submit your code!

Submit Open IDE