CMPS 2200 Recitation 09

Here are the slides we would use. Just for your reference.

BFS: https://github.com/allan-tulane/cmps2200-slides/blob/main/module-08-graph/graph-merged.ipynb

Shortest Path: https://github.com/allan-tulane/cmps2200-slides/blob/main/module-08-graph/graph-04.ipynb

1. Shortest shortest paths

Suppose we are given a a directed, **weighted** graph $G = (V, E)$ with only positive edge weights. For a source vertex s, design an algorithm to find the shortest path from s to all other vertices with the fewest number of edges. That is, if there are multiple paths with the same total edge weight, output the one with the fewest number of edges.

Complete the function `shortest_shortest_path` and test with the example graph given in `test_shortest_shortest_path`. Note that the `shortest_shortest_path` function returns both the weight and the number of edges of each shortest path.

2. Computing paths

a) We have seen how to run breadth-first search while keeping track of the distance of each node to the source. Let's now keep track of the actual shortest path from the source to each node. First, observe that the order in which BFS visits nodes implies a tree over the graph:

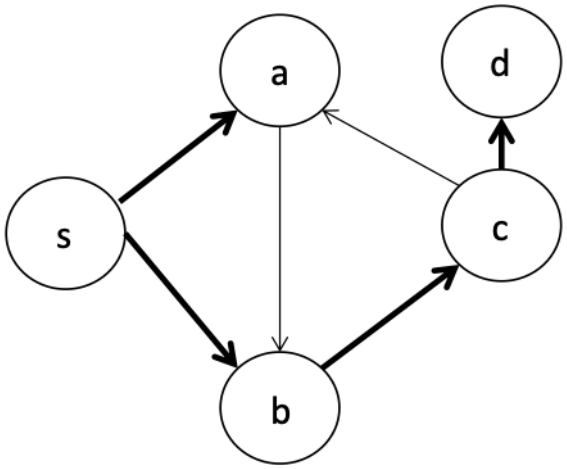

Figure 1: bfs.png

Here, the dark edges indicate all the shortest paths discovered by BFS. To keep track of the paths, then, we just need to represent this tree. To do so, we can store a `dict` from a vertex to its parent in the tree. In the above example, this would be:

```
{'a': 's', 'b': 's', 'c': 'b', 'd': 'c'}
```

Complete the `bfs_path` function to return this parent `dict` and test it with `test_bfs_path`. Your algorithm should not increase the asymptotic work/span of BFS.

b) Next, complete `get_path`, which takes in the parent `dict` and a node, and returns a string indicating the path from the source node to the destination node. Test with `test_get_path`.