Algorithms HW4

Blake Williams

010974710

1/26/2024

For the BFS algorithm we initialize the bfs with a declaration void bfs(Graph &G, int start, int destination). Then we utilize a Queue to store the nodes that need to be visited by calling Queue<int> queue, we then reset the graph state by doing G.reset(). Then add the start node to the queue by doing queue.push(start) starting the BFS. We then mark the start node so we do not revisit it again, then start the trace. A while loop is added to run continuously as long as there are no more nodes left to visit. Then we store the number of neighbors the current node has into the numberOfAdjacencyNodes variable, then get a pointer to the first neighbor in the list. The for loop itterates through each neighbor, updating the pointer to move through the list. For each neighbor v of us if its not visited it checks if v has not been visited. Then marks v as visited, then sets a trace path from u to v, then adds v to the queue.

The DFS algorithm operates has the same components but utilizes a different data structure. Most notably the change from a Queue to a stack.

Finally the RDFS. This uses Recursion to find the end node instead of storing. If a neighbor is not visited it starts the trace, then does a recursive call to itself to move to the next node.

BFS TEST:

DFS TEST:

RDFS TEST:

