Niall Sauvage

20334203

Final Project Design Document

CSU22012 – Algorithms and data structures ii

# 1)

“Finding shortest paths between 2 bus stops (as input by the user), returning the list of stops en route as well as the associated ‘cost’.”

In order to efficiently complete this task, I decided to use Dijkstra’s algorithm, as it is not only relatively simple and easy to implement, but it will find the shortest path between two vertices whereas Floyd-Warshall uses a cost matrix from every vertex to every other vertex. In order to represent my graph, I am using an adjacency list, which is an array of size V (where V is the count of vertices in the graph) with each entry representing a unique vertex. The array itself stores Linked Lists at every entry, which themselves store edge objects, representing the edges leading out from that vertex. Additionally, for convenience, I created a LinkedList which stores all stop (vertex) objects in the graph. I also used a hashmap so that I could look up a stop ID and have a stop object returned in 0(1) time. Finally, to represent the path itself, I used two arrays of size V, one which stored doubles called distTo[] and the other which stored stop objects called prevTo[]. distTo[v] stores the cost of the shortest path to v from the stop we’re starting from, while prevTo[v] stores the stop previous to v along the shortest path from the source. By simply following this back and then reversing it, we can find the shortest path to v.