Ranjan Behl

ECE 368 Project 3 Milestone 1

This project is very different from the last two since this project is based upon graphs a concept that I am completely foreign to compared to the last two projects which was just sorting and Huffman both of which I touched upon in ECE 264. In this project we are asked to be able to traverse graphs by using of the famous shortest path algorithms, in this case the algorithm is the Dijkstra's algorithm. The project's most important part is essentially using Dijkstra's algorithm to traverse a graph and find the shortest path between two given nodes at any point of time. Thus, the first thing I plan to do is learn Dijkstra's algorithm and make sure I am able to understand the pseudo code that is posted on Wikipedia. At the same time, I plan to improve my knowledge regarding graph theory. Once I understand Dijkstra's algorithm fully, I will then try writing its pseudo code based on the parameters given to us in the project. Once I am sure that I have got the algorithm and the pseudo code right, I will then start working on the other parts of the project.

For the parsing and reading the of the input file, I plan to use a either a multilinked linked list or a 2d array by storing all the relevant information for easy and practical access. If I go with the multilinked linked list, then each node of this linked list will store the x and y co-ordinates of the corresponding vertex and will contain pointers to all the neighbors of the corresponding vertex. Doing it in such a way will make implement the Dijkstra's algorithm much easier.

Now I will try to explain my understanding of Dijikstra's shortest path algorithm. In this algorithm we start by taking a vertex and assign weights to the distances between the current vertex and all the other vertices, which is infinity when we start. Now, we proceed further by examining the neighbors of the current node and find their distances from the current node. We then move to the one with the shortest distance from the current node and make this node the current node. We then keep travel of the nodes visited and we shouldn't revisit a node. We will repeat this process until either we finally reach the node that we wanted to or the distance from the current node to our destination node is infinity, in which case we go back to our starting node, choose an alternative neighbor and repeat the whole process.