ECE368 Project 3

Milestone 1

Xiao Xiao

The main purpose of this project is to implement Dijkstra’s shortest path algorithm for weighted undirected graphs. There are two input files that will be used to implement the algorithm. The first input file will represent a map, which is an undirected graph whose vertices are points on a plane and are connected by edges whose weights are Euclidean distances. The second input file contains a list of search queries with the first line containing the number of such queries and each of the following lines containing one query in the form of a source vertex and destination vertex pair. The goal is to complete the shortest path from each source listed in the query file to the corresponding destination using Dijkstra’s algorithm with the given input files.

Parse Input file

The first thing is to establish an adjacency list which will store the total number of points and the corresponding edges related to those points. The Euclidean distance will be calculated to determine the distance between two neighbor points. All these information will be stored in the adjacency list. The second part is to parse the queries file. After reading the starting point and the destination, the following part is to implement the Dijkstra’s algorithm to find the shortest path.

Find the shortest path

1. Assign to every node a tentative distance value. Set zero to initial point and infinity for all other nodes.
2. Set the initial node as current. Mark all other nodes unvisited. Create a set of all the unvisited nodes called the unvisited set.
3. For the current node, consider all of its unvisited neighbors and calculate the tentative distances. Compare the newly calculated tentative distance to the current assigned values and assign the smaller one.
4. When finished considering all of the neighbors of the current node, mark the current node as visited and remove it from unvisited set. A visited node will never be checked again.
5. If the destination node has been marked visited or if the smallest tentative distance among the nodes in the *unvisited set* is infinity (when planning a complete traversal; occurs when there is no connection between the initial node and remaining unvisited nodes), then stop. The algorithm has finished.
6. Otherwise, select the unvisited node that is marked with the smallest tentative distance, set it as the new "current node", and go back to step 3.