Project 3

Dijkstra’s Algorithm

CS241-01

Junda Lou

[lou@cpp.edu](mailto:lou@cpp.edu), [loudakkk@hotmail.com](mailto:loudakkk@hotmail.com)

3.9.2017

Project Description

In this project, we are asked to implement Dijkstra’s Algorithm. Besides it, we also need to implement a graph by ourselves. We have to put what we have learned academically into physical codes, and it also require us to use certain data structures that we have already learned in CS240.

Project Specification

A graph implementation can be based on two data structures: Array and Linked List. Each one has their own advantages and disadvantages. I choose Array as my data structure of the graph because it has only O(1) operation on the edges, which is the main part of this project. Secondly, Dijkstra’s Algorithm requires two Sets to store vertices from the graph, and it need to find the next vertex with the smallest relax value. A Priority Queue is a good choice here, but what I used is a Set from the Java internal library. I have to figure out my own method to find the next node, which takes O(n) time complexity because it has to iterate the Set to find the smallest value in it.

Testing methodology

It is the most difficult part of the project. The city and road data is too huge to put it on a hard copy. All I can do is trying to make sure every line of code of the algorithm is correct. When I was debugging it, I trace the prev[], relax[] and two Sets specifically because they are very important, and most of my errors are mostly small misses on the if conditions. Also, reading in the data is another big challenge, which I did not expect. When I tried split(“ “), it only ignore 1 space and leave the others. When I did .next(), it ignore the spaces, but some of the city names have space within themselves. Finally, I successfully split the data into different arrays.

Lessons Learned

Time complexity is a big problem when we are doing projects like this. This can be viewed as a mini-map, which is important to find the shortest path in the minimum amount of time. When the database gets bigger, it makes the problem even bigger. If it takes an hour to find the shortest path from one point to another, no one will use your program.