Report for Programming Problem 3 – Bike Lanes

Team: 2018283497_2015235572

Student ID: 2015235572 - José Miguel Saraiva Monteiro
Student ID: 2018283497 - Daniel Roque Pereira

1. Algorithm description

The approach used for this problem was to divide it in two parts, or subproblems using Tarjan's and Kruskal's algorithms.
Some speed-up tricks were used such as only executing Kruskal's algorithm when the number of questions was greater than two or when the strong connected components formed a circuit(more than 1 POI). Since the first two questions only rely on Tarjan's algorithm outcome, there is no need to waste processing time doing unnecessary calculations. Also on the Tarjan algorithm,since there is a recursive call, the data structures (vectors) were passed as reference to avoid excessive memory usage. In Tarjan's algorithm,stacks were used in order to achieve better performance when compared to queues on recursive calls.

1.1. Circuit identification

The first subproblem consists in finding *Strongly Connected Components* which answers the first and second question, using the Tarjan algorithm which generates multiple sets of points of interest from which it is possible to determine the total circuit number (number of sets of POIs) and the largest circuit possible (set with the (greatest POIs number).

1.2. Selection the streets for the bike lanes

The second subproblem was solved using Kruskal's algorithm to answer the last two questions, by finding the minimum spanning trees of the many circuits and from there calculate the longest bike lane, and the total bike lane length to be built(sum of the spanning trees of the strongly connected components).

2. Data structures

The data structures used were a global matrix with the vertex connections, where *X* and *Y* are the vertices, and the value on the index [*X*][*Y*] is the connection distance between the two, starting on point *X* and ending on point *Y*. A vector with strongly connected components was also used to save the multiple circuits, which consists in a vector of vectors in which every "subvector" is a POI circuit.

The Kruskal algorithm uses a vector of pairs which is used to store elements of type <pair<int,pair<int,int>> which can be translated into < distance , <u,v>> where u and v are the coordinates in the matrix.This server as a way of having fast access to any graph point.

3. Correctness

Considering the fact that both algorithms being used have been verified globally to be correct, in finding both the strongly connected components for Tarjan's and for finding the minimum spanning trees, correctly implementing the pseudo code that was given on the class slides, and adapting it to the data structures used, should give the expected right answer. This is true for Trajan's algorithm to find the total number of circuits, as well as the one with the most points of interest. Regarding Kruskal's algorithm, it gives the minimum spanning tree for a given set of vertices, in this case, the points of interest. Executing this algorithm for every possible circuit allows the calculation of the last two questions, which are the longest lane in the entire city to be built, and the total length of bike lanes the city will build obtained by adding every bike lane in every circuit.

4. Algorithm Analysis

Regarding time complexity, Tarjan's algorithm worst case performance is $O(|V| + |E|)$ while Kruskal's worst case performance is $O(E \log V)$, where V is the vertex number and E is the number of edges. The implementation done, the worst case takes $O(|V| + |E|)$ time plus $C * O(E \log V)$ since the Kruskal algorithm is run multiple times for the C circuits found.

The best case scenario in terms of performance would be if the number of questions is equal or less than two, so only the Tarjan's algorithm is run, so the program would take $O(|V| + |E|)$ time to execute.

5. References

- Class slides *Week 10* and *Week 11*