CARPOOLING ALGORITHM FOR DECREASE THE ENVIRONMENTAL IMPACT.

Felipe Sosa Patiño
Universidad Eafit
Colombia
fsosap@eafit.edu.co

Camila Barona Cabrera
Universidad Eafit
Colombia
cbaronac@eafit.edu.co

Mauricio Toro
Universidad Eafit
Colombia
mtorobe@eafit.edu.co

ABSTRACT
The problem is the traffic at the streets, because some people use their cars alone, then the number of cars is very big, and the city get overload of cars, their noise and their contamination. This situation has to be solved because the Medellin city has constant alarms for big levels of contamination and pollution at the air. Besides the respiratory diseases that the contamination can generate, other aspects of the quality of life are affected: the people get late to their jobs, to their schools or college, and get late to other important dates. This is because the public transport is usually very crowded and uncomfortable.

1. INTRODUCTION
The arrival of the personal cars and motorcycles made easier the way the people moves from one place to another, but the overcrowding of that vehicles at actual times, has been an obstacle for the mobility through the city, because in most cases every person has his own car and uses it alone.

2. PROBLEM
Each person at his or her car or motorcycle in a big city of almost 2'530.000 people, makes a chaos at the streets, because there are some accidents starred in most of the times by motorcycles because those vehicles are less balanced than the cars. Those accidents produce mobility stagnation and a lot of stress on people.

3. RELATED WORK
For the current report, an investigation was conducted of different data structures that promoted to give an effective solution to the problem posed.

3.1 Muddy City
This problem is based on enough streets must be paved so it is possible to travel from any house to any other house, possibly via other houses. The paving should be done at a minimum cost. For that, a possible solution is **minimum spanning tree (MST)**

The MST is a subset of the edges of a connected, edge-weighted undirected graph that connects all the vertices together, without any cycles and with the minimum possible total edge weight. That is, it is a spanning tree whose sum of edge weights is as small as possible.

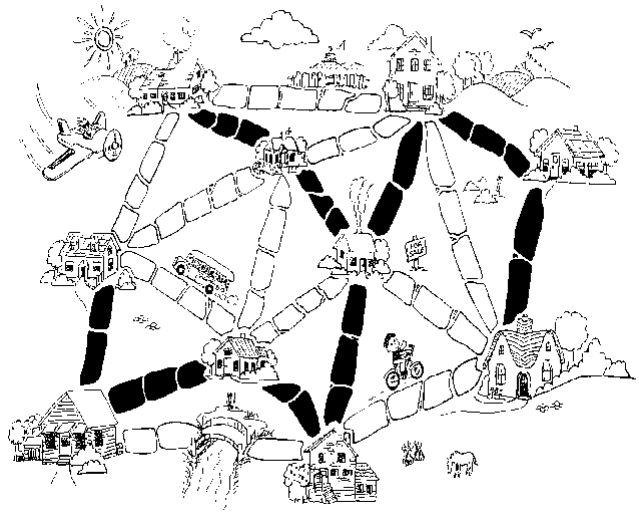

Figure 1.

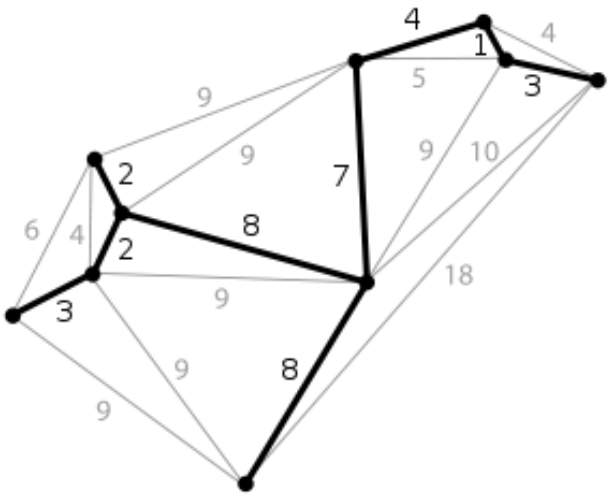

Figure 2.

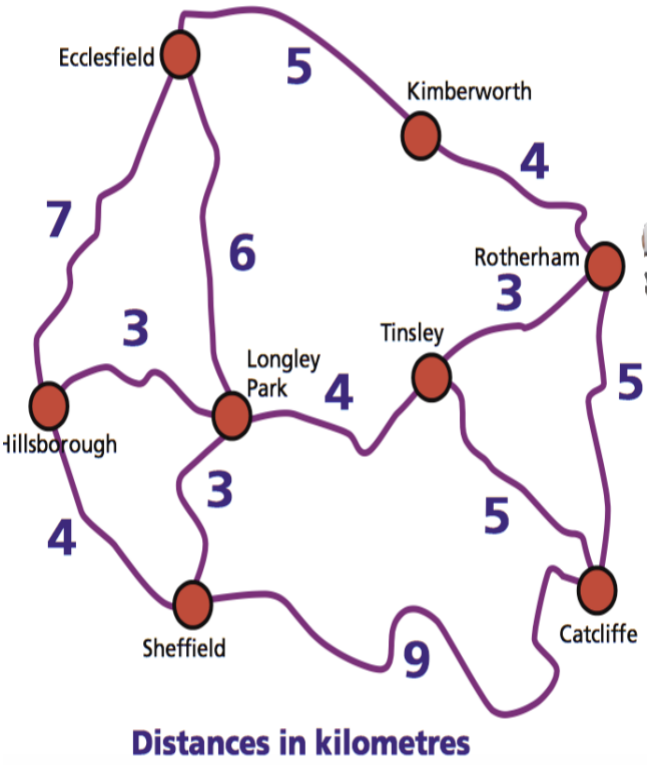

Figure 3.

3.2 Deliveries problem

This problem is based on find the best route to make deliveries to all the places marked in the map, starting from a special company saved time and money the shortest route may be found by trial and improvement methods, or by considering all the possible routes.

Prim's Algorithm is a possible solution of this problem. Is a greedy algorithm that fins a minimum spanning tree for a weighted undirected graph. The algorithm operates by building this tree one vertex at a time, from an arbitrary starting vertex, at each step adding the cheapest possible connection from the tree to another vertex.

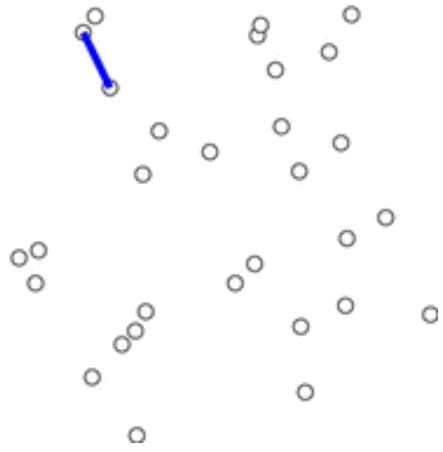

Figure 4.

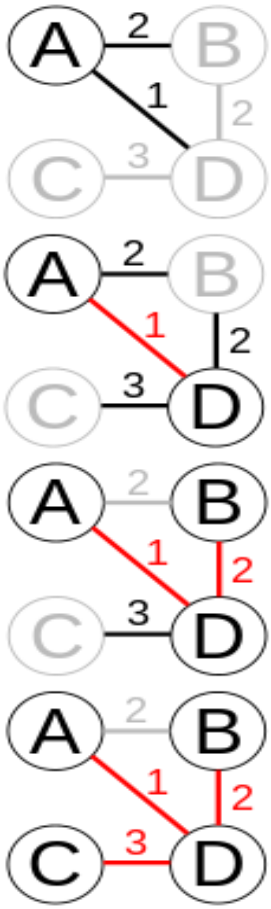

Figure 5.

3.3 Network design problem

This problem is about how a company with different offices can connect a phone lines with each other with the minimum total cost.

One of the many solutions of this problem is **Kruskal algorithm:** is an algorithm of graph theory to find a minimum covering tree in a connected and weighted grapdelih. It looks for a subset of edges that forming a tree, include all vertices and where the value of the sum of all the edges of the tree is the minimum. If the If the graph is not connected, then look for a minimum expanded forest (a minimum expanded tree for each related component).

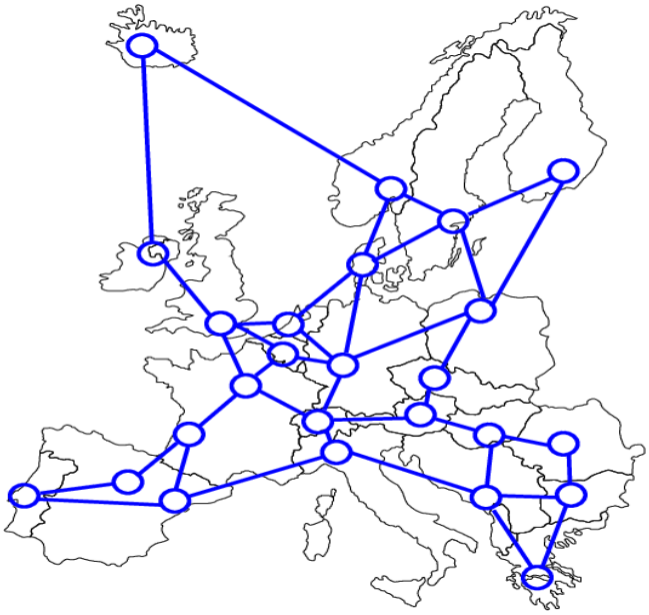

Figure 6.

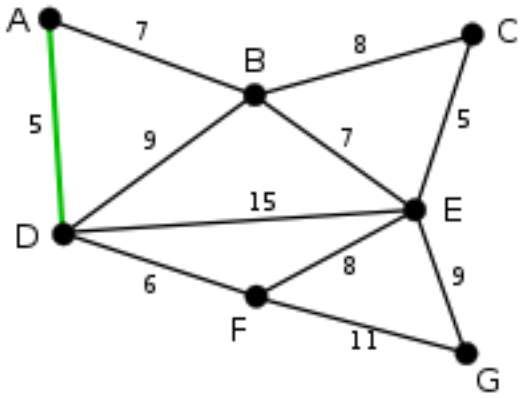

Figure 7.

3.4 Traveling salesman problem

The problem of the traveling salesman, problem of the traveling salesman, problem of the traveling agent or traveling salesman problem (TSP) answers the following question: given a list of cities and the distances between each pair of them, which is the shortest route possible that visits each city exactly once and at the end returns to the origin city?

One of the solutions for this problem is: **Boruvka algorithm**, The algorithm begins by examining each vertex and adding the lower weight arc from that vertex to another one in the graph, without taking into account the already added arcs, and continues uniting these groups in the same way until a tree that covers all the vertices.

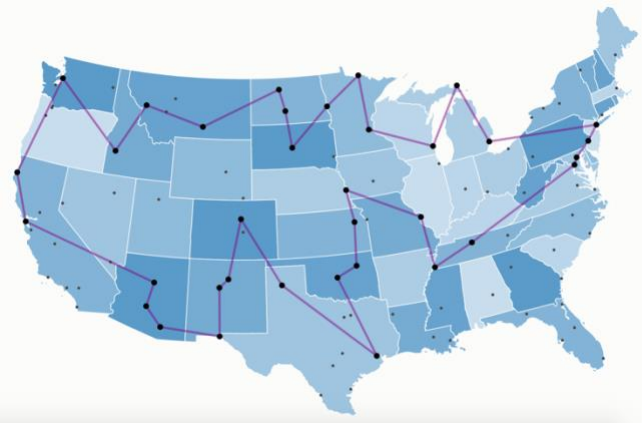

Figure 8.

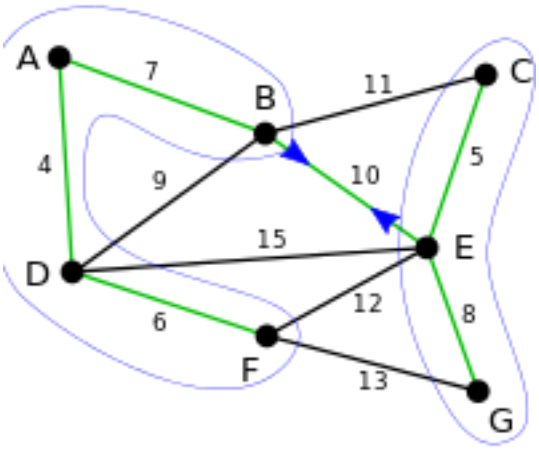

Figure 9.

REFERENCES

1.Minimum spanning tree,2019. Consulted on march 2 of 2019, Wikipedia, The free encyclopedia. Recovered of: https://en.wikipedia.org/wiki/Minimum_spanning_tree

2. Toy problems for real world. Consulted on march 2 of 2019. Tenderfoot: Unit 2. Recovered of: https://www.computingatschool.org.uk/data/tft/02/04Activity_Toy_Problems_Real_World.pdf

3.Prim's algorithm,2019. Consulted on march 2 of 2019, Wikipedia, The free encyclopedia. Recovered of: https://en.wikipedia.org/wiki/Prim%27s_algorithm

4. Kruskal algorithm,2019. Consulted on march 2 of 2019, Wikipedia, The free encyclopedia. Recovered of: https://es.wikipedia.org/wiki/Algoritmo_de_Kruskal

5. Krustal algorithm, 2012. Consulted on march 12 of 2019.

Graphs: Software for construction, edition and graph analysis. Recovered of: http://arodrigu.webs.upv.es/grafos/doku.php?id=algoritmo_nkruskal

6. Applications of MTS. Consulted on march 2 of 2019. GeeksforGeeks: A computer science portal of geeks. Recovered of: https://www.geeksforgeeks.org/applications-of-minimum-spanning-tree/

7. Traveling salesman problem, 2019. Consulted on march 2 of 2019. Wikipedia, the free encyclopedia. Recovered of: https://es.wikipedia.org/wiki/Problema_del_viajante

8. Boruvka Alforithm, 2017. Consulted on march 2 of 2019. Wikipedia, the free encyclopedia. Recovered of: https://es.wikipedia.org/wiki/Algoritmo_de_Boruvka

9. Figure 1: https://alyssea84.wordpress.com/2014/02/15/the-muddy-city/

10. Figure 2: https://en.wikipedia.org/wiki/Minimum_spanning_tree

11. Figure 3: https://www.computingatschool.org.uk/data/tft/02/04Activity_Toy_Problems_Real_World.pdf

Figure 4: https://en.wikipedia.org/wiki/Prim%27s_algorithm

Figure 5: https://en.wikipedia.org/wiki/Prim%27s_algorithm

Figure 6: https://raweb.inria.fr/rapportsactivite/RA2010/realopt/uid17.html

Figure 7: https://en.wikipedia.org/wiki/Kruskal%27s_algorithm

Figure 8: http://examples.gurobi.com/traveling-salesman-problem/

Figure 9: https://en.wikipedia.org/wiki/Borůvka%27s_algorithm