<u>Network Analysis Lab</u>

Jose Gomez

1. Introduction

This document summarizes the results obtained from the network analysis laboratory carried out in the Methods in Spatial Analysis course taught in the Geoformatics department of University of Salzburg. Throughout the process we will explore the well-known network analysis-based Dijkstra algorithm (Dijkstra, 1959) to find shortest and optimal path. The data used is a set of roads and hospital locations in the city of Bogota, capital of Colombia and the proposed problem is finding optimal paths and estimated travel times from arbitrary points to health facilities locations. For this, spatial data infrastructure, network topology and analysis algorithm were implemented in Postgis, the spatial extension of Postgres and the results were displayed in QGIS software.

2. Materials and methods

The data

Two main sources were used for this analysis both openly provided for IDECA (Capital Spatial Data Infrastructure, in Spanish. https://www.ideca.gov.co), which administrates the spatial information in Bogota city. These two layers were the “Integral Road Mesh. Bogota D.C” and “Health Provider Institutions Bogotá” which, for simplicity, are all assumed to be hospitals. The data was downloaded in shapefile formant and then converted to a spatial database.

Figure 1: Layers used for the analysis.

Dijkstra and TSP algorithms

The objective of Dijkstra algorithm is to solve the optimal path problem which consist of finding the path that represent less cost, that might be distance (shortest path) or other constrain, from an initial vertex called home, to another one called destination. The cost of a path is assumed to be equal to the sum of the cost of links between consecutive vertex on the path, and it’s called the accumulative cost. As a coarse description, the algorithm is based on a best immediate processor approach which selects the next vertex that minimizes the current aggregated cost until the destination vertex is reached (Sniedovich, 2006).

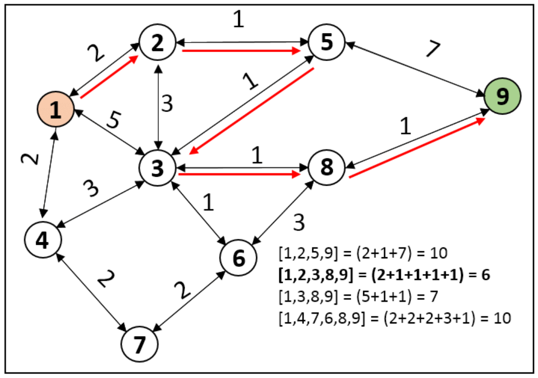
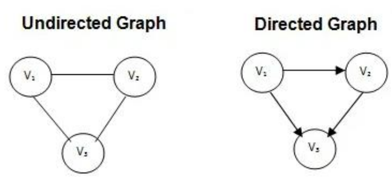


Figure 2: Illustration of Dijkstra algorithm from vertex 1 to 9. Taken from Rehman et al. 2019.

On the other hand, the Traveling Salesman Problem can be defined as: Given a set of nodes, let there be a salesman be located at a single depot node. The remaining nodes that are to be visited are called intermediate nodes. Then, the TSP consists of finding an optimal tour for the salesman, starting and ending at the depot, such that each intermediate node is visited exactly once and the total cost of visiting all nodes is minimized (Bektas, 2006). This problem can be solved with an iterative Dijkstra approach where all optimal path between intermediate nodes is estimated and then optimized into a single rout.

As we are interested in differentiating from walking and driving time and distances, it is mandatory to introduce the concept of directed and non-directed graphs. Shortly, in an undirected graph there is no direction associated with the edges that connect the vertices. In a directed graph there is a direction associated with the edges that connect the vertices. With this we ca assume that when a person is waking, no constrain on the road sense should be considered, different than when is comes to driving where a vehicle cannot go against the road scene. For this, as Dijkstra algorithm bases its estimations on cost, the way to direct a graph is to assign an exaggerated value when the path can not be considered due to its directionality, this is done with the attributes of the road layer. In the table, the total cost is an undirected graph while direct and reverse are the two sense a road can go (Figure 3). We can see that, when a path is only one way, the cost is defined as 1000000, and the actual value of distance only appears in one of the columns, reverse or direct.



cost_total double precision	cost_direct double precision	cost_revers double precision
0.00028922370802470816	1000000	0.00028922370802470816
0.0004359121102882684	0.0004359121102882684	0.0004359121102882684
0.0002717064396784065	0.0002717064396784065	0.0002717064396784065
0.000562703252967004	0.000562703252967004	0.000562703252967004
0.00017309152529825701	0.00017309152529825701	0.00017309152529825701
0.00043606354700943015	0.00043606354700943015	0.00043606354700943015
0.0003832282897198908	0.0003832282897198908	0.0003832282897198908
0.0001772266802561966	1000000	0.0001772266802561966
0.0008797922226949207	0.0008797922226949207	0.0008797922226949207
0.00043089095161227527	0.00043089095161227527	0.00043089095161227527
0.0005572631687679428	0.0005572631687679428	0.0005572631687679428
0.00043609102783835976	0.00043609102783835976	0.00043609102783835976
0.0007722773858605355	0.0007722773858605355	0.0007722773858605355
0.0015400551182602504	0.0015400551182602504	1000000
0.0004451243134538292	0.0004451243134538292	0.0004451243134538292
0.0004791944808840402	0.0004791944808840402	0.0004791944808840402
0.00027794869539707074	0.00027794869539707074	0.00027794869539707074
0.0008659460184599914	1000000	0.0008659460184599914

Figure 3: Difference and tabular representation of directed and non-directed graphs

3. Results

Shortest path

First, the shortest path between one arbitrary house and the closes hospital to it is estimated. For this, the steps are:

- Locate the node point of the hose
- Locate the point of the hospital
- Locate the closest nodes to those points
- Find shorts path with directed and non-directed graph

After that we can find two results with different paths. In the figure 4 it is shown the directions of the streets: red and green lines represent opposite directed lines while black lines are double sense ways. This explains the behavior of both directed and non-directed shortest paths. The undirected one, or walking path (blue line), can go against the sense of the main red line and trough the green ones as well while the directed path, or driving path (purple line), must look for a green or black path to go through as a car should (except for the first edge where there is no other option, likely a mistake in the data).

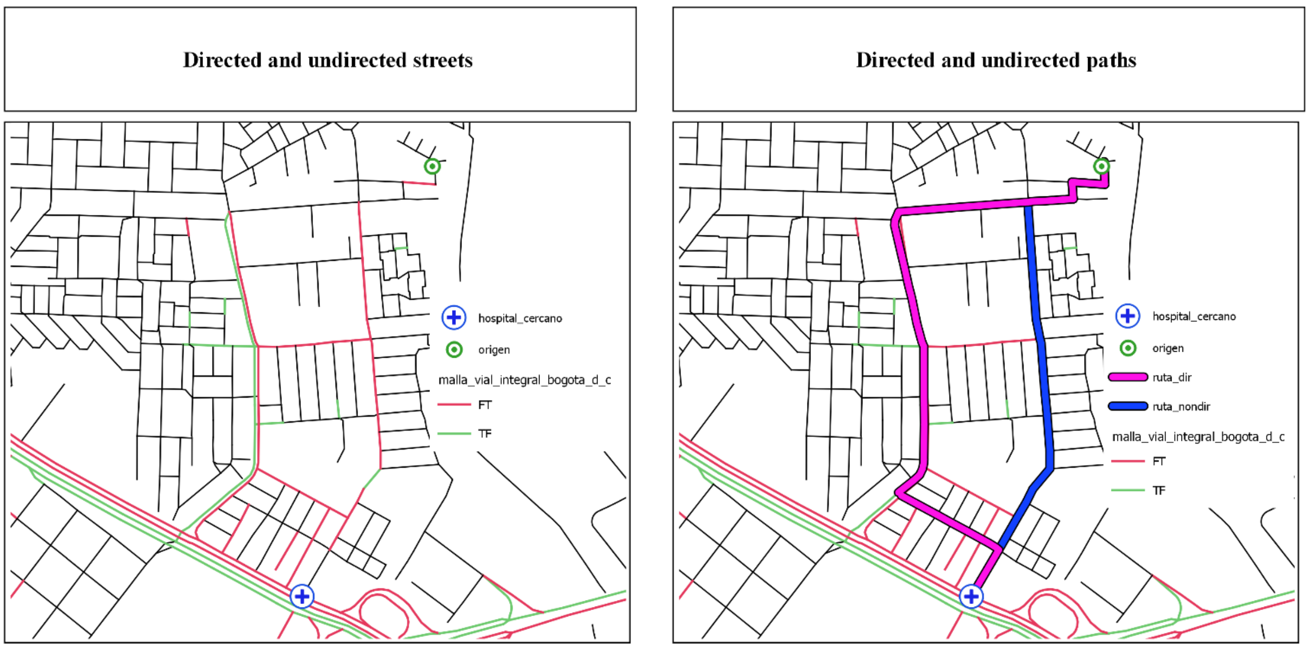


Figure 4: Shortest path to hospital

Optimal route, TSP problem

For this part of the exercise a simple situation in proposed: the origin node is a medical supply depot that is I charge of providing stuff for the 5 closest hospitals in the zone. For sake of simplicity, and to not extend the problem into a Multiple TSP, the depot only counts with one truck that must wind the best way to pass by all hospitals starting and ending at the depot. For this a cost matrix between all nodes, including the origin, is estimated and based on this, the PST algorithm finds the minimum aggregated cost of all the possible combinations.

The result is presented in the figure where we can see each journey in different colors. Although sometimes might seem counter intuitive as segment 2 this is in fact the optimal route to fulfill the task based on directed graph analysis.

start_vid bigint	end_vid bigint	agg_cost double precision
5074	17561	0.0200550108380322З
5074	58619	0.00515781795199904２
5074	79066	0.0130172000365206１
5074	81833	1000000.0168089095
5074	84338	0.012883423022390579
17561	5074	0.026305742239098127
17561	58619	0.015075540902913543
17561	79066	0.0383130124946115
17561	81833	1000000.0091356881
17561	84338	0.038179235480481466
58619	5074	0.013803495483361116
58619	5074	0.013803495483361116
58619	17561	0.014908320973938835
58619	79066	0.025810765738874492
58619	81833	1000000.0116622195
58619	84338	0.025676988724744464
79066	5074	0.022731639422804515
79066	17561	0.0311907043598246１
79066	58619	0.018178730391113535
79066	81833	1000000.0243899288
79066	84338	0.004388887553952545
81833	5074	0.02387421771494154
81833	17561	0.009722035645929009

Figure 5: Cost matrix

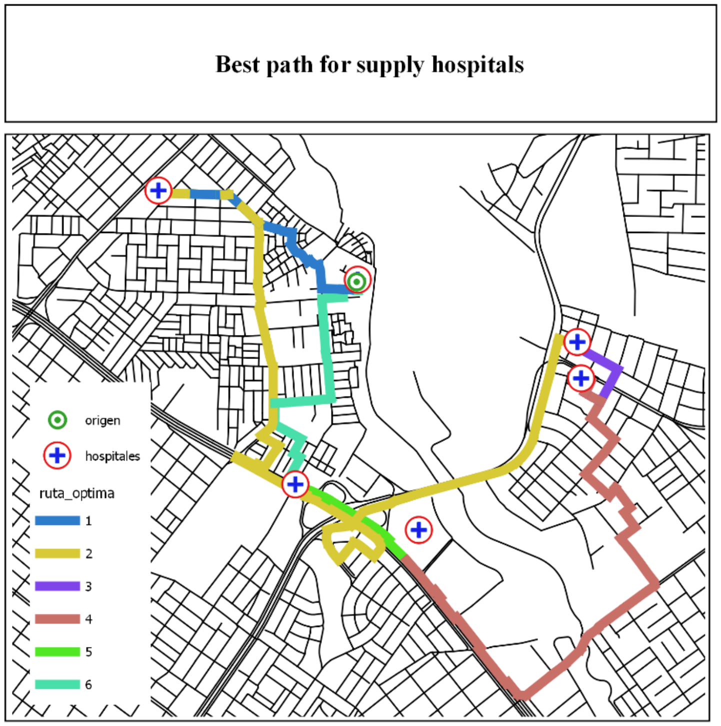


Figure 6: TPS solution to hospital supply problem

Catchment areas

A catchment area can be understood as a one zone where a facility, such has a hospital, attracts population that uses its services. It can be defined trough different metrics, in this example time will be used, that means that every point where driving distances is under a condition is considered to be inside the catchment area. For this, first distance to time conversion is needed and it is performed based on an average speed of 20 km/h.

To find the limits of the catchment area, the algorithm of driving distance of Pgrouting can be used. It estimates which nodes are located below some cost threshold, which in this case will be a time threshold. Three different thresholds where calculated: 4 (blue), 8 (green) and 10 (red) minutes. Wit this points, minimal convex polygon is created, and the catchment zones are defined. It is clear that, as the threshold increases, also the number of points that fulfil the condition increases which is translated in bigger catchment areas.

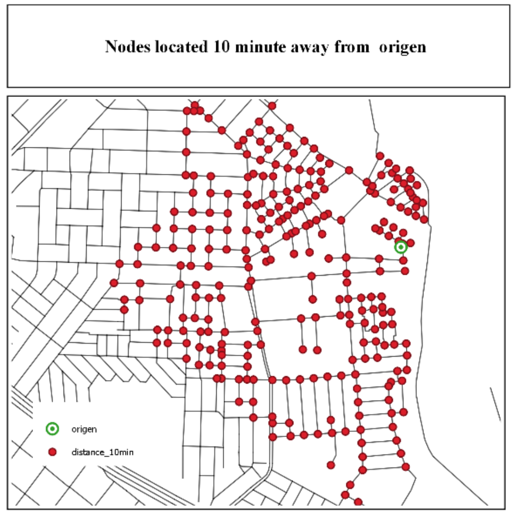


Figure 7: Nodes below threshold example

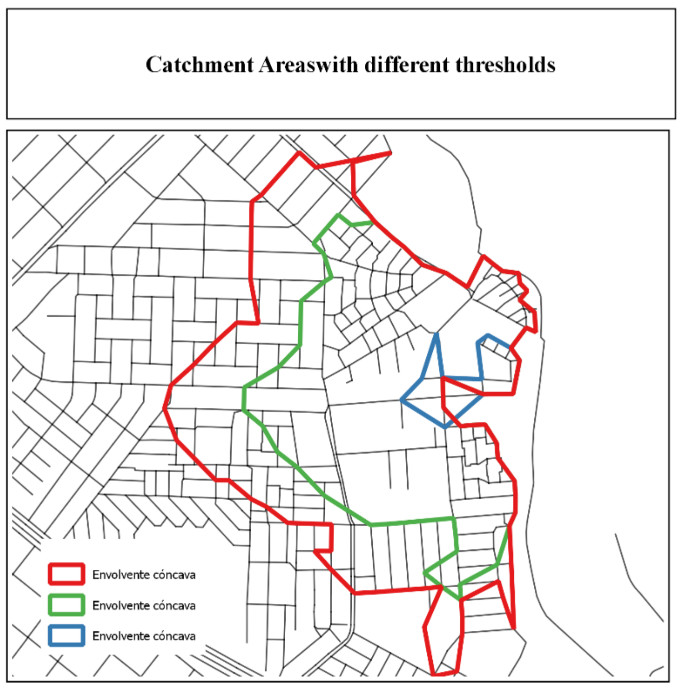


Figure 8: Cachment areas result

1. References

Bektas, T. (2006). The multiple traveling salesman problem: an overview of formulations and solution procedures. omega, 34(3), 209-219.

Rehman, A. U., Awuah-Offei, K., BAKER, D., & Bristow, D. (2019). Emergency evacuation guidance system for underground miners. In SME Annual Meeting 2019 (pp. 19-100).

Sniedovich, M. (2006). Dijkstra's algorithm revisited: the dynamic programming connexion. Control and cybernetics, 35, 599-620.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. Numerische mathematik, 1(1), 269-271.

pgRouting, pgRouting; http://pgrouting.org/.

PostGIS,PostGIS; http://postgis.net/.