DALARNA
UNIVERSITY

Working papers in transport, tourism, information technology and microdata analysis

From the road network database to a graph for localization purposes

First Author: Xiangli Meng
Second Author: Pascal Rebreyend

Nr: 2014:09

Editor: Hasan Fleyeh

Working papers in transport, tourism, information technology and microdata analysis

ISSN: 1650-5581

From the road network database to a graph for localization purposes

Authors: Xiangli Meng and Pascal Rebreyend[1]

Abstract: The problems of finding best facility locations require complete and accurate road network with the corresponding population data in a specific area. However the data obtained in road network databases usually do not fit in this usage. In this paper we propose our procedure of converting the road network database to a road graph which could be used in localization problems. The road network data come from the National road data base in Sweden. The graph derived is cleaned, and reduced to a suitable level for localization problems. The population points are also processed in ordered to match with that graph. The reduction of the graph is done maintaining most of the accuracy for distance measures in the network.

Key words: road network, graph, population, GIS.

[1] Corresponding author. E-mail:prb@du.se. Phone: +46-23-778921.

1. Introduction

Consider the *p*-median problems which allocate *P* facilities to a population geographically distributed in *Q* demand points such that the population's average or total distance to its nearest service facility is minimized. The number of possible solutions is usually enormously large and thus the problem is NP-complete (see Kariv & Hakimi, 1979). The difficulty of finding a good solution would increase when *P* and *Q* increases. Several heuristic methods have been proposed to derive a good solution (see Mladenović, Brimberg, Hansen & Moreno-Pérez, 2007). In our case, we are interested in locating public facilities such as hospitals, public services at the scale of the country Sweden. The objective measure we want to minimize is the average distance between citizen's residence and the closest facility nationwide. The scale of the problem is quite large. We can mention that similar problems occur in business with the same framework such as the gravity *p*-median problem (see Huff 1944, 1946). Regardless the specification of the problems we want to address, solving such problem is based on computing often distances between candidate locations and all or a subset of points representing where people are living. Therefore all the candidate locations should be connected in a graph and the corresponding distance matrix should be in an appropriate scale that enables it for further calculation. For previous researches of location-allocation problems, most of the work done are using the Euclidian distance as a measure. With Euclidian distance, it is much easier to get the distance between any pair of points. Also it is easy for computer to handle. Nonetheless, the Euclidian distance measure is often not accurate, especially for two points that are not directly connected and need some detour. Recent works also show that in rural and semi-rural areas the Euclidian distance is performing badly, in comparison of the network distance (see Han & al, 2013).

In the work of Han & al (2013), locations have been improved by using the road network to compute distances. To use a real network instead of the Euclidian distance affects a lot the way on how we can solve the problem. To directly use coordinates and methods based on location of points in a 2-D space is not anymore possible. As example, the best location of an hospital for a set of cities can be different if by using the Euclidian distance we found that this location is in the middle of a lake or on the top of high mountains. Especially, their results show effect of natural barriers and the effect of the distribution of the population. Since it is not suitable to use the fast methods designed to take profit of Euclidian distances, they have used general methods from combinatorial optimization such as Simulated Annealing. Therefore, the drawback of their approach is that the complexity of the problem (NP-class) and its size leads to sub-optimal solution. Nevertheless, even the sub-optimal solutions are found to outperform previous solutions. Naturally there is a need to find some approach that can produce both faster and better solution.

The work by Han & al (2013) provides us with a good direction to deal with the location problems. A key issue in their work is to derive the road network distance. However, a common situation is that the real data which researchers obtained in the road network database is not the suitable graph for the framework. It may be in different format such as coordinates of the points in the network or polylines, therefore they will not give direct measure of the distance between pair of points. Another common problem to the data set is that it contains often uncompleted information of points or road segment that seems to be isolated and unconnected with the other parts of the graph. Such kind of problems needs to be dealt with before going to further analysis. Dealing with the data in an inappropriate way would lead to the wrong results in finding good solutions to location problems.

We encountered the problems above in our research. The road network data set we used has been provided by the National Swedish Road Agency (Trafikverket). The data comes from the National road data base (NVDB) which is operated by the Swedish Transport Agency, the Swedish Transport Administration and a few other departments. The road network data for the whole country together with its previous versions have been used for other researches as well as business usages, such as road maintenance, traffic management, navigation, etc., see Lundgren (2000). Since the location problems have special requirements for graphs, the previous transformation techniques on the NVDB data do not fit in our framework. There are few researches in location problems use the road network distance at this large scale. Although not nationwide, some researches use part of the data for *p*-median and Gravity *p*-median problems. See Carling etc. (2012) for an example. However the whole data set would have many more problems than data only on a single small area. So there is no way to apply their dealing process directly.

We are aware that our data set is based on national level; therefore the scale itself to this problem would cause trouble for further analysis. Due to the fact that *p*-median problems require the distance between any pair of nodes, the distance matrix corresponding with the graph would have very large dimensions, and it would be beyond the ability of the computers to process it with heuristic algorithms or other methods. Thus we need to transform the data appropriately in a way that we reduce the data to an affordable level for computers to process without losing too much the accuracy at the same time. In this way we can improve speed and accuracy of algorithms for our location problem.

In this paper, we propose our transforming procedure for the road network data for whole Sweden. We will explain the problems we met in processing the data set and how we deal with them. First we will introduce how we clean and process the data set, and then we will explain how we process the population and match it to the road network. After that we will present how we transform the large graph obtained from the previous step to a small and usable graph. During these process, we avoid to introduce approximation unless it is necessary, e.g., missing information. The paper is organized as follows, Section 2 introduces the road data structure and our pre-processing procedure; Section 3 introduces our process in dealing with population data; Section 4 presents the graph reduction; Section 5 presents the experiments with matrix distances; Section 6 gives the conclusions. At Section 7 we talk about the future work.

2. Road data

The raw data we get are a shapefile describing the road network. Shapefiles are used in Geographical Information System (GIS) to represent spatial data. The basic entity in the provided shapefile is a polyline representing a road segment. A road segment is a small portion of a road having the same parameters such as the same road ID, the same speed limits, etc. Every road segment is represented by a set of points (polyline) (x, y and z). For each road segment some attributes are provided such as the direction and the speed limit. The first task is to read all these segments and find out the connection between them.

The raw data consists of 34,180,478 nodes. Among them, 19 are not taken into account since they are outside the country as well as outside the boundaries of the shapefile. After removing duplicates points, we end up with 28,061,739 different points. Speed limits are important information since later we will use it to estimate traveling time. As we can see on the Table 1, more than 80% of the Swedish road network has a speed limit of 70.

If we look on how polylines and segments are, we end up with 31,406,511 edges. The average length of an edge is 21 meters. The maximum length of a segment is 3602 meters and 26206 edges are more than 200 meters. Only 844 of them are more than 500 meters. Having such high number of short road segments is due to the fact that the database from which the shapefile is extracted is used to represent the geographical information associated with a road. Therefore, long segments represent straights sections of the road.

Table 1. Lengths of roads on different speed limits.

Speed limit	Length (km)
5	183
20	24
30	18,958
40	4,535
50	51,104
60	1,726
70	547,866
80	23,962
90	19,581
100	8,137
110	3,977
120	753

In our work, all programming has been done in C, using gcc in a Linux Environment (64 bits) on a Desktop computer with 32GB of ram with an Intel I7-3770 CPU. Parsing the shapefile has been done by using the library called shapelib.

2.1 Data structure

A key to have efficient software in C is to have good data structures which provide a good compromise between speed and space. In our case, during all the processes, starting from reading the road network up to finalizing the reduced graph, we need to store informations about nodes and their connections. Since our main goal is to represent a graph, we have created two main structure types in C, one to represent a node and one to represent an edge. The node structure contains informations like the coordinates, ID of edges connected to it, and the edge information about the length, travelling time.

But, we need also to have a data structure where we can quickly find all nodes in a small given area. This is needed already from the start when we create the graph from the road network. In the process, for each node we want to add, we need to know if a node exists at or closed to the given location. For this purpose, we have decided to use a grid to be able to find quickly all nodes in a given part of the country.

In order to speed up computation, a 2-D grid is applied on the map. All the cells have exactly the same shape and are rectangular. Since some points can later be added outside of the grid, some specific fields are used to store them.

By default, our grid is composed of squares of 500m by 500m. Thus, we have 1295 cells in the X-coordinates and 3043 in the Y-coordinates.

In order to check quickly if a node exist or to identify nodes in the neighbourhood, a grid structure is created where each cell is a 500 meters by 500 meters squares. Each cell contains the list of nodes which belongs to it. Thus, we have 1295 cells in the X-coordinates and 3043 in the Y-coordinates. Overall, the UNIX process needs around 6GB of memory to run with the Swedish road network.

2.2 Cleaning and pre-processing

The first problem we encounter is the lack of direct information about crossings. In our case, we have identified them by ourselves.

Obviously, two different segments going through exactly the same node are crossing at this node. But only few real crossing can be detected by this approach, since in most cases two points from different segments differ by few centimetres. The chosen approach is to, for each node, round the coordinates to the nearest multiple of 2 meters (for all three axes). The choice of 2 meters comes from the general size of a one-way street as well as the height needed for a bridge or tunnel.

After these steps, we end up with 27,922,796 different nodes and a graph since we are able to identify crossing.

2.3 Connectivity

The graph we have now is representing well the road network and its structure. But when we are considering the whole country we faced a problem: not all nodes are connected to each other. Although most part of the graph is connected, there exists some parts of it that are disjoint to the rest of the graph. Those parts could not be neglected.

Thus, the next step is to find out the different strongly connected components (we have directional edges) in the graph. The strongly connected component is a well-known problem in graph. Therefore, we are using the Tarjan's algorithm (see Tarjan, 1972), we have identified 3057 components. The mainland is clearly recognized as the one having the highest number of nodes. 559 components have more than 10 inhabitants, 171 more than 100, 111 more than 200 persons.

Others components are disconnected from the mainland for two main reasons. The first one is that in Sweden we have islands without any bridge to the mainland. It's for example the case the second biggest component in our case which represent the island of Gotland and its 236 235 inhabitants. It's also the case for a lot of small islands, both in the archipelago and island along the coasts as well as small islands on lakes. In most cases, communications to the mainland are done by a ferry line.

Another problem in the data itself exists where we miss crossing or road. Such problems occur by wrong or inaccurate values in the database, especially regarding the altitude. This can be detected when the distance is shorter than a threshold or when the connection between the components is possible only in one direction. We choose here to have a generic solution instead of using threshold. That is because previous researches on this data set gives us no information of the setting for threshold, and inappropriate threshold would lead to over or miss detection.

In order to take into account the people living on these “islands”, we will add to the graph virtual edges representing ferry lines or other means of transportations.

A virtual edge will be added between the closest pair of nodes for which the distance between the mainland and the island is the shortest. We can notice that in case of Sweden, most of real islands are directly connected to the mainland.

We can also mentioned that parts of the networks wrongly classified as islands will not affect so much results since virtual edges added are short (often less than 100m).

The algorithm we have used to find these edges is not optimized since the algorithm is not taking into account geographical localization of points. Since this algorithm only runs once and we store theses edges in a file, to have a running time which is roughly one day is not an important issue.

3. Population

3.1 Population data.

Another important factor in our location models is the population. The population are required to be geo-coded so that we are able to identify the population in each grid and find out the distance from the grid to the facility. The census population data comes from the Statistics Sweden (SCB) from the year 2012. The data is well organized and so we skip the steps of pre-processing it.

In total we have 188,325 points representing 5,411,373 persons. Thus, each point represent in average almost 29 persons. The most populated point represents 2,302 persons. In our case, the population of Sweden is represented by a set of points. Each point represents the number of people between 20 and 64 years old living in a square centred to the point. The size of the square if often 500m by 500m but variations occur between big cities and the landscape area. The population points are derived after aggregation. It means that all the persons in the same square would be assumed to have the centre of the square as the same starting point, and their distances to the same facility are the same.

It should be mentioned that here we use the population to represent the demand for the facilities in location-allocation models. So we directly use the original data in our graph. This representation makes sense when it comes to locating public facilities and retailing centres, etc., but when the facilities located are not targeted at public, the population will not fit in the data frame. In that case the demand points should be changed to the ones that have usage on the facilities. However, even for locating public facilities, the population quite often is not equal to or exactly equal to the demand in many problems, because the demand of public facilities varies among different types of people. Therefore certain transformations of the population data are required according to the specific problems. For example, if we deal with the data for locating retailing centres, it would be more appropriate that we put small weight on the population with very small ages, and if we deal with locating hospitals, the babies should be more reasonable bearing more weight than young people. Usually giving the population different weight requires auxiliary information of the population such as age. Here we do not concern this problem here and just propose how we connect the population data points to the graph. For the location problems with special requirements should be done in a similar way.

3.2 Matching population points with road points

The population of each population point is connected to the closest road node. Since segments between nodes are really short, we don't add look for the closest distance between the population point and the closest point of a segment. We have 497 points representing 1424 persons which are more than one kilometre to the closest road node. (resp. 6031 and 33,779 if for more than 500 meters). Approximations done here are due to the inaccuracy in data provided. Since most of the population points provided represent the aggregated population on 500 by 500 meters squares, we may have an error of maximum $500\sqrt{2}/2$ which is about 353 meters (Euclidian distance) between where a person is living and the point used to represent this person. Although this is a basic assumption for p-median models, it would result in some errors in the finial graph and may lead to errors in the future. Current and Schilling (1987) point out that there are 3 types of error sources, A, B and C in this kind of aggregating processes. Source A comes from the difference of distances from a real population location inside a grid to a facility outside that grid when we “relocate” the point to the centre of the grid, i.e., the persons in the grid do not actually live in the centre of the square. Source B is the distance between the real population location and the facility in the grid where the facility is located, which is treated as 0 in modelling process while it is not 0 in reality. Source C comes from the points at the borders of some grid, which are assigned to a further facility because centre of the grid is close to that one. In our data with the grid defined before, it suggests that for Source A and B errors, the maximum error for a person should be at most 353 meters. That might be important if we locate facilities in a small but high populated area, but it is a small distance error when we consider locating facilities in a national level. Carling etc. (2012) give the mean distance in Dalarna part of Sweden for people travelling to hospitals, and it is around 40 kilometres. Thus in our case, the Source A and B error will not have big influence the quality of our graph, and will not affect the further research result in most cases. The only exception is that if researchers are locating a large number of facilities in the country and the resulting average distance to the facility is not large enough to neglect the error as large as 353 meters, then some methods should be taken to improve that. For details of those kinds of methods, see Current and Schilling (1987). As to Source C error, they will mostly happen to the grids that have similar distance between 2 or more facilities. In that sense the difference between 2 facilities should be within 353 meters. Consider the level of the whole country, it should not influent the result in a large scale.

Finally, we can add that it's easy and quick to update our approach to keep and use this distance in measurement. It’s only requiring to add an extra value to nodes representing this fixed distance.

4 Graph reduction

At this point, we have a strongly connected graph representing our road network. The nodes in the road network have some overlap with the population nodes. Thus some of the nodes are also representing where people are living.

Since the goal of our work is to minimize distances between where people are living and facilities (the closest or a set of facilities in the case of the *p*-gravity model), we need only to keep the information needed to compute distances between people's residence and candidate nodes. According Hakimi (1964), in case of the p-median problems, it is locating facilities on nodes which are crossing or representing where people live that gives the best solution to the problems. Locating facilities outside these nodes and into some segment will only provide sub-optimal solutions. Consequently we have informations in our graph which are not used to compute distances and therefore we can remove them. In general, to compute distances in the network, we will use the Dijkstra algorithm or a modified version to compute from a point (typically a population point) distance to all or a subset of candidate points.

In this part, we will explain which information can be removed, how and the benefits.

4.1 Removing dead ends.

In our graph, we have nodes with degree 1, i.e. nodes which are connected to only one neighbour. Such nodes can be called dead-end nodes. If to this node is associated some population, we should obviously keep this node in our system. On the other hand, if not a single person is connected to this node, we can remove this node from our graph since this node will never been used to compute the distance matrix. Obviously, the neighbour can end up to be a new dead end and therefore, in our algorithm, we parse all node and if a node is a dead end, we recursively analyse and maybe remove its neighbour until the graph becomes stable.

By applying this, we can remove 9,873,764 nodes which represent more than 35% of the nodes. This is a huge reduction of our graph. Also by doing this we barely lost any information in the population or road network. In practice, these deleted nodes mainly represent small roads on the country side, where people move away but the road information is still stored in the data base.

4.2 Removing useless node of degree 2.

If the original graph contains a lot of nodes, it's not to represent it structure but to represent the shape of the road (curves, hills, etc.). We are not directly interested by such information but only the accurate information about the real distance by the network between nodes as well as the traveling time based on the speed limit. Since lot of nodes are representing the shape of the road only and not its structure or population, we can reduce even more our graph by removing them. Nodes like this usually are connected to 2 neighbours, thus are denoted as degree 2.

Our algorithm works as follow: For each node, we check if this node represents some population's residence. If not, we check if this node is not a crossing by checking if this node has exactly 2 different neighbours. If this node has 2 neighbours, we will analyse the direction of edges and remove them to disconnect this node from the graph. Then, we add in our graph a new edge between the two neighbours. The distance and traveling time of this new edge is the corresponding sum of the two removed edges. We repeat this process until not further reduction is possible.

We can notice here that the speed limit may not be any more an available information since the two aggregated edges may have different speed limits. Anyway, since the two measures we want to use are distances (in km) and traveling time, modifications done in our graph by this algorithm do not affect them.

At the stage, we can remove 10,347,191 nodes.

By applying these two graph reductions, we have been able to remove 20,220,955 nodes (72% of the nodes) and now, the graph has only 7,701,841 nodes.

5 Experiments: Matrix distances

This work has been done in order to work on the p-median problem used to investigate the facility localization problem. Different approaches and algorithms can be used but as soon as we want to use the network distance, we need to use the reduced graph to compute distances. The first experiments we have done in this case are to compute the distance matrix between the 188,325 points and a set of candidate locations. As a first trial, we have used as candidate nodes 1938 nodes which are the closest to the centre of the 1938 settlements in Sweden.

The graph itself provides us with distance between some pairs of nodes, but not all of them. The route from a node to another one that is not adjacent will have many choices and the distance will have many values. We aim to find the shortest. In the first part of the experiment, we were using the classical Dijkstra algorithm (Dijkstra, 1959) with a non-optimized queue. Using such methods leads to an average of 12 days of computations.

5.1 Optimized Dijkstra method.

A way to optimized computations, aside using well the C language, is to have a better data-structure for the queue used by the algorithm. The Fibonacci heap is therefore used for this purpose.

The Dijkstra algorithm is an efficient algorithm and a run of this algorithm is computing distances from (or to) a points to (or from) all others. This algorithm works by starting from the source nodes and exploring in an iterative fashion nodes around. Each time, the closest node which is unvisited is chosen.

For later optimization, we can even add an early stopping criterion to this algorithm and stop once distances for all nodes within a certain distance have been computed.

6. Conclusions.

To deal with real data leads to face different problems. Aside the side of data, we need to check carefully them. To have wrong or inaccurate data is often result in incorrect analysis results. After deriving the clean and accurate data, another problem is to reconstruct information we need based on it. In our example, we have mainly rebuilt the structure of the road network by identifying crossing based on a 2-meters approximation. The approach we proposed has been used both on full Sweden and as prototype on the province of Dalarna and in both cases results are encouraging. The missing information of the nodes has been added by linking it to the nearest neighbour and very little error generated by doing so. It should be noticed that for the nodes in Gotland, there are no bridge or tunnel connect them to the main land, and we use virtual link based on the ship speed and time required to get to the main land. In such procedure we actually underestimate the distance between the main land and Gotland in the sense that people need some time for transferring and waiting between ships and cars. However we do not have further information on the (average) time for the passengers. The graph would make more sense if we have those time data and add that to the virtual link.

Our approach is built to be as general as possible. The graph after identifying crossing and connecting the whole network is not able to be handled efficiently by computers. Therefore the second goal of our approach is to reduce as maximum as possible the number of nodes. It should be mentioned that it is possible to skip the distance matrix and go directly to searching for the shortest distance between 2 nodes and just store the edges in the graph. However in our trial, that approach takes very long time for computer to process. Thus we are not quite flexible to choose good heuristic methods; neither can we apply the algorithms in an efficient way. By reducing the graph, we reduce the computational time by a huge factor, leading to more efficient algorithm and better results.

In our data file, we have the speed limit for different segment of the road, therefore that factor are taken account in our process. We get a distance matrix and a time distance matrix. But the two dealing processes have quite small difference, thus we combine them together to give our process without separation. It would also be interesting to see if there are some difference in the solutions to location with the 2 matrixes.

7. Future work

In our work, we were focusing of removing useless information to reduce the size of the problem. Since to compute distance most of approaches are using the Dijkstra algorithm, we can investigate if some pre-computations can be made in order to speed up the Dijkstra algorithm. The main idea is that long distance travels will use and reuse the same set of main roads and therefore we can add virtual edges representing them.

Our approach has been tested with the Swedish road network based on the official public road agency. Other tests can be done by using information from other sources. To test our approach with countries is also a challenging test. Obviously, questions regarding island, level of correctness will be addressed carefully.

The graph we derived is based on our needs for location problems. It could also be used for other kind of studies such as Travelling Sales Man problems. The techniques we used in this process are not confined to the unique data set. However, for other data set that having the similar scales, the problems in transformations maybe different. Some new problem may pop out. Therefore it would be interesting to see how the other data could be processed. Another issue that need to be mentioned is that we have not apply the heuristic algorithms to the graph we derived, therefore we have no idea of how the performance would be, i.e., if the solutions of *p*-median are accurate enough, and what characterises this graph would have. In the process of solving location problems, we would get better understanding of the important features of this graph. That knowledge would help up improve the processing technique.

Reference

Carling, K., Han, M., & Håkansson, J., (2012). Does Euclidean distance work well when the p-median model is applied in rural areas? Annals of Operations Research, 201:1, 83-97.

Current, J. & D. Schilling. (1987). Elimination of Source A and B errors in p-median location problems. Geographical Analysis 19, 95-110.

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

Hakimi, S.L., (1964). Optimum locations of switching centers and the absolute centers and medians of a graph, Operations Research, 12:3, 450-459.

Han, M., Håkansson, J. & Rebreyend, P. (2013). How do different densities in a network affect the optimal location of service centers? Working papers in transport, tourism, information technology and microdata analysis, ISSN 1650-5581; 2013:15

Huff, D.L., (1964). Defining and estimating a trade area. Journal of Marketing, 28, 34-38.

Huff, D.L., (1966). A programmed solution for approximating an optimum retail location, Land Economics, 42, 293-303.

Kariv, O., & Hakimi, S.L., (1979), An algorithm approach to network location problems. Part 2: The p-median. SIAM Journal of Applied Mathematics, 37, 539-560.

Lundgren, M-L (2000), The Swedish National Road Database – Collaboration Enhances Quality, Proceedings of the Seventh World Congress on Intelligent Transport Systems, 6-9 November, Turin, Italy.

Mladenović, N., Brimberg,J., Hansen, P., and Moreno-Pérez J.A. (2007). The p-median problem: A survey of metaheuristic approaches. European Journal of Operational Research, 179 (3) (2007), pp. 927–939

Tarjan, R. E. (1972), "Depth-first search and linear graph algorithms", SIAM Journal on Computing 1 (2): 146–160, doi:10.1137/0201010