

Accurate solutions for real instances of the traveling salesman problem using Google Maps APIs.

Leonardo G. Hernández-Landa, Rosa Elena Mata-Martínez
Industrial Engineering and Administration Department
Universidad Autónoma de Nuevo León
Nuevo León, México
leogabrielhdz@gmail.com

Abstract

There are a large number of variants for the traveling salesman problem, as well as the solution methods such as linear optimization, heuristics and metaheuristics that provide close or optimal solutions for the problem. When we deal with real instances, these solutions usually become invalid due to problems of accuracy and/or update of the digital maps, mainly when we want to create routes in a neighborhood due to dominant factors such as the sense of the streets, the closeness between the points, or the geospatial data of the map. Google Maps is the most accurate and accessible database in terms of spatial information. In this work we present a methodology to calculate precise routes for the Traveling Salesman Problem. We use the google maps APIs to obtain information, then preprocess the data in order to convert it into a valid graph, and finally the instance is solved by a proposed metaheuristic. The obtained solutions are compared between digital maps (INEGI, INE of Mexico) and the google maps API showing the advantage of use the APIs.

Keywords

Traveling Salesman Problem, Google Maps, Heuristics, Routing.

1. Introduction

Every day scientific research is closing gaps with real-world applications and a rapprochement and cooperation between research and industry is taking place in the word. The Travelling Salesman Problem (TSP) is a relatively old problem, it was documented early in 1759 by Euler, whose interest was in solving the knights' tour problem. A correct solution would have a knight visit each of the 64 squares of a chessboard exactly once on its tour. The term 'travelling salesman' was first used in 1932, in a German book written by a veteran travelling salesman. The TSP was introduced by the RAND Corporation in 1948. The Corporation's reputation helped to make the TSP a well-known and popular problem. The TSP also became popular at that time due to the new subject of linear programming and attempts to solve combinatorial problems, (Larranaga et al.,1999). The TSP objective is to find the shortest route for a travelling salesman who, starting from a home city, has to visit every city on a given list precisely once and then return to the home city. The main difficulty of this problem is the immense number of possible tours: $\frac{(n-1)!}{2}$ for n cities.

The TSP is a problem that has been widely studied and there is a wide range of scientific literature and variants of the problem. (Reinelt,1994) Is a complete book dedicated to the TSP overview and application. First the theoretical prerequisites are summarized. Then the emphasis shifts to computational solutions for practical TSP applications. (Laporte, 1992) integrated overview of some of the best exact and approximate algorithms so far developed for the TSP, at a level appropriate for a first graduate course in combinatorial optimization. (Reinelt, 1991) provides researchers with a broad set of test problems from various sources and with diverse properties. (Larranaga et al., 1999) made a review of the different attempts made to solve the Travelling Salesman Problem with Genetic Algorithms.

There are a lot of literature to know how to solve the TSP such as (Miliotis, 1976; Burkard, 1979; Raman and Singh Gill, 2017), but not how to manage the real info in order to obtain quality solution. We can use digital maps directly provided by governmental institutions such as the case of Mexico is The National Institute of Statistics and Geography (INEGI). The first problem which we usually face is the updating of information. Small changes in the map could make our solutions invalid, among other problems that we have to face such as the accuracy of the map and the roads direction. The more detailed tool that we can trust about the streets and updated roads is Google Maps. There are other

online public maps such as Open Street Map, however they are still in development and have lack complete information.

In this research we present a methodology that aims to take advantage of the benefits of using Google Maps APIs to solve real applications of the TSP. Although Google Maps APIs can be used to solve the TSP directly it has certain kind of problems that must be addressed in order to reduce the use of resources. We mainly refer to the combinatorial nature of the problem and the economic and computational cost. For example, for a instance that has 200 nodes, trying to obtain a matrix of distances directly from the Google Maps API represents making 40,000 calls from the API, which raises the costs and time to obtain the information.

Google Maps was introduced in February 2005 which revolutionized the way of see the maps on the web. Almost immediately there was a need to obtain information that was not directly from the graphic map and the first Google Maps API emerged. The API consist of JavaScript files that contains classes with methods and properties that we can use for request updated information of the map, (Svennerberg, 2010).

Then the proposed methodology is based on the interaction of several elements that allow solve the problem with real data without losing quality in the information and reducing costs and times.

- The geospatial maps are taken as start,
- a transformation of the data is made, and the coordinates are extracted,
- Obtaining the visibility graph,
- calculation of distance matrix using the Google Maps APIs,
- application of Floyd-Warshall algorithm to complete the graph and
- the solution to the TSP is found

2. Methodology

We focus on instances where it is difficult to accurately calculate distances, in example routes that need to travel through neighborhoods or electoral sections are where it is often more complicated to make routes or tours. The problem then would be to go through all the corners like the collecting waste problem.

TSP is defined in (Laporte, 1992) such as a graph $G = (V, A)$ where V is a set of n vertices. A is a set of arcs or edges and let $C = (c_{ij})$ be a distance matrix associated with A . The TSP consists of determining a minimum distance circuit passing through each vertex once and only once. Such a circuit is known as a tour or Hamiltonian circuit (or cycle). In several applications, C can also be interpreted as a cost or travel time matrix. It will be useful to distinguish between the cases where C (or the problem) is symmetrical, i.e. when $c_{ij} = c_{ji}$ for all $i, j \in V$, and the case where it is asymmetrical. Also, C is said to satisfy the triangle inequality if and only if $c_{ij} + c_{jk} \geq c_{ik}$ for all $i, j, k \in V$. This occurs in Euclidean problems, i.e. when V is a set of points in \mathbb{R}^2 and c_{ij} is the straight-line distance between i and j .

2.1 Data extraction

With the geospatial information of the maps provided in this case by the INEGI, with the help of a GIS software we extract the polygons and roadways of the map on which we want to work in detail. We use the RGDAL library (Bivand et al. 2015) on R programming language for the purposes of information extraction, due to the flexibility in the manipulation of the information. Figure 1 shows different levels of detail that can be extracted from geospatial maps.

We use the information of the coordinates and polygons to establish points of interest and define an initial graph. At the same time, a preprocessing, cleaning and elimination of duplicates of the data is carried out.

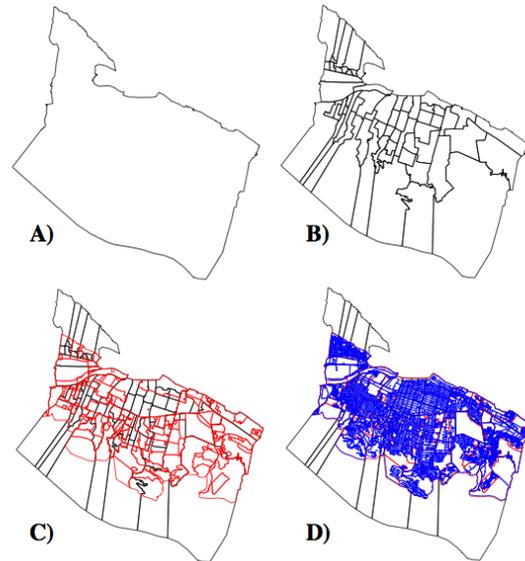


Figure 1 A) Layer of municipality, B) layer of electoral sectioning, C) layer of neighborhoods, D) layer of blocks

2.2 Visibility graph

Once the n points of interest are located, a matrix of Euclidean distances is created we proceed to create the visibility graph G_{vis} , which determines where the connections from i to j are guaranteed as long as there are no obstacles.

(De Berg et al. 2008) Define the visibility graph as let S be a set of disjoint polygonal obstacles in the plane with n edges in total. To compute the visibility graph of S , we have to find the pairs of vertices that can see each other. This means that for every pair we have to test whether the line segment connecting them intersects any obstacle. Such a test would cost $O(n)$ time when done naively, leading to an $O(n^3)$ running time.

Algorithm 1 shows the general procedure to obtain the visibility graph from a point against the obstacles.

Algorithm VISIBILITYGRAPH(S)

Input. A set of disjoint polygonal obstacles.

Output. The visibility graph $G_{vis}(S)$.

1. Initialize a graph $G = (V, E)$ where V is the set of all vertices of the polygons in S and $E = \emptyset$.
2. for all vertices $v \in V$
3. do $W \leftarrow \text{VISIBLEVERTICES}(v, S)$
4. For every vertex $w \in W$, add the arc (v, w) to E .
5. return G

Algorithm 1. Visibility graph search procedure, (De Berg et al. 2008)

Where $VISIBLE\ VERTICES(v, S)$ is a procedure where we proposed to create a m restricted candidates list according to the distance matrix as we can see in Algorithm 2.

Algorithm $VISIBLE\ VERTICES(v, S)$
Input. v, A set of polygonal obstacles.
Output. List of visible vertices
1. Sort in ascending mode the m closest vertices to v according to distance matrix
2. For every vertex i in l do
3. $L \leftarrow IS\ VISIBLE(v, i)$
4. Return L

Algorithm 2. Visible Vertices algorithm

Where $IS\ VISIBLE(v, i)$ It is a function applied directly from the visibility library (Obermeyer and Contributors)

2.3 Using Google Maps APIs

Once obtained the graph of visibility with respect to the geospatial data with the help of the (Cooley et al. 2018) library in R we obtain the distances from Google Maps, due to the previous process we can reduce the number of calls from the APIs up to 70%. The process is simple, for each pair of nodes visible within the visibility graph we make a call to the API. Once the process is completed we must ensure that the status is complete for all pairs of nodes.

To obtain the complete graph we will use the Algorithm 3. The Floyd-Warshall algorithm is a simple and widely used algorithm to compute shortest paths between all pairs of vertices in an edge weighted directed graph. This algorithm has a worst-case runtime of $O(n^3)$ for graphs with n vertices, (Hougardy 2010).

FLOYD-WARSHALL Algorithm
Input: A digraph G with $V(G) = \{1, \dots, n\}$
Output: An $n \times n$ matrix D such that $D[i, j]$ contains the length of a shortest path from vertex i to vertex j .
1. $n \leftarrow rows[G]$
2. $D^0 \leftarrow W$
3. For $k \leftarrow 1$ to n
4. Do For $i \leftarrow 1$ to n
5. Do For $j \leftarrow 1$ to n
6. $d_{ij}^k \leftarrow \min(d_{ij}^{k-1}, d_{ik}^{k-1} + d_{kj}^{k-1})$
7. Return D^n

Algorithm 3 The Floyd-Warshall algorithm

2.4 Solving the TSP

With the complete distance matrix, we can proceed to generate the route that will pass through the points of interest through the use of any solution algorithm for the TSP. In this case we use a simple heuristic constructor using the nearest insertion, algorithm 4, and to improve we use a local search using the 2-opt procedure, algorithm 5.

For the following, we will assume that there is a complete undirected graph K_n with edge weights c_{uv} for every pair u and v of nodes. For ease of notation we will denote the node set by V and assume that $V = \{1, 2, \dots, n\}$.

Insertion procedure

1. Select a starting tour through k nodes v_1, v_2, \dots, v_k ($k \geq 1$) and set $W = V \setminus \{v_1, v_2, \dots, v_k\}$.
2. As long as $W \neq \emptyset$ do the following.
3. Select a node $j \in W$ according to the shortest distance from the tour.
4. Insert the node that has the shortest distance to a tour node, i.e., select j with $d(j) = \min\{d_{\min}(l) \mid l \in W\}$.
5. end of insertion

Algorithm 5 Constructive method through nearest insertion

2-opt Procedure

1. Let T be the current tour.
2. Perform the following until failure is obtained.
3. For every node $i = \{1, 2, \dots, n\}$ Examine all 2-opt moves involving the edge between i and its successor in the tour. If it is possible to decrease the tour length this way, then choose the best such 2-opt move and update T .
4. If no improving move could be found, then declare failure.
5. end of 2-opt

Algorithm 4 Improvement method for the TSP initial solution

Experimentation

The procedure was executed for some specific zones, Figure 2 shows an example of the neighborhood Palo Blanco in the municipality of San Pedro Garza Garcia, Mexico. The extraction of the information and its final route result is observed in Figure 3. Keep in mind that the final solution incorporates roads that are not within the area to be explored, however they are necessary routes for the solution of the problem. The algorithms were executed in an iMac computer with 2.7 Ghz of processor, 8 Gb of RAM in macOS Sierra 10.13.

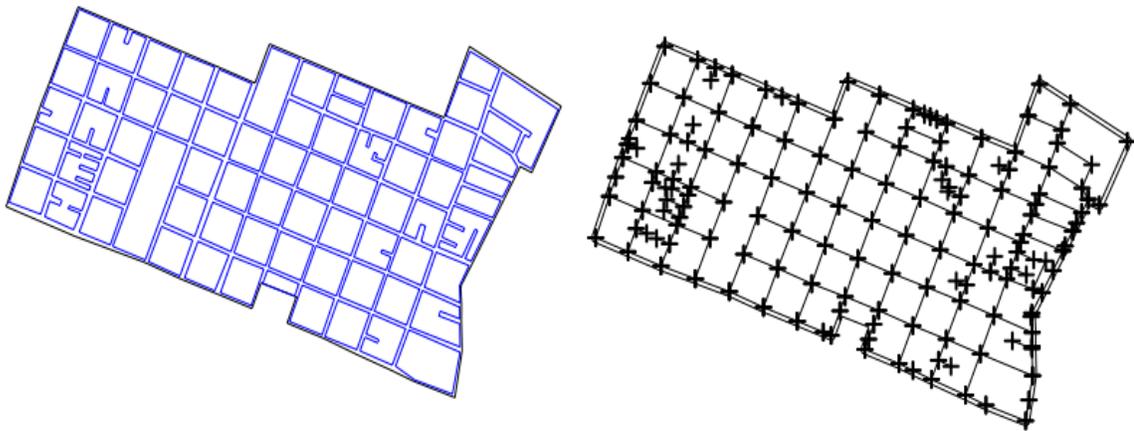


Figure 2 Palo Blanco neighborhood and the destiny points for the TSP

Table 1 shows a comparison of different instances evaluated with the methodology, you can see that in column 2 is the information if we only consider the use of digital maps without resorting to google against column three that shows the real distance with respect to the use of APIs. Then column 4 clearly shows the difference in precision of the actual distance using google APIs to solve the TSP against the only use of digital maps.



Figure 3 Real solution for Palo Blanco TSP route using Google Maps APIs

Table 1 Comparison of results for solved instances with digital maps and the use of google maps

INSTANCE	Digital map solution	Using GM APIS	GAP %
1	20167	23465	16.35
2	16775	20103	19.83
3	14545	18654	28.25
4	13663	16096	17.80
5	13445	15434	12.88
6	12123	13930	12.97

3. Conclusions

The use of google maps APIs allows to obtain precise and real solutions for the problem of the traveling agent. Regardless of the variant of the TSP and the restrictions, this methodology can be applied for any specific case. As a main recommendation, we should point out that we must obtain the exact coordinates of the points of interest to visit, since the quality of the solution depends on these points, mainly in instances where complete neighborhoods have to be traveled. Another recommendation that we can make is that the quality of the solution will depend greatly on having well identified roads and blocks to find a good solution for the visibility graph, from which the distance matrix is obtained.

It is observed that the obtained solutions can be more expensive in distance or time of the solutions obtained directly from geospatial maps, nevertheless the precise solutions for the real instances are guaranteed.

As future work, different variants of the TSP will be compared, as well as the extension of the application of the TSP to the vehicle routing problem (VRP) and its variants.

4. References

- Bivand, Roger, et al. "Rgdal: Bindings for the Geospatial Data Abstraction Library." *R Package Version 0.8-16*, 2015, pp. 1–49, doi:10.1353/lib.0.0050.
- Burkard, Rainer E. "Travelling Salesman and Assignment Problems: A Survey." *Annals of Discrete Mathematics*, vol. 4, no. C, 1979, pp. 193–215, doi:10.1016/S0167-5060(08)70827-6.
- Cooley, David, et al. *Googleway-Vignette*. R package version 2.7.0, 2018, p. 93.
- De Berg, M., et al. "Computational Geometry: Algorithms and Applications." *Computational Geometry*, 2008th ed., vol. 17, 2008, doi:10.2307/3620533.
- Hougardy, Stefan. "The Floyd-Warshall Algorithm on Graphs with Negative Cycles." *Information Processing Letters*, vol. 110, no. 8–9, 2010, pp. 279–81, doi:10.1016/j.ipl.2010.02.001.
- Laporte, Gilbert. "The Traveling Salesman Problem: An Overview of Exact and Approximate Algorithms." *European Journal of Operational Research*, vol. 59, no. 2, 1992, pp. 231–47, doi:10.1016/0377-2217(92)90138-Y.
- Larranaga, P., et al. "Genetic Algorithms for the Travelling Salesman Problem: A Review of Representations and Operators." *Artificial Intelligence Review*, vol. 13, no. 2, 1999, pp. 129–70, doi:10.1023/A:1006529012972.
- Miliotis, P. "Integer Programming Approaches to the Travelling Salesman Problem." *Mathematical Programming*, vol. 10, no. 1, 1976, pp. 367–78, doi:10.1007/BF01580682.
- Obermeyer, K. J., and Contributors. *VisiLibity: A C++ Library for Visibility Computations in Planar*. 2008, <http://www.visibility.org>.
- Raman, Vikas, and Nasib Singh Gill. "Review of Different Heuristic Algorithms for Solving Travelling Salesman Problem." *International Journal of Advanced Research in Computer Science*, vol. 8, no. 5, 2017.
- Reinelt, Gerhard. "The Traveling Salesman : Computational Solutions for TSP Applications." *Computational Solutions for TSP Applications*, 1994, doi:10.1007/3-540-48661-5.
- . "TSPLIB—A Traveling Salesman Problem Library." *ORSA Journal on Computing*, vol. 3, no. 4, 1991, pp. 376–84, doi:10.1287/ijoc.3.4.376.
- Svennerberg, Gabriel. *Beginning Google Maps API 3*. 2010, doi:10.16373/j.cnki.ahr.150049.

Biographies

Leonardo G. Hernandez-Landa holds a BSc. in Industrial Engineering from ITSPe in Veracruz, México and earned his PhD in Engineering from the graduate program in System Engineering at Department of Mechanical and Electrical Engineering, Universidad Autónoma de Nuevo León (UANL). Leonardo is currently a Professor of operations management at Department of Industrial Engineering Administration, UANL in San Nicolás de los Garza, México, where he joined in 2016. Dr. Hernandez' research has primarily focused on methods for solving difficult discrete optimization problems arising in logistic, routing and transportation systems. Previously, he has conducted funded research on vehicle routing problems with accessibility and route design. Dr. Hernandez is a SNI Fellow second highest country-wide distinction granted by the Mexican System of Research Scientists, where he has been a member since 2017. Dr Hernandez live very happily with his wife and family in Apodaca Nuevo León, México.

Rosa Elena Mata-Martínez holds a BSc. in Chemistry Pharmacist Biologist from Universidad Autónoma de Nuevo León (UANL) in Nuevo León, México and eamed her Master of Superior Education from the graduate program in Facultad de Filosofía y Letras. Rosa Elena is actually a Professor of Chemistry, Materials Technology and Manufacturing Process at Department of Industrial Engineering and Administration, UANL in San Nicolás de los Garza, México, where she joined in 1992. Ms. Mata Mtz. Research has focused on Education Problems, Company Situations on Engineering Area and Pharmacist Aspects. Ms. Mata Mtz. has a PRODEP (Programa para el Desarrollo Profesional Docente, para el tipo Superior) distinction where she has been a member since 2009. Ms. Mata Mtz has one daughter and lives San Nicolás de los Garza, N.L.