Masters Program in Geospatial Technologies

SCHEMATIC BUS TRANSIT MAPS FOR THE WEB USING GENETIC ALGORITHMS

Marcelo de Lima Galvão

Dissertation submitted in partial fulfilment of the requirements for the Degree of *Master of Science in Geospatial Technologies*

NOVA IMS Information Management School
Universitat Jaume I
Westfälische Wilhelms-Universität Münster

SCHEMATIC BUS TRANSIT MAPS FOR THE WEB USING GENETIC ALGORITHMS

Dissertation supervised by

Prof. Dr. Francisco Ramos Romero
Dept. Lenguajes y Sistemas Informaticos
Universitat Jaume I
Castelló – Spain

Co-Supervised by

Prof. Dr. Angela Schwering
Institute for Geoinformatics
Westfälische Wilhelms-Universität Münster
Münster – Germany

Dr. Mauro Castelli
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa
Lisbon – Portugal

February 2016

ACKNOWLEDGMENTS

"Non Mihi Solum."

I would like to thank first my supervisor and co-supervisors: Francisco Ramos, Angela Schwering and Mauro Castelli. Their trust, help and orientation in the realization of this research was indispensable.

I also leave my most sincere gratitude to the Professors Joaquín Huerta, Michael Gould, Marco Painho, Christoph Brox, and Christian Kray for the excellent organization of the Master of Science in Geospatial Technologies and the thoughtfulness they have with their students.

During this master program I have learned from 18 different Professors. Thank you all for sharing your valuable knowledge. I would like to leave a special memory for Professor Ricardo Quirós Bauset. I will never forget your courage, work and love for life.

My gratitude stays as well with the European Union Commission EACEA for the realization of the Erasmus Mundus Programme. You have given me a unique opportunity to grow as a professional and a human being.

Thanks Dori Apenewicz and Karsten Höwelhans, always kind and helpful. Without your support the pace in this journey wouldn't be the same.

Thanks to my 14 master colleagues that I now call friends. We have spent good times together although, but it was in the difficult ones, far from our families, that your presence was vital. I will never forget you.

From Brazil, I will always be grateful to Professor Pastor Willy Taco, Professor Marcus Lamar, and Professor Clovis Zapata. Your support in this project was crucial.

Finally, I thank my family for their love and care. In special, I thank the most important people in my life: Paulo, Lavínia, Leandro and Henrique. I would never have made it without your unconditional love and support.

SCHEMATIC BUS TRANSIT MAPS FOR THE WEB USING GENETIC ALGORITHMS

ABSTRACT

The octilinear schematic map, layout recognized worldwide in metro maps, is an important transit informative tool. This research investigates how algorithms for the visualization of schematic maps can be availed in mobile web devices context in order to empower the efficiency in transmitting information of bus transit maps. A genetic algorithm for path octilinear schematization technique has been used and tested to create the schematic data. Location-based and interactivity functionalities were embedded to the resulting digital maps in order to create personalized maps to meet specific user needs. A prototype of a web application and real transit data of the city of Castellón in Spain was used to test the methodology. The results have shown that real time schematizations open possibilities concerning usability that add extra value to schematic transit maps. Additionally, suggested improvements have been made to the genetic algorithm and performance tests show that genetic algorithms are adequate, in terms of efficiency, to sketch bus transit maps automatically.

KEYWORDS

Transit Map

Schematic Generalization

Octilinear Graph

Genetic Algorithm

Location-based

Digital Map

Web Visualization

Public Transportation

Information Systems

ACRONYMS

API – Application Programming Interface

GA – Genetic Algorithms

GIS – Geographic Information Systems

OSM– OpenStreetMap

POI – Point of Interest

PT – Public Transportation

REST – Representational State Transfer

SQL – Structured Query Language

SVG – Scalable Vector Graphics

UJI – Universitat Jaume I

UML – Unified Modeling Language

W3C – World Wide Web Consortium

WGS - World Geodetic System

INDEX OF CONTENT

INDEX OF TABLES

INDEX OF FIGURES

1 INTRODUCTION

A public transportation (PT) map is a graphical representation of the geographic elements of the services provided by the PT system. It has as fundamental purpose not only to inform the services available but mainly to allow users to elaborate itineraries using these services to reach a specific location. This kind of information tool is vital for the profusion of the PT system usage since difficulties on elaborate itineraries leads people to choose different means of transportation and by consequence, lacks of transit maps contributes to the underutilization of the services (Allard, 2009).

The metro map layout (Figure 1.1), firstly conceived for the underground network in London, is one of the most famous designs used by cartographers to create transit maps. It is a more diagrammatic representation of the network aimed to enhance connections and sequence of stations in sacrifice of precise geographic information, making it a more efficient information tool for transit needs (Avelar, 2002). Because of this unfaithful geographic representation, the metro map can also be called schematic transit map and we also use the term "octilinear" (Nöllenburg, 2005) for this specific configuration. The octilinear layout is widely used for drawing metro maps network, even though scarce because of its complexity, they are also used for bus network.

Production of schematic maps still today requiring the hand work of professional cartographers, therefore, not only a costly task but inappropriate to the modern mobile digital devices. For example, Google Maps, popular digital map app for transit information (installed in more than one billion mobile devices until July 2015), does not provide schematic visualization services. Many reasons are involved in the difficulty of the full automation of the schematic map production. The main reason it is the computational complexity of making an octilinear simplification without dissolving the topology of the network. Secondly as cited by Wolff (2007), is that cartographers use their background knowledge regarding the region to produce a

more pleasant and balanced schematic maps that facilitate the assimilation of information by the user.

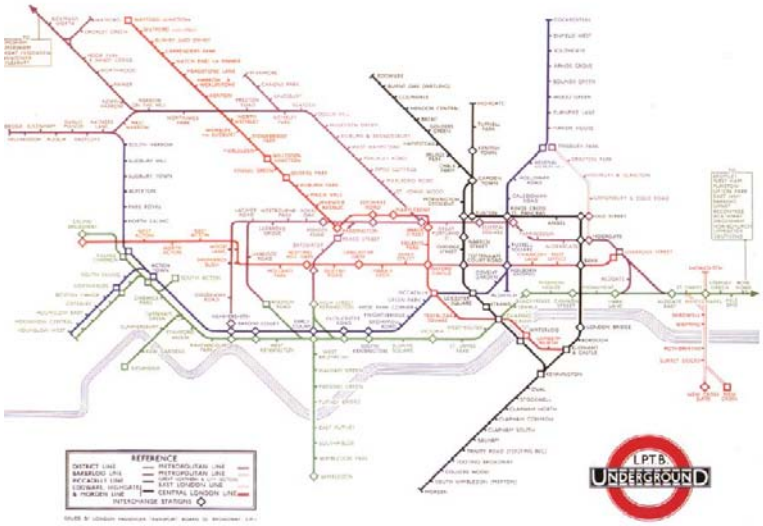


Figure 1.1 First official publication of an octilinear map. Henry Beck 1933.

In the past 15 years, researches addressing the automatic metro layout problem have been prolific, and, with the help of growing computer power, the results are every day closer to results of professional handmade maps. This whole context, together with the emerging mobile communication technology, opens new possibilities for the usage of schematic maps.

This research intends to explore the modern mobile devices interactivity and its location awareness as a platform for the use of automatic schematic drawing algorithms. Based on the visual information-seeking "mantra" proposed by Schneiderman (1996), we suggest that maps can be created on demand with minimal information to meet the specific needs of a person or a group and, by interaction, the user can deep in more detailed information. In other words: Personalized interactive schematic maps.

Static transit maps are usually designed to present complete information of a network. Such a map for a bus network of big city becomes, if not impractical to produce, surely unpractical to use. Imagine an schematic map for the city of São Paulo (Brazil) bus network with its all 1,387 different operative lines (São Paulo

Transporte, 2016). The idea behind creating personalized schematic maps consists in selecting only the parts and lines that might be relevant to a specific user in order to have a cleaner and more readable map. Since schematic maps differs from sketched maps by being a more complete source of information for general purposes (Avelar, 2002). We can also call the personalized schematic map as "sketch map" because they are prepared for a specific purpose.

In order to produce personalized schematic maps we assume that the process of producing schematic data must be made in real time. This efficiency in the process is crucial. First, it helps information in the map to be update to the last changes in the operations of the service. Second and more important is that for different areas and different set of lines it is required a different schematic layout. Changes in the area results in changes in scale and in the dimension of the map. Variation in the set of lines implies density variation in the elements on the map. Since an adequate transit schematization depends of dimension, scale and density, a new schematization is required for every personalized map produced.

We aimed our methodology to bus networks because, first, it is the most common model of PT, second, schematic maps are scarcer for this modality, third because bus networks are usually more complex in its morphology and size, in a way that a solution meant for bus should work for metro or others PT means.

We have improved the genetic algorithms (GA) for octilinear schematization developed by Galvao (2010) to generate the schematic data. The GA for path schematization was chosen because of its control over the execution time. To validate the results in a practical case we developed a web application enabling the remote use of the generated schematic maps. Real transit data of Castellón de la Plana (Spain), a typical medium size European city, has been used to produce the final sketches.

1.1 Objectives and delimitations

The main objective of this project is to explore new usages for real time schematization algorithms in the context of the hypermedia interactivity and mobile

devices. We approach this objective by testing software applications and functionalities that might add extra value to the traditional concept of schematic maps.

The second objective, but as well as relevant, in this project is to identify the advantages and limitations of using genetic algorithms as solution for the automatic generations of schematic maps in real time. To meet this objective we first implemented some of the future work suggestions made by Galvao (2010) to improve the visual quality of the resulting maps and submitted the algorithm to systematic performance test.

It is out of the scope of this project issues related to routing algorithm, journey planner or similar processes. Although some of implemented functionalities depend of this kind of process we opted to use already available external services of directions. It is also out of the scope to develop an algorithm aimed to produce high quality schematic maps similar to the ones made by professional designers. Although, we understand and also work on the concepts of completeness, correctness and elegancy for schematic maps, the main focus of this project is related to schematic map algorithm performance.

1.2 Thesis outline

This research contributes and investigates the computational production and visualization of schematic maps of bus networks for the Web. The structure of the thesis is organized as follow:

Section 2 is theoretical framework required for the development and understanding of the project. It is a brief introduction to the disciplines, techniques and terminology used on the methodology developed.

Section 3 is a review of academic works related to the topic of this research. A brief introduction of the emerging of the topic as a scientific field, then a table format summary of the main academic works related to automatic schematic maps is presented, and third we will analyze the selected works that were more influential to the content of this thesis.

Section 4 is a full description of the methodology from the data preparation, passing through the schematization process itself until the visualization of the schematic data in web. The methodology is always concerning the specificities related to bus networks schematization and interactivity with their components.

Section 5 is the presentation of the results through a novel of the effects of the techniques described in the methodology in the resulting maps. Additionally, it is presented the results of the performance tests made in relation to the schematization process, in special to the GA.

Section 6 is the final remarks concluding the thesis.

2 THEORETICAL REVIEW

2.1 Public Transportation

Transit maps are graphic representation of the main components of a Public Transportation (PT) Network. PT, or mass transit, is a shared modality of transport system that allows the translation of people from one point to another, usually in an urban area on which the vehicle used is owned by third parties. The services might be provided by public or private companies; however it must be available to the general public. The most commons means of public transport are buses, trams, subways, trains or ferries.

According to Vuchic (2005) urban transport influences the shape of cities and their inhabitants' quality of life. Experiments have shown that mass transit has great influence in reduce congestion on urban roads, thus improving the economic, social and environmental traits of a city.

The main component of a PT system is the transport line. The lines are the paths where vehicles travel according to a predefined schedule. A transit line is composed by stops (stations), connections, and terminals. A stop point is a place on the line where the vehicle parks for the landing and boarding of passengers. A connection is a stop that belongs to more than one line. In a connection point, a passenger may perform a transfer from one line to another. Terminal, in this context, can be defined as the start or end stop of a line. A set of transit lines is called network (Vuchic, 2005). Figure 2.1 represents those elements as a network.

2.1.1 Bus Transit Networks

Bus is the most serviceable mass transit modality around the world. Bus lines can use the street infrastructure already existing in the cities, making it convenient and flexible for a PT system to be implemented. However this convenience and flexibility provides the formation of more complexes and multiform networks in comparison with others kinds of PT modalities.

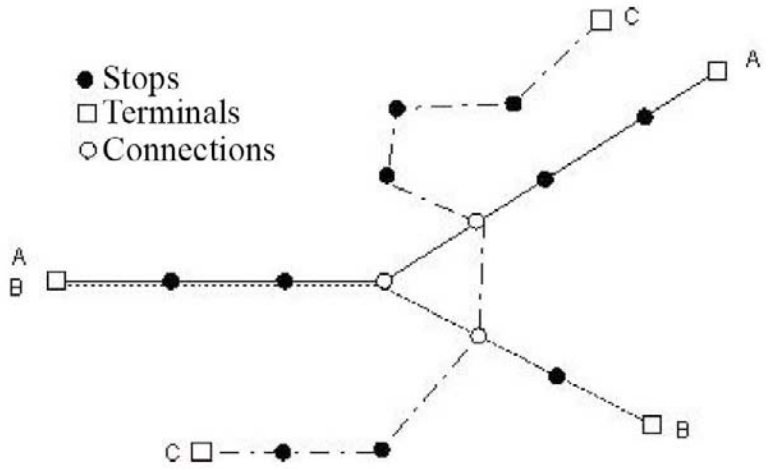


Figure 2.1 Public transportation network components.

That way, bus networks have different characteristics in relation to other mass transit systems, and they need to be taken into consideration by professionals that are working on bus transit information systems. Those differences, listed below might not be exclusive for bus systems but they are more evident in it and together represent its peculiarity (Rainsford, 2002).

Bus networks are dynamic: the frequency, shape, or availability of a line can change throughout the day. For example, night bus lines or periodic route change due to urban roads with reversible lanes.

Integration through walk: connection between lines might require a short walk that sometimes includes street crossing, walk around the corner or to another street to board in a different stop point. Connections might contain two or more stop points (not necessarily in the other side of the street).

Roundtrip lacks of equivalence: the outwards and return routes of a bus lines often are partly or totally different. One-way lines are also possible. This differs from metro lines that both-ways can be represented as a single line. Those lacks of equivalence were illustrated by Huang (2003) (Figure 2.2).

Stop skipping: It is not unusual to see bus lines that ignore a stop point on its route. Some lines intend to be more express by not allowing boarding at some stop points. Moreover the jumped stops might be variable along the day, e. g. rush hours.

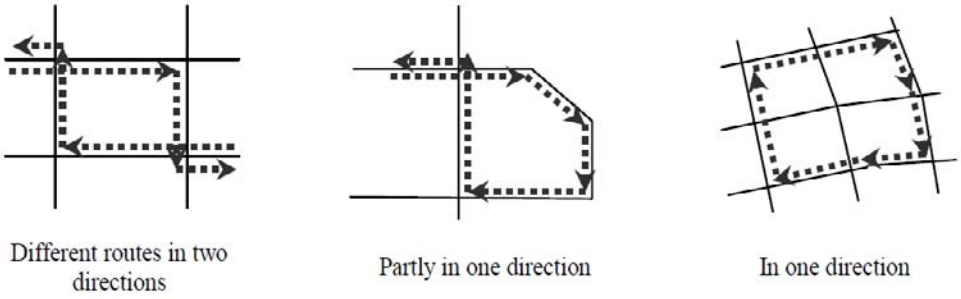


Figure 2.2 Bus line routes way types (Huang, 2003).

These peculiarities represent challenges on modeling and representing bus networks (Huang, 2003). The following section shows on how designers have been dealing with the graphic representation of transit networks as information tool for passengers.

2.2 Public Transportations Maps

Public transportation maps are informative tools useful not only to make people aware of the available services but crucially to provide information on how to use those services efficiently. In a transit map passengers should be able to identify, first, their origin and destination location and finally decide the line or combination of lines that best fits to reach their final destination. Difficulties on performing this task, i.e., elaborating trip itineraries, promote the underutilizations of the transit services available (Allard, 2009).

Public transportation maps can be conceived in three different topological classes: Conventional cartographic (topographic), schematic (or diagrammatic) and hybrid design.

Conventional cartographic transit maps (Figure 2.3) represent the transit elements (stops, lines, connections) in a faithful manner to the real world, e.g. the distances between stop points are preserved. They are also called topological correct maps since the geographical relationship among the elements are correct. Cartographic transit maps serve not only for transit information but could be used as reference for walking routes.

Figure 2.3 Example cartographic transit map (*Consorcio Transportes* Madrid 2009).

Schematic transit maps (Figure 2.4) intend to be a more conceptual representation of a network. They are designed to be more functional in terms of transit. There is no commitment with scale maintenance and the transit elements are simplified, e.g. lines are straightened. However, the topological connectedness must be preserved. The goal of schematic maps is to provide transit information efficiently by allowing easy identification of connections and easy following the sequence of stops in a path and at any rate are meant to serve for walking reference. The most known schematic transit map format is the octilinear, recognized in the metro map of many cities, and more details on it are given in a separated subsection.

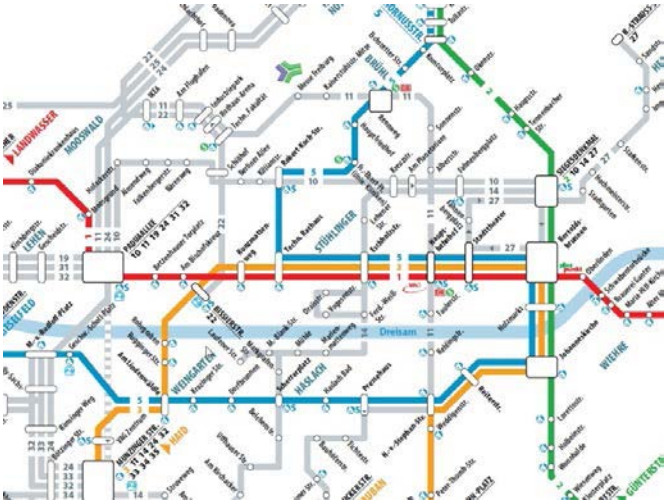


Figure 2.4 Example schematic transit map (VAG Liniennetz Freiburg 2014).

Hybrid design transit maps combine aspects of both, schematic and cartographic (Figure 2.5). In hybrid maps, lines can be straightened and the connectivity highlighted, with simultaneously, being showed together with topographic elements where the general scale of the map keeps mainly constant. Although it is not meant for walking reference, street elements are shown for better localization references. Some hybrid maps designs may be made part schematic and part conventional.

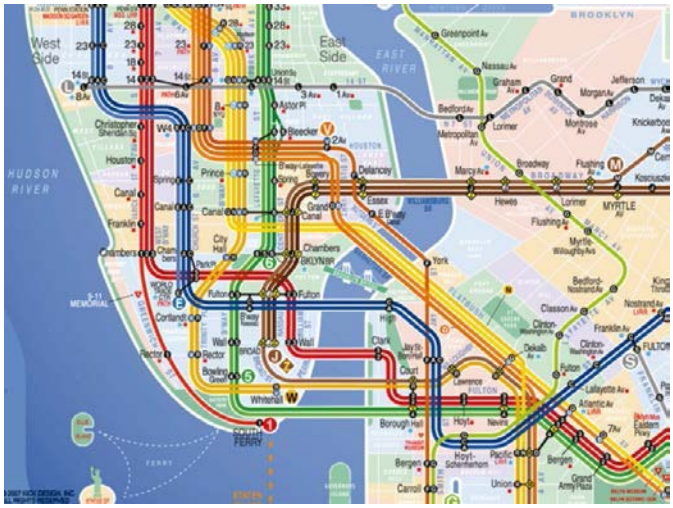


Figure 2.5 Example of hybrid transit map (New York Kick Map 2007).

2.2.1 Schematic Octilinear Maps

The schematic octilinear map, commonly known as the "metro map layout" is the most used diagrammatic format for transit maps. As it was presented previously, it is a functional informative tool for trip elaboration in transit services and it is characterized by having all line segments orientation restrict to vertical, horizontal and both diagonals (octilinear orientations).

The octilinear map emerged in the first half of the XX century when electricity allowed the growth of subway and railroads networks. Realizing that, the available maps were overloaded with information impairing their readability. Henry Beck elaborate in 1933 a new map design for the London subway network, probably, inspired in the digital circuits diagrams used by engineers. The resulting map (Figure 1.1) succeeded. Immediately, all copies of its first edition were out stocked. Over the years, other cities around the world adopted the same design for their networks, and

the same design still being widely used without much modification (Garland, 1994). It turned into a design classic.

2.2.1.1 Rules of Beck's diagram

The most remarkable characteristic of Beck's layout was allowing segments orientated in modulo 45° only, However other common characteristics having been studied and identified in octilinear maps used around. Nöllenburg and Wolff have been listing those rules in the context of automatic drawing of metro maps. The general rule can be listed as follow:

- Do not change the network connectedness topology. It means, the planarity of the network must remains consistent. Extras edge crossings should be avoided.
- Restrict edge orientations to the octilinear angles (0°, 45°, 90° 135°, 180°, 225°, 270°, 315°)
- Ensure that adjacent and non-adjacent stations keep a certain minimum distance. And, preferably, keeping distances between adjacent stations as uniform as possible.
- Keep minimal the number of bends along the lines. If bends cannot be avoided, obtuse angles are preferred over acute angles.
- Preserve the relative position between subway stations. For example, a station being north of some other station in reality should not appear below that station on the map.
- Make sure that dense regions of the map get a larger share of the available space.
- Color each edge according to the lines to which it belongs. This assumes that each line has a unique color, and if many lines share a same edge, a parallel for each line must be draw.
- Label stations with their names and make sure that labels do not obscure other labels or parts of the network.

2.2.2 Digital Transit Maps

Digital transit map is the representation of spatial transit information in digital devices. Nowadays, the most used digital devices for passenger looking for transit information are laptops, tablets and smart phones, and in this context, digital maps can also be called mobile maps. Navigation, route finders and other location-based services are consolidated applications for this kind of devices.

Digital maps have brought many benefits to map makers and map users when compared to traditional paper maps. Digital maps can be more adaptive; they are

easy to reproduce, update, edit and personalize and data sets from different source can easily be joined together to create a new map. Digital maps provides new ways of interactions and visualizations and it can be integrated with others services. And, with the emerging of mobile devices, a digital map can now be used on the go.

Together with these benefits new challenges emerged on producing transit maps in the mobile map context. Mobile maps, in contrast with traditional cartography, offer only a small screen to present the necessary information which implies difficulties in presenting the complete information with clarity and readability. Allied to it, despite recent progress, mobile devices have limitations regarding memory, processing and communication bandwidth.

GIModig (2015) addressed the challenge of real-time adaptive generalization of geospatial data to small screen devices (Sarjakoski, et al., 2002). Generalization are functions that simplify, aggregate and reduce geospatial information in map representations and, according Sarjakoski, et al. (2005), those methods are essential in the mobile maps context because increases the readability of information, what is vital for small-display cartography. Others techniques can also be cited as adequate for small display cartography, like angular schematization, fish-eye view and off-screen visualization.

Angular schematization like generalizations, aims to reduce unnecessary information, and emphasizes some aspects of the topology (Avelar, 2002) , thus generating a cleaner, summarized and easy reading map. For this reason it can be used in the context of mobile maps.

Fish-eye view is variable-scale visualization technique analogous to a magnifier glass (Figure 2.6). The idea of this approach is that a focus area (usually center) takes progressively space from peripheral areas. This causes the congested areas of the map to be presented in larger scale and, at same time, to keep the surrounding areas in the visual bounds but in smaller scale. This technique, besides increasing the readability of the features in the focus area, it avoids extras zooming and scrolling by the user (Plaisant, Carr, & Shneiderman, 1995) .

Figure 2.6 Variable scaled map of Boston's road map. (Haunert & Sering, 2011)

Off-screen visualization is a technique that gives indication of off-screen objects locations by cues in form of geometry (arrows, triangles, circles) arranged in the border of the screen which allow users to recognize the locations of the objects outside the limits of the map. This approach allows users to pan and to zoom the map without losing the context of the map viewport and the objects of interest. Figure 2.7 illustrates an example of off-screen object visualization techniques.

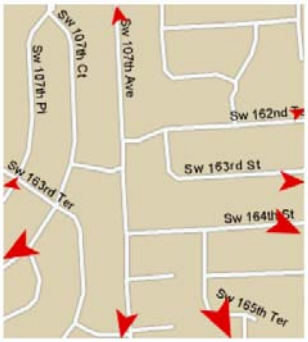


Figure 2.7 Off-screen objects located by Arrow. Distance proportional to arrows size. (Burigat & Chittaro,, 2011)

2.3 Graph Theory

Transport networks can be naturally represented as graphs where the vertices represent stop points and the edges the topological connections between the stops.

A simple graph denoted below as G, is a structure formed by two kinds of objects: vertices and edges.

$$G = (V, E)$$

Where V represents a finit set of vertices, and we say that $|V| = n$. And E represents the set of edges of the graph, and $|E| = m \; and \; E \subseteq V^2$

One can say that an edge is an unordered pair of vertices. An edge connecting the vertices v and w is denoted simply by vw or wv, in this case we say that the edge falls in v or w, and v and w are neighbors or adjacent vertices.

The set of all vertices adjacent to vertex v is called neighbors of v, and can be denoted by $N(v)$. The degree of v is $d(v) = |N(v)|$, i. e the number of vertices adjacent to v.

2.3.1 Adjacency list

In graph theory, an adjacency list is a data structure that represents all the edges of a graph in a list. It is a data format for a graph.

Each entry in the adjacency list corresponds to a vertex v in the graph, followed by the set of vertices adjacent to it. The graph in Figure 2.8 can be represented by the adjacency list in the Table 2.1.

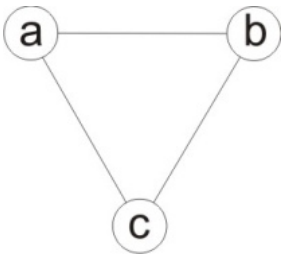


Figure 2.8 Simple graph example.

VERTEX	ADJACENT VERTICES
A	b, c
B	a, c
C	a, b

Table 2.1 Adjacent list example

2.3.2 Path, cycle and cover

A path P in a graph G is a sequence of distinct vertices $(v_0, v_1, \dots, v_k)$ of G, such that $v_i v_{i+1}$ belongs to E for all $i = 0..k-1$. The length of this path is represented by $lhp = k$ (number of edges). A $vw - path$ it is a path that starts at v and ends in w, if the $vw - path$ exists, it means that v and w are accessible. A $vw - path$ added with the vw edege i called cycle.

A set $\mathbb{L}$ of path and cycles in G is called coverage, if for all v of V_g, v belongs to at least to one of the elements of $\mathbb{L}$. In this case is said that $\mathbb{L}$ covers G.

2.4 Graph Algorithms

In order to examine all the vertices and edges of a graph G systematically some graph search algorithm is required. There are several known types of search algorithms. Each search algorithm is characterized by its search strategy of the elements, the order the vertices are visited, the data structure used and the results obtained at the end of the algorithm execution.

2.4.1 Depth-first search

Depth-first search performs a progressive search through expansion of adjacent vertices, and deepens more and more until it encounters a vertex that has no other adjacent vertex that has not yet been visited. So, search goes back to the previous vertex and resumes the search for vertices not yet visited.

As a result, after the execution of the algorithm, all vertices are visited in order that allow the given graph to be divided in a minimum set of paths. This property is especially useful to create a topological data structure of ordered vertices, what is especially useful in the context of this project because it let us treat the graph in separated paths. Algorithm 2.2 is a depth-first search algorithm that makes use of a stack S to control the visited order of the vertices.

```
1   DepthFirstSearch(G,r):
2      create empty stack S
3       i = 0
4      S.push(r)
5      while S is not empty:
6            current = S.pop()
7            if current was not visited:
8                current.order = i++
9                for all vertex v adjacent to current:
10                    S.push(v)
```

Algorithm 2.1 Depth-First-Search pseudo code.

2.5 Graph Drawing

A convenient way to understand the graph is to represent it geometrically. Graphs drawing deals with the task of representing geometrically a graph to better meet their practical needs. Typically, the obtained drawing should be as readable and clean as possible. The practical value of graph drawing is so pertinent that it owns its unique discipline within the computer graphics field of study.

The graph drawing is motivated by study fields such as topologies organizations, electrical circuits, software engineering, paleontology, sociology and pretty much everything that deals with entities and their relationship (Battista, Eades, Tamassia, & Tollis, 1998). For instance, the Figure 2.9 shows how Arcmap takes advantage of graph drawing to represent models of complex geoprocessing tools.

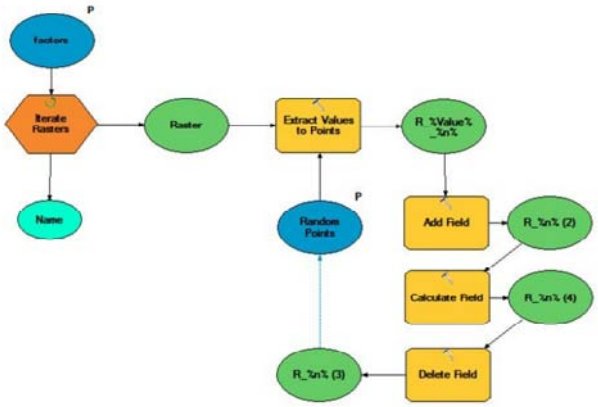


Figure 2.9 Graph drawing application example.

A drawing Γ of a graph G is a function that maps each vertex v of G to a coordinate in a space by Γ(v), and the polyline format of each edge vw of G by Γ(vw). If no drawing Γ(vw) of any distinct edge vw intersects we say that the drawing Γ is planar. If a graph G accepts a planar drawing, we say that G is a planar graph.

In order to formalize the good drawing concepts for graph drawing algorithm, the drawing properties of a graph can be classified into drawing conventions, aesthetics, and constrains (Battista, Eades, Tamassia, & Tollis, 1998). The conventions are rules that the drawing must satisfy, for instance: all Γ(vw) must be octilienar. Aesthetics is a property that a drawing pursues as much as possible, for instance: maintain the

length of any Γ(vw) as uniform as possible, even if the drawing does not accept all the edges to be at the same length, the variance of the lengths must be minimized. Constrains are rules or criteria of the drawing that are applied to only part of the graph, for instance: draw the given path P of G as straight as possible

2.6 Genetic Algorithms

This section is a basic introduction to the concepts of Genetic Algorithms(GA). We use GA in our methodology to simplify paths in octilinear configuration, and the terminology here introduced is used contently in further sections.

Many computational problems require a search for a solution within a wide range of possible solutions (Melanie, 1999). Genetic algorithms (GA) are a search heuristic method for those kinds of problems, suited when optimizations are not implementable or performance is a requirement. The solution found by GA is usually considered an adequate solution, being impossible to prove that it is an optimal solution.

GA are search algorithms inspired by the mechanism of natural selection of Darwin. The basic concept of natural selection is that favorable heritable traits become more common in successive generations of a population, and the less favorable characteristics become less common. Darwin's argument is that individuals with favorable phenotypes are more likely to survive and reproduce than those with less favorable phenotypes, so the genomes associated with favorable phenotypes are in greater numbers in the next generation of a population. Over successive generations, individuals have produced a high degree of adaptation to an ecological niche because their favorable genes passed from their ancestors.

Basically, what a GA does is to create an initial population of possible solutions to a problem and then the solutions of a population are used to create a new generation of solutions. This process is driven by the expectation that a population has better solutions than the previous generation. This expectation stems from the act of choosing the most appropriate solutions to produce the next generation of solutions.

This process is repeated until a criteria like found good enough solution or a time limit was reached.

The first and most important step when using a genetic algorithm is to choose how to represent an individual in a population (Melanie, 1999). An individual is the fundamental unit of a genetic algorithm since they represent a possible solution for the given problem.

In genetic algorithms, individuals are referred by the term chromosome, i.e., a set of genes. The most elementary way to encode a chromosome is through a bit string, but depending on the nature of the problem other forms of encoding genomes may be used. Just for illustrative purposes, Table 2.2 shows the representation of two chromosomes encoded in bits string.

NAME	GENES CODE
Chromosome *A*	1011000100110110
Chromosome *B*	1101111000011110

Table 2.2 Example of encoding chromosomes.

Once the format of a chromosome is defined, an initial population of chromosomes should then be generated to be submitted to the evolutionary process that consists of the following steps:

Evaluation: This step consists in verifying the suitability of chromosomes as a solution to the problem. The evaluation is usually made by a fitness function. The fitness function takes a chromosome as input, and outputs an index used to assess how well the chromosome fits as a solution.

Selection: This step consists in identifying the chromosomes that will be selected for breeding, it means, to select the chromosomes that shall pass their genes to the next generation. The probability of a chromosome to be selected must somehow proportional to its suitability as a solution to the problem. The suitability of a chromosome is made in the evaluation process. There many techniques used for selection, each one are best suited for a kind of problem.

Crossover: The crossover process is analogous to the process of chromosomal crossover in the sexual reproduction. It is important to promote the diversity of the individuals in the next population by the exchange of genetic material. This operation consists of two chromosomes recombine randomly generating new chromosomes. Table 2.3 illustrates a crossover between chromosomes and the new generated individuals.

NAME	GENES CODE
Chromosome A	10110|00100110110
Chromosome B	11011|11000011110
Chromosome A'	10110|11000011110
Chromosome B'	11011|00100110110

Table 2.3 GA crossover example.

Mutation: This operation consists of randomly change some of the genes in the chromosomes. Table 2.4 illustrates the mutation of two chromosomes generated from the crossover in the previous example.

NAME	GENES CODE
Chromosome A'	1011011000011110
Chromosome B'	1101100100110110
Mutated chromosome A'	1011010000011110
Mutated chromosome B'	11111 00100110010

Table 2.4 GA mutation example

Update population: After the selection, crossover and mutation operations are completed, a new set of chromosomes will be created. This step consists on establishing a next generation from this new set of chromosomes.

Stop criteria: After the population of the new generation is updated, a test should be made to order to check if the stop criteria was satisfied or not. The stop criteria might be a limit in time or in generations passed, or even a set of constrains to be satisfied.

If the test is satisfactory, the process stops, and the best evaluated solution is selected as final solution, otherwise, the GA returns to the evaluation step.

The above described process consists of the basic operations of a genetic algorithm. The techniques chosen to be implemented of these operations will depend on the nature of the problem. For example, if the initial population is generated in a uniform or non-uniform manner, one can also opt for elitism where n best individuals are always kept to the next generation, or even defining the probability of a gene to be mutated. One way or another, the GA are very adaptive, and they are applied in fields like bioinformatics, game theory, scheduling problems, etc.

3 RELATED WORKS

Computed cartography emerged from the automation of processes usually done manually by cartographers in the past. The evolution of such processes like generalization (line simplification, smoothing, aggregation) are now part of the Geographic Information Systems (GIS) field of study and contributed to the arise of many digital maps tools in the Web, like OpenStreetMaps (OSM) or Google Earth (Allard, 2009). However, digital web maps that provides topological information in schematic layout still scarce.

In the past 15 years many researchers have been made addressing the "The metro layout problem" (Hong, Merrick, & Do Nascimento, 2005). Each research address the problem in a specific way for a specific purpose and the techniques developed are as diverse as possible.

The Table 3.1 lists some authors and their method of schematization. This variety of academic research shows, as said by Allard (2009), how prolific the automatic generations of schematic maps of public transportation has been as a study field.

Each method has its pros and cons. Besides the technique used, we include extra information about the purpose and results obtained. We classify them by the kind of algorithm if it is aimed for high quality maps, if the correct topology is guaranteed, if it's aimed for real time schematization, if all edges are octilinear, and the aimed transport mode. More details and comparison of results can be found in surveys like by Wolff (Drawing subway maps: A survey, 2007) or Nöllenburg (A Survey on Automated Metro Map Layout Methods).

In despite of the works mentioned on Table 3.1, there are more academic papers related to topic, and it will be unnecessary to review all of them in the scope of this document since topic here is slight different. However, it's worthy to mention the more relevant ones for this research.

AUTHOR	ALGORITHM	HQ	TOPO.	REAL TIME	FULL OCT.	MODAL
(Avelar, 2002)	simulated-annealing	No	Yes	No	No	Bus/Metro
(Cabello, de Berg, & van Kreveld, 2005)	combinatorial approach	No	Yes	Partially	Yes	Bus/Metro
(Stott & Rodgers, 2004)	hill climbing	No	Yes	No	No	Metro
(Hong, Merrick, & Do Nascimento, 2005)	spring embedded	No	Yes	No	No	Metro
(Nöllenburg, 2005)	mixed-interger programming	Yes	Yes	No	Yes	Metro
(Merrick & Gudmundsson, 2007)	*C*-Directed Path Simplification	No	No	Yes	Yes	Metro
(Galvao M. d., 2010)	genetic algorithms	No	No	Yes	No	Bus
(Li & Dong, 2010)	stroke-based	No	Yes	Not reported	Yes	Bus
(Wang & Chi, 2011)	least-square	Partially	Partially	Yes	Yes	Metro
(Ti, 2014)	scale-adaptive stroke-based	No	Yes	Partially	Yes	Bus/Metro

Table 3.1 Summary of researches dealing with the metro map layout.

Avelar (Avelar, 2002) presented in her doctoral thesis one of the first complete studies about computational octalinear maps for public transportation. The relevance of this work extends the algorithm in itself. Even being a computational research, important design aspects are covered. Like definitions of schematic maps, map styles classification, symbology conventions and design considerations for aesthetically pleasant maps. Moreover, Avelar handles data modeling aspects essential to the topological characteristics of transports maps and objects of interest.

Regarding the algorithm, (Anand, Avelar, Ware, & Jackson, 2007) , Avelar's method first simplify the lines using Douglas-Peucker, then uses simulated annealing to iteratively move the vertices to conform with octalinear positions without violating the network topology. The algorithm aims to create octalinear schematization of road networks. The presented results showed success in preserving the topology of the

Zurich's road network and having most of the edges in octalinear positions (some edges could not be octalinear placed). However, in terms or performance they were not adequate for real time solutions.

Nöllenburg & Wollf (2006) pushed forward the field of automatic generation of schematic maps. Since they aimed their research to high quality results, their academic works are full of orientations on how to pursue a high level design. Nöllenburg also proved elegantly that the metro layout problem is part of the NP-Complete class of decision problems (Nöllenburg, Automated drawing of metro maps, 2005). It means that the time to solve this problem grows exponentially proportional to the size of the instance. These results drive computer scientist whom deals with the problem to use heuristics searches, treat instances differently, change the definition of the problem, or to use optimizations.

Given this knowledge, Nöllenburg (2006) models the problem in mixed-Integer programming (MIP), a mathematical programming that uses linear and integer constrains. That way, and using a commercial MIP solver design for efficiency (CPLEX), Nöllenburg obtained results of visual quality similar to ones made manually by professional designer. Additionally, his solution was able to create space to label all stations properly. However, even using CPLEX, the execution time cannot be guaranteed to be short, varying from couple of minutes to hours depending on the complexity of the network. More recently, increments on the of the MIP solution, like presented by (Oke & Siddiqui, 2015) succeed on reducing the time of execution.

Galvao (2010) implemented a genetic algorithm for path simplification in an octalinear format. The genetic algorithm intents to find an octalinear format for a polyline in a way to avoid change of directions (especially acute ones) and at the same time to avoid distortion and displacement of the stations. However, different from the works cited above, the genetic algorithm doesn't guarantee the correct topology when applied to network graphs (especially complex ones) because it treats the paths individually. The genetic algorithm solution fits the purpose of this work because the execution time of the algorithm can be controlled, thus suiting for

schematization on demand in web, which required real-time solutions. More details on the genetic algorithm will be given in the section 4.3.2.

As it was previously explained, this work does not have as primary objective the development of algorithms to the automatic drawing of schematic maps, instead is about how those algorithms can be availed in the interactive environment of remote web devices like laptops, tablets and smart phones in order to add value to automatic generated personalized maps for bus transit networks. Although the number of publication about algorithm in itself has been substantial, the same doesn't apply to "on the fly" schematic maps.

Swan et al (2007) adapt Avelar's simulated annealing technique to a prototype to generate schematic polylines data as a Web Feature Service (WFS), an Open Geospatial Consortium (OGC) service specification for geometry features. Although the paper lists some practical applications of the web service, nothing is said on how the schematic information can be presented on web clients.

Wang and Chi (2011), as far it is from our knowledge, are the first to aim the research of automatic schematic maps into personalized solution for mobile devices. They assume that for better readability in small displays, the route of interest of the user should be more prominent in the resulting schematic maps. They use energy function to give priority to a specific route and find the final layout by a least square method. Their results prove that a real time solution for the problem is able to add extra value to schematic maps, opening a new door of possibilities of functionalities to be explored in schematic maps. However, they did not test their solution in web environment.

4 METHODOLOGY

In Section 2 we have explored fields of study that serve as background for the automatic generations of personalized bus transit maps. From theoretical topics of public transportation, passing through digital cartography and graph theory, the approached topic of this study characterizes itself by being multidisciplinary. This section describes the implementation of the methodology used to obtain the final results. First, we describe the steps to get the data ready to be consulted, second, we describe process related to query the data based on the user given information, then we show the details of the process used to transform the geographic data into schematic data and finally we describe how the schematic data is passed to a web client (browser) and the details of the web client implementation aimed to better present and control the behavior of the schematic.

4.1 Getting Data Ready

This section mainly concerns about relevant issues regarding the treatment of the spatial data necessary to produce schematic data, it means preparing geographic data to represent network topology. We discuss as well concepts specific for bus services map information systems.

4.1.1 Data modeling

The data model delimits the boundaries of the operations that can be performed and the results that can be obtained by a computer system. It describes the way real word or abstract elements will be represented by a computer, being the domain where the application can translate the data into the maps. The resulting UML scheme illustrated in the diagram of the Figure 4.1 describes the geographical elements as well as the topological connectedness attribute of the transit networks required for the automatic generations of schematic maps in this project. The geographic entities are stereotyped with their respective geometry, and all its spatial attributes are stored in the WGS 1984 standard.

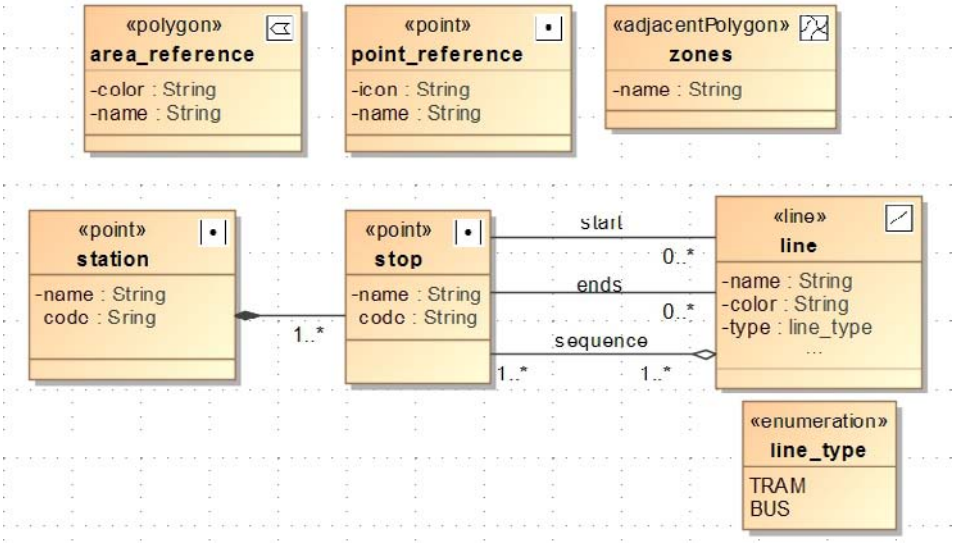


Figure 4.1 Application data model.

The *line* (paths where vehicles travel according to a predefined schedule) is the main component of a public transportation system. Its main attributes are its name, its color for its visual identification on the map, and its type, that represents the transport model performed in the line, for our case they are bus or tram. The *line* is also stereotyped as geospatial feature (geometry) line to store the path performed by the vehicles in the transit line. It is worth mentioning that this line geometry is not used to create the final sketched maps at all, but rather the ordinal sequence of stop points is used. Figure 4.2 illustrates the difference between the line route (in blue) and the polyline that connects its sequence of stops (in red).

A transit *line* is a composition of one or more ordered *stops*. A *stop* (place on the line path where the vehicle parks for landing and boarding of passengers) has as attributes its name and code, and its geometry point that indicates its geographic coordinates. It is worth mentioning that this many-to-may relationship between line and stop will produce a third table in the database logical model.

Due to the lack of connectivity of outward and return path of bus lines, many proposed data model for bus networks opt for dividing a line as a set of two directional paths, one path representing the outward and a second representing the return (Huang, 2003) (Rainsford, 2002). However, we opted to model the *line* with both directions without separating it in two, it means that a line is defined by the

sequence of stops of its route until the first stop has been reached again. This way any morphology of bus route can be represented in our data model. In order to identify the end of the outward way, each line is related with a start and end $stop$. The $stop$ point where the line ends, means the end of the outward way. In case of circular lines the start and end $point$ are the same.

Figure 4.2 Line route (blue) vs. Line sequence of stations (red).

Another specific known concept for bus network data model is the concept of the, here called , $station$. Called "link" by Rainsford (2002), and differentiates from "route stop" (here stop points) as "bus stop" by Huang (2003), the $station$ is a set of one or more $stop$. The stops belonging to a same set are within a minimal walk distance x from each other, and a stop is related to one and only one station. Additionally, any two stop of the same station cannot serve in the same line route in consecutively sequence. It means that no vehicle will stop consecutively in two stops belonging to the same station.

The attributes of $station$ are its name, its code, and its point geometry. The point geometry of $station$ usually correspond to the geographic coordinates of center of the polygon formed by its $stops$.

The station concept becomes necessary for bus networks, first, because it represents stops with potential for transfer between lines and second because it enables us to identify as a single stop two stops on opposite sides of street where a line share its outward and return way. It is important to preserve both, $stop$ and $station$ in the

data model because stops represent the exactly position of the stops in a route and are useful for locating them in cartographic reference maps, and the station holds the connectivity of the network allowing its integration in a journey planner, the point, as a result, the station geometry is more useful for schematic transit maps.

Line, stop, and station form the group of the essential entities in our data model. The $point\_reference$ and $area\_reference$, and $zone$ are not essential part of the transit information. However, they are considerable references for sketched maps because they help providing agility to human self location. The $point\_reference$ has point geometry as attribute because it represents a point feature that can be included in the map, for example, a city hall, a train station, or a cathedral. A point_reference comes as well with its name and a link to icon image as attributes. The area_reference has polygon geometry as attribute, because it represents a polygon feature that can be included in the map, for example, parks, a campus. A $area\_reference$ come as well with its name and its color as attributes. The $zone$ has the same attributes as $area\_reference$, but they are different entities, $zone$ is a bigger area that delimits city parts as neighborhoods, peripheries, or satellite cities. It is drawn below every other entity in schematic maps and are many times used not only as geographic but as tariff reference for the system.

4.1.2 Data collection

This project has used real data from the bus and tram transit network of the city of Castellón de la Plana, Spain, to testify the here proposed methodology. Usually, geodata collection is one of the most time consuming processes of a geospatial project. Fortunately, the raw data used in this project was already collect by the Geotech Institute in UJI, still, the data format indicates that the collection was made in two steps: stop points collections and line route collection. Stop points collection consists into attribute a name and register the geographic coordinates of each stop. The line route collection consists in, for each line, recording consecutives geographic coordinates in the vehicle during the full trip of a line. The recorded consecutives coordinates will form the polyline of a line route.

The coordinates of the stops and the polylines of the lines form the set of raw geospatial data necessary for the generations of schematic maps. The Figure 4.3 is the geo visualization of the whole raw data overlaid in an OSM base map. In total are 258 stops (green points) and 20 lines (blue polylines).

Figure 4.3 Castellón transit raw data.

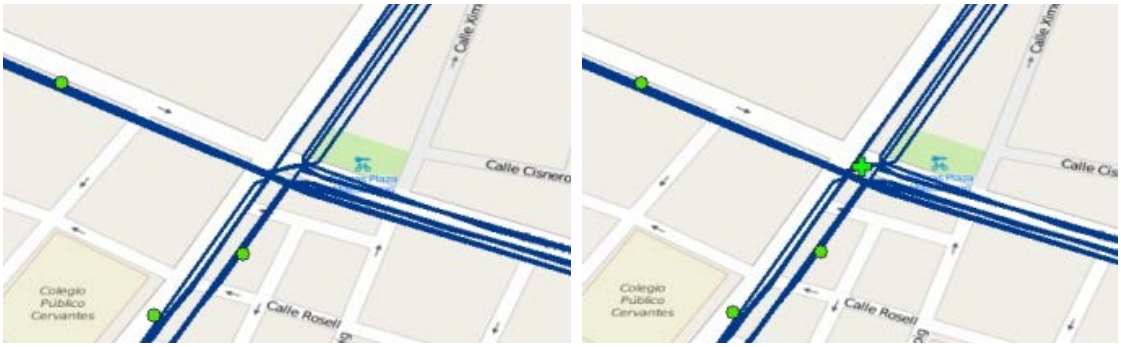
In the next session it is described how to process this data to feed the data model presented in the previous section. It shows how to link the lines with their respective sequence of stops and how to create the stations with their set of stops.

4.1.3 Topological data preparation

With the raw geospatial data (polylines of lines and points of stops) some geospatial process can be performed in order to, first, create the topological order of stops of each line and second, group the stops into stations. Before we proceed to those steps it is important guarantee that the resulting network topology will be planar graph. A planar network is important because facilitates a more precise drawing of network in terms of its topology.

The process of planarization of a graph consists basically on adding a fake vertex where edge crossing cannot be avoided. So, in order to obtain a planar graph of the proposed data model, it makes necessary to add a fake stop at any line route or junction crossing that is not already connected by another stop. The Figure 4.4 illustrates such a case where 7 different lines intersect themselves. A new stop, the

green cross, was created to represent topologically this connection between the lines. Those fake stops are set with the name "ghost" to indicate that they should not be represented with a symbol in schematic maps. A total of 19 ghost stops were necessary to be created to guarantee a planar graph, increasing the number of stops from 258 to 277.



a) **b)**

Figure 4.4 Creation of ghost points to planarize graph.

The next step is the creation of relations between lines and stops (including the ghost ones). The process to relate each line with its sequential group of stops consists of, for each line l, selecting from the table of stops, the stops that intersects the polygon route of l. The Figure 4.5 illustrates the selected stop points (in cyan) intersected by a specific line. The ids (primary key) of selected stops were used to form a new table that is added with a column with line id that they intersect. A third column with the order of the stops visited by the line on its full route (outward and return) is then added to the table. The tables created by each line are merged to form a new relational table between lines and stops that allow the representation of the connection topology of the transit network. The resulting logical relational model tables are:

LINE(lineId, name, color, vehicletype, polyline, startpointid, endpointid)

STOP (stopId, name, point, stationId)

LINESTOPSEQUENCE(stopId, lineId, order)

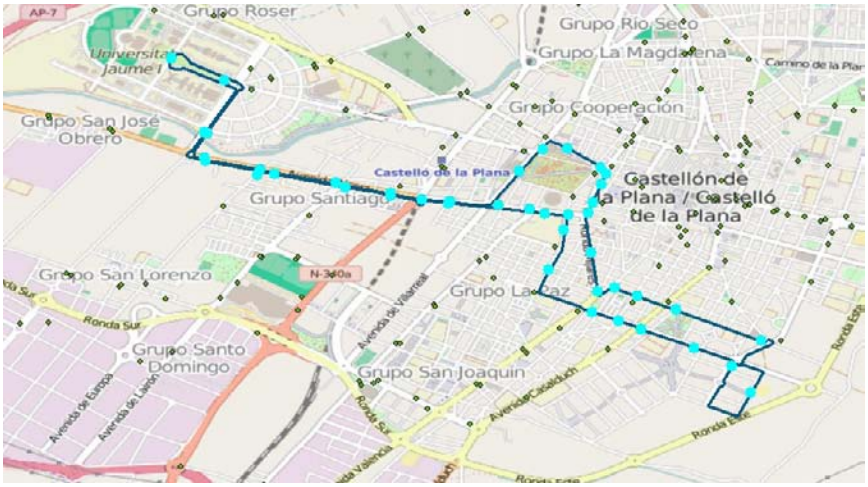


Figure 4.5 Selection of stops in a line route.

The creation of *station* is a similar process to the process of identifying connections between lines in journey planning algorithms in transit networks. The stop points within a determined walking distance from each other are grouped, after that, the most distant stops from a same group serving in the same line in a consecutive manner are removed from the group to form an independent station.

To accomplish that and feed the data model with stations, firstly, a 30m (halfway of the maximum walk distance) buffer was applied to each stop (blue circles in Figure 4.5 b). The buffers of the stops are dissolved to form a single polygon if they are intersecting each other (pink polygons in Figure 4.5 c). Each stop will receive the id of its dissolved buffer that will correspond to its *station* id. Finally, the dissolved buffer is converted to a point geometry feature in manner that its point geometry is located in the center of it (the red dots in Figure 4.5 d). This point geometry will represent the station on our data model. If two stops belonging to a same station are consecutive stops of a line, one of them (usually the one further from the center) must be removed from the group and a new station must be created for this stop.

a) **b)**

c) **d)**

Figure 4.6 Creation of stations.

The attribute name of each station was taken randomly from one of its stops. As a result, the station of a ghost stop will also be "ghost". At the end, every stop, including the ghost stops, is associated to one and only one station, and every station is associated with one or more stops. This way a line can be expressed not only as sequence of stops, but as a sequence of stations as well. After this process the total number of stations is 173 (not counting ghosts) against 258 stops. The final resulting logical relational model tables are:

STOP (stopId, name, point, stationId)

STATION(stationId, name, point)

The *point_reference*, *area_reference*, and *zone* (the rest of entities of our model), are not relevant for theoretical process of automatic creation of schematic transit maps described in this project. They are not topologically connected to the

others transit entities (although they could be), instead they are present only to server as location references in the schematic maps. It is worth record that geospatial data for those entities were created manually using a GIS with the support of a base map. For the $point\_references$, features as the cathedral, train station, and the general hospital were model. For the $area\_reference$, the features modeled were the central park, the campus, and the Mediterranean Sea. For $zone$, the historical city center, and the boundaries of the city were modeled.

4.2 Location Based Data retriever

In the previous sections was proposed a data model and has been showed how the spatial data can be collected and prepared to feed the data base. This section consists in presenting techniques on how the proposed relational data model can be queried in order to obtain the correct data structure that will enable the production of schematic maps. Retrieve the right data and in the correct other is critical for the quality of the resulting schematic maps.

Bus transit networks are usually bigger and more complex then subway networks. For example the bus network of Wuhan, a 7 million citizen city in China, had 240 bus lines in 2003, and many of routes overlapping each other (Huang, 2003). The creation of an easy to read transit schematic map with all 240 lines is a challenging task even for the most skilled cartographers. For the automatic generation of schematic maps a bigger size of the instances represents more data to be transferred through web, and, especially for schematization, considerable more data to be processed. Moreover, a good readability cannot be guaranteed.

The concept proposed in this work states that the user should receive a minimal but sufficient set of information. Therefore, the data to be transfer and processed will be reduced and the resulting map will become more clear and personalized.

Usually a routine of a person in a city is limited to a few spots. For example, a student goes from home to the campus, from the campus to the sport center, a from the sport center to home. A tourist goes from the airport to the hotel, from the hotel to a monument, from a monument to the hotel. Why it makes necessary the whole set

of 240 lines be presented to tourist in Wuhan? Digital schematic maps allow us to construct maps with a limited set of information that will attend the needs of an individual. This is more or less the concept of the handmade sketch maps as an alternative to the all-in-one paper maps.

In this section we show an example of how a specific set of lines can be determined to a user. We describe as well how this set of lines is used to query the data base in order to obtain the correct data structure for the generation of schematic maps. Moreover, we introduce a concept of not only reducing the number of lines to be presented, but omitting as well part of the line that are not useful to connect the POIs of the user.

4.2.1 Indentifying lines using Google transit API

To identify the set of lines that are valuable to a user, first, it makes necessary to identify the points of interest (POI) of this user. The POIs are usually locations in the city where the user often goes or intends to go. This data can be obtained from mobile devices in the favorite places stored or current locations of the user for example. The prototype made for this project allows the users to manually add markers to a map indicating their POI. Figure 4.7 exemplifies the selection of three POIs (red markers) by a specific user.

The next step consists in using the POIs to obtain the transit lines that connect them. No routing algorithm was developed to this project because is out of its delimitation, instead the *DirectionService* provided by the Google Maps JavaScript API has been used. This direction service allows transit lines to be set as travel mode and, for every steps of the found route, if a transit line has been used, it is possible to identify the proposed line name.

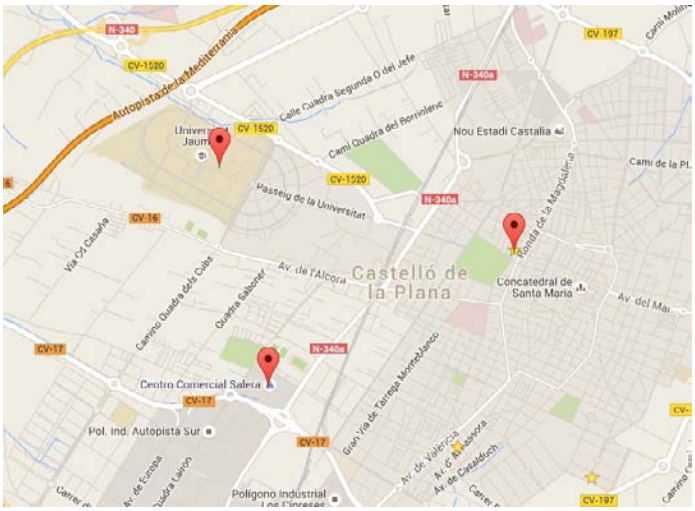


Figure 4.7 Selection of POIs.

The routing algorithm is made between the POI. Consequently, the number of POI increases squared the number of requests of routing because it is a simple arrange in two. Since routing algorithms are usually costly processes, is better keeping the number of POIs small. Even the Google API limits the number of routing requests per second.

The lines names found by the routing algorithm are collected and the resulting line set and POIs are the parameters to execute the queries in the database.

4.2.2 Querying Data

The line names and the positions of the POIs are used to form the URI parameters of the HTTP GET method in a REST architecture. The line below exemplify the URI of a GET method containing four lines (L16, L10, L11, L7) and three POIs (Home, Universitat Jaume I and Commercial Center) as parameters.

```
http://localhost:9080/pubrest/network/x?line=L16&line=L10&line=L11&line=L7&poi=Call
e Universitat Jaume I(-0.06737709045410156 39.9937584258632)&poi=Home(-
0.04248619079589844 39.988234687450756)&poi=Comercial Center(-0.06325721740722656
39.97955362458796)
```

In possession of the parameters, the server application can use them to query the database and obtain the necessaries data structures to generate the schematic data. First, the server application instantiate lists of objects for all the entities of our data model. Those are straight forward query consulting to the data base, and should

result in a list of stops, stations, lines, references and zones. In order to have a spatial sequence of the geographic elements the stops, and station lists are queried ordered by the distance of the focal point selected as the city center. This order will imply later in selection order of the schematization algorithm.

The next two necessary data structure required for the schematization are, first, the adjacency list and second, the list of edges.

The list of adjacency represents the connection topology of the network. It shows which stations are direct linked. In other words, an entry in adjacency is a station and its adjacent stations, it means, a list of all pair of stations that are in consecutive order for specific lines.

The query to obtain the adjacency list of stations will be a relational join between a pair of joins between station and stops joined with line. The join between line with the pair station/stop is made by the relational table *linestopsequence*. Finally to relate stations only with the ones adjacent to them a condition implying consecutiveness must be stated. It means that the difference of the order in a line of their stops must be equal to 1 or -1. The SQL query below exemplifies this. The final result set is a table with a column with a station id and a column with a station id adjacent to station in the first column. Since line routes overlap each other, this pair of station is grouped in order to avoid repetitions.

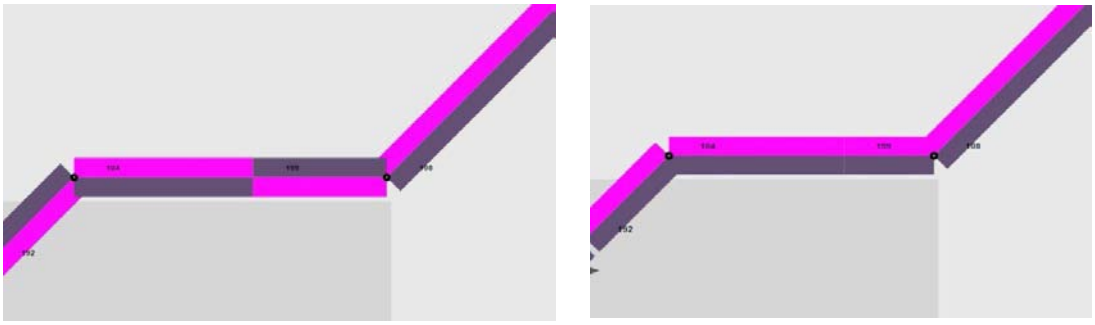
```
SELECT s.id stationid, t.id adjstationid
FROM linestopsequence a
JOIN linestopsequence b ON a.lineid = b.lineid
JOIN stop x ON a.stopid = x.id
JOIN station s ON x.stationid = s.id
JOIN stop y ON b.stopid = y.id
JOIN station t ON y.stationid = t.id
JOIN line l ON a.lineid = l.id
WHERE (b.sequence = a.sequence - 1 OR b.sequence = a.sequence + 1)
      AND (s.id <> t.id)
GROUP BY s.id, t.id
```

This result set is read by the server application in order to construct the adjacency list of stations. The adjacency list of stations will be used further by the depth first search algorithm to divide the network into paths that will be sent individually to octilinear simplification, which will be explained in the next section.

The other important data structure is the list of edges. The list of edges could easily been obtained from the list of adjacency since an edge is formed by two adjacent stations. However, in the context of schematic transit maps an edge shares one or more line segments and the shared lines by an edge must be identifiable. It means that a segment for each line serving an edge must be drawn in parallel and each segment must be colored with its specific line color.

```
SELECT s.id stationid, t.id adjstationid, l.id lineid
FROM linestopsequence a
JOIN linestopsequence b ON a.lineid = b.lineid
JOIN stop x ON a.stopid = x.id
JOIN station s ON x.stationid = s.id
JOIN stop y ON b.stopid = y.id
JOIN station t ON y.stationid = t.id
JOIN line l ON a.lineid = l.id
WHERE  (b.sequence = a.sequence - 1  OR b.sequence = a.sequence + 1)
  AND ( ST_Distance( ST_GeomFromText('POINT( -0.042666 39.987593)',4326), s.geom)
      < ST_Distance( ST_GeomFromText('POINT( -0.042666 39.987593)',4326), t.geom) )
ORDER BY   stationid, adjstationid
```

The query to obtain the list of edges is though very similar to the one made for the list of adjacency. There are only two basic differences. First, the aggregation "group by" is removed from the SQL query in order to obtain the repeated adjacency relation between stops that represent the lines that overlap in as single edge. Second, a third column with line ids is added in order to retain the information about which line is represented by this segment. The SQL query below illustrates this. The spatial condition where the station with a shorter distance from a focal point has priority over farther stations is included because it promotes a coherent order of the line segments of an edge. This condition tends to avoid, for example, when the edges are composed together that a segments of a same line appear in different positions in the edges, like illustrated in Figure 4.8.



A b

Figure 4.8 Incoherent line segments positioning.

The list of relevant lines passed as parameter from the client to the server application should be included as conditions in WHERE statement to limit the records only for those selected lines in the adjacent list query as well as for the edge list query. The SQL line below is an example of how these conditions are included for the set of lines L16, L11, L10 and L7.

```
...WHERE  ... AND  (  l.name = 'L16' OR l.name = 'L11' OR l.name = 'L10' OR l.name
                                    = 'L7' ) ...
```

The POIs besides of serving as reference in the final schematic map, they can be used for reducing even more the information to the user. The parts of the lines routes that not connect the POIs are not indispensable to final schematic map. The POIs can be used in the queries to delimit the relevant part of the lines. The next section deals with this issue.

4.2.3 User Influence zone

Most of the time a passenger, while performing a transit journey, uses only part of a transit line. Rarely ever a passenger boards in the start station and leaves in the end station, thus the parts of the line that are not relevant can be omitted from the map. Therefore, the necessary elements to be presented are the paths that connect the POIs, still, a way to identify the start and the end station of each line is important.

The techniques used in this project to limit the part of the lines to be presented are based in spatial queries. The use of spatial queries for this purpose proved to be problematic and a topological solution should fit better as solution for this case. Even

not being a complete success, it's worth mentioning the details here since some results will be presented with this limitation of the line routes.

```
SELECT s.* FROM station s, (
SELECT ST_UNION(poisbuffer.geom, stopscircle.geom) AS geom
FROM ( SELECT ST_Buffer(ST_Collect( ARRAY[ST_GeomFromText('POINT(-
0.06909370422363281 39.99421871723377)', 4326), ST_GeomFromText('POINT(-
0.04231452941894531 39.98797164114611)', 4326), ST_GeomFromText('POINT(-
0.06334304809570312 39.97935631488548)', 4326)] ), 0.012) AS geom
) poisbuffer,
( SELECT ST_MinimumBoundingCircle(ST_Collect( ARRAY[ST_GeomFromText('POINT(-
0.06909370422363281 39.99421871723377)', 4326), ST_GeomFromText('POINT(-
0.04231452941894531 39.98797164114611)', 4326), ST_GeomFromText('POINT(-
0.06334304809570312 39.97935631488548)', 4326)] )) AS geom
) stopscircle
) poisinfluencearea
WHERE ST_CONTAINS(poisinfluencearea.geom, s.geom)
```

The spatial solution adopted here consists in using the POIs to construct a polygon that determines the user influence zone. The idea is to select only the geographic transit features contained within this polygon. The polygon that presented the best result was a union of the POIs minimum bounding circle with the buffers of the same POIs. This composition reduces fragmentation of lines because it includes in the influence zone not only the area between the POIs but the area surrounding them as well. Figure 4.9 is a graphical illustration of the influence zone creation and the logical spatial query below exemplifies how to select stations inside the influence zone. In Figure 4.9(c), it is possible to see in cyan the stations selected to be relevant to the user.

The problem of this spatial restriction lies on the fact that a line route that connects to POIs might exceed the limits of the influence zone, like illustrated Figure 4.10. In this case the representation of the line, in green, will be fragmented because some of its stations connecting the POIs lie out of the influence zone. Bigger influence zone tends to reduce this problem, but more irrelevant data will be loaded into the data structures.

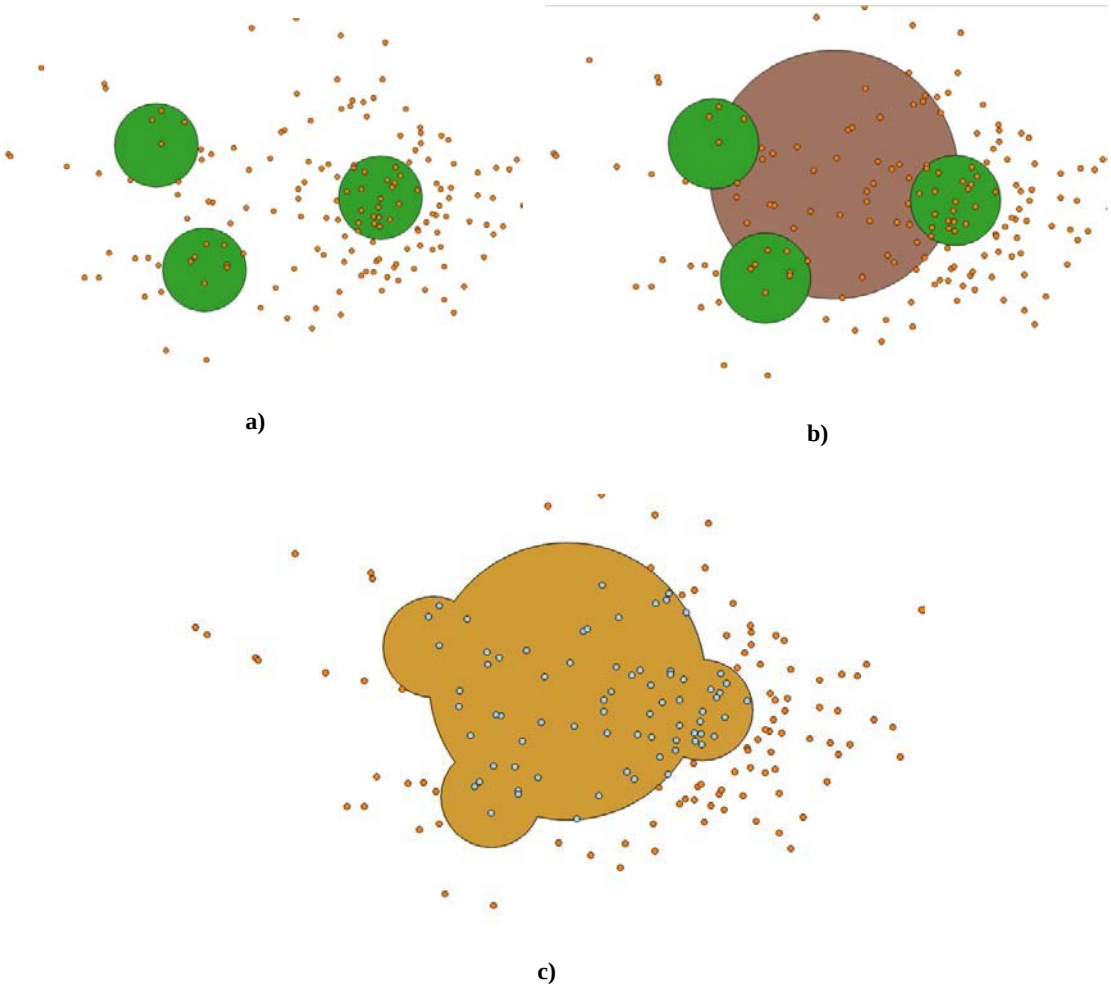


Figure 4.9 Illustration of a spatial creation of influence zone.

A possible final solution to query only the relevant parts of the line would be a topological restriction rather than a spatial one. A topological restriction consist in first, identifying the closest stations to the POIs and to use them in the relational table between line and stop (*linestopsequence*) to select the stations that are part to the line paths that connects those stations. This selection will require techniques similar to the ones used in routing algorithm what is out of the limitations of this work and will be left as future work proposal.

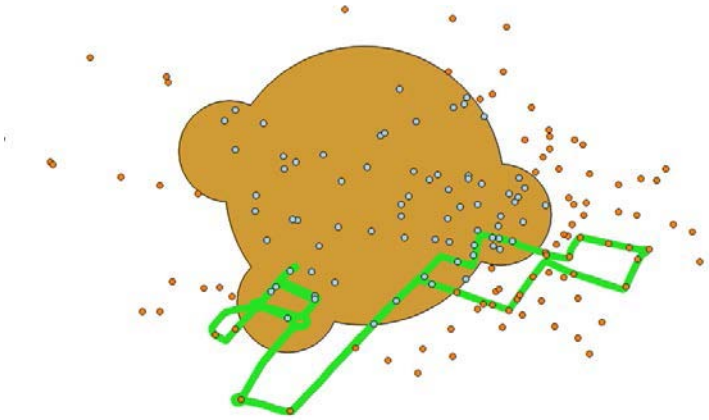


Figure 4.10 Example of line fragmentation due to spatial influence zone.

In this section it is presented details about the data structure to generate the schematic maps, additionally, it shows how the data model is queried in order to select only information that is relevant for the user. At the end, the data structures are station list, a stop list, reference list, an adjacency list, and, finally, an edge list. In the next section it is presented how those data structures are used to transform the data into schematic data.

4.3 Generation of schematic data

This section is the core of the methodology. It is where the geographic data is shaped into schematic data. The last section defines and shows how to obtain the data structure that will serve as input for the schematization process. In order to make this transit data attend the desired properties of a schematic map it makes necessary to perform a set of operations that includes conversions, transformations, and generalizations.

Since the spatial data is stored in the WGS1984 (an ellipsoidal system that describes position in latitudes and longitudes), the first step is to convert from this ellipsoidal system into a two-dimensional Cartesian system, i.e., a map projection. Computations in Cartesian systems are far less costly than in ellipsoidal systems. Additionally, schematic maps are traditionally 2D, what makes the Cartesian systems more suitable for schematizations.

The conversion from WGS1984 to Cartesian 2D system is an arrangement of the earth surface in a flat surface. This is done by a projection of the ellipsoid into a developable surface and it is a regular operation in cartography that requires a set of trigonometric equations. The conversion used here was held with the support of Java API called *JMAPViewer* that converts the ellipsoid coordinates to the Mercator projection system.

This conversion is made for all the geographic data and after an affine transformation is performed in order to adjust the origin, scale and rotation of the system to better fit the data in canvas of device screen. The origin of the system is moved to the top left corner of the bounds of the data. The scale is changed to enable the whole data to include in 1000x1000 pixels square. A slight rotation is made in order to make the street lines of Castellón closer to vertical and horizontal grid. This anticlockwise rotation of 16 degrees is not essential but will result in a map with less bends in the transit paths, but the north direction will be changed.

This affine transformation does not change the scale factor of the Mercator projection. It means that the relative distances between the geographic features are preserved. In the next subsection the transformation deals with the variable-scale visualization adequate to schematic transit maps.

4.3.1 Fisheye Transformation

One of the characteristics of schematic transit is that regions with more density of stations get larger share of the available space than regions with less density, i.e., variable scale visualization. This makes the stations to be better distributed in the map and, as result, the space for labels are better distributed making the visualization of the maps more pleasant.

Fish-eye, as explained in the theoretical review of this document, is a variable scale transformation where the scale is progressively bigger when closer to a given focal point, and, as a result, areas further from the focus will have a smaller scale. Sarkar & Brown (1994) identifies usages of fisheye transformation to graph vizualization. They calculate a position of vertex v in the fisheye view from a function using the distance of its original position to a focal point.

Their implementation allows only one focal point and all the data must be inside a determined bound. The equation $f(x)$ is an example of a rational function that can be used to calculate the fisheye coordinates of a node. Like illustrated in Figure 4.11, this kind of function indicates how the scale factor is alternated while the distance from focal point grows. The constant c represents the level of distortion to be applied. If c is zero no distortions is applied because linear equations have a constant slope.

$$f(x) = (c * x + x) / (c * x + 1)$$

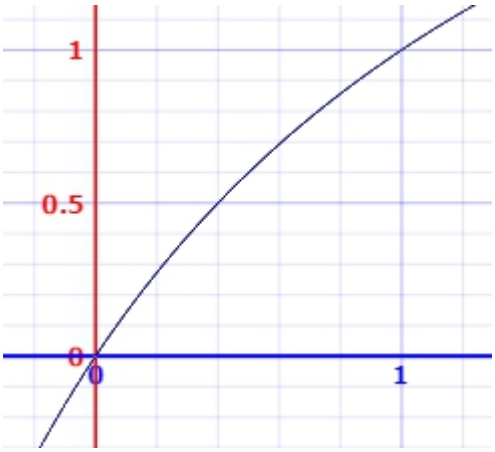


Figure 4.11 Rational function for fisheye scale distortion.

Using the most congested area of Castellón (center) as focal point, all the spatial data is submitted to a fisheye transformation. The impact on the scale factor throughout the limits of the data bounds can be visualized in as Figure 4.12 (Sarkar & Brown, 1994).

This simple mono focal fisheyes transformation attends the goal of this dissertation, but the station density distribution in a city is not always regular. Although, the centers of cities are normally more dense areas than peripheral ones, more complexes distributions of stations can be found, especially in metropolis. Jenny (2006) made a cartographic analysis of London´s Underground schematic map to understand the distortions provoked by the designers when sketching schematic maps. Figure 4.13 illustrates throughout a grid the kind of resulted distortion in handmade schematic maps.

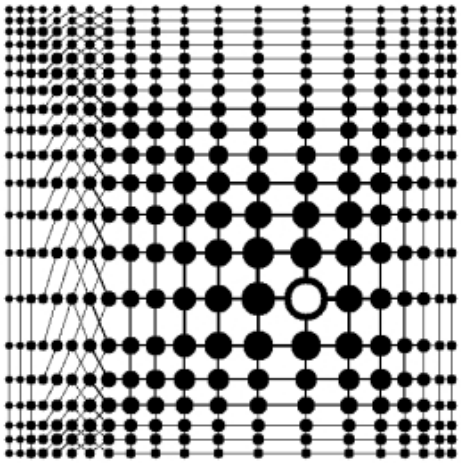


Figure 4.12 Fisheye scale distortion effect.

Ti & Li (2014) developed a schematization algorithm that is able to identify automatically the congested areas and, in addition, a multi focal fisheye transformation is executed through the spatial data. This approach is more adequate to automatic drawing of schematic maps because it reduces the dependency of human interference and, because it is multi focal, it allows a better distribution of stations in metropolis composed with satellites cities. However, Ti & Li didn't present the detail of the execution time of their transformations algorithms and it is not possible to affirm how its computational cost will affect real time schematizations.

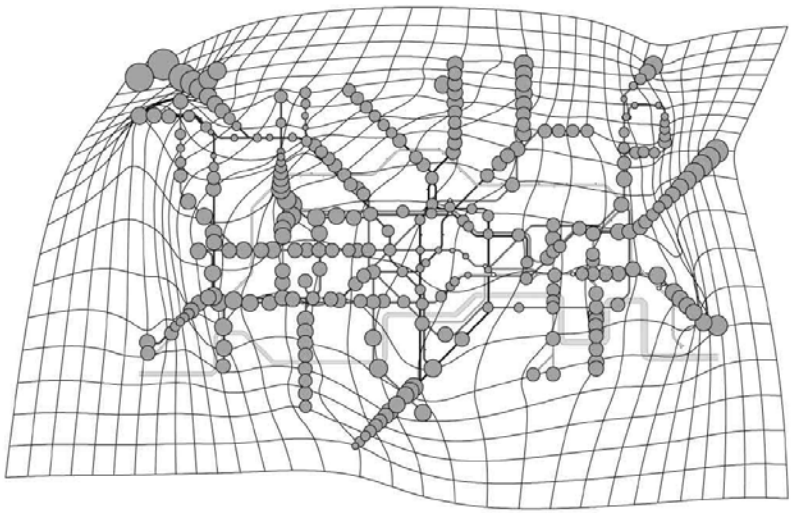


Figure 4.13 London tube map scale variation (Jenny, 2006).

4.3.2 Octalinear Path Simplification using Genetic Algorithms

The goal on this dissertation is to explore the use of automatic generation of schematic maps into real time web applications in order to produce customized schematic maps. Since the metro layout problem has been proven to be NP-Hard (Nöllenburg, 2005) and, by a consequence, optimal solutions for the problem still a very timing consuming task, we opted to adopt a path octilinear schematization solution that uses genetic algorithms to reduce the bend along the path, limiting its distortion in relation to its original shape. Although path schematization approaches to the metro layout problem does not guarantee the topological correctness in the resulting layout, this constrain was sacrificed for performance purposes, since the combination of preserving the topology, octilinear layout, and bend reduction is responsible for the NP-Hardness of the problem.

In spite of not guarantee the correct topology, the algorithm tends to avoid distortions (distance) in the resulting layout. It means that the topology of the original input is used as information for some topology correctness in the output. That, allied to the fact that instances tend to be small, in the specific case here, contributes to reduce the topological rupture in the final layout.

Path schematization requires first that the transit graph must be disjoint into non-overlapping paths that cover the whole graph. To obtain a minimal set of paths, a depth-first search algorithm is performed in the network. The resulting set of path is, then, sent to be schematized by the GA, and a new schematic position for each station is calculated.

The GA for path schematization developed by Galvao (Galvao M. d., 2010) has been presented in details only in Portuguese, making important a description of the main parts of its implementation. We start describing the encoding of chromosome. Second, we show how the mutation and cross-over process are performed. After, we show how the populations are generated, and finally the fitness function and the selection process.

4.3.2.1 Encoding the chromosome

An important step in developing a genetic algorithm is choosing how to encode each individual's chromosome. Individuals from a given population are the fundamental units of a genetic algorithm. They identify a possible solutions to a given problem, and at this present study, an octalinear path is considered an individual.

Due to the geometric nature of the problem, the most intuitive way of encoding a chromosome is using the Cartesian coordinates of stations. However, in order to facilitate other operations involved in a genetic algorithm, instead of using Cartesian coordinate system, it was used a sort of relative polar coordinate system.

Relative polar coordinate system represents one path with n points, selecting as origin point p_0 one of the extremes of the line. p_0 is represented by its Cartesian coordinate, i.e., its (x, y) values The only point adjacent to p_0, p_1 is represented by the polar coordinate relative to p_0, i.e., a value (r, θ) where r represents the distance between p_0 and p_1, and θ the angle formed by the segment that unites the points p_0 and p_1 and the axis x. The other point adjacent to p_1, p_2, is represented by relative polar coordinate to p_1, i.e., a value (r, θ). This representation made by relative polar coordinates goes until point p_n forming the chromosome of the path. Figure 4.14 illustrates an example of a path with its chromosome value.

CHROMOSOME
(15.0, 36.0)
(25.0, 45.0)
(21.55, 315)
(21.55, 315)
(21.55, 315)
(21.55, 315)
(19.13, 0)
(37.64, 0)
(24.48, 0)

Table 4.1 Chromosome genes of path in Figure 4.14

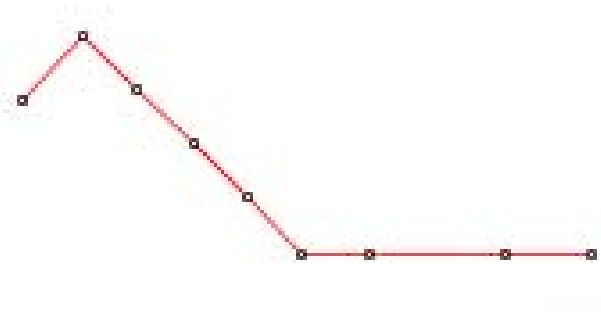


Figure 4.14 Example of octilinear path

4.3.2.2 Mutation and crossover

After encoding the chromosome, the next step is to define the other basic operations of a genetic algorithm. Crossover and mutation help guarantee diversity for further generations.

The mutation operation consists of randomly change one of the chromosome genes. The mutation process helps prevent a permanent fixation on a local minimum in the solution space. Since a gene is represented by the values (r, θ), the mutation of a chromosome corresponds to alter the r values and θ of a gene. In order to avoid disproportional distortions in the process, the mutation process alters randomly the value r inside a margin of 50%, and θ mutation is done by adding or reducing 45° to its original value, thus keeping octilinearity of a chromosome.

The crossover operation consists in combining randomly two chromosomes to generate a new one. The crossover process chosen for this project is the crossover between two points. Having as input two chromosomes, the crossover process select two random points of the path as cutting points, then two parts of the first chromosome are joined together with the complementary missing part of the second chromosome. Table 4.2 exemplifies a crossover of paths chromosomes.

CHROM. 1	CHROM. 2	CHROM. 1 X CHROM 2
(31.00, 97.00)	(31.00, 97.00)	(31.00, 97,00)
(16.15, 90)	(16.15, 90)	(16.15, 90)
(26.51, 0)	(26.51, 0)	(26.51, 0)
(24.36, 0)	(26.51, 45)	(26.51, 45)
(24.36, 0)	(18.26, 45)	(18.26, 45)
(24.36, 0)	(32.97, 0)	(32.97, 0)
(19,00, 315)	(34.24, 315)	(19.00, 315)
(41,5, 270)	(34.25, 225)	(41.50, 270)
(30,87, 0)	(30.87, 0)	(30.87, 0)
(16,88, 0)	(16.88, 0)	(16.88, 0)

Table 4.2 GA Crossover example .

It is worth mentioning that for both, the crossover and the mutation, if the input chromosomes represent octilinear paths, the resulting chromosome will be octilinear,

since the mutation and crossover are the only way to generate new individuals. If the initial population is composed only by octilinear members, all the individuals produced by the GA will be octilinear.

4.3.2.3 Generation of populations

The input of the GA process is a path, in other words, a sequence of station Cartesian coordinates. From this input, the GA generates n valid chromosomes. Those n chromosomes represent the elementary chromosome group and are used as individuals on the first population as well. The number of individuals per population remains constant in the evolutionary process and the population size n for the experiments in this project was ten.

To create the elementary group of chromosomes the path is submitted to n Douglas-Peucker line simplification and for each simplification a different distance dimension is set as parameter. As a result, n different paths with possible different number of bends are created. From each of those paths an octalinear version from them is created in a way that all curves that connect two bends is restricted to the octilinear orientation, and the stations between two bends are distributed in a equidistant way through the curve.

To generate an offspring from a population, the individuals from this population are selected to mutate and cross between themselves to form new individuals. In order to guarantee diversity and avoid local minimum fixation, an individual of the elementary group is randomly selected to be added in the offspring as an intruder.

4.3.2.4 Selection process and fitness function

Individuals with favorable genotypes must be more selectable to reproduce so that their genes are in greater numbers in next generation. The goal of the fitness function is to evaluate a chromosome according to its degree of adaptation as a solution to a problem, thus, the fitness function returns an index that represents a level on how good this chromosome is as a solution.

As explained in Section 2.2.1.1, that, to facilitate to be followed with the eyes, the paths in a schematic map should avoid as possible bends on its way. However,

reducing bends from a path causes displacements from its original layout and a disproportional distortion must be avoided as well. In order to find a balance between those two aesthetics factors, the fitness function calculates the distance from the individual to its original shape and a weighted arithmetic mean of the number of bends in the individual to form a single resulting index.

The distance factor is the mean of the distances between the respective points from chromosome path and its original path. Let $L = (a_0, a_1, a_2, \ldots, a_n)$ be the original path, and $L' = (b_0, b_1, b_2, \ldots, b_n)$ the octilinear path to be evaluated, and let $d(a, b)$ be the Pythagorean distance between point a and b, the distance factor $\bar{d}$ can be calculated as:

$$\bar{d} = \frac{\sum_{i=1}^{n} d(a_i, b_i)}{n}$$

The bend factor is a weighted mean of the number of bends in the schematic path in a way that obtuse bend have more weigh then oblique ones. Let $|x°|$ be the number of bends with x degree in a schematic path, the bend factor Z can be calculated as:

$$Z = \frac{1 * |45°| + 3 * |90| + 5 * |135°|}{n}$$

Using the distance factor and the bend factor the fitness function return the final index η as followed.

$$\eta = a * \left(\bar{d} + b * \sigma\right) + Z$$

Where σ is the standard deviation of $\bar{d}$, and a is the constante that determines the weight of the bend factor in relation to the distance factor. And b is a constant to weigh σ in relation to $\bar{d}$. As a result lower index values are better solution than bigger ones.

With the fitness function the GA evaluates each chromosome of a population, and orders them according to their indexes. The probability of an individual to be selected to pass its genes to the next generation is proportional to its position in the position in list, in other words, individual evaluated with lower index has more chances to be chosen than the ones worse evaluated. This kind of selection is called ranking selection. In order not

to lose the best solution of a generation, the selection process is also elitist. It means that the best solution of a population is copied to its offspring.

4.3.2.5 Evolutionary process

Due to the evolutionary process, it is expected that the best individual of a determined generation has a better or equal index values than the best individual of its previous generation. The experiment made by Galvao et al (2014) shows graphically a sequence of the best different solutions over successive generations. In Figure 4.15 is possible to note this morphological evolutionary process and how the algorithm tends to reduce the number of bends without disproportional distortions to its original shape.

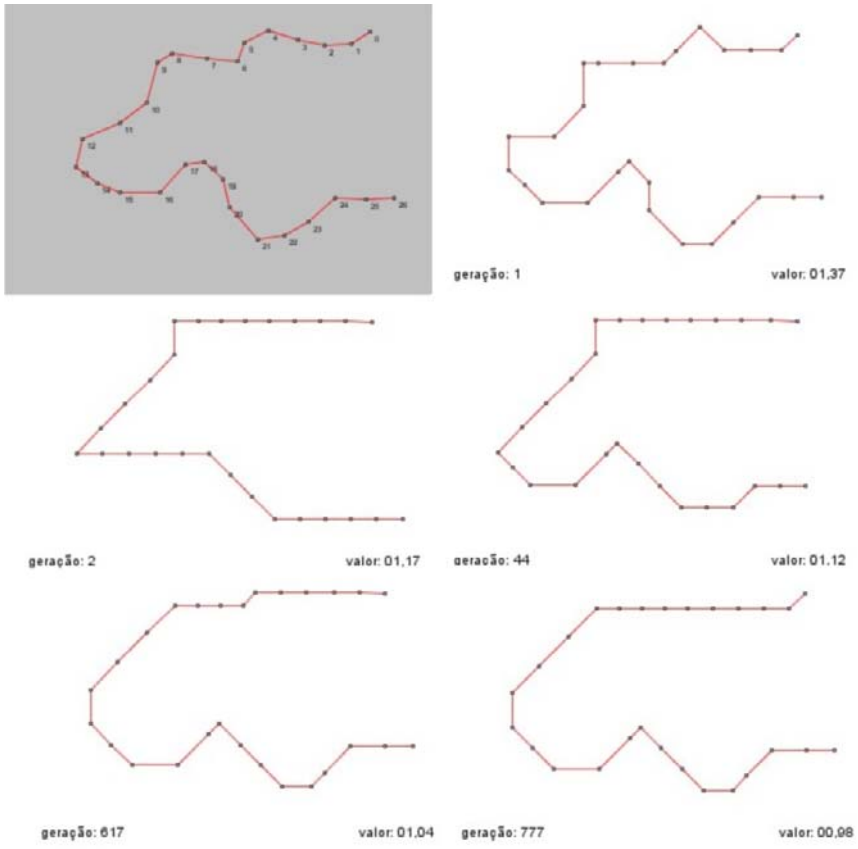


Figure 4.15 Illustration of octilinear path evolutionary adaptation in GA.

4.4 Web Schematic Data Visualization

Static visualizations, like the ones of a paper map or in an image digital format (JPEG, PNG, etc) limits a data set to be present in a single format and any different

perspective of this same data set will require a new media to be represented. However, digital devices (laptops, tablets and smart phones) allow interactivity with the visual data, granting visual tools, like maps, with new possibilities to explore the data according to specific needs of the user. Therefore, the visual information-seeking "mantra" proposed by Schneiderman (1996), "overview first, zoom and filter, then details on demand", is every day more valid for schematic maps too.

The biggest advantage of web visualization is its power to make the information broadly accessible. Due to its standard protocols, the visualization on the web can be accessed to any connected device in the world independent of operational system and the device itself. Another advantage of web visualization is its hypermedia essence, which allows not just a non-linear interaction with the visual elements, but to third-party data services to be explored as well, empowering the dynamic and richness of the visual information tools.

In order to generate interactive schematic maps, it is required to represent the spatial data in a vector base graphic. Scalable Vector Graphics (SVG) is a specification of World Wide Web Consortium (W3C) for vector visualization that supports hypermedia interactivity in web. SVG is currently the most used and supported vector based graphic tool for the web, its visual elements, such as lines, circles and rectangles are expressed through XML tags.

After the schematization, the data is ready to be sent to the client and plotted to the user. The server responds the GET method with text in JavaScript Object Notation (JSON) page containing all the schematized spatial data and its attributes. Due to the fact that we are dealing with spatial data, GeoJSON, an extension of JSON for geographic data has been used to represent the data instead.

All the data is contained in single *FeatureCollection* object. FeatureCollection servers as a container for spatial features of any kind of geometry type. In the FeatureCollection, every single segment of an edge is represented by a LineString geometry having as attribute the line it represents, its color and its direction. Every station is represented by a geometrical point having as attribute its name, its id and

its geographic coordinates. The text below exemplifies how those features are represented as a feature object.

Edge segment:

{"type":"Feature","geometry":{"type":"LineString","coordinates":[[249.8440606474748
6,743.849080150498],[166.07848129605222,743.849080150498]]},"properties":{"linename
":"L16","color":"#669BBC","direction":3}}

Station:

{"type":"Feature","geometry":{"type":"Point","coordinates":[300.53184393210677,556.
3116197764097]},"properties":{"stationid":85,"type":"station","name":"Ctra. Alcora
(Mas Blau)","lat":39.9863876045044,"lon":-0.05506556252567}}

Using the JavaScript library Data Driven Documents (D3), the GeoJSON data is read to dynamically generate the SVG tags and insert them into the body of the document. D3 allows not only JSON to be read but also allows all the style, attributes, and behavior of each element to be set. That way, interaction events like pan, zooming and click can be controlled for all elements of the schematic map, making it a powerful information tool and in accordance with the visual information-seeking mantra.

5 ANALYSIS AND RESULTS

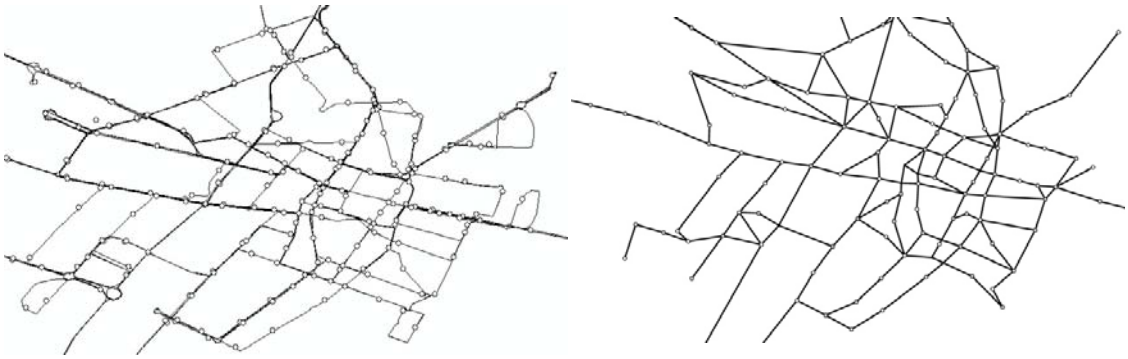
In the last section we have presented a specific methodology to the creation of customized bus transit schematic maps for web environment. In this section we use the PT data of Castellón, Spain, to present, throughout a novel, the effect of the models and techniques developed in the methodology of this project.

We start with the raw data showing the effect of node reduction with the station concept. Then, we show how our data structure supports a more adequate visualization to schematic maps on demand. Next, we present the effect to readability of the information reduction based on user POIs. After that, we show how the fisheye transformation affects the position of the transit spatial features to produce a more balanced map. Finally we add the GA path schematization that restricts all segments to the octalinear orientation. To sum up, we give examples on how digital schematic maps can be explored in web platforms in order to increase its power as an information tool. The last subsection is dedicated exclusively to treat about the performance aspects related to the schematization process.

5.1 Data model and structure analysis

The Figure 5.1(a) is a representation of the raw transit data of Castellón. It is a combination of the all 20 lines routes and all stop points. Although this representation is useful to show the whole network cover and the exactly paths through the streets, it lacks on information about the lines connectivity, what is not adequate for schematic transit maps. Using the concept of station, explained in Subsection 4.1.3, Figure 5.1(b), represents the same network but with its topological connectedness.

This graph representation, besides allowing the identification of connectivity, it requires considerably less data then the lines and stops representation. While Figure 5.1(a) requires 257 stops and 3264 edges, Figure 5.1(b) requires only 173 nodes and 296 edges.



a) b)

Figure 5.1 Illustration of graph creation after introduction of stations.

Figure 5.1(b), although it represents the transit data in a topological manner, it is impossible to identify each line individually. Since to color each edge according to the lines to which it belongs is a rule of the schematic transit maps, we use the edge list data structure, explained in Section 4.2.2, to draw individually each line segment of an edge in parallel and color then accordingly. Figure 5.2 shows how the network is represented using the edge list to allow, one by one, the identification of the lines.

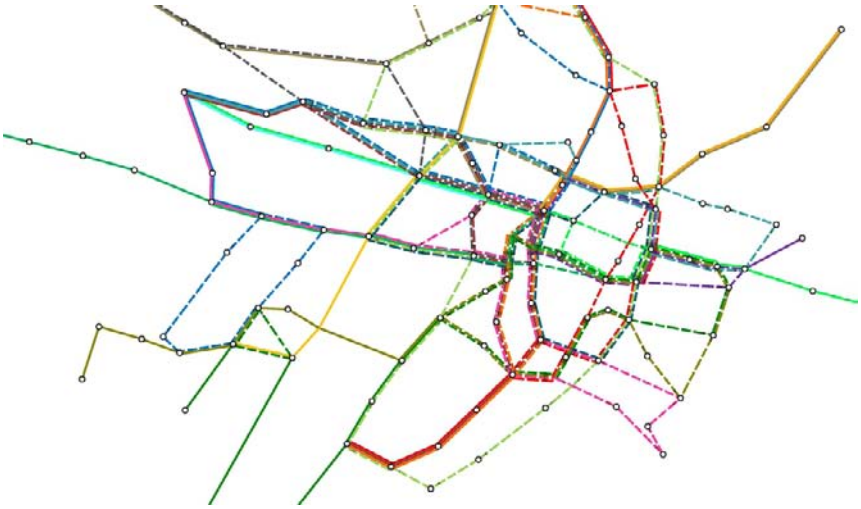


Figure 5.2 Graph with individual colored line segments for each edge.

Bus lines, in contrast with from metro lines, have poor connectivity between the out and return way (Section 2.1.1). This causes some of their segments to be single directions segments. Those segments are distinguished from round trip segments by dashed stroke style complemented with an arrow indicating the direction which the

vehicle travels (Figure 5.3). This symbology allows the user to identify, line by line, the whole trip performed by a vehicle.

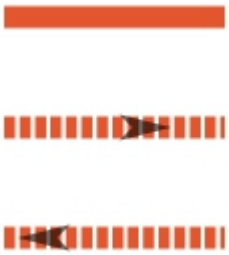
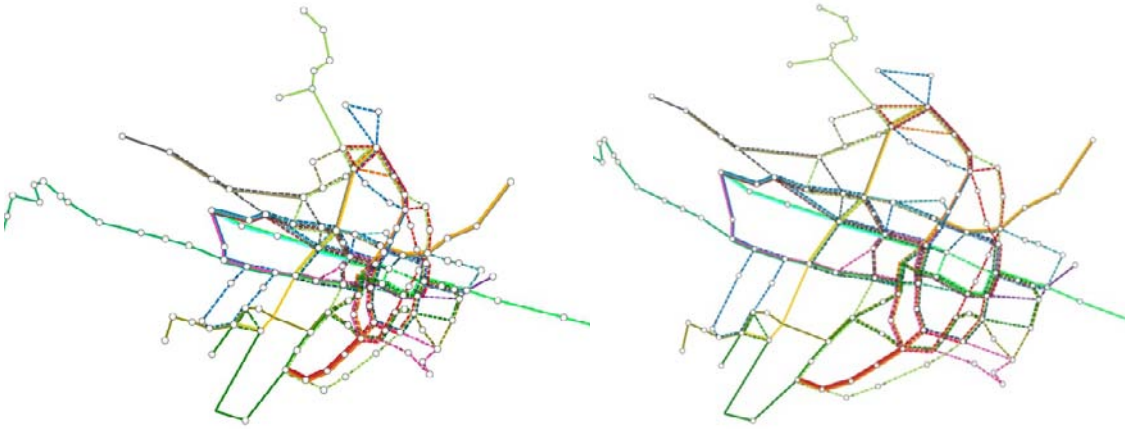


Figure 5.3 Bus route direction indication. Non-dashed stroke means service in both directions.

5.2 Schematization analysis

The examples on Figure 5.1 and Figure 5.2 are merely a projected representation of the features by their geographic coordinates. The fisheye effect and the octilinear layout distort the positioning of the spatial features through the map, what makes it schematic.



a) **b)**

Figure 5.4 Fisheye effect in the network of Castellón.

Figure 5.4 shows the result of applying the fisheye transformation described in Section 4.3.1. It is possible to notice that in the resulting map, the dense central area takes space from the less dense peripheral areas. This results in a more uniform distribution of the station through the map, what is not only in accordance with the

Beck's maps rules, but it is a more adequate visualization to small screen devices once it reduces pan and scrolling when exploring the map.

The octilinear schematic layout has as main goal to increase its visual simplicity by restricting the orientation of its edges. Figure 5.5 shows the resulting network after applying the GA for octilinear path schematization. An anti-clockwise rotation of 16 degrees has been applied before the running the algorithm in order to create a better alignment of the segments with the octilinear orientations.

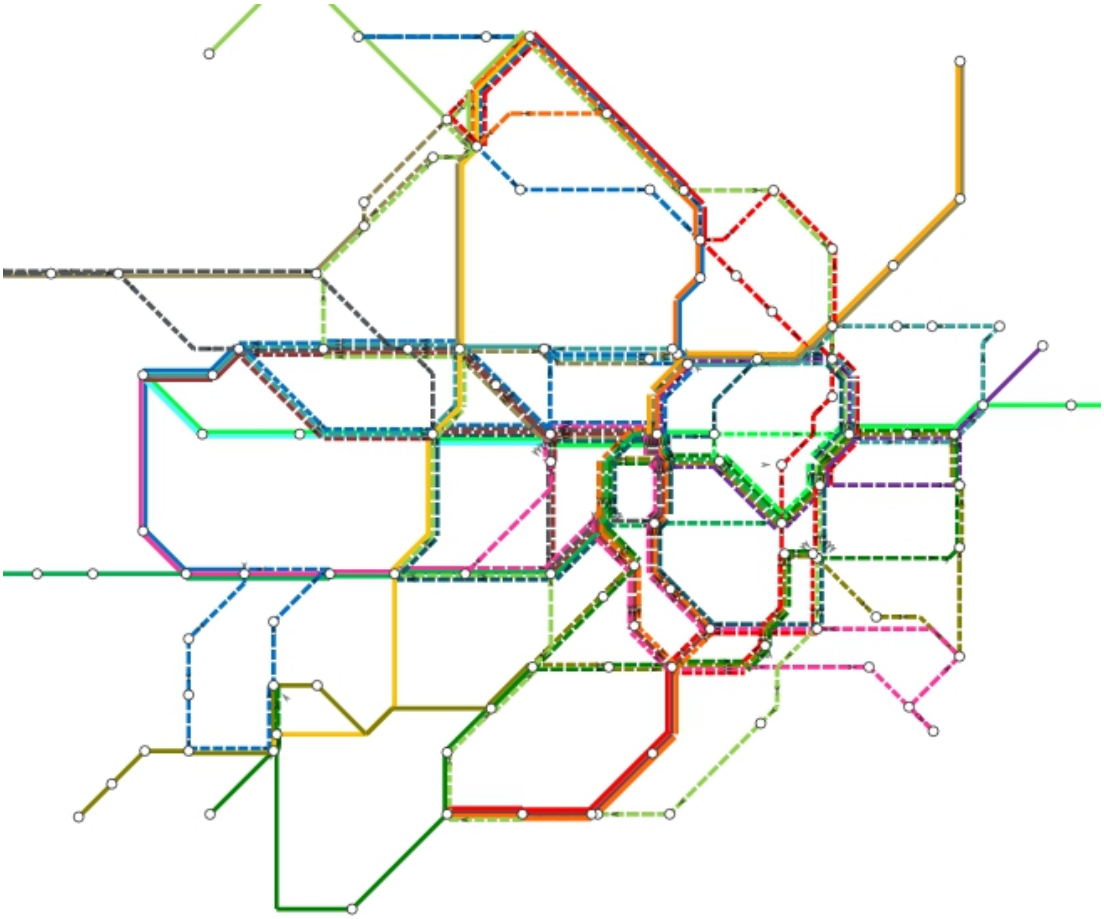


Figure 5.5 Final result after applying the GA for octilinear path schematization in Castellón transit network.

The octilinear layout has been proof as a useful configuration for schematic transit maps and the success of first Beck's map confirm this. However, the evaluation of specific octilinear graph layout solution for a given transit network representation is

trickier and, in many aspects, are subjective requiring it to be surveyed for a more precise analysis.

Nöllenburg (A Survey on Automated Metro Map Layout Methods) has been trying to systemize design principles that help in the evaluation of octilinear maps. The evaluation here made is based on those rules and the comments are referred to Figure 5.5.

The topological correctness is not guaranteed by the GA path schematization. However, edge crossing and changes on the circular edge order is possible, those violations are scarce and many times null in produced map. The given example shows this even for a bus network with multiples cycles embed in its graph configuration. In opposite to extras edge crossing, the resulting configuration presents edges overlaying, which means when two edges lay over a same axis. Most of those violations have been identified programmatically and a solution and future work suggestion can include the overlapping edges in the same edge structure to be drawn in parallel.

The edge orientation restriction is not violated. All edges have been drawn in one of the octilinear orientation. This is even an improvement to the technique initially developed by Galvao (2010) and by him suggested as future work.

Minimization of bends is an aesthetics principle and it is impossible to draw all lines in a straight line. This is even more difficult for bus networks that are in the street level and have more complex morphology than metro networks. The GA fitness function helps keeping the number of bends small and avoids sharp turns, additionally, it is possible to set the depth-first search to give straightening preference to a specific path or line over others as illustrated in Figure 5.5 where the light green line (horizontal) is set to have preference to be rectified over the others.

The geometric distortion is minimized by the GA as well, but it plays a conflicting role with the bends minimizations. Changing the parameter in the fitness function it is possible to drive the results to fewer distortions or to less bends. Figure 5.6 illustrates the effect of this parameter in the fitness function. The figure in the right gives more priority to less distortion, while the one in the left to less bends.

Figure 5.6 GA octilinear schematization with different weights for bend reduction.

The last worthy to comment rule is keeping the edge lengths uniform. Although, the GA tries to keep the distance between adjacent stations equal in a straight path, many times it provokes considerable displacement of the stations, what makes the final result to violate the distortion rule in the fitness function. An uniform edge length factor could be include in the fitness function or a stroke-based positing of the stations on a path, like the one applied by Li and Dong (2010), could by applied here, what would give better results in term of this rule.

5.3 Context and digital-web platform issues

This subsection discloses the automatic generated schematic maps as a digital tool for web-platforms. In opposite to static (paper) maps, the experience of using a web map can be enhanced by context-aware information and hypermedia integration. We first present personalized schematic maps created based on user information. Second, we list developed techniques regarding the exploration of interactivity and service-oriented architecture.

Static maps tend to be an all-in-one informative tool. They are meant to the general public. Automatic schematic map drawing algorithms, as long as they can produce results in real time, can be used to produce personalized informative tool In other words, a map that meets a specific user or group of users. We used instances of POIs to, first, spatially limit the bounds of the network and then to automatically select the

relevant lines for those POIs. Figure 5.7(a) illustrates those POIs in a cartographic map. Figure 5.7(b) shows a schematic map with the full network together with the POIs. Figure 5.7(c) shows a new schematic map with all lines but limited by the influential zone (Section 4.2.3) of the POIs. Figure 5.7(d) shows a new schematic map with only the lines that connect those POIs and limited by their influential zone.

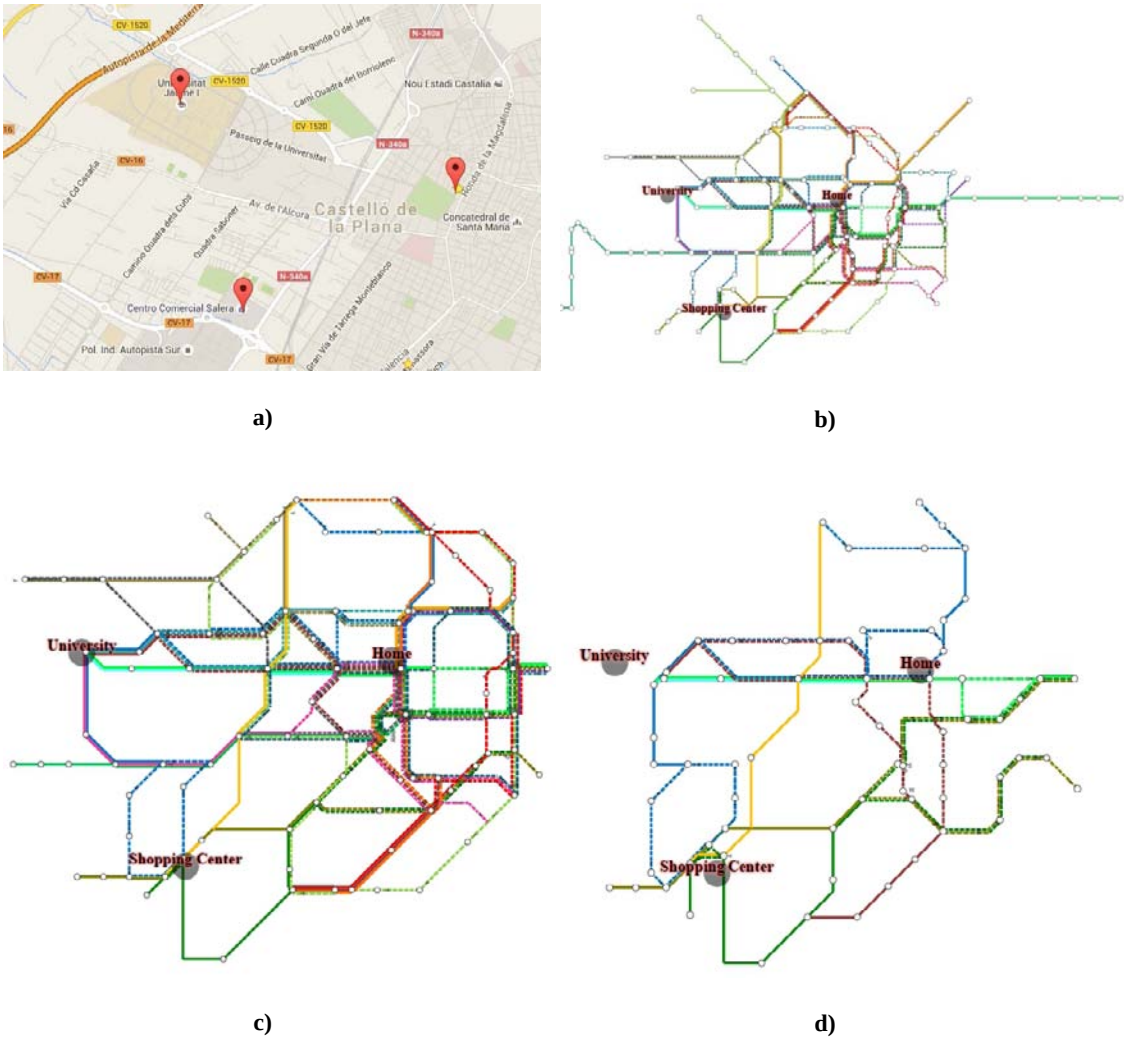


Figure 5.7 "Less is more". Schematic map information reduction by influence zone and lines selection.

As a result, we can say that the influential zone limitation reduces the dimension of the resulting map, grabbing the attention of the user to the relevant area. The line reduction removes useless information, making the map more efficient for journey planning. Regarding the influential zone technique two more points are worthy to be commented. First, the effect of the influential zone will be bigger for larger cities, for

instance, a user is interested only in the transit of a satellite city in a metropolis. All the rest of the neighborhoods could be removed. The second point is that the effect of the influential zone could be increased using a topological limitation instead of a spatial one (Section 4.2.3).

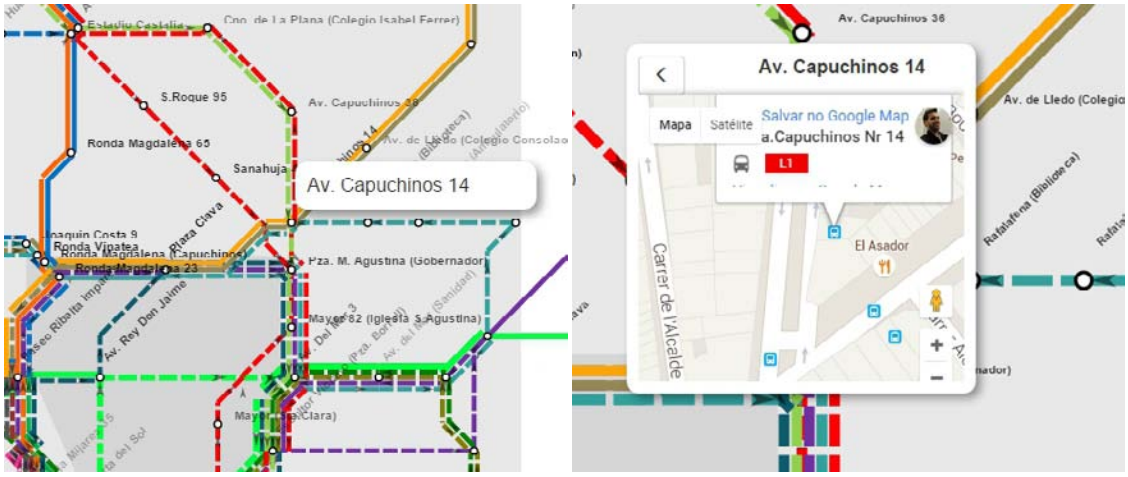
Productions of personalized maps are probably one of the main advantages of automatic schematic maps on demand. However, to use them in web platforms creates a new range of manipulation possibilities. We implemented, as example, three interaction processes to illustrate how a digital vector-based map can be more useful as an information seeker tool than static maps.

a) **b)**

Figure 5.8 Line enhancement by interaction.

Many times is difficult to find a line of interest in a schematic map, especially for bus networks that tend to be bigger and with a lot of line sharing a same route, and it is common to lose the focus on a line when following it with the eyes. A line (path) enhancement can be made on demand on digital schematic maps by a click on the line or by getting the information from routing algorithms. By a click on a line or in its legend symbol our implementation highlights a line and minimizes the others. Figure 5.8 illustrates how this kind of interaction helps a passenger to keep the focus on its line of interest.

Schematic transit map is useful for seeking transit information, such as connections between lines and when to board or leave a vehicle, but it lacks on information to serve as walking references. One example on how the interactivity allows obtaining more detailed information on demand is to allow a rapid switch between the schematic and a cartographic map. Figure 5.9 shows how a web visualization of schematic maps can be used on this sense. By a click on a station of interest, the application automatically translates and zooms into this station and opens over it a cartographic map locating the stop points of the station.



a) **b)**

Figure 5.9 Interaction for seeking information. Click in a station to zoom to it and show its cartographic map.

Most of the current mobile devices are coupled with GPS and modern browsers that can also use IP address to determine a location with enough degree of accuracy, thus schematic maps for web platforms can take advantage of those services to be an even more efficient navigation tool for transit services. Our implementation animates with blue growing rings to the user closest stations. Figure 5.10 illustrates the resulting effect and a live experiment on performing a journey proves its usefulness to self-location while inside a vehicle or to indentifying the right station to get off.

The given implementations above are only three examples on how digital schematic maps can be explored in the hypermedia environment of the web. Many more features could be added to an schematic map application like showing public services

associated with an specific station, or drawing the schematic map in a oblique views for navigation purposes (like car navigation system) rather than orthogonal or integration with time tables or vehicles real-time GPS data.

Figure 5.10 Schematic map and Location-based services. Using device positioning service to create a navigation perspective.

5.4 Performance issues

Performance is crucial for automatic generation of transit schematic maps that are aimed to produce personalized maps. The advantage of the GA for octalinear path schematization is that its total execution time can be controlled by stop criteria.

The stop criteria used for the maps created in this section was not a time limit, instead, it was controlled by a number g of generation passed without improvement detected. This number of g of generations is proportional to the length l of the path. For example, a path with 40 stations ($l = 40$) has g equals to 500, it means that if 500 generations is passed without a better index returned by the fitness function the algorithm stops, and if after 500 generation a better index is found the GA submit the path to more 500 generations. In the same way, for l smaller than 4, g is equal to 10; for l smaller than 10, g is equals to 50 and so on.

This stop criteria was used to generate the schematic map on Figure 5.5. This graph with all the 173 nodes (stations) and a set of 19 lines had as total execution time of 1.34 seconds (from request to submission) , in which , the GA was responsible for only 0.39 seconds of the whole process (spatial queries to the data base consumed most of the time). Table 5.1 details for each key operations, the total execution time spent.

OPERATIONS	TIME OF EXECUTION
6 × Query database and create list of spatial features	0.89 s
Depth-first-search	0.01 s
Fisheye transformations (all features)	0.01 s
GA for all paths (octilinear schematization)	0.39 s
GeoJSON composition	0.02 s
Others	0.02 s
Total	**1.34 s**

Table 5.1 Example of distribution of the total execution time.

This results show the efficiency of the GA, representing less than 30% of the total time of execution. However, in order to have a deeper understating of its performance the GA was submitted to experimental executions.

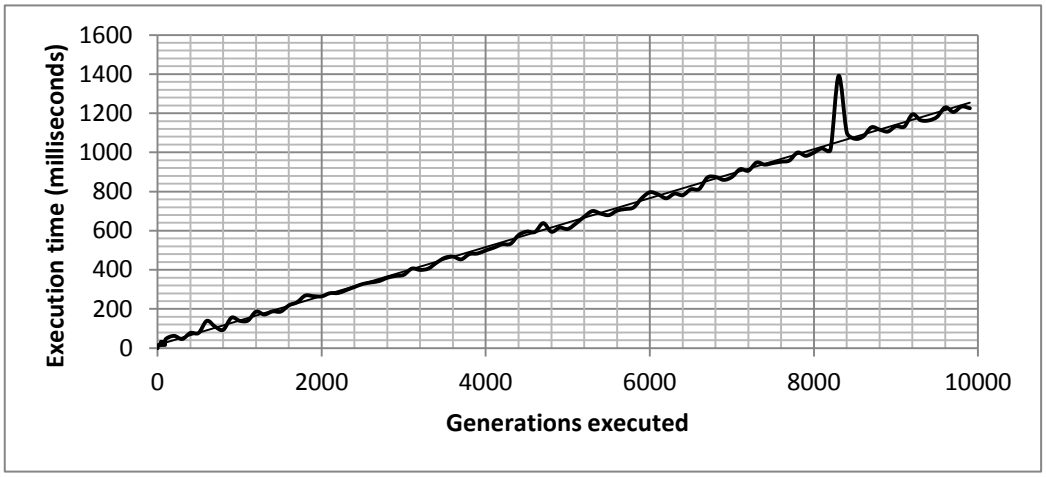


Figure 5.11 Relation generations executed and total execution time.

Two factors that affect directly the performance of a GA are the size of its instance and the number of generations executed. Two experiments were made to measure the effect of those factors in the performance of the GA. In the first experiment, the size of the instance remained constant while the number of generations grows progressively. In the second experiment, it was the number of generations that remained constant, while the size of the instances grown progressively.

The graphic in Figure 5.11 is the scatter plot of 110 successive GA executions of a path with 51 stations with variable numbers of generations per execution. The graphic shows a linear relation between number of generation and execution time. For example, the execution time of 9900 generations of a path with 51 stations was 1.23 seconds.

The graphic in Figure 5.12 is the scatter plot of 50 successive 200 generations GA executions with a variable size of instance per execution. The graphic shows a linear relation between number of stations and execution time. For example, the execution time of 200 generations for a path with 735 stations toke 0.39 seconds.

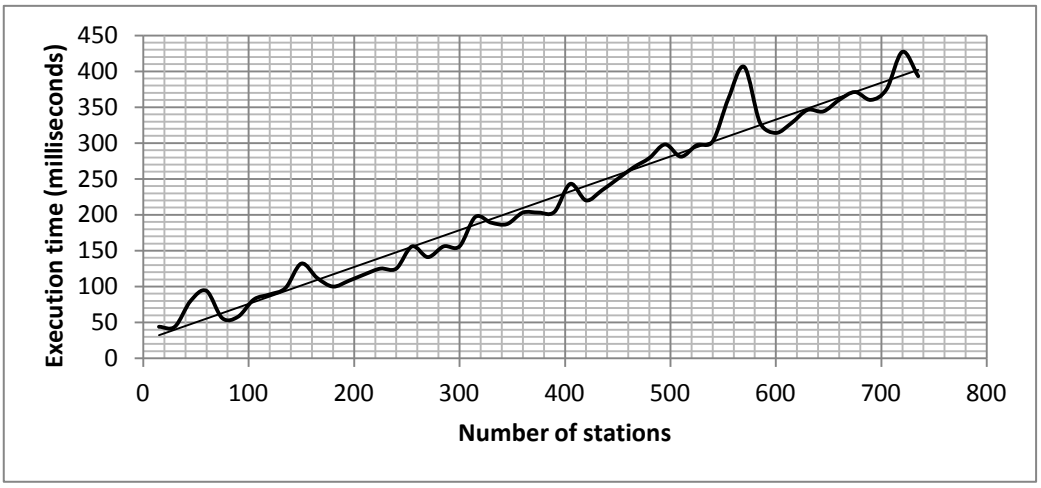


Figure 5.12 Relation instance size and execution time.

Another factor that interferes in the execution time of the GA is the size of the population per generation. This factor is important because is related to diversity in the population and helps avoiding fixation in a local minimum solution. No performances tests have been made regarding population size and they are leave as suggestion for a future work.

The fact that the total time of execution can be controlled by the stop criteria of the GA, its low cost execution, and the linear relation between number of generations and instance size with time of execution helps to prove, in terms of performance, that GA are adequate for octalinear map schematization.

6 CONCLUSION

Production of schematic maps still today requiring the hand work of professional cartographers, therefore, not only, a costly task but inappropriate to the modern mobile digital devices. Moreover, the preparation of octilinear maps for bus networks (most used public transportation mean) is even more challenging due to their complex morphology.

This research investigated, with a special focus in bus networks, new usages for real time schematization algorithms in the context of the hypermedia interactivity and mobile devices. We provided a methodology concerning the data model, data structures, schematization algorithms and web visualization of schematic data. We understand that our proposed method was able to generate the right data structures and compose the schematic transit information to full visualization and interactivity in the web environment.

Using real transit data from Castellón, a web application was developed to a practical test of the methodology. Several functionalities were included allowing the user specify his/her needs and interact with the schematic data in order to increase the power of schematic maps as a transit information source. Real time computed schematization allows a transit map to be personalized and more adequate to the visual information-seeking "mantra" proposed by Schneiderman (1996), "overview first, zoom and filter, then details on demand".

We used GA for path octilinear schematization (Galvao, Lamar, & Taco, 2014) to transform the geographic data into schematic. For this, we have implemented the suggested improvements by the authors thus resulting in a better final visual quality of the schematic maps. We also made performance tests to the GA proving its utility to computer schematic maps in terms of efficiency. As a final remark, we conclude that the proposed data structure and the genetic algorithm can be used to create schematic maps for bus networks, especially in cities that lacks of this kind of informative tool or professional designers.

7 BIBLIOGRAPHY

Allard, J. (2009). The Design Of Public Transport Maps. *Doctor thesis*. Politecnico di Milano.

Anand, S., Avelar, S., Ware, J. M., & Jackson, M. (2007). Automated schematic map production using simulated annealing and gradient descent approaches. *GISRUK*. Citeseer.

Avelar, S. (2002). Schematic maps on demand: design, modeling and visualization. *PHD Thesis*.

Barkowsky, T., Latecki, L. J., & Richter, K.-F. (2000). Schematizing maps: Simplification of geographic shape by discrete curve evolution. In *Spatial Cognition II* (pp. 41--53). Springer.

Battista, G. D., Eades, P., Tamassia, R., & Tollis, I. G. (1998). *Graph drawing: algorithms for the visualization of graphs.* New Jersey: Prentice Hall PTR.

Cabello, S., de Berg, M., & van Kreveld, M. (2005). Schematization of networks. *Computational Geometry*, *30*(3), 223--238.

Galvao, M. d. (2010). Esquematizações infográficas automáticas para transporte público: usando algoritmos genéticos. *Trabalho de Graduação*. Universidade de Brasília.

Galvao, M. d., Lamar, M. V., & Taco, P. W. (2014). Desenho automático de mapas octalineares de rede de transporte pú}blico utilizando algoritmo genético. *TRANSPORTES*, 21--30.

Garland, K. (1994). *Mr Beck's Underground Map.* Capital Transport.

GiMoDig. (2015, July). Retrieved from GiMoDig project: http://gimodig.fgi.fi/

Hong, S.-H., Merrick, D., & Do Nascimento, H. (2005). The metro map layout problem. *Graph Drwaing: Springer Berlin Heidelberg*.

Huang, Z. (2003). Data integration for urban transport planning. *Dissertation*. Utrecht University.

Jenny, B. (2006). Geometric distortion of schematic network maps. *Bulletin of the Society of Cartographers, 40*(1), 15--18.

Li, Z., & Dong, W. (2010). A stroke-based method for automated generation of schematic network maps. *International Journal of Geographical Information Science, 24*(11), 1631-1647.

Li, Z., & Dong, W. (2010). A stroke-based method for automated generation of schematic network maps. *International Journal of Geographical Information Science, 24*(11), 1631--1647.

Melanie, M. (1999). *An Introduction to Genetic Algorithms.* MIT Press.

Merrick, D., & Gudmundsson, J. (2007). Path simplification for metro map layout. *Graph drawing* (pp. 258--269). Heidelberg: Springer.

Nöllenburg, M. (2005). Automated drawing of metro maps. *Master Thesis*. Universitätt Karlsruhe, Fakultät für Informatik.

Nöllenburg, M. (n.d.). A Survey on Automated Metro Map Layout Methods. *2014*.

Nöllenburg, M., & Wolff, A. (2006). A mixed-integer program for drawing high-quality metro maps. *Graph Drawing.* Springer Berlin Heidelberg.

Oke, O., & Siddiqui, S. (2015). Efficient automated schematic map drawing using multiobjective mixed integer programming. *Computers & Operations Research 61*, (pp. 1-17).

Plaisant, C., Carr, D., & Shneiderman, B. (1995). Image-browser taxonomy and guidelines for designers. *Software, IEEE 12.2*, 21-32.

Rainsford, D. A. (2002). Mobile journey planning for bus passengers. *Geographic Information Science. Springer Berlin Heidelberg*, 228-242.

São Paulo Transporte, S. (2016, 1 25). *Indicadores: SPTrans*. (DT/AST, Producer) Retrieved from SPTrans: http://www.sptrans.com.br/indicadores/

Sarjakoski, T., Sarjakoski, L. T., Lehto, L., Sester, M., Illert, A., Nissen, F., . . . Ruotsalainen, R. (2002). Geospatial info-mobility services-a challenge for national mapping agencies. *International Archives Of Photogrammetry Remote Sensing And Spatial Information Sciences 34.4*, 356-360.

Sarkar, M., & Brown, M. H. (1994). Graphical fisheye views. *Communications of the ACM, 37*(12), 73--83.

Shneiderman, B. (1996). The eyes have it: A task by data type taxonomy for information visualizations. *Visual Languages, 1996. Proceedings., IEEE Symposium on* (pp. 336--343). IEEE.

Stott, J. M., & Rodgers, P. (2004). Metro map layout using multicriteria optimization. *Eighth International Conference on Information Visualisation, 2004. IV 2004.* IEEE.

Swan, J., & al, e. (2007). Automated schematization for web service applications. *Web and Wireless Geographical Information Systems* (pp. 216-226). Springer Berlin Heidelberg.

Ti, P. (2014). Automated generation of adaptive schematic network maps. *PHD thesis*. The Hong Kong Polytechnic University.

Ti, P., & Li, Z. (2014). Generation of schematic network maps with automated detection and enlargement of congested areas. *International Journal of Geographical Information Science, 28*(3), 521--540.

Vuchic, V. R. (2005). *Urban transit: Operations, planning, and economics.* Hoboken: Wiley.

Wang, Y.-S., & Chi, M.-T. (2011). Focus+ context metro maps. *Transactions on Visualization and Computer Graphics, 17*(12), 2528--2535.

Wolff, A. (2007). Drawing subway maps: A survey. *Informatik-Forschung und Entwicklung, 22*, pp. 23--44.