University of Porto

Doctoral Thesis

Automated Generation of Context-Aware Schematic Maps: Design, Modeling and Interaction

Author:
João Mourinho

Supervisor:
Prof. Teresa Galvão
Co-Supervisor:
Prof. João Falcão e Cunha

A thesis submitted in fulfilment of the requirements
for the degree of Doctor of Philosophy

in the

Department of Engineering and Industrial Management
FACULTY OF ENGINEERING OF THE UNIVERSITY OF PORTO

July 2015

Declaration of Authorship

I, João Mourinho, declare that this thesis titled, 'Automated Generation of Context-Aware Schematic Maps: Design, Modeling and Interaction' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

UNIVERSITY OF PORTO

Abstract

Faculty of Engineering of the University of Porto
Department of Engineering and Industrial Management

Doctor of Philosophy

Automated Generation of Context-Aware Schematic Maps: Design, Modeling and Interaction

by João Mourinho

Spider Maps are schematic maps with enhanced features which make use of the conceptual spider maps design principles. They share a set of characteristics with traditional schematic maps, but they include innovative features such as a spider architecture which makes them an enhanced vehicle for spatial context communication for public transport networks. This thesis presents a comprehensive state of the art analysis regarding the cartographic evolution through time, the automated generation schematic maps and contextual aspects of map use. It also defines the concept of Spider Map, its design properties and presents a proof of concept regarding their advantages in comparison with traditional diagrammatic maps, both in what concerns to learning enhancement by reducing extraneous cognitive loading and to support user actions in a transportation network. These evidences were supported by a usability test conducted with real users. As most of the production of schematic maps is currently done manually or by assisted methods, being both approaches resource (time and money) expensive, this thesis proposes an approach which includes an innovative algorithm based on the tabu search metaheuristic to effectively generate them. A set of tests were conducted with real world maps, with the results showing excelling results in what concerns to execution speed, quality of solutions and the ability to support geographical constraints (such as rivers, lakes, mountains, etc). It is also shown show that the automated generation of spider maps can improve Location Based Services and interaction in Public Transportation systems.

Universidade do Porto

Resumo

Faculdade de Engenharia da Universidade do Porto
Departamento de Engenharia e Gestão Industrial

Tese de Doutoramento

Geração Automática de Mapas Esquemáticos Sensíveis ao Contexto: Desenho, Modelação e Interação

por João Mourinho

Os Mapas-Aranha de transporte são mapas esquemáticos com características melhoradas que utilizam os princípios de desenho dos mapas-aranha conceptuais. Apesar de partilharem um conjunto de características com os mapas esquemáticos tradicionais, incluem também particularidades inovadoras como a arquitectura em aranha. Os Spider Maps são um meio melhorado para a comunicação de contexto espacial em redes de transporte público. Esta tese apresenta uma análise completa do estado da arte no que toca à evolução cartográfica através do tempo, da geração automática de mapas esquemáticos e dos aspectos contextuais do uso dos mapas. Define também o conceito de Mapa-Aranha, as suas propriedades de desenho e apresenta uma prova de conceito no que toca às suas vantagens em comparação com os mapas diagramáticos tradicionais, tanto no que diz respeito às melhorias de aprendizagem pela redução da carga cognitiva como no apoio às ações do utilizador numa rede de transportes. Esta prova é apoiada por um teste de usabilidade com utilizadores reais. Como a maior parte da produção de mapas esquemáticos é atualmente feita manualmente ou de forma semi-automática, acarretando custos em termos de dinheiro e tempo, é também proposta uma abordagem inovadora para a sua geração de forma eficaz e eficiente que inclui um algoritmo baseado na meta-heurística pesquisa tabu. Esta abordagem foi testada através de um conjunto de mapas reais com bons resultados no que toca ao tempo de execução, qualidade das soluções e capacidade de suportar os acidentes geográficos (tais como rios, lagos, montanhas, etc). Esta tese mostra também que a geração automática de mapas esquemáticos pode também melhorar os Serviços Baseados na Localização e a interacção em sistemas de transporte público.

Acknowledgements

Special thanks to my advisor Professor Teresa Galvão and to my co-advisor Professor João Falcão e Cunha for their great inspiration, support, help and true friendship along this project. I want to thank all the support and sympathy i had from all the Department of Engineering and Industrial Management people, in particular from Professor José Fernando Oliveira and Professor José Sarsfield Cabral. To my office colleagues and friends who walked along with me, a big thanks. They were always friendly, supportive and exemplar in all aspects of our daily work. I would also like to thank the people of OPT[1] and FWT[2] for their collaboration and to INEGI[3] for funding this research project.

[1]OPT is an company based in Porto which develops IT infrastructures for Transportation Services. http://www.opt.pt

[2]FWT is a company based in London which produces maps for transportation networks. http://www.fwt.co.uk

[3]INEGI is a research institute based in Porto

Contents

List of Figures

List of Tables

Abbreviations

CPU	Central Processing Unit
FWT	London Based Company that produces the London Bus Maps
GIS	Geographic Information System
GPS	Global Positioning System
GRASP	Greedy Random Adaptive Search Procedures
GUI	Graphics User Interface
HPPO	Heuristic Point Placement Optimization
ICA	International Cartographic Association
IPC	Inter-Process Communication
KPI	Key Performance Indicator
LBS	Location Based Services
MIP	Mixed Integer Programming
OPT	Optimização e Planeamento de Transportes
PT	Public Transportation
SGDS	Simple Graph Data Structure
SRDS	Semantically Rich Data Structure
SVG	Scalable Vector Graphics
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UML	Unified Modeling Language
VLSI	Very Large Scale Integration
XML	Extensible Markup Language

To my parents for their unconditional love,
to my sister and my two little nephews for their joy,
to my shining angel Isabel,
and to God, to whom i owe everything.

Chapter 1

Thesis Overview

1.1 Motivation and Problem Definition

Public transportation systems are fundamental to support efficient mobility and cities' livability. With the increasing complexity of public transportation networks, the existence of clear and easy-to-read transportation maps helps users to learn and understand the underlying network thus increasing ridership in public transports. The main questions when one looks at a transportation map are *"Where am I?"* and *"From here, where can I go to?"*. Improving passenger information does not mean to increase the amount of information provided to user. Instead, by eliminating superfluous information and entropy, it is possible to increase map learnability. A good transportation map will provide to the user the correct space context to make the map easier to understand and thus speed up trip planning. Schematic maps, due to their apparent simplicity, have been used to depict transportation networks on public transportation systems. They are diagrammatic representations based on highly generalized lines, usually not following precise geographic accuracies. They use a non-linear scale that emphasizes some details while omits or removes relevance from others. A good example of a schematic map is the one produced by Harry Beck for London Underground in 1933 [25].

A particular type of schematic map called Spider Map is particularly well suited to improve user learning due to its inherent context improving features. Spider maps are suited to depict bus or underground transportation networks in a city or region. Instead of representing the entire network, spider maps represent a particular area of specific context, but they are still able to provide the user with the complete transport network overview. The diagrammatic representation has a central area, named Hub, from which all the schematic routes leave. This central area may depict a geographic representation of a borough, a tourist area or a transport interface. The Hub adds context to the

information provided to the user, as it provides details about the area where the user currently is, such as streets names, bus stops locations or relevant touristic places (see figure 1.1).

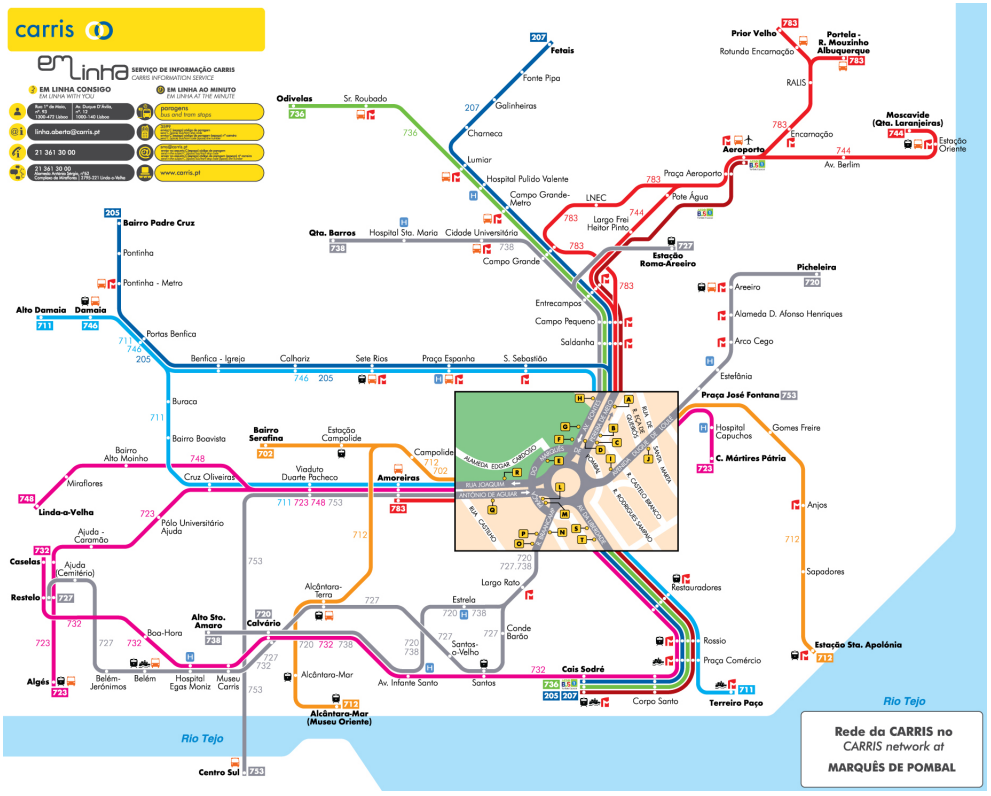

FIGURE 1.1: A Spider Map in Lisbon showing the bus network available from Marques de Pombal

Several cities around the world offer schematic maps to the citizens, mainly for metro networks or bus networks. Teams of cartographers and designers produce these schematic maps by hand. This manual work involves high costs, especially when the transportation network suffers frequent changes. In 2009, MBTA (Boston, USA) launched a 2-year project for replacing the existing schematic map with a total cost of $500,000 for labor and production [45]. The production of schematic maps is, therefore, an expensive and time consuming task. As a consequence, schematic maps cannot be effectively used for powering Location-Based Services (LBS) where near real time agile schematic map production is needed, nor the contextual advantages of spider maps can be fully extracted. Consequently, there is the need to develop methods to automatically generate spider maps in an effective and efficient way.

In this research work we propose an approach to automatically generate Spider Maps through a comprehensive information system that includes an innovative algorithm based on the tabu search meta-heuristic that can produce ready-to-publish quality spider maps in soft real time. The automated production of spider maps opens a new world of context

dependent applications and drastically reduces production and maintenance time and costs. To our best knowledge, the automatic generation of transportation spider maps has not been addressed in literature, and all the research made on the topic of the generation of schematic maps fails to model the multidisciplinary nature of the problem in some aspect.

With this research we also intend to define spider maps, their properties and to demonstrate that they are more effective than traditional schematic maps in the communication of public transports information, user learning and interaction, testing them with real users and real world maps.

If generated automatically, spider maps can be an effective tool in powering LBS, increase information quality available to users and increase Public Transportation (PT) ridership.

1.2 Research Questions

The research questions that guide this thesis are the the following:

- **Q1** What are the significant features that define the spider map, concerning its mathematical properties and visual presentation?
- **Q2** Can we prove that Spider Maps are better than traditional Schematic Maps concerning to user experience in wayfinding?
- **Q3** How to effectively generate Spider Maps through an information system addressing the issues of the current state of the art?

1.3 Objectives

This research work has the following objectives:

- Describe the state of the art in what concerns to the problem of the design, modeling and interaction of schematic maps.
- Define and systematize the set of features that comprise effective context-aware schematic maps (spider maps).
- Test the validity of the spider map concept and its advantages regarding traditional schematic maps both conceptually and in practical use in public transportation networks.

- Build a theoretical framework that supports the assumption that spider maps improve LBS and PT ridership
- Develop an effective and efficient system of computer algorithms capable of generating good context-aware schematic maps in soft real time, by mapping into the algorithm knowledge from different areas.
- Develop a software framework that could support and power the algorithms in the task of automatically generate context-aware schematic maps
- Evaluate the performance and quality of the algorithms that were developed.
- Impact real world public transportation, both socially and economically, bringing advantages both to transportation company ecosystem and end users.

1.4 Methodology

In order to be successful, this investigation followed a mixed approach, as literature revision showed us that the design, modeling and interaction of context-aware schematic maps generated by automatic means is not complete and there are many voids in what concerns to its state of the art. The knowledge, therefore, needs to be obtained not only by a objective/quantitative approach (mathematics, computing algorithms) but also by a qualitative vision, as it deals with subjective concepts (like human perception, user satisfaction, information quality) which are inherently subjective and context-dependent. In addition, this is a complex multidisciplinary problem that involved a great deal of research and information integration from different (but often intersecting) areas of knowledge such as computer science and software engineering, mathematics and operations research, artificial intelligence, cognitive psychology, linguistics, services, human-computer interaction, arts and cartography.

1.4.1 Automated Design of Context-aware Schematic maps

In what concerns to the problem of the automated generation of context-aware schematic maps, it was important to understand the nature of the problem, why the problem is important and how this research could contribute to improve the actual state of the art. In order to achieve that, a comprehensive literature and state of the art analysis was carried out along with a critical analysis of the advantages and disadvantages of several researches on the field. The strong points and shortcomings of each approach to this problem were summarized. This study can be classified as exploratory and descriptive, as the investigation questions found helped to shape our investigation. The solution of this

problem, more than purely academic or theoretical, needs to have a strong connection to real world (and have social and economic consequences). Therefore, the close cooperation with real world business experts in the transportation field such as OPT and FWT was crucial to mold the course of the research as real world presents some challenges that are not usually found in purely conceptual problems. The implementation of the information system that could generate context-aware schematic maps in an automatic way was not a sequence of activities with some backtracking from one activity to another. It was based on a spiral model instead[46], as depicted in figure 1.2, for the development of the main algorithms, mixed with an incremental delivery model (figure 1.3) when small improvements were produced for each of the main algorithms.

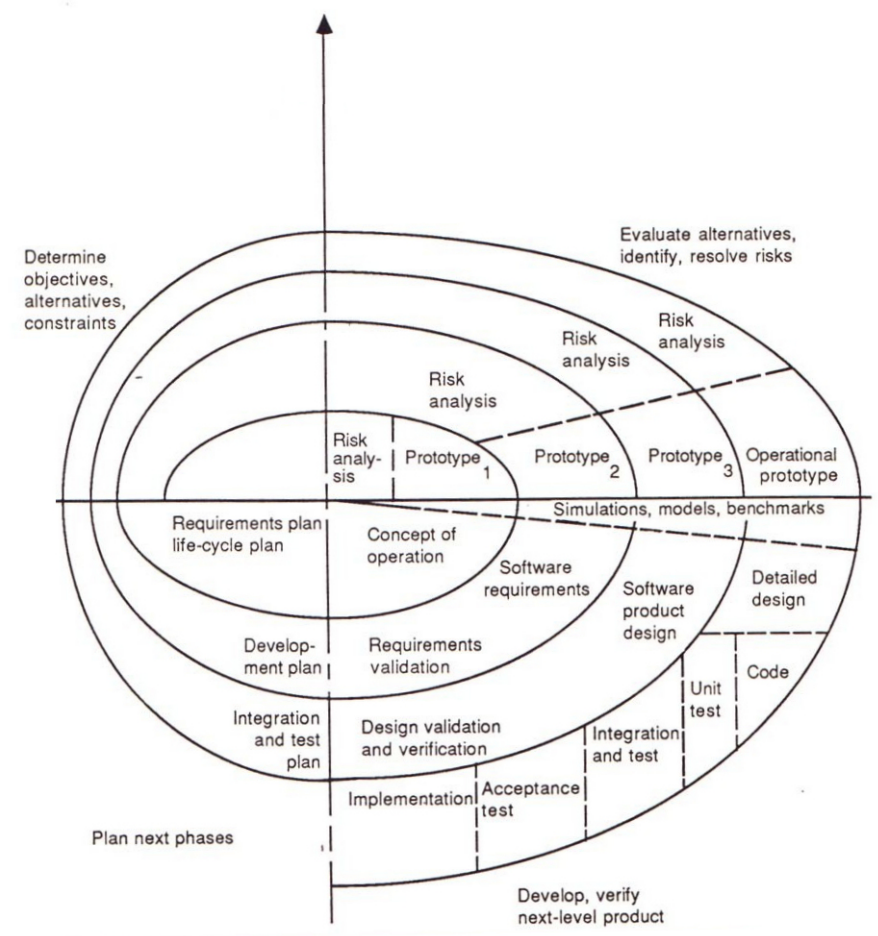

FIGURE 1.2: Boehm's spiral model of the software process (c) IEEE 1988 [1]

This mixed development model allowed us to combine the advantages of both models: for the main algorithms, each spiral cycle allowed us to elaborate objectives such as performance and functionality. Alternative ways of achieving these objectives and the constraints imposed on each of them are then enumerated. Each alternative is assessed against each objective and the advantages and risks of each option were identified. The identified risks were then addressed through the literature revision or, if no literature

on the issue existed, creative ways of tackling the problem were developed, followed by more detailed analysis, prototyping and simulation. Having done this, the development of the software was carried one (algorithm coding, unit tests, integration, integration tests.

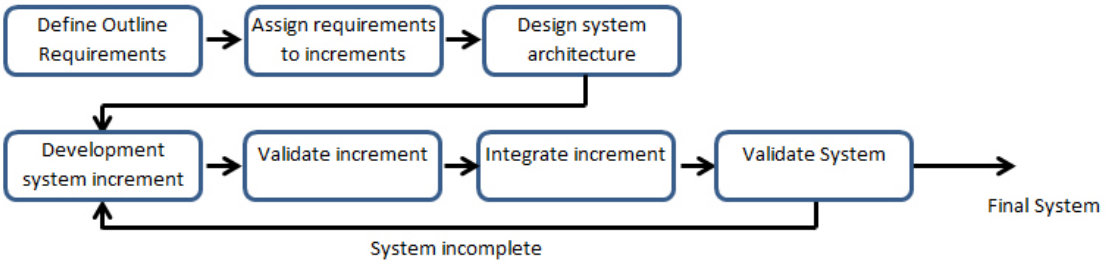

FIGURE 1.3: Incremental Delivery Software Development Model [2]

When an algorithm was developed through the spiral model, before proceeding to the next spiral cycle, it entered in an incremental delivery model: making small improvements and delivering them till the improvement potential of the algorithm was exhausted. After that, the algorithm proceeded to the next spiral model cycle.

The computer software framework supporting the algorithm system for the development of context-aware schematic maps was specified through a complete descriptive method, as the framework properties and architecture needed to be described (software engineering requirement elicitation). The properties of the framework and the generated maps needed to be tested through a pure experimental plan, where several schematic maps (intentionally varying in its properties and characteristics) were tested.

1.4.2 Definition and Interaction of Spider Maps

As the concept of Spider Maps for as a context-aware type of schematic map for depicting transportation networks was non-existing in the literature, the first task was to systematize their features and elaborate their definition. To prove the validity of this definition and all the advantages concerning traditional schematic maps, a user test based on the best quality *versus* cost examples of usability testing found in the literature was conducted, through a field experiment. This assured the external validity of the results. The satisfaction level of the users was measured through a brief written interview, with a couple of open questions, in order to gather valuable feedback from the users. The interview was semi-structured as there was a simple predefined question structure but there was space given to users if they wanted to disclose further details about their user experience. Learning times and other relevant indicators were measured through the observation method and the use of measuring instruments. Then, the information

was put together with the interview results. The intention was not to get an absolute measurement of user friendliness, but a relative comparison measure between traditional schematic maps and spider maps and classify their comparative quality.

1.5 Thesis Outline

The research in this thesis investigates and contributes to the design, modeling and interaction challenges of context-aware schematic maps applied to transportation networks. The thesis is organized in seven chapters, with chapter 1 being the introduction.

In Chapter 2 the foundations on where our research work was developed are described. The state of the art regarding the evolution of maps and the characteristics of schematic maps and the spatial communication exercise in which they take part are exposed. The evolution of the research on the automated generation of schematic maps is also presented, with a relative evaluation of each approach in what concerns to its advantages and shortfalls. Then, important previous work about cognitive psychology, human interaction and context aspects in maps is detailed. A description of the Location Based Services and their characteristics is also presented.

In Chapter 3 the context-aware enhanced schematic map, called Spider Map, is defined. Its formal definition, characteristics and particular advantages in what concerns to traditional schematic maps are presented. In this chapter it also made proof of its concept, through the description of a user testing process regarding the use of spider maps *versus* traditional schematic maps. The chapter ends by highlighting the theoretical bases on how spider maps can enhance Location Based Services.

Chapter 4 thoroughly presents the modeling of the problem of the automated generation of Spider Maps, starting with the description of the problem. The Spider Map data structures and multicriteria optimization modeling are portrayed.

The approach to automatically generate spider maps is proposed in Chapter 5, regarding its architecture and algorithms.

The tests conducted on real map instances with the algorithms developed based on our approach are presented in Chapter 6, together with the description of the GenX software framework developed along this research work. The tests are described and their results are presented and analyzed.

Finally, the conclusions and contributions of this thesis are highlighted and the insights for future research are presented in chapter 7.

Chapter 2

Literature Revision

After starting the research work, it was soon clear that the problem of automatically generate context-aware schematic maps involved integrating a set of sparse knowledge from several areas. Therefore, this section presents the state of the art and evolution of knowledge in the areas of science that have been identified as being relevant and that served as basis for all the research work presented in this thesis.

2.1 Maps and Cartography

2.1.1 Map Definition

According to the definition of International Cartography Association (ICA) [47], "*a map is a symbolized image of a geographical reality, representing selected features or characteristics, resulting from the creative effort of its author's execution of choices, and is designed for use when spatial relationships are of primary relevance*". The term "map" is likely a model of what it represents, a model that allows us to understand the structure of the phenomenon represented. Therefore mapping is more than just rendering; it is also getting to know the phenomenon which is to be mapped [48]. Maps are a graphic representation of a geographical setting: they are a collection of symbols used to represent a portion of the space[23]. Maps are concerned with locations and attributes and possess a set of intersecting definitions:

- Maps are reductions of reality and therefore require scale conversions
- Maps are transformations of space

- Maps are abstractions of reality and therefore require generalization of the information
- Maps use signs and symbolism.

The concept of map as an abstraction of geographic reality is also considered by Board[49]. In fact, all maps are, to some extent, simplified or generalized [50]. Being so, it is easy to understand how the faithfulness of the representation of reality can be compromised through a balance between their connection to reality and scale, abstraction, and symbolism requirements[23]. As we know, it is impossible to show the real world in detail without sacrificing scale. At the limit, we would have 1:1 scale maps, which would be as large as reality itself, which is clearly undesirable and useless. Jorge Luis Borges, a a famous Argentinian writer once wrote a fictional story called "on Exactitude in Science" which tells of an Empire where *"(...) the Art of Cartography attained such Perfection that the map of a single Province occupied the entirety of a City, and the map of the Empire, the entirety of a Province. In time, those Unconscionable Maps no longer satisfied, and the Cartographers Guilds struck a Map of the Empire whose size was that of the Empire, and which coincided point for point with it. The following Generations, who were not so fond of the Study of Cartography as their Forebears had been, saw that that vast map was Useless, and not without some Pitilessness was it, that they delivered it up to the Inclemencies of Sun and Winters. In the Deserts of the West, still today, there are Tattered Ruins of that Map, inhabited by Animals and Beggars; in all the Land there is no other Relic of the Disciplines of Geography."* [51]. As this curious story shows, scaling has clear practicability and usefulness determinant advantages, at cost of smaller space between map features, less detail and reduced lengths. Features may become too small to see, patterns may disappear as features coalesce, labels can overlay each others reducing readability. Cartographers need to make compromise decisions in what concern to the selection of the number and size of features that are presented on the map.

2.1.2 Maps as Communication Vehicles Through Times

Since the most remote times, man has tried to dominate the world by capturing the two dimensions of his existence: time and space. As he made watches to represent time, he also made maps to represent space. There is clear evidence that mapping precedes both written language and systems involving numbers [52], so it may be said that spatial communication preceded written communication. With the exception of a few number, there have been no mapless societies in the world at large. Despite all the gaps in

temporal or geographical records (due to nonsurvival of evidence), it is widely accepted that the creation of maps accomplished four main functions in ancient societies:

- Graphical wayfinding and description of the real world
- Sacred and cosmological representation of the religious world
- Promotion of secular ideologies
- Aesthetic function or decoration

The evolution of maps was not uniform across all cultures. Map perspectives for example, differed from culture to culture, and varied according to their main functions. Another example is the map limits and map shape. While to the modern map user the bounding frame announces completeness and limits the map representation from the surrounding space, this concept has only appeared in modern maps, as previously the map boundaries were defined by the shape and characteristics of its physical support. The oldest known map (fifth century BC), the Babylonian *Imago Mundi* [3], was made of clay and had no geometrical shape at all (figure 2.1), while the most of the ancient Greek maps possessed a circular form, such as Hecateus of Milethus map [4] (figure 2.2) or the Shield of Achilles [5] which is acknowledged to present the Earth, sky and sea, the moon an the constellations (figure 2.3). At this time, map shape started to differ, mostly because the campaigns of Alexander the Great, as maps took new shapes, such as the Ptolemy world map [6] as shown in a reconstruction from the fifteenth century in figure 2.4.

The Peutinger map in figure 2.5, which dates from the twelfth or early thirteenth century A.D. may be taken to indicate the strong likehood that graphic itineraries existed in the Roman period, although this map itself is distant from any demonstrated Roman original. The manuscript can be dated to the twelfth or thirteenth century, but it is clear that it is a copy of a much older original [7].

Although the Peutinger Map is too compressed to be a real rendering of the landscape, as its predecessors, its creator did not attempt to show places and distances at the same scale, nor is the north always at the top of the map (In fact, it is better compared to the maps of the subways in modern cities, showing how one has to travel and mentioning the distance between the stops, but without pretension to show with precision the surface of the earth). The development of cartography during the middle ages was based mostly on religious interpretations and travels reports. One example is the Tabula Rogeriana [8] (figure 2.6) produced in 1154 by an Arab geographer called Muhammad al-Idrisi who merged reports from several Arab merchants with classic geographical knowledge. As it is possible to see it is oriented south up, featuring Europe, Asia and Africa upside down.

FIGURE 2.1: *Imago Mundi* [3] (5^{th} century B.C.), the oldest known map

FIGURE 2.2: Hecatheus of Miletus world map (modern reconstruction) [4] (4^{th} century B.C.

FIGURE 2.3: The Shield of Achilles Map [5], modern recreation of the shield described at Homer's Iliad (13^{th} century B.C.)

FIGURE 2.4: Ptolemy World Map [6] (2^{th} century A.D.

China map development was considerably more advanced than in Europe, and featured sophisticated mapping techniques that its western counterparts were only able to match on the fourteenth century. One famous map was the Da Ming Hum Yu Tu map [9] (figure 2.7, which benefited from information brought by Mongolian trade routes with the west. Painted in silk in 1389, it featured disproportional vision of the world, depicting a oversized China with the rest of the world looking tiny.

FIGURE 2.5: The Peutinger Map (first sheet), showing roman routes [7] (From the 13th century A.D., believed to be a copy from a much older original).

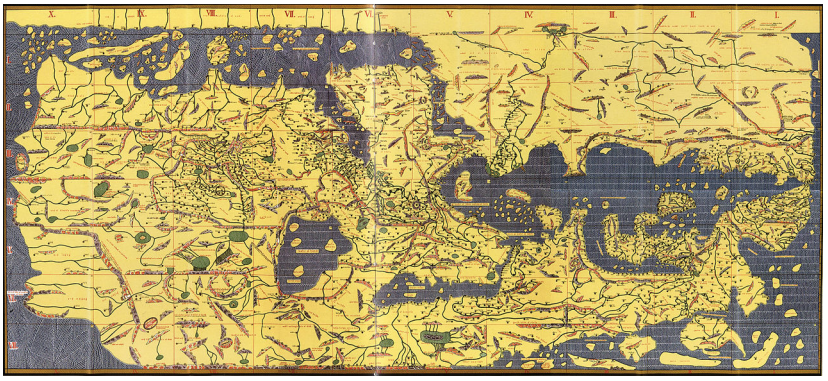

FIGURE 2.6: The Tabula Rogeriana [8] (12th century A.D.)

FIGURE 2.7: The Da Ming Hum Yu Tu map [9] (14th century A.D.)

Before the Renaissance in the fifteenth century, Venetian and Genoese map makers used rounded shame maps (circles or ellipsis) to depict world, shown in figures from 2.8 to 2.11 [10] [11] [12] [13].

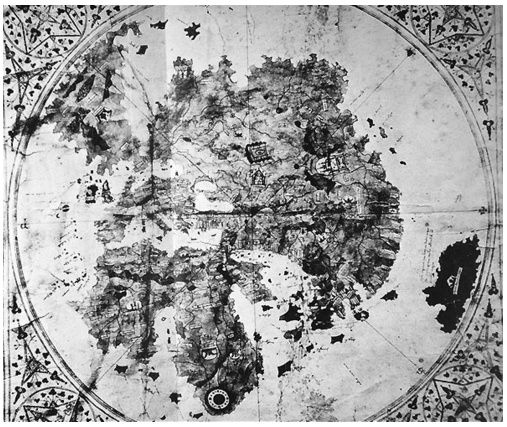

FIGURE 2.8: De Virga World Map [10] (early 15th century A.D.)

FIGURE 2.9: Bianco World Map [11] (early 15th century A.D.)

With the Renaissance and the Age of Discovery, the physical world was gradually being mapped, and cartography was increasingly important. At this time, maps acquired the standard orientation we are now used to (north up), as seen in the world map of Diogo Ribeiro (figure 2.12), a Portuguese Cartographer from the sixteenth century.

Refinements from the age of discovery maps were limited to the detail of the discovered world. Modern concepts such as the "paper map" and the "north up" orientation were inherited from those days and became standard features. Through time, it is possible to observe a set of cognitive transformations leading towards the *"the idea of the map"* [52] as a basic form of human communication and involving changes in modes of thinking

FIGURE 2.10: Genoese World Map [12] (late 15th century A.D.)

FIGURE 2.11: Fra Mauro World Map [13] (late 15th century A.D.)

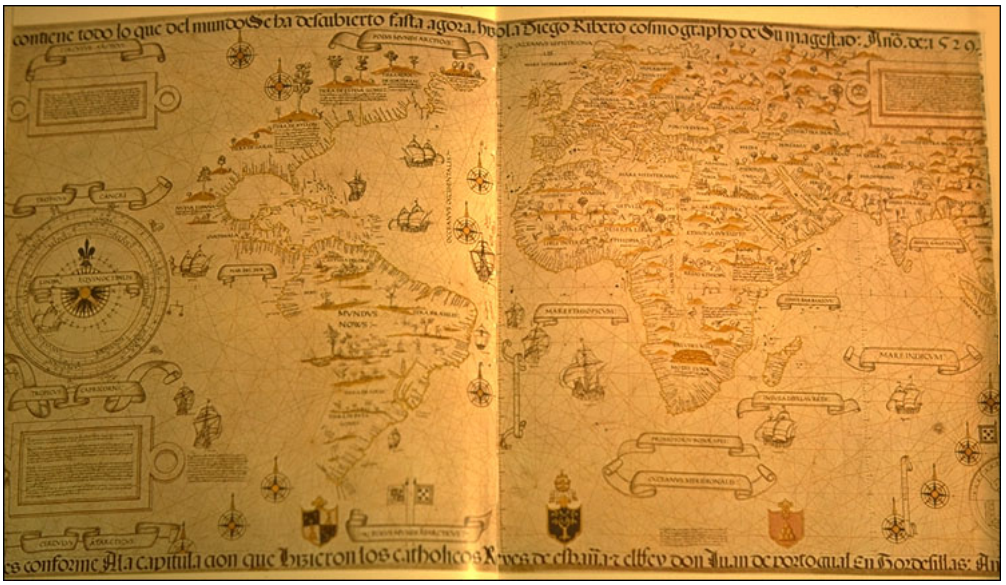

FIGURE 2.12: Diogo Ribeiro World Map [14] (16^{th} century A.D.)

about, and graphic representation of, the world at various scales. One of those cognitive transformations was the recognition, by groups of people, that what today we call a map could record and structure human experience about space. Whether this was intuitive or conscious, a graphic language of maps was being developed. Other cognitive transformation derives from the former one: maps have become graphical designed artifacts with distinctive geometrical structures and arrays of signs recognizable to the intended viewers. We can think of the syntax, semantics and pragmatics here, if we are describing the use of maps as a communication process. With the evolution of maps, a set of common geometric elements appeared: all maps share a number of common elements, as the frame, which bounds the area of the map. This was an innovation of the Renaissance era, as it was not present in the Peutinger Map (figure 2.5, where the area of the map was limited by the format of the scroll on which it was drawn. A second geometric element was the nonexistent notion of uniform scaled image, as in many of those maps some areas were given significantly different weight and map space, according to the map function and political or religious importance. Other geometric feature was the centering of maps: a point of higher importance was chosen to be the center of the map. This reflects the same sort of manipulation of the geometry of the map to fit a specific perception of the world. Another very important geometric feature which is fundamental to influence the cognition of space is map projection and orientation, as a transformation from real coordinates to map coordinates had to be performed. More than a mirror of society, maps are a reciprocal part of cultural growth and influence the pattern of its development.

Cartographic activity has increased with the transition from the Renaissance to the Enlightenment era, as the exploration of the globe was being executed at high speed and geographical findings were being rationalized and analyzed together with other types of knowledge. Mercator, Ortelius, Cassini and many others improved the cartographic science on that time frame, and thus maps begun to reflect that integration in the work of newly created scientific institutions to bolster cartographic works. Government and administrative institutions increasingly relied on maps in order to manage and control their territories, although this remained very much an unstructured process until the beginning of the nineteenth century. Expanded map consumption resulted from increasing economic stability and growth after the seventeenth century. This in turned spurred increased literacy, allowing the middling sort to engage in cultural and political criticism within the "public sphere". An increased widespread print and visual culture produced maps that adhered to a common aesthetic of layout and design [53]. An example is the Atlantic Neptune atlas by Joseph DesBarres, an important collection of maps of North America made for the British Admiralty depicted in figure 2.13.

FIGURE 2.13: Joseph DesBarres Atlantic Neptune Map [15] (18^{th} century A.D.)

At the late 18^{th} century, the industrial revolution brought a wide set of scientific, economic and social changes which pushed new developments, namely in the transportation field. [54] The geographical world had already been discovered but the transportation systems (railways, roads, airways, underground systems, high-speed trains) have been

growing till today, and they are expected to continue to grow. Large urban areas appeared and needed complex transportation systems, combining different transportation types. The need of highly efficient, easily understandable (and thus, practical) transportation maps pushed the evolution of the traditional maps, and new forms of cartographic representation have emerged. The importance of route maps on modern society has been increasing. At the nineteenth century, railway and route maps started to be more common. One example is the 1841 Pigot and Co.'s New Map Of England and Wales With Part Of Scotland, which depicted mail roads, turnpike roads, cross Roads, rail roads, rivers and navigable canals (figure 2.14).

With the massification of the automobile in the twentieth century and increasingly dense transportation networks, transportation maps (such as road maps) became widespread. An example is the Michelin Road map collection [17]. The second half of the twentieth century has seen the rise of computing technology, and computers became an important base of all human activity. The first Geographic Information Systems (GIS) was developed in the 1960 decade in Canada by Tomlinson [55] and their use become generalized in many activities such as public service planning (including public transportation networks) and military applications. The GIS conceptualized by Tomlinson was a digital analogy of the physical multilayer geographic maps previously used. Modern GIS technologies use digital information merged and filtered from several sources (aerial or satellite imaging, digitization of geographic points, etc) to present a true integrated information system. Other important development was the Global Positioning System (GPS) which helped to add the variable "time" into GIS and turning them in Real Time GIS [56].

The development of personal computing, portable and wearable devices, together with the onset of real time GIS and GPS democratized the access to GIS. New types of navigation and wayfinding appeared, such as the personal navigation systems with user-centered perspective [57] [58], as seen in figure 2.16.

Parallel to these developments was the use of the World Wide Web as communication platform. For many years, it has been the medium to acquire and disseminate geospatial data. This resulted in new mapping techniques as well as new possibilities for uses not seen before with traditional printed maps and most on-screen maps. A true revolution was caused in 2005 by the introduction of the programmes Google Earth and Google Maps [48]. Everyone with internet access could display satellite data and maps for free on a level of detail that was not heard of before (figure 2.17)

From the early Babylonian maps to the current map applications, it is possible to see that mapping provided a vehicle not only for space presentation but also for the implicit and explicit transmission of ideas. Maps are part of the communicative process intended to

FIGURE 2.14: Pigot and Co.'s New Map Of England and Wales With Part Of Scotland (detail of London) [16] (19^{th} century A.D.)

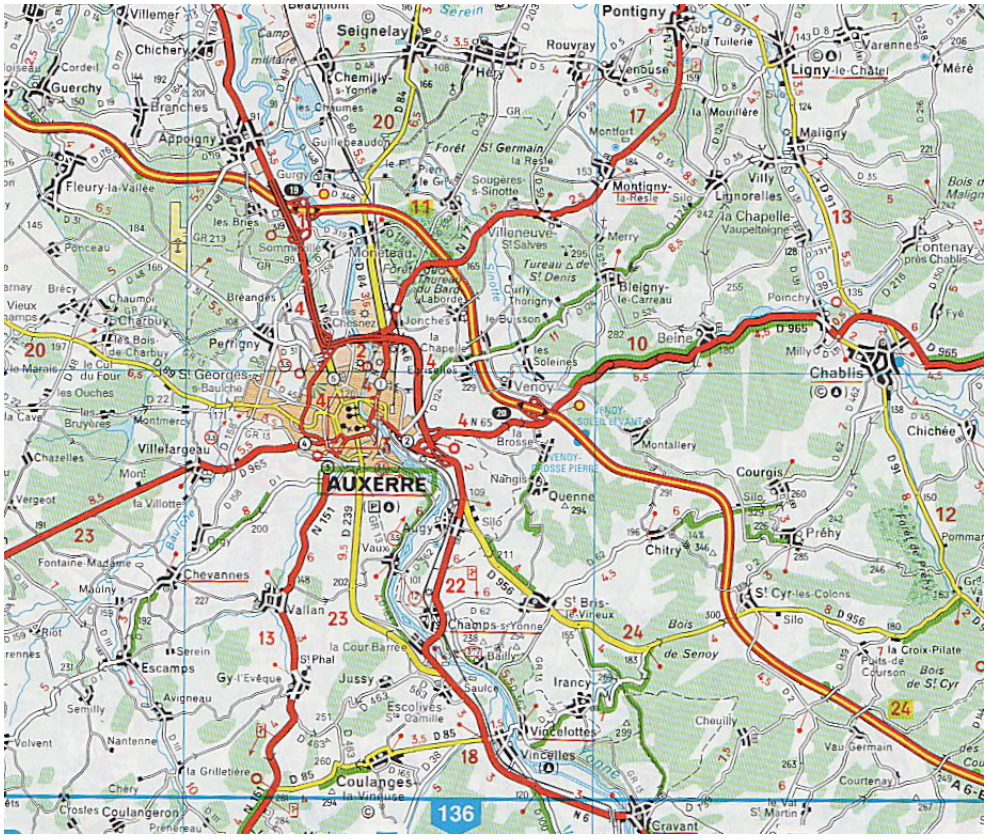

FIGURE 2.15: Michelin Road Map (detail of Auxerre) [17] (20^{th} century A.D.)

FIGURE 2.16: Smartwatches with GPS Personal Navigation [18]

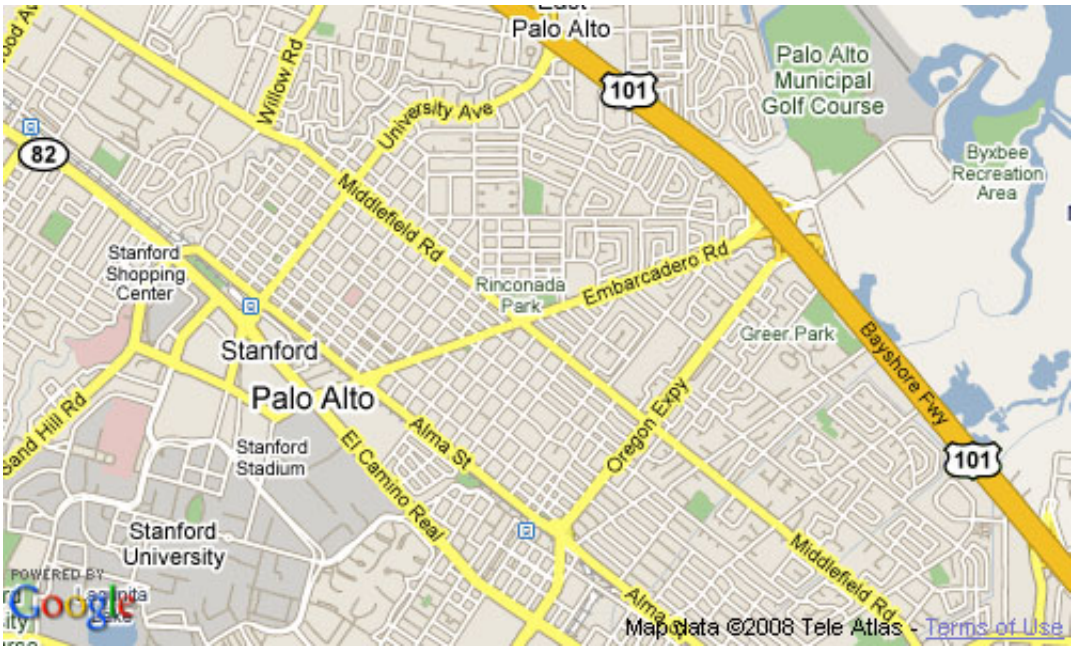

FIGURE 2.17: Google Maps - Palo Alto area [19]

communicate space information: the producer of maps (i.e. the sender) communicates to the receiver the message (map). The making of maps was only possible through the use of symbols and abstraction (which serve as a language), as maps are mainly intended to communicate space information. They use a language, a conceptualization of the reality. However, *"(...) we must accept, although our general position is founded in semiology, that precise scientific analogies to the structure of language may be impossible to sustain."* [52]. Being so, as with every translation from a real domain to a conceptual domain we accept that inevitably there is loss of information. A similarity is the conversion of an analog signal to a digital signal: the analog signal is continuous and contains the "truth" about the signal, while the digital signal is a discretization of the continuous signal, and therefore it is a simplification. The fidelity level of the digital signal is related to its sampling frequency and resolution. The same happens in cartography: if we increase the fidelity level of a map, we need more information resolution. In practice this means magnified scales (the more magnified, the closer to reality till reach the 1:1 scale) and less schematization. Of course this implies, for example, a larger piece of paper as a map basis (or a larger screen, if we talk about digital visualization). As its dimensions increase beyond the reasonable, the advantages of the map start to fade. Therefore, the schematization and simplification of detail is an inherent and necessary part of cartography. In some special cases, the geometric accuracy in the map will be less important than functional relationships between mapped features. To serve this purpose, mapmakers sometimes distort map's geography to show relations that are more meaningful to the map users [23]. Thus the degree of information loss (distortion

or dissimilarity with reality) is related to the trade-off between connection to the reality and practical use of the map.

2.1.3 Map Taxonomy

Although there is not an explicit agreement regarding the taxonomy of maps, they can be divided into two main classes with non intersecting characteristics: topographic and thematic[59]. While topographic maps communicate the general image of the surface of an area (figure 2.18), thematic maps represent the distribution of one or several particular phenomena (figure 2.19) or represent the relations between phenomena (for example, conceptual maps for learning purposes). Topographic maps feature some degree of distortion of area [60], and distortions reveal information about the places that would otherwise be difficult to observe [50]. From the two main categories we can create a new one that mixes characteristics from the first ones in differing degrees. Nowadays, most of commonly used maps mix topography with thematic approach to improve readability and include clues to tailor geographic features to special purposes (figure 2.20).

In the mixed category it is possible to include climate maps, economic maps, political maps, and transportation maps. Transportation maps, in turn, are a generic definition for maps which serve the purpose of communication transport network information, and encompass the route maps, metro maps, railway maps, transit maps, schematic maps. Figure 2.21 summarizes this categorization.

Maps can be also classified regarding its visual presentation. Diagrammatic maps are simplified maps, they are intended to form a general idea of the phenomenon or event shown in graphic form on the map and to emphasize its fundamental characteristics. Schematic maps increase simplification to a higher degree, using highly generalized lines that conform to a specific and finite orientation schema. Spider maps are a special type of schematic map that appeared recently and follows visual architecture resembling the layout of a spider (figure 2.22).

2.1.4 Transportation Maps

In this thesis we pay attention to transportation maps. They depict paths between location points and are nowadays one of the most common form of graphic communication. One famous map is the London Tube Map (figure 2.23) that has seen many developments and updates since the nineteenth century.

Map producers use a wide array of generalization techniques to improve the clarity of the map and to emphasize important information that needs to be communicated[61].

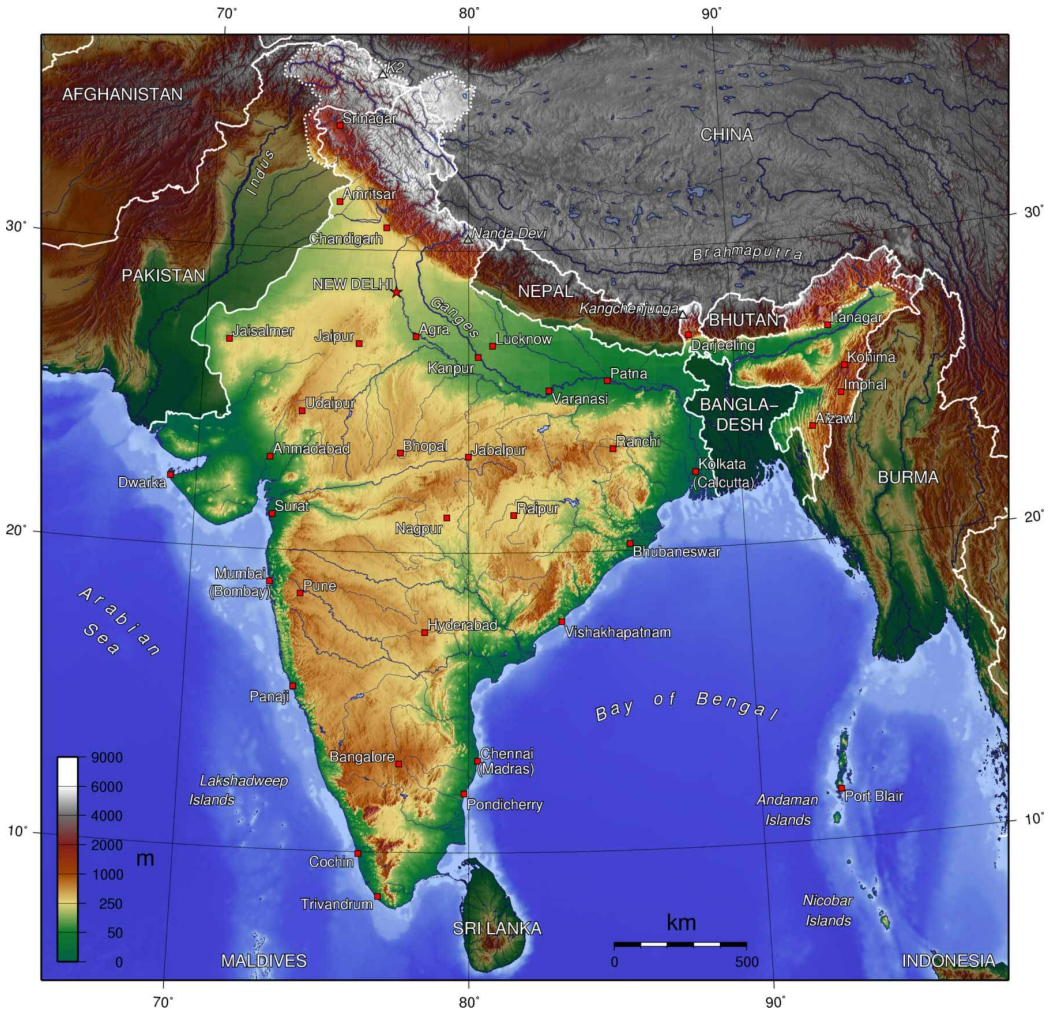

FIGURE 2.18: India Topographic Map [20]

Although creating a simple sketch depicting a simple route for personal use may be a simple task, when we talk about transportation maps, the underlying design is quite complex. Map makers use a variety of cartographic generalization techniques to improve the clarity of the map and to emphasize the most important information [62]. The most common generalization techniques are presented in the following list[23], and illustrated in figure 2.24.

- **Simplification** - Selectively reducing the number of points required to represent an object
- **Smoothing** - Reducing angularity of angles between lines
- **Aggregation** - Grouping Points locations and representing them as aerial objects
- **Amalgamation** - Grouping of individual areal features into a larger element

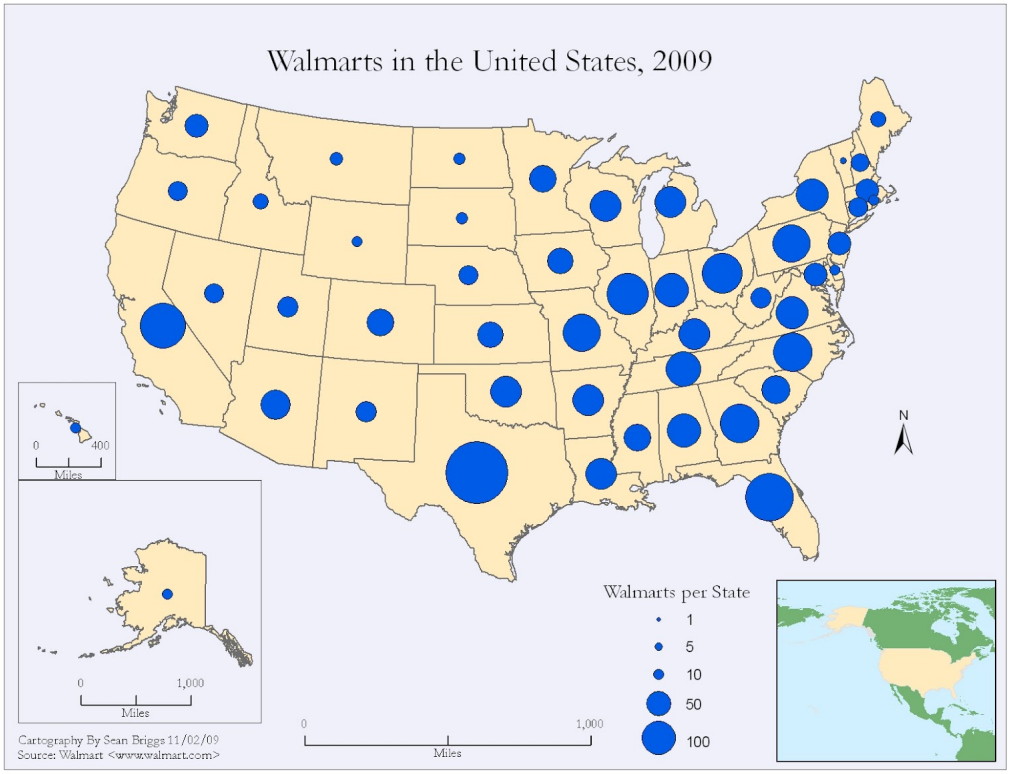

FIGURE 2.19: Thematic map, showing the number of Walmart stores per state in the United States in 2009 [21]

- **Collapse** - Replacing and objects physical details with a symbol representing the object
- **Merging** - Grouping of line features
- **Refinement** - Selecting specific portions of an object to represent the entire object
- **Exaggeration** - To amplify a specific portion of an object
- **Enhancement** - To elevate the message imparted by the object
- **Displacement** - Separating objects

Generalization of transportation maps causes them to be "diagrammatic" maps, by their inherent sparseness and imperfection, but those factors facilitate learning of the map by the user, by being more straightforward and not overwhelming user imagination with the completeness of the concrete visual world or with details shown in non-diagrammatic maps [60] (figure 2.25). Metro maps depict routes as as compromised spatial and logical transformation, where only relevant data is presented to users. Generalization techniques, therefore, are essential to simplify the complexity of the route system for presentation. Being so, metro maps often present qualitative spatial concepts adapted to

FIGURE 2.20: Mixed Cartographic-Thematic map, showing both the geography of France and the dates and places of the Tour de France of 2011 [22]

common characteristics of mental knowledge representation [63]. This means that in what concerns to transportation maps, it is more important for users to capture the basic structure of the network than to show accurately physical locations on the map. Cognitive psychology research shows that an effective route map must clearly communicate all the turning points on the route[64], and that precisely depicting the exact length, angle, and shape of each transportation line is much less important[65], although when traveling at earth's surface, the use of important clues is proved to improve user learning [66] [67].

2.2 Schematic Maps

The need of highly efficient, easily understandable (and thus, practical) transportation maps pushed the evolution of the traditional maps, and new forms of cartographic

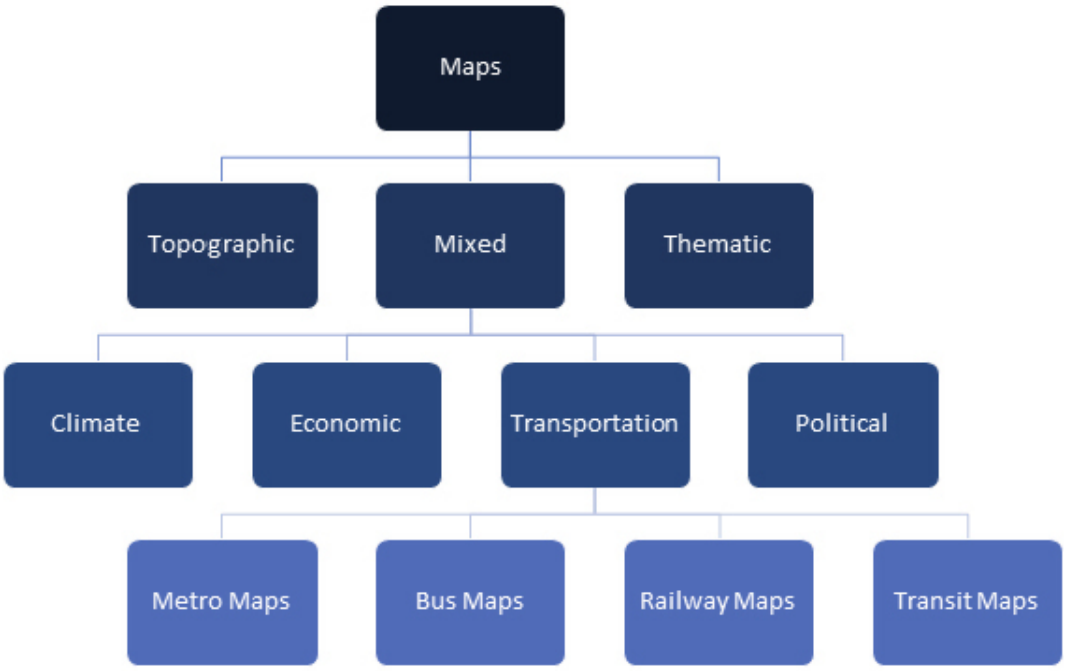

FIGURE 2.21: Map Taxonomy according to its function

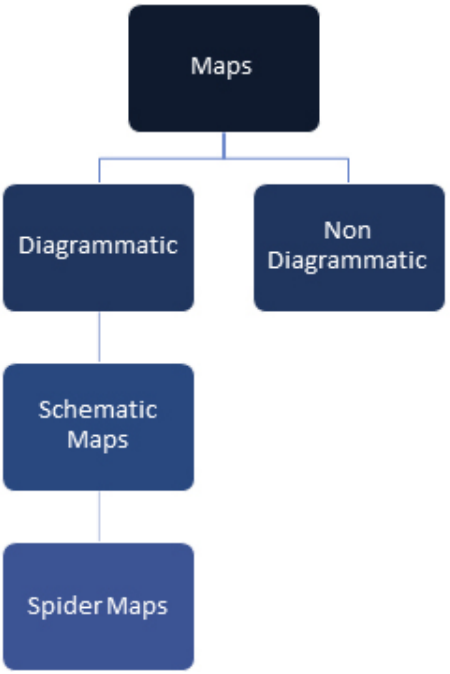

FIGURE 2.22: Map Taxonomy according to its visual presentation

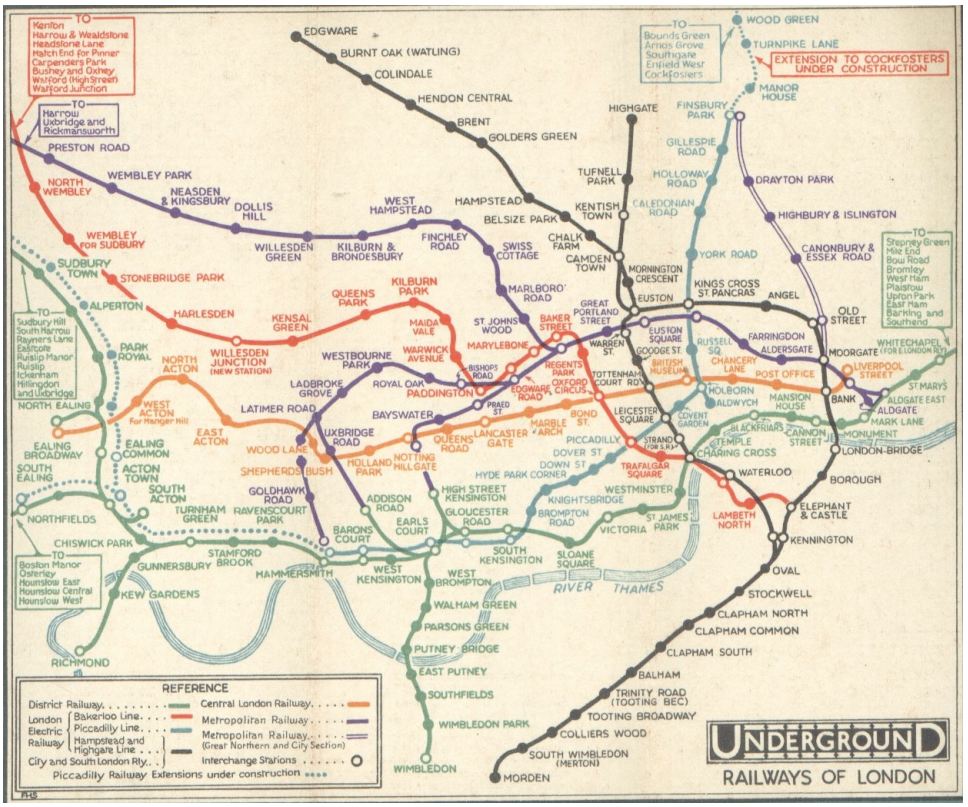

FIGURE 2.23: London Underground Map, from the beginning of the twentieth century. *Source: http://www.chadsayshello.com*

representation have emerged. Among the new forms of cartographic representation that have emerged are the schematic maps. Schematic maps were a new kind of map which appeared in response to the need of better and simpler transportation maps to describe complex public networks. [54]. Among the new forms of cartographic representation that have emerged, schematic maps were probably the most bold. One famous schematic map applied to a public transportation network was the Harry Beck's London Underground diagram, depicted in figure 2.26.

Despite being bold and including some new and controversial features, this map was considered an innovation, as for the first time lines were drawn either horizontally, vertically or diagonally at 45°. This map also uses a non-linear scale, so the central area of the diagram is shown at a larger scale than the extremities. It shall be noted that although it does not mimic the geography of London, this map gives the traveler some clues about the terrain features (ex: river) and his/her location. Avelar [68] defines schematic maps as *"an easy-to-follow diagrammatic representation based on highly generalized lines which is in general used for showing routes of transportation systems, such as subways, trams and buses, or for any scenario in which streams of objects at nodes in a network play a role"*. The most important advantage of schematic maps is that they

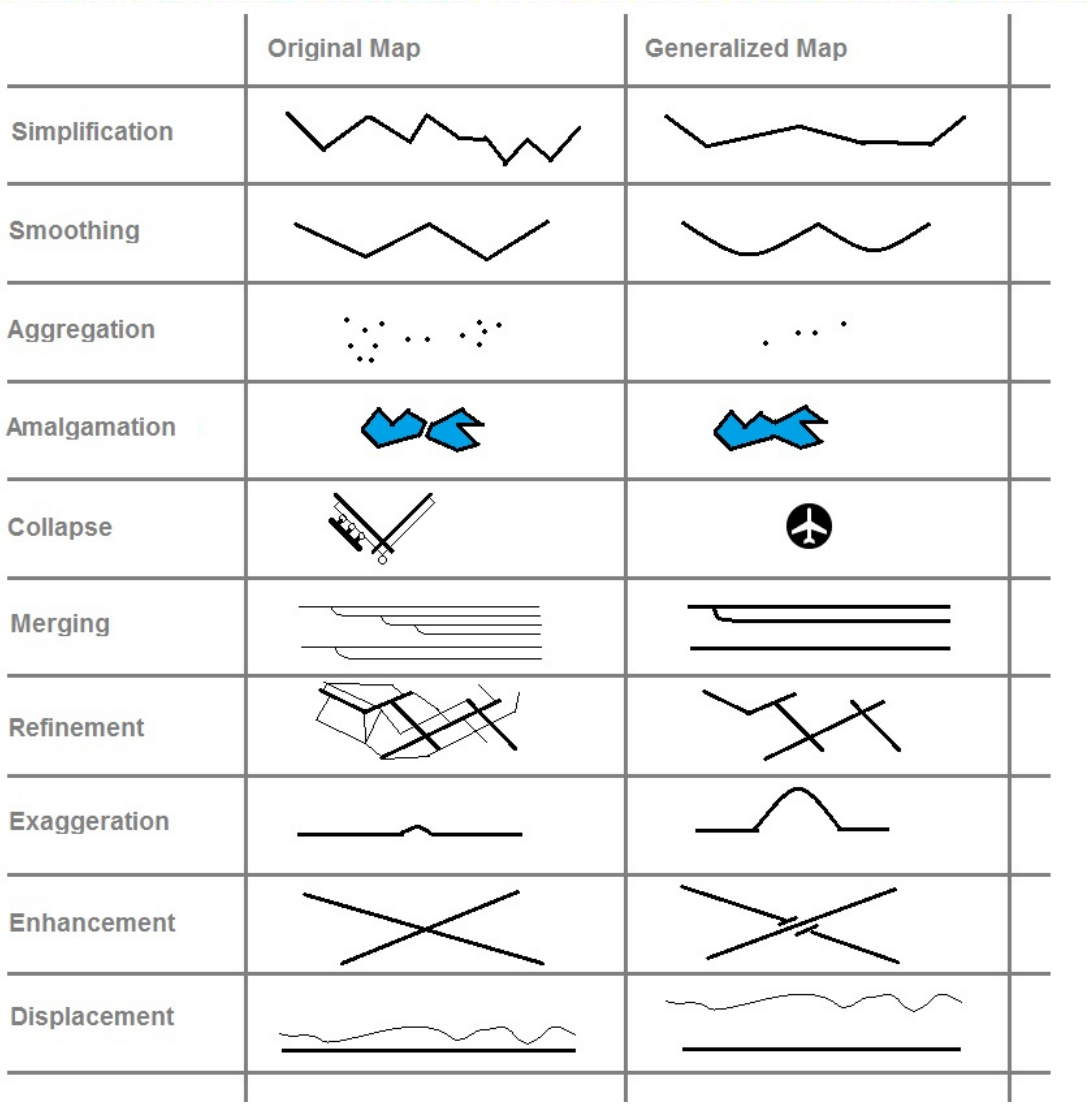

FIGURE 2.24: Most common generalization techniques [23] (adapted)

provide a quick view of the layout of the network by removing unimportant information like the detailed shape of the connections.

Schematic maps have been increasingly used in response to the need of better and simpler maps to describe complex transport networks. This apparent simplicity is achieved through a sequential process where choices are made regarding the level of detail and schematization choices. This process, called "schematization process", is (most of the times) still a manual process being carried away by teams of expert designers and cartographers, despite efforts to automate the process through the use of computers. This happens as there is the lack of thorough algorithms who can mimic designer and cartographer decisions to effectively generate schematic maps with enough quality to be presented to public. Through the schematization process, certain map details are emphasized while others are deemphasized. It is fundamental, however, to present the

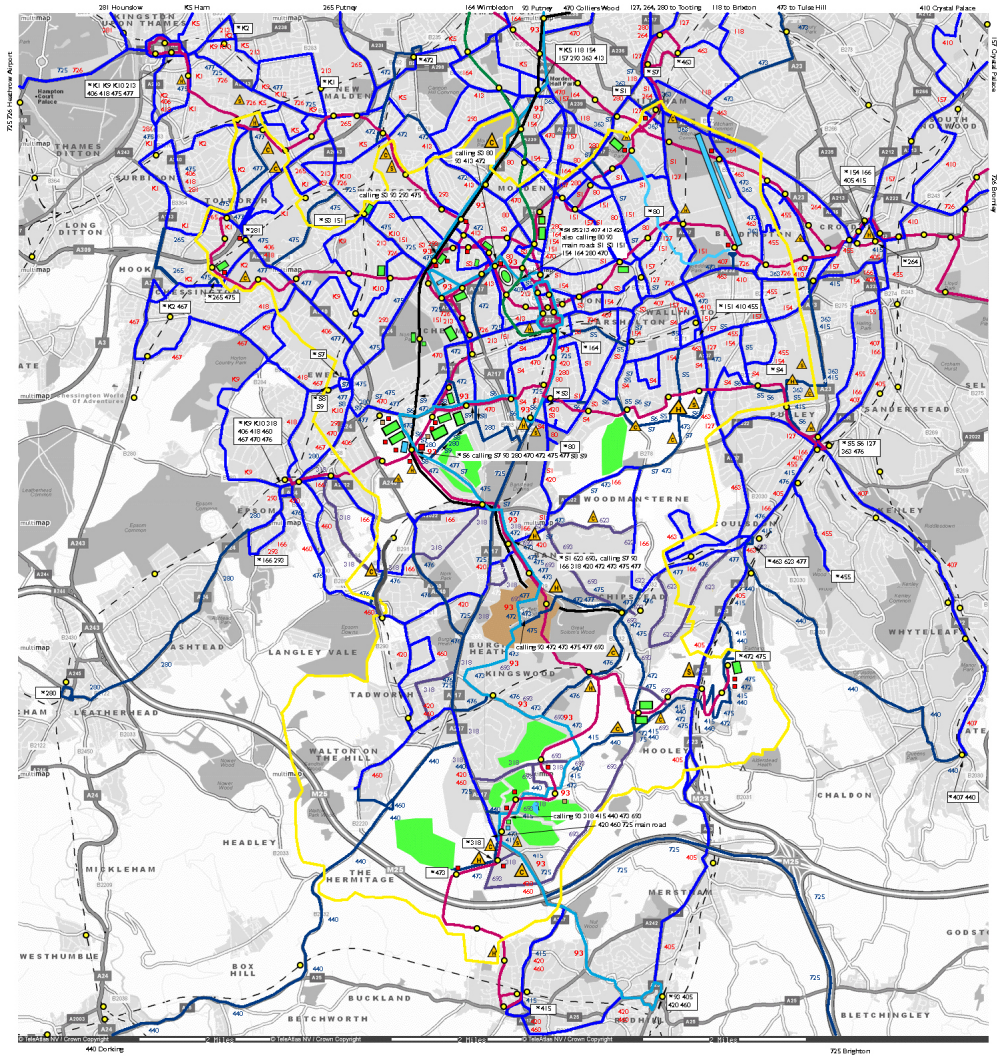

FIGURE 2.25: Route Map: Cheam Bus Route Map [24]

smallest information amount the user needs to learn the map: the more information presented, the higher the learning time will be. Latto defends that information shall be reduced to its basic components to achieve that goal [69]. Usually, the schematization process is a sequential process that starts with the input of a non-diagrammatic" map, involves some sort of pre-processing (like aligning stops to a pre-defined grid, for example). Then, an incremental optimization of the layout of the map (simplifying line geometry, moving stops, etc) is performed. Finally, at the end, the map may be subjected to a post-processing phase (where colors and shapes of visual elements are worked on, visual artifacts are placed or removed and other aesthetic work may be done). The output of this process is a schematic map (figure 2.27).

In fundamental terms, schematic maps [70] are advanced abstractions of the physical reality. They are seen as conceptual representations of the environment, and provide a suitable medium for representing meaningful entities and spatial relationships between

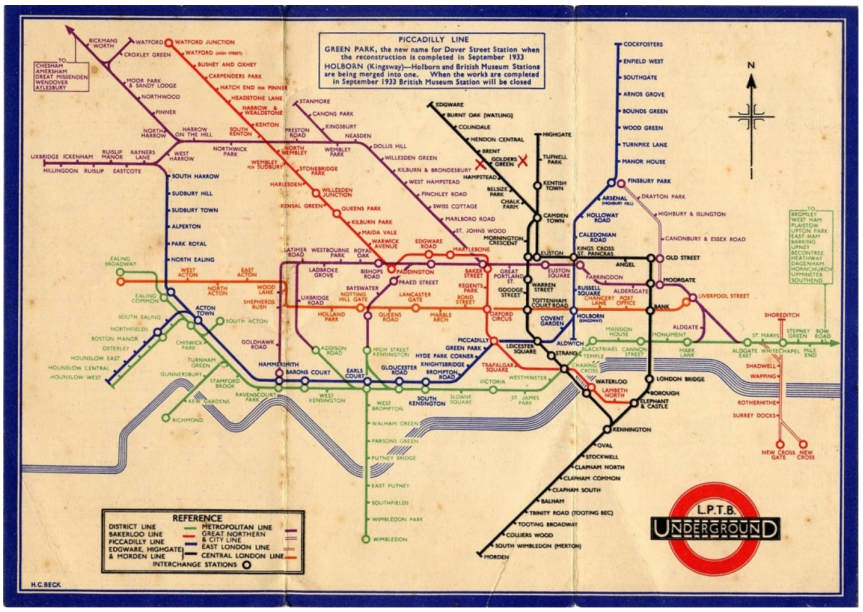

FIGURE 2.26: London Underground Diagram by Harry Beck [25]

FIGURE 2.27: Schematization Process Steps, from the initial map to the schematic map

entities of the represented world [71]. Moreover, they relate concrete and detailed spatial information from the physical environment to abstract and conceptual information stored in our brain. Being so, schematic maps are related to the transformation of mental maps of verbal directions into a physical map (however, an instrumental condition is that a suitable relationship must exist between schematic maps and the represented world). The more clearly we relate schematic maps to the critical elements of the represented environment, the higher usability quality the map has. This happens because humans represent spatial information like they perceive and conceive of the environment. As schematic maps mimic more closely the way humans understand the spatial reality, they improve tasks such as wayfinding in comparison with traditional maps [72].

Schematic maps are mixed maps which mix the traditional geographical structure with a set of thematic aspects of the user domain that are relevant to the intended use, while removing other features that may distract user attention and divert the attention from the main objective of the map. They relax some spatial constraints, while preserving the ones that help users in executing specific tasks. Schematic maps, such as public transportation schematic maps can be considered precompiled general wayfinding plans that

stop short of verbalization. For example, routes can be derived more easily than from more general maps, but certain spatial and other relations cannot be derived [73]. This is clearly intended in order to help users to quickly understand the relevant geographic and route information that allows them to easily make quick decisions in what concerns to their wayfinding tasks through the transportation network. Nevertheless, schematic map quality and adequacy is always highly subjective, as the mental representations of the world by individuals has a subjective dimension and is distorted by emotions, knowledge, capabilities and experience. Avelar [60] summarizes a set of characteristics the schematic maps have:

- **Geometric inaccuracy** - The shapes do not follow precise geographic accuracies. This means that length, orientation, resolution and fidelity of lines are not tied to geographic reality. For user learning purposes, those aspects are not desired as they may increase cognitive overload and may only be roughly preserved, while topological information of the line network needs to be preserved. We have a case here where function and use configure the visual appearance of the map.
- **Finite set of discrete line orientations** - Schematic maps line orientation follows a fixed orientation schema, commonly of 45 and 90 degrees (such as Beck's London Tube map) or of 30, 60 and 90 degrees, though the orientation interval may vary from map to map for some applications. [74].
- **Overlapping lines are in general separated by a minimum legibility distance** - This distance may be zero or other predefined positive constant
- **Lines are usually straight** - Bends tend to be as few as possible but they may be introduced were needed to improve map visualization, improve user reasoning by introducing important details that can more closely follow the geographic reality.
- **Scale variability** - Maps do not have a constant scale factor across the map. Differential scaling is commonly used to magnify denser map parts and to condense sparse map parts, in order to balance visual information density and allow user to be able to understand denser areas while keeping the map canvas at the same size. This is obviously translated into map distortion, which is totally desirable and intended as it allows better visual balance of the map. Of course, schematic map makers can use differential scaling to guide user attention and to *lie* to the user (intentionally or not), in what concerns to distances. As Wyckoff says, *"There's no escape from the cartographic paradox: to present a useful and truthful picture, an accurate map must tell white lies"* [75].

Avelar also includes a set of other aesthetic features, but they are not considered to be included in the commonly accepted definition of schematic map, such as the line coloring and shape scheme, line corner aesthetics and styles as they are not specific and exclusive characteristics of schematic maps.

2.3 Automated Generation of Schematic Maps

Nowadays, schematic maps are widely used in transportation networks, and they keep most of the design principles introduced by Harry Beck. Nevertheless, not much has been written about them. Some authors [60] defend that the production of schematic maps can be be manual, assisted and automatic. The manual design of schematic maps involves a spiral iterative labor-intensive process where sketches are produced entirely by hand in the search of the most pleasing solution, and incremental steps toward that direction are taken from sketch to sketch. The assisted method is the currently most used method since the advent of computers and computer assisted design software. In this method, the geographical coordinates transportation network points are digitized (through GPS or other assisted measurement tools), the geographical features are used as the background of the map, and re-placement of the visual elements such as lines and stops is done with manual input from the user. Although this method is faster than the pure manual production, it requires as much visual evaluation and iteration from the map producer. The automatic method is based on the automation of the whole process of the production of schematic maps, by implementing strategies to automate each of the phases of the schematization process (figure 2.27. Automating the process has been, and continues to be, the subject of much research [76][77][78], nevertheless those approaches have yet to support real world complexity and interaction. Nowadays, most of the maps are still made by teams of expert designers, with a mix of manual and assisted methods[79].

The roots for the automated generation of schematic maps can be traced to the development of computer graphic geometry. In 1973, Douglas and Peucker published a scientific article proposing an algorithm for the reduction of the points needed to represent a line[27]. This paper also predicted what would be one of the major components of schematization algorithms by stating *"line reduction will form a major part of automated generalization"*. Line generalization involves simplifying the line such that the overall form of the line is maintained as much as possible. The Douglas and Peucker heuristic is used as a basis for many schematization algorithms. This heuristic recursively subdivides the original line at the node which is furthest from a line between the two end-points until all nodes are within a certain error criterion (fig. 2.28).

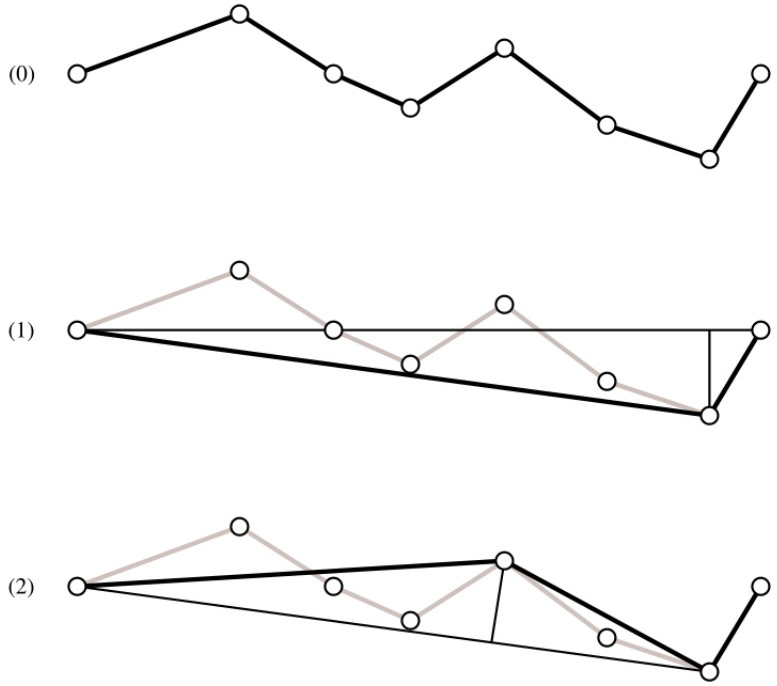

FIGURE 2.28: Example of the Douglas and Peucker Algorightm [26] [27]

Douglas and Peucker algorithm runs in O(n log n) time, although it may not find the optimal solution (keeping the overall form of the line). By this time, map schematization was focusing on relaxations of the network structure while keeping topological accuracy. Routes and junctions were symbolized abstractly [80]. Elroi [81] enhanced the process by proposing a three step algorithm:

1. Simplify lines to their most elementary shapes
2. Re-orient lines to conform to a predefined regular grid, to obtain a 0, 45 and 90 degree schema
3. Apply differential scaling to allow congested areas to be magnified while sparse areas are condensed

With this research work, Elroi laid the theoretical foundations of the schematization process, despite not providing any real world examples. He pioneered the idea of embedding the map layout on a regular grid[82]. At that time, Weibel and Brassel published a conceptual framework for automated map generalization [83]. In what concerns to the magnification of crowded areas through differential scaling, Sarkar and Brown [28] developed a method based on the idea of a very wide angle lens that magnifies nearby objects while shrinking distant objects, called *fisheye* as a valuable tool for seeing both

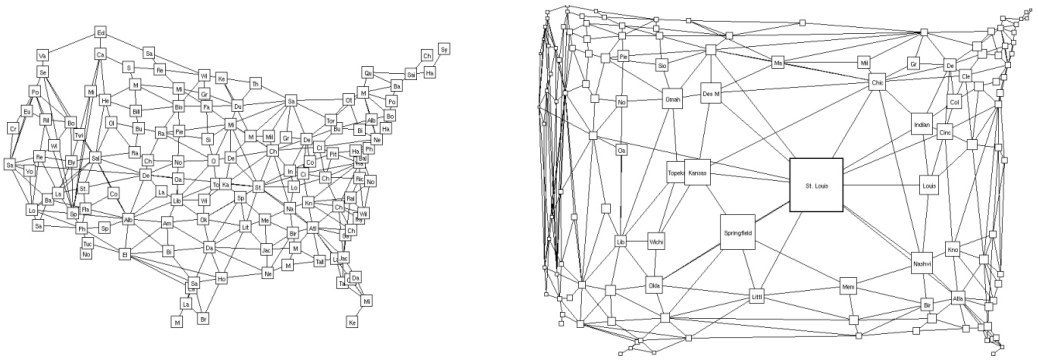

FIGURE 2.29: Fisheye Algorithm Example [28]

local detail and global context simultaneously. In fact, this may be considered a focus-and-context technique for graphs that serves the schematization purpose of emphasizing important map details while de-emphasizing unimportant details. This research systematized a formal model for differential scaling concepts only using one map view, thus combining the advantages of having different views (one for the focus, and one for the context) into the same view, while not having any of the disadvantages of having two different views. Basically, generating a fisheye view involves magnifying the vertices of greater interest and correspondingly demagnifying the vertices of lower interest. The position of all vertices and bend points must also be recomputed in order to allocate more space for the magnified area so that the entire map does not overflow the map frame or boundary. Figure 2.29 shows an example of this algorithm, on a graph that represents the major cities in the United States with the edges representing paths between neighboring cities. The fisheye algorithm was generated with focus on a particular city (St. Louis) It is possible to observe how this area is magnified, while the others are demagnified. Differential scaling is widely used in the design of visually unbalanced schematic maps which have crowded areas and uncrowded areas as it allows better visual balance of the map elements. Differential scaling can also be used to manage the emphasis of some map aspects through scale. It introduces geometric distortion on the map, and this distortion on the map produces visual changes that may be useful to enhance map readability (at the cost of precise geographic accuracies).

The topic of geometric distortion of schematic network maps was later revisited by Jenny [29], in an analysis made to the London tube map through a software called MapAnalyst. He analyzed the differential distortion that occurred in the London tube map in comparison with the real geography of London (figure 2.30). He developed the idea of using distortion grids or displacement vectors during the design process, as seen in the figure 2.31. The displacement vectors could be used to systematize the adequate scaling of the map, and to allow further interaction possibilities for map makers in assisted and automatic schematic map production.

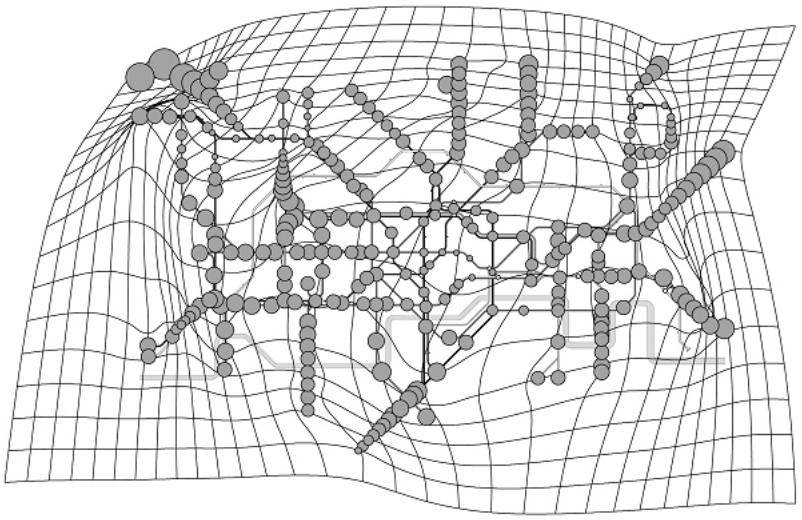

FIGURE 2.30: The current London Underground diagram with an overlaid distortion grid and displacement circles. The circles' areas are proportional to the distances to the correct locations [29].

FIGURE 2.31: Displacement vectors. Arrows point at the correct geographic location of each station [29]

Line simplification was again researched by Latecki and Lakämper[30] [84] through the vertex deletion on polygons to simplify their shapes. Through discrete curve evolution strategies, their algorithm achieves the following goals:

- It leads to a simplification of shape complexity
- It does not introduce any blurring (i.e. shape rounding)
- There is no dislocation of relevant features
- It is stable with respect to noisy deformations
- It allows to find line segments in noisy images

This algorithm keeps the same planar position of a set of predefined points. Figure 2.32, shows how this algorithm reduces a graph to its elementary shape by keeping the position of the squared/circular points (predefined points).

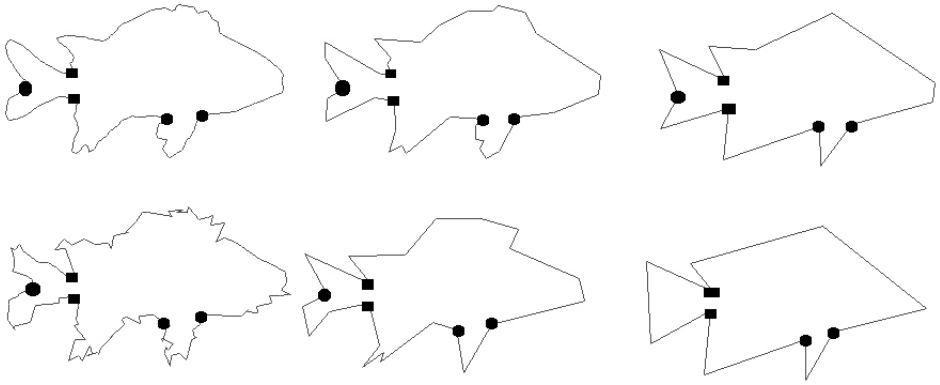

FIGURE 2.32: Example of the Discrete Curve Evolution Technique of Latecki and Lakämper[30]

The Discrete Curve Evolution approach was later improved by Barkowsky [63] having this research as a basis. Nevertheless this approach does not take in consideration the distance between stations, nor restricts possible edge directions. In addition, it does not provide a mean to magnify crowded areas to improve readability nor considers station labeling occlusion possibility. Therefore this algorithm is more suitable for simplifying road maps than for schematizing public transport maps.

Neyer [85] also studied a line simplification problem where each simplified line segment must be parallel to one of a set of given orientations. This algorithm forces the segments of a polygonal chain P to be in compliance with a finite set of orientations O. The objective is to find another polygonal chain which is a C-oriented approximation of P,

i.e. the distance between P and Q in the Fréchet metric[1] [86] needs to be within a certain threshold value. Neyer gives a dynamic programming algorithm to compute C-oriented line simplifications in $O(kn^2logn)$ time, where N is the number of vertices on P, k the number of segments of Q and the number of orientations $|C|$ is constant. In what concerns to practical schematization of transportation maps, C would contain the usual orientations (0, 45 or 90 degree). The problem with this algorithm is that only considers one path in the transportation map at a time, so there is no interaction between the lines within the whole graph. This may introduce undesired features (segment intersection, change in network topology, vertex disconnection, etc) [87]. Generalization techniques were improved by Agrawala and Stolte [61] merging some knowledge from cognitive psychology. They consider that an effective route map must clearly communicate all the turning points on the route and that precisely depicting the exact length, angle, and shape of each road is much less important. While those considerations also hold true for generic schematic maps, their generalization methods do not take in account several specific schematic map problems such as unbalanced maps with non uniform density and they have only applied those generalization methods to one route (line) at once, so they do not take in account generalization of complex transportation maps and interaction between lines. Avelar and Müller [88] developed an algorithm to enhance the aesthetic criteria of a transportation network by moving the vertices of the line segments while preserving topological relations. The method of preservation of map topology was made through three conditions:

- No absence of line crossings that were present in the input map
- No line crossings that were not present in the input map
- Cyclic order of outgoing connections around any node agrees with the ordering of connections in the input map

Although this algorithm was successfully applied to the road map of Zurich, not all line segments could be drawn octilinearly because vertex positions are influenced by several potentially conflicting terms. Other considerations such as labeling were not included in this research.

Further research carried by Avelar [60] identified and summarized two key factors for producing what she calls *good* schematic maps: a set of Aesthetic Criteria, a conceptual model for a database containing geographical and topological information about the map, capable to address user queries. She proposes an iterative schematization algorithm which handles the conflicting objectives by establishing a compromise between

[1] The Fréchet metric is a measure of similarity between curves that takes into account the location and ordering of the points along the curves.

them at each iteration, and stated a set of constraints related to length, angle, and shape of edges to evaluate the quality of solution. This research, however, falls short of integrating cognitive psychology and computer science research knowledge into the algorithm, centering much of the research into the so called "aesthetic factors". Also, it uses a constant scale factor which may force shorter lines to shrink to a point. Her algorithm also relies heavily on user parametrization. Data about computational efficiency are also not given.

Another algorithm for schematizing road maps is presented by Cabello and Kreveld [89]. The algorithm draws the edges of the input network as octilinear paths with two or three links while preserving the input embedding. The user can restrict the links to 0, 45 and 90 degrees and vertex positions keep their original position. The algorithm runs in $O($ $n\ log\ n)$ time as long as input edges are drawn monotonously. It also determines if a schematized map can be produced. However if it cannot, the algorithm fails and no map is generated, which is a disadvantage. Other drawback is that vertices (stops) keep their original positions[89]. In real-world transportation maps, moving stops is crucial, as not doing it leads to many unnecessary bends in the layout and crowded downtown areas remain confusing. Although efficient, this algorithm does not generate effective schematic maps. This research work was later completed [90] with a combinatorial approach to the problem of aligning points. Given a set of points, the task is to align as many points as possible horizontally, vertically or diagonally, where each point can be placed somewhere in its own, given region (for example a circle, rectangle or Voronoi cell around each point). The points represent the vertices of an underlying graph and alignments are only considered for adjacent vertices. The goal is to place vertices such that as many edges as possible are drawn as straight, octilinear line segments while roughly preserving their positions. After modifying the vertex positions, the remaining non-octilinear edges could for example be simplified with the previous method of Cabello and Kreveld [89]. Although being an improvement, this method still has the problem that after being placed, the vertices remain at the same position. This is a big limitation as as the algorithm progresses, much improved results could be obtained if there was the possibility of displace the vertices to avoid local minima.

Nöllenburg [87] formulated the problem as a Mixed Integer Programming (MIP) problem. Given a planar graph G of maximum degree 8 with its embedding and vertex locations and a set L of paths or cycles in G (e.g. transportation network lines) such that each edge of G belongs to at least one element of L, draw G and L nicely. He first defines the concept of "niceness" a map by listing a number of hard and soft constraints. This method optimizes a weighted sum of costs corresponding to the soft constraints, while enforcing the compliance with the hard constraints. This approached was tested with real world city maps, and the results were compared with the actual maps used in

those cities. This algorithm proved that high quality solutions can be found, but the time to find them is a serious concern, although good intermediate solutions could be found easily. Introducing labeling constraints produced a big MIP search space making the processing time a big concern. In addition to this, this algorithm may fail to produce a solution if there is none. These factors prove that this algorithm is not suitable for soft real time and/or dynamic environments where response time and finding a solution are fundamental (such as location based services). The MIP approach was latter revisited [91] [92], with this latter research improving some details. Nevertheless, the maps tested were generated within a 10-12 hours timeframe, which is still an enormous processing time. The authors confirm that their method is unable to produce good labeled maps instantaneously.

Other approaches to the problem of generating schematic maps involved the use of multicriteria optimization techniques. Stott et al [93] used a hill climbing algorithm to improve the starting layout of a map. The algorithm starts, therefore, with an initial layout placed on a regular grid. At each iteration, the algorithm measures the map by calculating a number of criteria. These criteria are weighted and summed together: stations are moved if the sum of the weighted criteria is reduced (thus minimizing the objective function value). An iteration of the method consists of attempting to move each station in the map. Some advantages of this method in regarding to MIP were the faster processing time and the fact that a solution is always obtained, even if is very suboptimal. Despite this advantages, the initial version of the algorithm had some problems such as typical local minimum management problems (for example, overlong edges would not be moved as this would need moving several vertices at the same time), the relative position of stations was not kept and occlusion problems could occur with labeling. Therefore new refined versions of this algorithm were developed [26] [94]. These refined versions solved the disadvantages by being capable of moving sets of stations at once if it advantageous. The set of criteria and labeling was also improved. Despite these improvements, this algorithm had some downfalls: parametrization is not automated, so the user has to set up the criteria weightings, computational efficiency was not the focus of the algorithm, and some of the criteria use quadratic complexity which can cause problems. The placement on an initial grid is too simple, which can avoid denser area stops to be successfully placed as no contention management strategies were employed, and this can jeopardize all the algorithm. Uniform grid scaling is also employed, and this factor can cause lower quality map production. Octilinearity is also not guaranteed.

Generalization by vertex displacement has been investigated using a number of meta-heuristic techniques, including simulated annealing [95], genetic algorithms [96] and tabu search [97]. The application of of memetic algorithms to automated schematization was researched by Swan et al [98]. This algorithm was also based on a set of seven constraints:

- **Topological:** the original network and derived schematic map should be topologically consistent;
- **Orientation:** if possible, network edges should lie in a horizontal, vertical or diagonal direction;
- **Length:** if possible, all network edges should have length greater than or equal to some minimum length (to ensure clarity);
- **Clearance:** if possible, the distance between disjoint features should be greater than or equal to some minimum distance (to ensure clarity);
- **Angle:** if possible, the angle between a pair of connected edges should be greater than or equal to some minimum angle (to ensure clarity).
- **Rotation:** an edge's orientation should remain as close to its starting orientation as possible
- **Displacement:** vertices should remain as close to their starting positions as possible

A prototype algorithm was built and a mockup result was presented, but no details were given in what concerns to performance and no analysis on the result quality was made, so little is known about this algorithm, its results and implementation. This set of constraints, however, was the base to other research works on the automated generation of schematic maps. A simple simulated annealing algorithm for producing schematic maps for mobile applications was implemented [78] but it had the typical limitations: the annealing schedule had to be set manually and finding an appropriate set of parameter values can be time-consuming. No tests were performed with real data and few data was provided regarding the ability of the algorithm to escape local minima. Dong [99] also designed a schematization algorithm based on those constraints, but he admitted that in a schematic map design of a complicated transport network, more map design aspects should be taken into account, such as overlap of a number of road segments, and false and true intersection points in the road network, such as having an intersection but not having any stations. The algorithm has not been tested with real world data and complexity, no details were given on performance, nor an analysis was made on the results other than the visual presentation of the map.

A recent research work was performed in order to automatically produce destination maps. Although destination maps may not be schematic maps in their traditional definition, some aspects are also important on what concerns to the design of schematic maps. This research [100] merged some cognitive psychology knowledge into aspects

such as hierarchical navigation (emphasize the more important transportation routes and remove the non-important ones to avoid information overloading, for example). Although the objectives of this research were not to produce traditional schematic maps in their strict definition, but more enhanced "navigation sketches", easy to learn and use by people when traveling across a transportation network, this work proved that cognitive psychology considerations play a fundamental role if we want to automatically generate effective schematic (or other type of) maps for navigation purposes.

Figures 2.33 and 2.33 provide a summary of these approaches through time.

2.4 Context Awareness and Cognitive Psychology

2.4.1 Map Creation as a Communication Process

In a communication process exercise, beyond the essential speech terms (sender, receiver and message), there are other elements that are necessarily present. These elements, called linguistic elements (as they depend on the language used in the speech) are the syntax, semantics and pragmatics [101]. Syntax is synonymous with "grammar" and it is related to the orthography and phonology, while semantics is the study of the meanings of linguistic expressions (as opposed to their sound, spelling, etc.) [102]. Semantics has four inner concepts [103]:

- **Reference** or **extension**: the object or set of objects to which an expression applies
- **Truth** and **falsity**
- **Intension**: what determines the extension of an expression; often regarded as a function from possible worlds to extensions)
- **What a competent user of an expression must know** (although this is a very important concept, there is no term that unambiguously expresses it)

Pragmatics has to do with context-dependent features of language and also includes things people can do with words or sentences that go beyond the literal meaning of the expressions involved. These three linguistic elements have a powerful influence over the message configuration and also in the particular understanding of the reality by the sender of the message. More important, the same happens for the understanding of the message by the receiver.

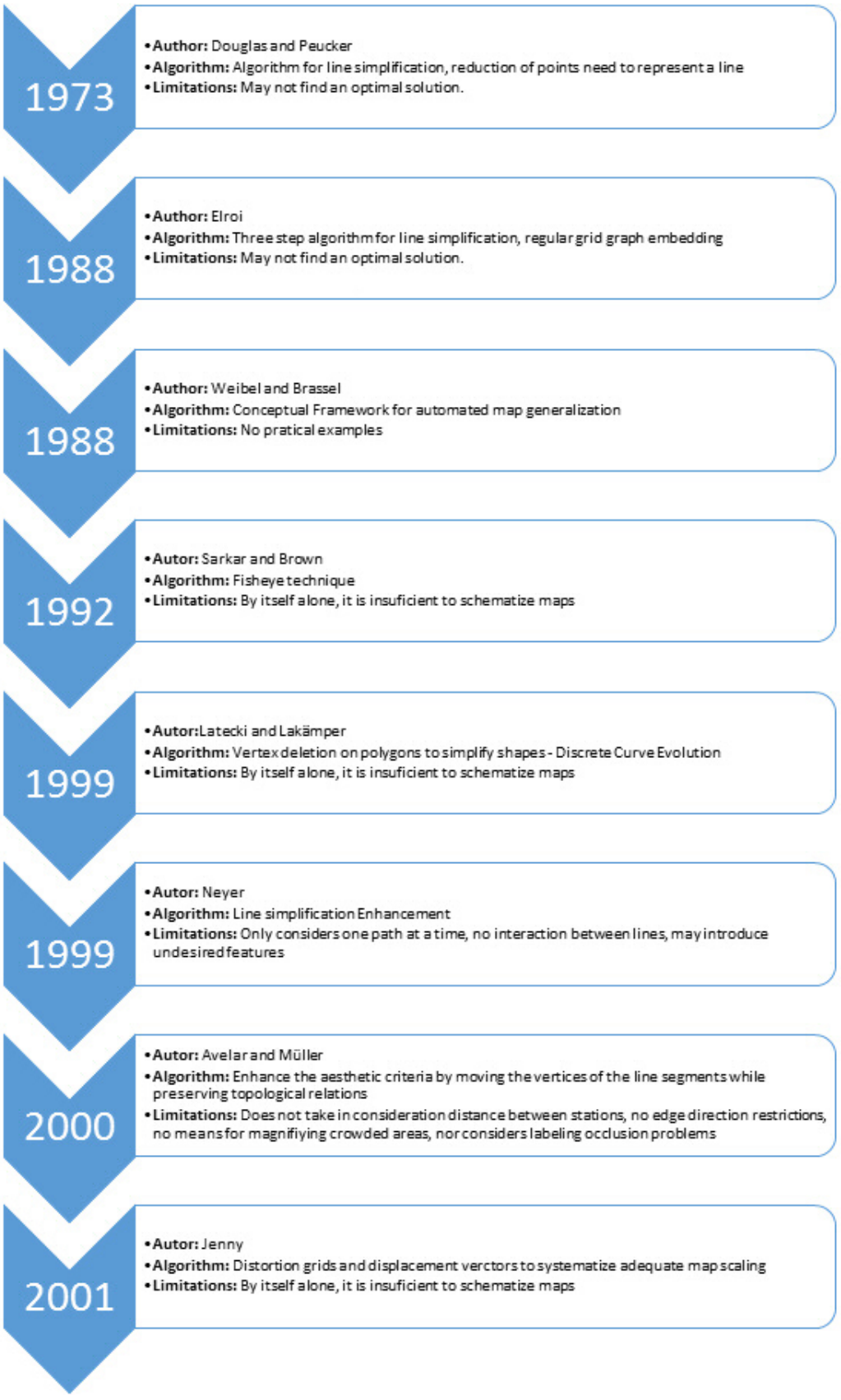

FIGURE 2.33: Summary of the automated schematization process approaches (1973-2001)

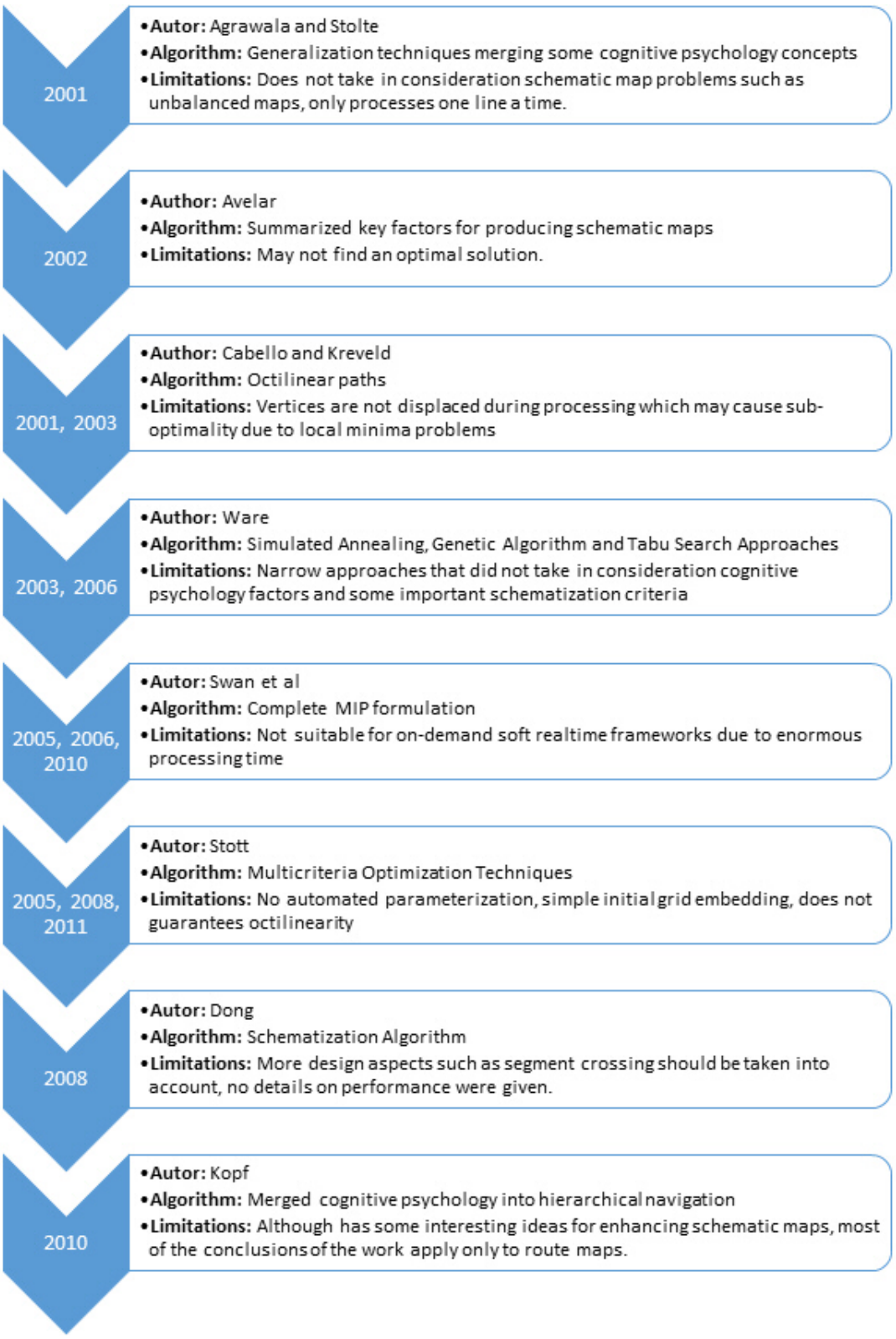

FIGURE 2.34: Summary of the automated schematization process approaches (2001-2014)

In the particular case of the schematic maps, we have also a communication exercise, using a set of organized symbols (lines, labels, colors, dots) for the formation of the message, which is the map. Presently, the creation (either manual, assisted and automatic) of schematic maps is mainly focused in the first two linguistic elements: the syntax and the semantics. Current literature focuses the type of symbols and abstractions to use, the relations between them (ex. line characteristics, line cross orientations, etc) and their meaning (ex: the use of different colors to change the meanings of the standard language). And although these studies are necessary and represent improvements over the traditional map generation, they lack focus on pragmatics (context). The language domain (syntax and semantics) is necessary but not enough to the understanding of the message sent by the sender. Therefore it is essential to know the enunciation context to improve map learning by the users. The participants in the communication influence the meaning of the message. This influence is related to the experience each participant has and to the way the expressions are used. The understanding of the meaning of a message heavily depends on the circumstances of its use. Being so, context is of fundamental importance if we talk about space communication and about map creation. To assure map information is clearly and quickly understood by its users, it is crucial to explore and optimize context.

Dey defines context as [104] *"any information that can be used to characterize the situation of an entity, where entity means a person, place, or object, which is relevant to the interaction between a user and an application, including the user(...)"*. In what concerns to maps, some authors enumerate several types of distinct contexts [105][106], but their classifications match in three of them: user context, space context and time context. Other types of context, such as physical surroundings, can be included as a subtype of one of these three.

User context refers to the physical, cognitive, or perceptual abilities, with personality differences, habits, etc. Space context may include orientation, location, physical surroundings. Time context is related the temporal conditionants. All these three main types of context are present in space communication in map use, nevertheless current maps do not pay much attention to context. Although in many knowledge areas such as Human Computer Interaction context has been extensively studied [105], the converse is true for map science, (with some exceptions regarding map use in mobile devices or in location-based services [59] [54]).

2.4.2 Concept Maps as Precursors of Schematic Maps

Human communication of any type of knowledge is always preceded by mental representation [107]. Therefore, it is fundamental to understand how humans mentally represent knowledge and establish relations between its subparts. Most cognitive theories share the assumption that concept interrelatedness is an essential property of knowledge [108], and that a well structured knowledge structure (with adequate relations) defines competence in a domain. Concept maps (also called *mind maps*), which are visual presentations of human mental knowledge, present some remarkable similarities with schematic maps. According to Dochy [109] a concept map is a structural representation consisting of nodes and labeled lines. The nodes correspond to important terms (concepts) in the domain. The lines denote a relation between a pair of concepts (nodes), and the label on the line tells how the two concepts are related. The combination of two nodes and a labeled line is called a proposition. A proposition is the basic unit of meaning in a concept map and the smallest unit that can be used to judge the validity of the relation (line) drawn between two concepts. These characteristics are also present in schematic maps, in the way that they are abstractions from reality where stops and connections between transportation lines are very important. On the other hand, from a cognitive point of view, investigating the structure of maps can provide a promising access to the way humans perceive, represent, and interact with their spatial environment. The existent *de facto* standard of mapping techniques has gradually evolved over a long time in a purposeful way; thus, the way people construct and interact with geographic maps has to be regarded as a valuable clue to the properties of the underlying mental structures and processes for spatial cognition [73]. There has been much discussion on how concept maps may improve learning [110]. Some authors have found that concept maps can improve learning, as they are graphic organizers, visual representations of knowledge, concepts or ideas. They are intended to enhance learning and selective reading [111], by mimicking the way human brain maps information (fig. 2.35). They have become one valuable tool to assist learning at schools, and include a growing set of diagrams which are helpful to schematize and summarize ideas and knowledge. Defined by ISO/IEC 13250:2003, concept/topic maps present several advantages in what concerns to learning [112]:

- Efficient context-based retrieval
- Acquiring new topical knowledge: Learners can gradually build their knowledge through natural and intuitive topic-aware content browsing.
- Deeper understanding of the domain conceptual relations

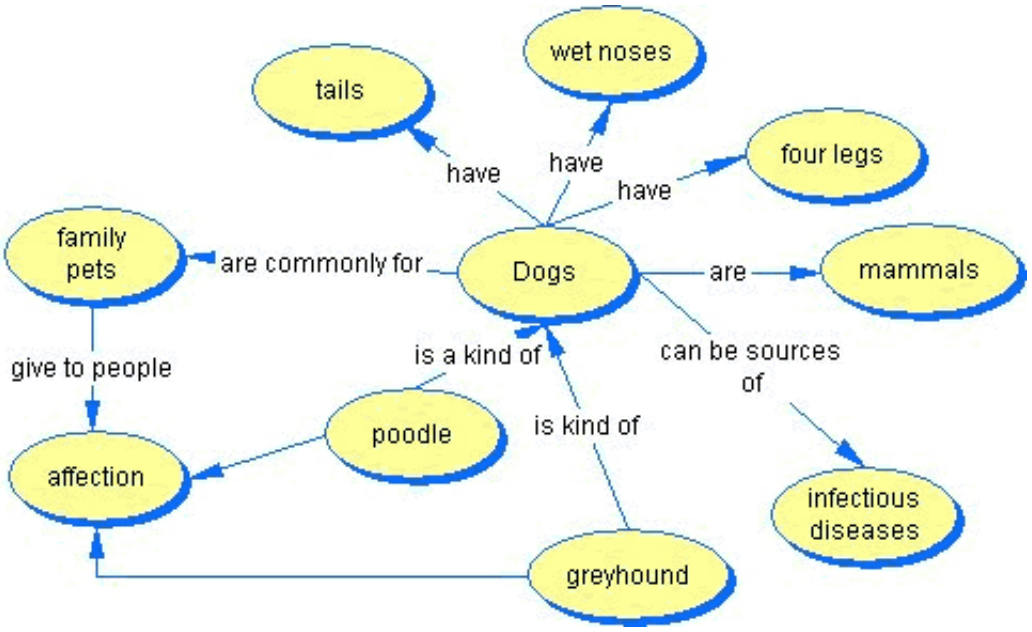

FIGURE 2.35: Excample of a concept map [31].

- Information visualization: Both the visualization and domain navigation facilities help learners to get quickly and easily orientated within the subject domain and build up their own understanding and conceptual associations. It supports their visual thinking and imagination and helps them create their own problem-solving paths.
- Customized views, adaptive guidance and context-based feedback. Topic maps-based applications can reason over learner's performance thus allowing for customized views on the same set of resources, adaptive guidance and context-based feedback depending on the current learner's task and goals.

These advantages seem to be consistent with other types of mind maps such as the conceptual spider maps, as they are used in K-12 (primary and secondary education) schools to improve learning [32]. A conceptual Spider Map (although there is no formal definition in classic literature about it) is used to describe a central idea: a thing, a process, a concept, a proposition. For example, a spider map may be used to organize ideas or brainstorm ideas for a writing project.

The focus of a spider map is the central area, representing the first and most relevant idea. From the central idea, the main ideas are attached, and subsequently the details are attached to the main ideas. Being so, the usual questions the spider map tries to answer to are the following:

- What is the central idea?

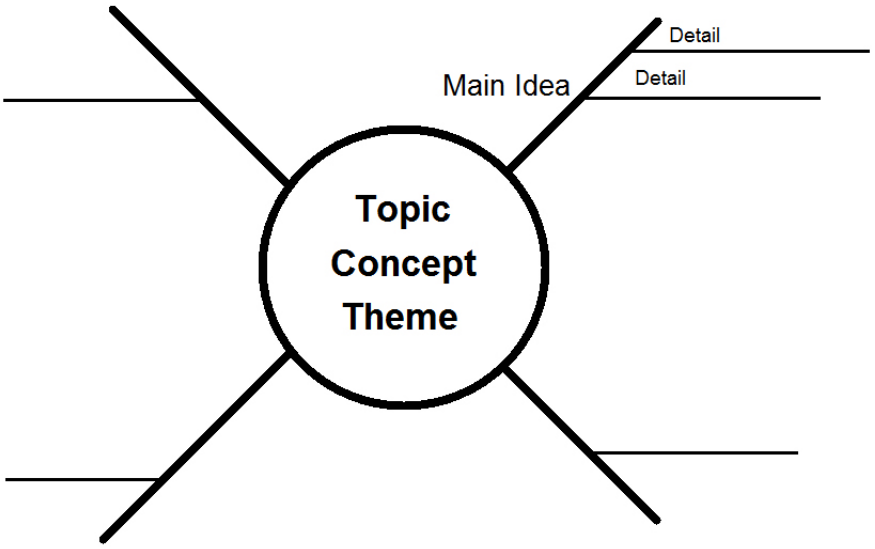

FIGURE 2.36: Skeleton of a spider map graphical organizer. [32]

- What are its attributes?
- What are its functions?

If we look at schematic maps as visual representations aimed at enhance learning (compared with traditional maps), we can notice that there are functional similarities between them and conceptual spider maps. In this work we intend to explore the features of conceptual spider maps to the design of transportation schematic maps.

2.4.3 Context Enhancement Techniques

When a map is used, it serves a specific purpose. Whether it may be wayfinding, locating, or other specific task, the better the map is suited to its purpose, the easier and faster will be for the user to accomplish that task. For this reason, the question that arise is *"Which features, relations, structures, and other properties support the specific process and which impede the process?"*[72]. Determining a cognitively adequate compromise between a "faithful" and a schematic map representation for a given class of tasks presents an interesting and difficult challenge that is related to the area of diagrammatic reasoning in artificial intelligence. Such a compromise reflects a trade-off between spatially presented information and conceptually presented information. In fact, there may be no compromise which will be agreed upon as "best compromise" by everybody, as there are individual preferences for dealing either with spatially represented information or with conceptually/propositionally represented information. This leads to the conclusion that

the context (either user, spatial or time context) is the clue for an adequate balance of map features, design and layout.

In what concerns to user context, two main enhancement strategies found in the literature: "user-centered design" and "user-adapted interaction".

Porathe conducted an experiment where a user-centered map (a map viewed in the user's perspective) was compared to other type of maps (paper map, north-up map and head-up) ([57]. The results suggested that for route guidance, user-centered maps are more efficient (faster decision making), less erroneous, and more user-friendly than maps displayed in the traditional exocentric perspective. However, one limitation of the study was that the map display types were not tested for route planning or judging distance.

Used-adapted interaction is related to the discovery of the user characteristics and preferences, either automatically or manually through user input. Automatic discovery of user characteristics may be obtained through stereotypical assumptions through user behavior, sensors, or other type of pervasive device that can capture user characteristics [113], serving as a base for user profile inference through meta-operators[114]. User characteristics may then be used to design specific maps tailored to each specific type of user (for example: maps with specific color schemes for colorblind people or school maps for children) which can speed up their spatial learning. User context, by adapting the map to the people (instead of being the other way around) can speed up their learning and improve user experience.

Designers and artists also have always tried to discover new forms of improving user experience by directing user attention to the relevant spots in their artworks and consequently managing context accordingly to their intentions. Studies on how to direct user's attention to specific points on paintings have been conducted [115], and they showed that the following techniques can be used to direct user's attention:

- Ink use: the quantity of ink can be used to draw user's attention
- Blur: the focus of attention shall be always sharp and the picture shall be blurred gradually as we move far from that focus.
- Using non-realistic rendering (as it removes detail in unimportant detailed areas)
- Selectively reducing detail according to an importance scale
- Using luminance bright/dark contrasts to focus user attention
- Reducing color detail (using less bits to map color)
- Reducing edge number (similar to a sketching process)

- Differential space scaling

Those generic techniques have been used for other purposes, including map creation through the study of the aesthetic factors [60]. Another relevant issue is the visual pleasure a graphical representation causes. If a schematic map is not pleasant to the eyes, it is highly probable it will be ignored or rejected by the user. To minimize this probability, the factors that influence a good representation have to be investigated. A study on this topic has been conducted on a set of famous paintings [69]. The results show that users like the paintings that are based in vertical and horizontal contours, instead of the paintings based in oblique lines. The author of the study calls this the "aesthetic oblique effect", and formulates a possible explanation for it: we like looking at what we are good at seeing. This may well be the reason why the 0, 45 and 90 degree line orientation schema has been used and commonly accepted by the public in schematic maps to depict transportation networks.

There are other context enhancement approaches that cognitive psychology and human-computer interaction show to be effective in improving map reading. Focus+context techniques are fundamental hybrid user/space context enhancement techniques that provide a visual layout that combines a focus area, where the data of most interest is displayed at full size or with full details, and a context area, which is a peripheral zone where elements are displayed at a reduced size or simplified. This way, the most relevant information is presented to the user while not overloading him with all the information the map has. There is a variety of widely used focus+context techniques and we mention here only a few ones:

- **Fisheye** - This technique assigns more display space to a a portion of the map (focus), magnifying and distorting it, while the rest of the map is presented with an increasingly diminishing scale as the map components depart from the focus [116] (figure 2.37).
- **Hyperbolic Geometry** - Similar to the fisheye technique, the essence of this scheme is to lay out the map components in a uniform way on a hyperbolic plane and map this plane onto a circular display region [34] (figure 2.38).
- **Spiral Representation and Augmented Context** - This technique uses a spiral layout divided into sectors to display items, and the higher DOI (degree of interest) objects may be magnified [117] as shown of figure 2.39.
- **F+C technique for Metro Map** - A special technique from Wang *et al*[35] that can be used for real time interaction with schematic metro maps: the best route

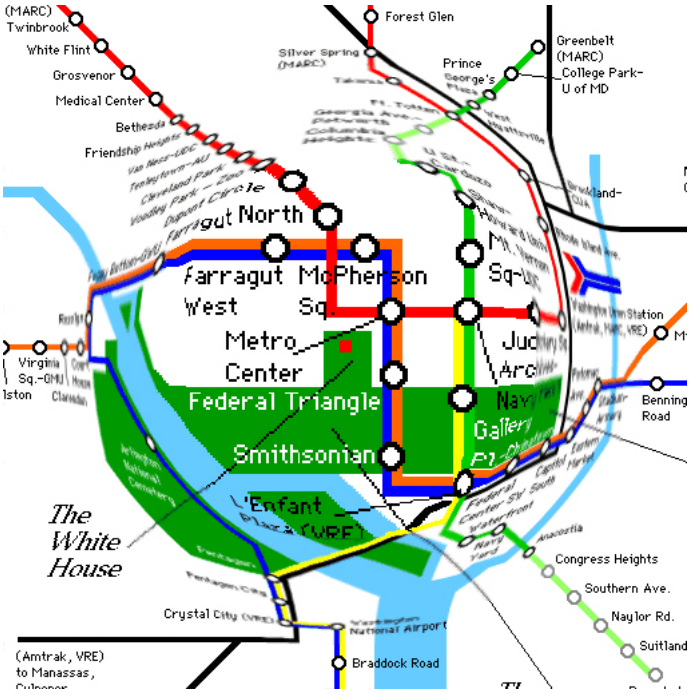

FIGURE 2.37: Fisheye view applied to central Washington D.C.. The Focus is the White House [33].

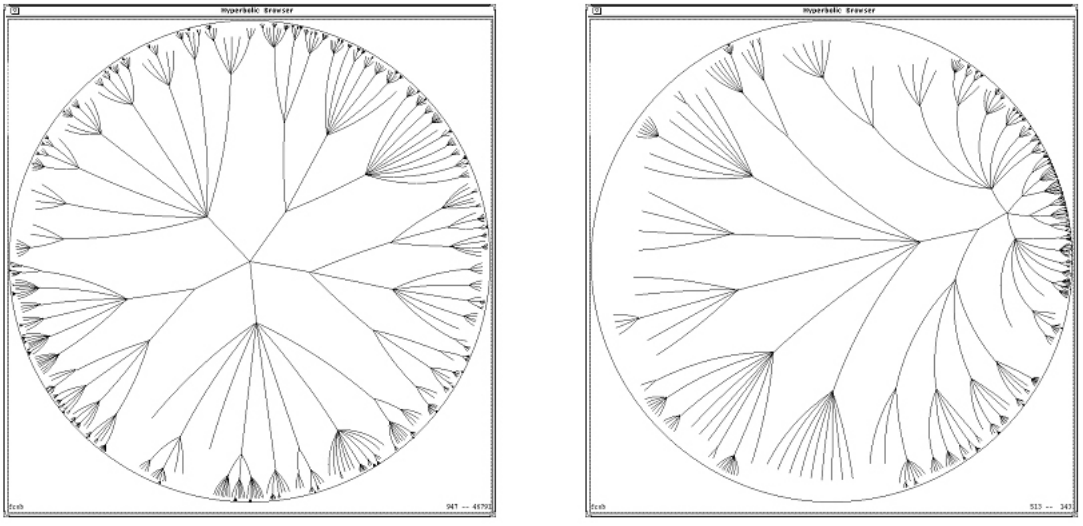

FIGURE 2.38: Change of focus on the Hyperbolic Geometry Focus+Context technique [34].

FIGURE 2.39: Spiral Representation and Augmented Context [34].

to the destination, which can be obtained from the arrival time of trains, is highlighted. The stations on the route enjoy larger spaces, whereas the other stations are rendered smaller and closer to fit the whole map into a screen. To simplify the navigation and route planning for visitors, the authors formulate various map characteristics such as octilinear transportation lines and regular station distances into energy terms. Then they compute the optimal layout using a least squares technique. In addition, the names of stations that are on the route of a passenger are labeled according to human preferences, occlusions, and consistencies of label positions using the graph cuts method. [35]. A comparison of this method with the traditional fisheye is depicted in figure 2.40.

FIGURE 2.40: (left) The official metro map of Atlanta city. (middle and right) The focus+context metro maps obtained using the fisheye and the F+C technique for Metro Map from Wang et al [35], respectively. The official map would become too small if it is displayed on a small area.

- **Semantic Depth of Field** - This method is based on the depth of field (DOF) effect used in photography and cinematography, and is therefore both familiar to

users and perceptually effective. Because this method blurs objects based on their relevance rather than their distance, it's called semantic depth of field (SDOF). [118]

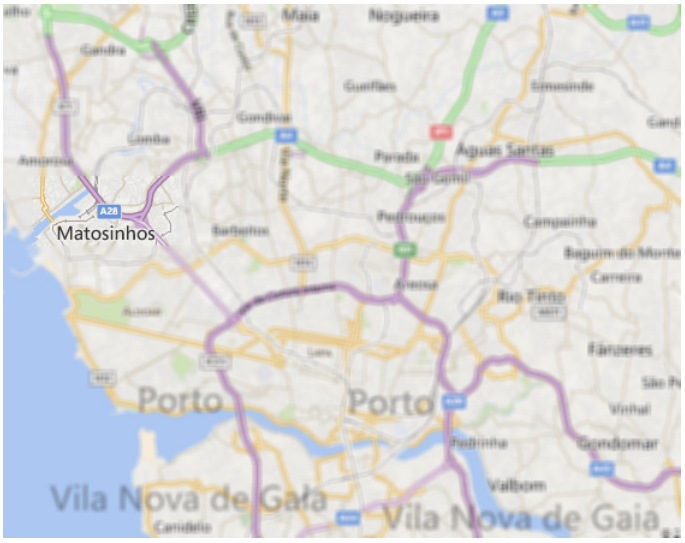

FIGURE 2.41: Semantic Depth of Field: the less relevant area is blurred while the important. It is possible to see that the important area (Matosinhos) looks sharp. Adapted from Bing Maps [36].

Other focus+context techniques appeared recently, associated with the onset of the mobile devices use and interaction, such as the fingerglass technique [37]. This technique lets the user to interactively define a viewport using one hand while the other hand can simultaneously interact with objects in the scene. The contents of this viewport are shown twice on the screen: in a global zoomed-out view stretched out across the entire screen, retaining contextual information, and as a magnified copy on top of the zoomed-out view.

FIGURE 2.42: FingerGlass focus+context technique[37]: The user specifies an area of interest with one hand and interacts with the magnified objects with the other hand. During the interaction, a new area of interest can be defined. Releasing all fingers makes the tool vanish.

Techniques like the fingerglass have the potential to highly improve map interaction in mobile devices and modern location-based services.

2.5 Location Based Services

Location-Based Services (LBS) are information services accessible with mobile devices through the mobile network which have the ability to make use of the location of the mobile device [119]. This definition is also accepted by the international Open Geospatial Consortium [120]. Some authors state that LBS are an intersection of several technologies as internet, mobile devices and geographic information systems (GIS) [38] [121]. This model is shown at figure 2.43.

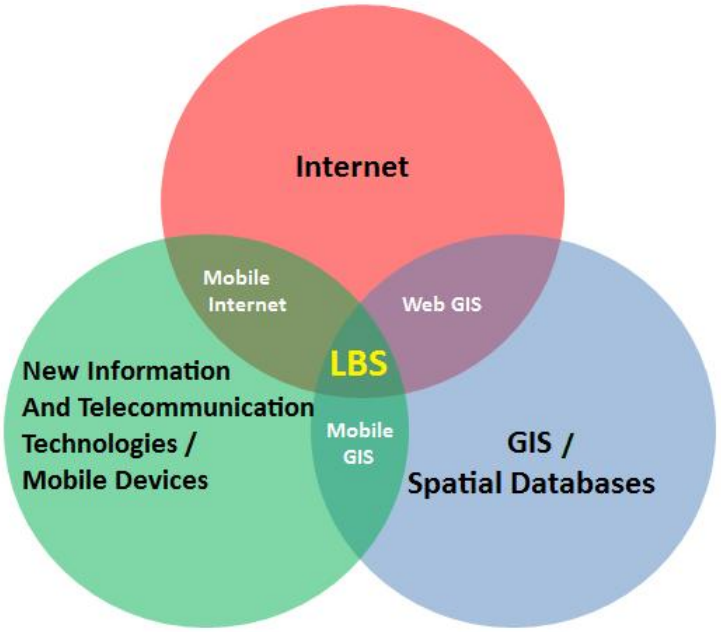

FIGURE 2.43: LBS as an intersection of technologies [38]

LBS allows the establishment of two way communication and interaction: the user tells the system his actual context, intention and/or preferences (or the system may obtain them in a pervasive way) which can help the provider of such location services to deliver information tailored to the user needs [106]. As it is also possible to observe in figure 2.43, there is a relationship between GIS and LBS. Both handle handle data with geographical reference and spatial analysis functions in order to answer the questions *"Where am I?"*, *"What surrounds me?"* and *"Where can i go to?"*. Nevertheless, their focus is completely different: while LBS targets large non-professional user groups, GIS can be seen as traditional "professional" systems to be used by a restricted group of expert

people. Schematic maps and Spider Maps in particular may be used to improve both GIS and Location-Based Services [54] and therefore their relation is specially important.

According to Steiniger [106], LBS have the following components:

- **Mobile Devices:** The apparatus serving as the physical interface for the user to access the service. It can be a smartphone, tablet, mobile embedded systems or toll payment systems (ex: in automobiles).
- **Communication Network:** The communication network that transfers data between the service provider and the user mobile device.
- **Positioning component:** This is the component that determines the position of the user. It can be a Global Positioning System (GPS) unit, or a triangularization technology that makes use of the wireless access points position (or GSM/CDMA antennas) to determine mobile device position, or any hybrid combination of both (as happens with assisted GPS (AGPS)). If this automation component is not present, the user has to specify its position by other mean.
- **Service and Application Provider:** The service provider offers a set of services to user.
- **Data and Content Provider:** Many times, the service and application provider needs to obtain certain information it does not own (such as geographical information, event information, etc) and needs to produce the contextual information that matters to the user.

LBS applications need to be aware of a set of details, such as the type of mobile user, the context of the user, the user needs (can be gathered by questioning the user or by pervasive means, automatically), the search and spatial analysis, the user interface, the visualization properties of the device and a wide set of technological questions(how to transmit and store data, technical protocols and details). These details allow the user to effectively use the services.

Reichenbacher [122] enumerated five possible mobile actions users usually execute when using LBS:

- **Locating:** This is the most obvious action: user wants to know where he is.
- **Searching:** User may want to search for persons, objects or events
- **Navigating:** User may ask for the way to a location

- **Identifying:** Involves asking information about a location
- **Checking:** User may look for events near or nearby some location.

From this five possible actions it is easy to understand that all of them depend on the context. We can divide the context into three types of context [106], as previously stated:

- **Spatial Context:** Where the user is,
- **Temporal Context:** When it is using the service,
- **User Context:** What is he using the service for.

Other authors [105] increase this list with other context types such as navigation history, orientation, purpose of use, social and cultural situation, physical surroundings and system properties. However, those context types can be viewed as subtypes of the three main context types proposed by Steiniger.

As it can bee seen, context is a main concept regarding LBS and as its importance is reflected in all the five kinds of mobile user actions, specially the spatial context.

2.5.1 Quality of Information in Mobile Services

Although mobile services are gaining popularity in contemporary life, there are some types of mobile services which are not effectively grasping their users. Public transportation services are among these services [123].

A comprehensive research [124] was made on how different dimensions of information quality affect consumers' satisfaction towards mobile information services and eventually the acceptance of these services. The fact that there are so many features which influence consumers' perceptions towards mobile services, called for a more precise theoretical framework. One promising way to consider the factors that affect the perceived quality of mobile information service from the consumers' point of view is the information quality framework of Chae et al.[39]. This framework identifies four dimension of information quality:

- **Context:** Although context was already mentioned previously, it is worth to mention that the definition from Dey [104] is adequate not only LBS but also generic mobile services context: *"any information that can be used to characterize the situation of an entity"*. Here an entity refers to a person, a place or an object that is considered important to the communication between the user and an application of the mobile service.

- **Content:** Content quality is the value and utility or usefulness of the information provided by mobile services [125]
- **Connection:** Connection refers to the link between the several components of the mobile service which allows the flow of information.
- **Interaction:** Interaction may be defined as the communication between a site and its users [126]. Mobile services achieve a high interaction quality if they are able to provide easy and efficient ways of interaction [124]. Being so, interaction is closely related to ease of use.

According to the framework, those qualities influence directly the user satisfaction. This influence has been proven by several studies [127] [128] [129] [130]. Based on the framework, Koivumaki et al [124] also show that all the four dimensions of quality in information services are positively related to user satisfaction. Another relevant conclusion of those studies is that user satisfaction is positively related to intention to use the service. The studies also state that although content quality is the most important factor, user satisfaction is affected by the set of the four factors, and consequently, the success of a mobile service depends on the form of all quality factors. Figure 2.44 shows the complete model.

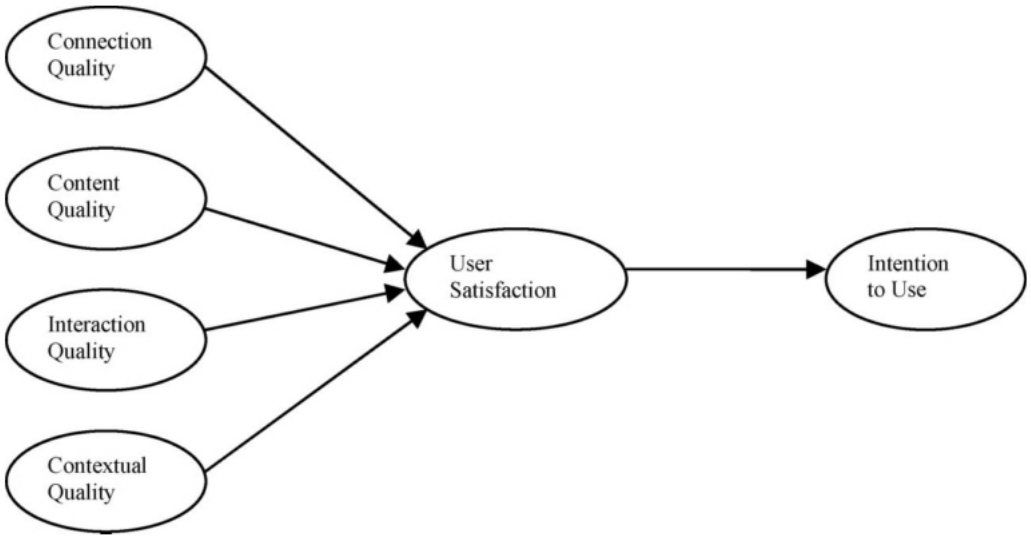

FIGURE 2.44: The theoretical model of information quality, source: [39]

Chapter 3

Spider Maps

Spider Maps are a type of schematic map (fig. 2.22) used in a few cities (London, Lisbon, Porto, etc) to depict public transportation networks. They are inspired by the London metro map, but they use enhanced context features and an improved visual presentation. This section focuses describes Spider Maps and how they reduces information overload, and how their advantages are, in fact, translated to real use through a usability test.

3.1 Concept and Definition

Bus spider maps are schematic maps with features that replicate some of the ideas of the conceptual spider maps. An example of a real spider map is depicted on figure 3.1, which illustrates a real spider map in use in London. A Spider Map is composed of two parts: a hub and a schematic map (fig. 3.2). Alongside with the spider map a route finder table is usually also found.

The Hub is a detailed rectangular area that depicts the geographic place where the map user is currently at (spatial context). This includes the contour of the surrounding buildings and all labels for the stops where the user can take a public transport to any of the reachable destinations departing from the hub. The hub is surrounded by a frame with points connecting the routes inside the hub with the schematic lines outside the hub (fig 3.3).

The schematic map outside the hub frame comprises a set of lines. Each line is formed by a sequence of segments which are characterized by a start and an ending stop. Segments shared by different lines are drawn together (usually side by side) if they follow the same path. Different lines can share stops(nodes) and segments. All the lines have a 0, 45 or 90 degree orientation and do not necessarily follow the geography of the city. Each node

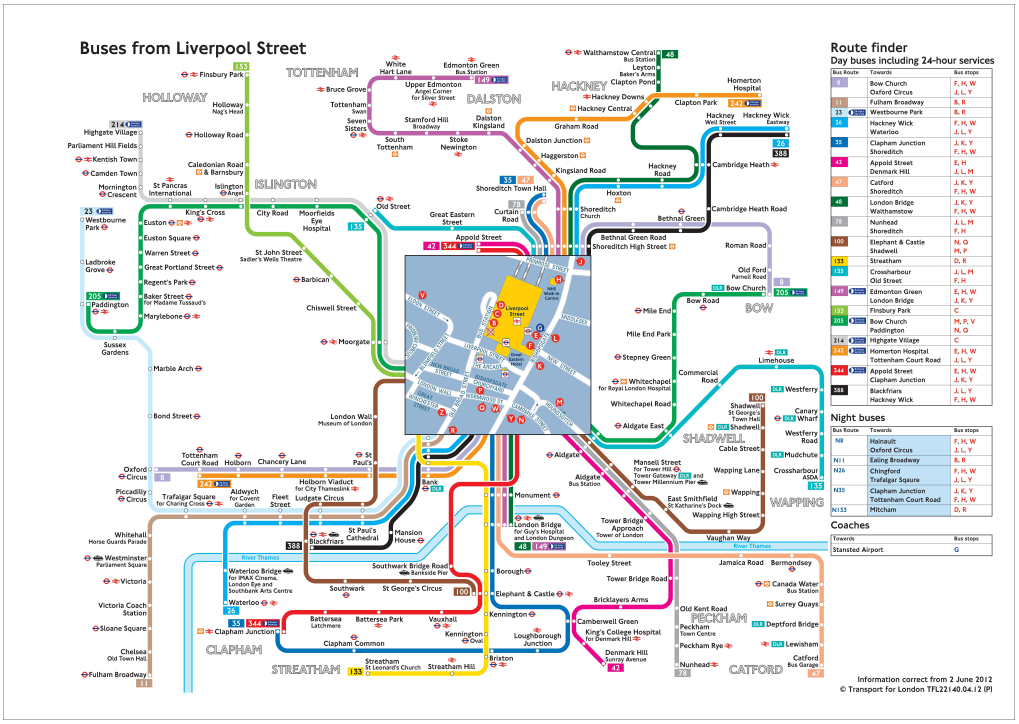

FIGURE 3.1: London Bus Spider Map (Buses from Liverpool Street) *Source: http://www.tfl.gov.uk*

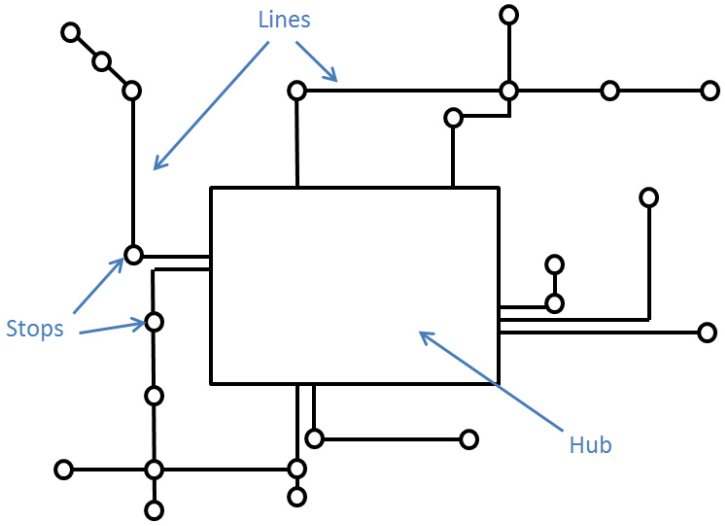

FIGURE 3.2: Spider map architecture schema

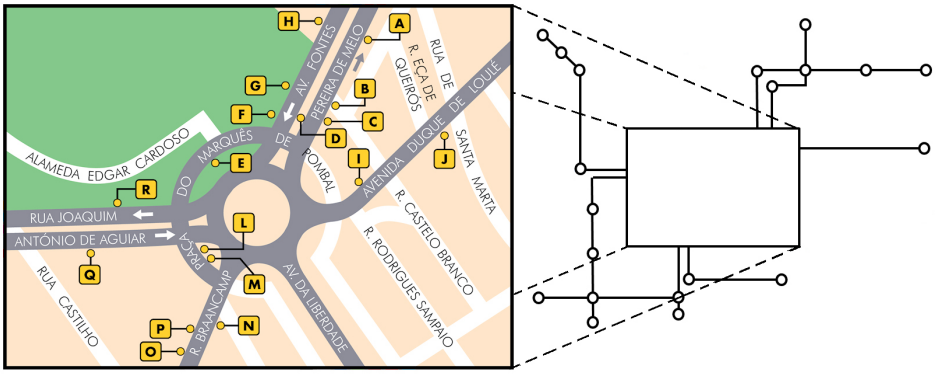

FIGURE 3.3: Example of the higher scale hub map, detailing all the stops (yellow squares with the identifying letters) [40]

corresponds to a stop or to a group of stops geographically close and handled together. Nodes and lines are identified by several visual elements such as shapes, colors and labels or images next to them. This implies that the drawing process needs to consider space for labels or other relevant visual elements. Geographical accidents (rivers, mountains, seashore, etc) may be also depicted if they are considered to be relevant.

A route finder table is usually placed next to the spider map and shows the direction and route that are associated to each stop inside the hub.

The properties of the spider map are visible in a real spider map example, such as the one shown in figure 3.1 which shows a spider map bus transportation network at a specific location, in this case, Victoria Station (London). It is possible to observe the spider map innovations in comparison with London underground maps of figures 2.23 and 2.26, most notably the hub depicting the user's spatial context, with the detail of the buildings and the stops in the neighborhood, with the transportation lines organized in such a way that they are grouped together, even if in reality they are not. This further increases the degree of abstraction removing undesired detailing. It is possible to see that the lines are schematized similarly to Beck's map, following the same orientation schema. Another conceptual and practical difference is the fact that a dense transportation network needs several spider maps, according to different locations of the user, as if the spatial context of the user changes, the map needs to reflect that. Being so, while a dense transportation network may be depicted by only one schematic map, the converse is true regarding spider maps. Therefore, for the same transportation network, the visual presentation of its spider maps may be very different depending on location of the hub of each particular map.

3.2 Design Guidelines

This section describes the set of main design guidelines of spider maps to reduce information overload (also called "Cognitive Load"[131]) in the production of spider maps:

- **Line simplification:** Reducing the number of inflection points on the lines reduces the complexity of the map, which has an improvement effect in map clarity [61]. Several techniques proposed for line simplification in schematic maps can still be applied to spider maps, such as Discrete Curve Evolution [84] [30] [87] or the Douglas and Peucker [27] algorithms, for example. Figure 3.4 shows an example of an iterative line simplification.

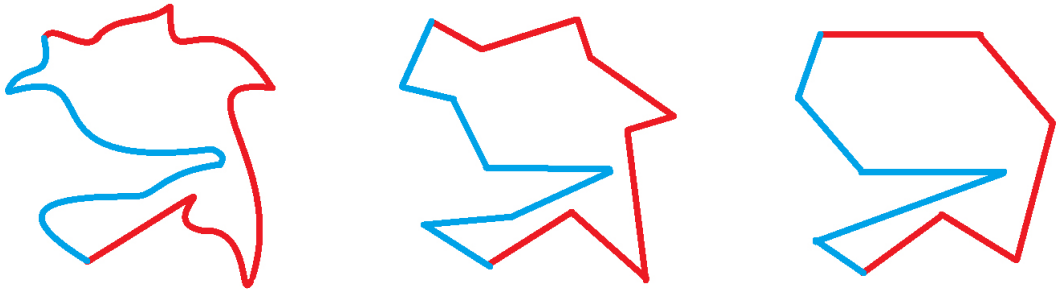

FIGURE 3.4: Example of an iterative line simplification, from left to right

- **Grouping nearby stops into map points:** This principle means that some stops that are very close to each other (and have similar names) are grouped together into a single map point. This option must be used carefully as it may eliminate information that some users consider relevant, but it is an effective measure in what concerns to decrease map cluttering. In fact, grouping stops not only reduces the number of points in the map, but also the number of needed labels. Figure 3.5 presents an example of point simplification [30] [63].

FIGURE 3.5: Example of point simplification.

- **Differential zooming in crowded areas:** This technique is used to achieve higher readability and uniformity in map reading, mainly in crowded areas [60]

[92] [132]. Points belonging to crowded areas are moved to neighbor areas with more free space, keeping the relative topology (Figure 3.6)

FIGURE 3.6: Example of differential zooming in crowded areas

- **Decreasing environment features detail:** This means, for example the simplification of geographical accident shapes. This technique is particularly visible in the shape of Thames river in London Tube [60] [29]. Although the real geography of the river is simplified and relaxed, there are important aspects to keep, for example, the topological relations of the stops and lines on each side of the river, to avoid user disorientation. Several generalization techniques have been researched in the literature, as shown in figure 2.24.

- **Grouping edges when they have the same start and end nodes:** Keeping lines tied together wherever possible reduces map entropy, makes line paths easier to understand and saves space for other map elements, as stated in chapter 2 and summarized in figure 3.7.

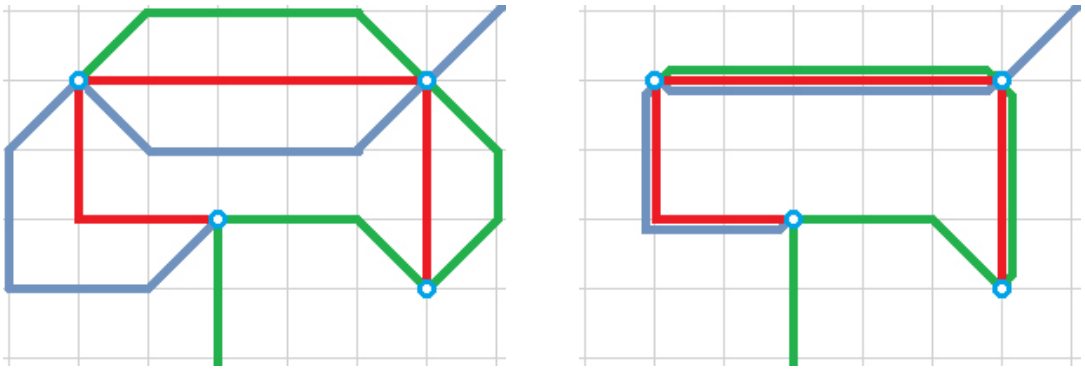

FIGURE 3.7: Grouping lines when following the same path.

- **Promoting simplicity instead of completeness of information**, depending on the particular purpose of each map.

- **Making use of differential symbol shapes/colors** to emphasize or deemphasize certain aspects. This topic was extensively studied for schematic maps [60], and spider maps can also take full advantage of it.
- **Direct user attention to the most important aspects**. Featuring a hub, user attention can be directed to user current spatial context which considerably reduces the time to perform map use tasks, and avoid superfluous information search.

3.3 Proof of Concept

There is almost nothing related to spider maps in literature but a couple of sparse references to their existence. As we build our research work on the automated generation of context-aware schematic maps, mainly spider maps, it is fundamental to demonstrate that spider maps are more effective than traditional diagrammatic maps in the communication of public transports information, due to their "by design" context awareness superior capabilities. Therefore we wanted to test a set of hypothesis that support our assumption that spider maps are a clear step forward regarding traditional diagrammatic maps in what concerns to the communication of transportation networks information. The hypotheses we formulate are the following:

- **Hypothesis 1:** Spider architecture is advantageous concerning to learning time, memorization, spatial reasoning and concept relation learning compared to traditional maps.
- **Hypothesis 2:** Spider Maps are an enhanced vehicle for providing passenger information in comparison with traditional diagrammatic maps.
- **Hypothesis 3:** Users prefer to use Spider Maps instead of traditional diagrammatic Maps

3.3.1 Test Design

Due to the close relation between mind maps and spider maps, we decided to assess the advantages of mind maps over common concept maps before comparing the performance of bus spider maps with bus diagrammatic maps. We designed a test composed of two phases. In phase 1 we use concepts maps to assess if the spider map architecture improves learning time, memorization, spatial reasoning and concept relation learning. Each phase has three different types of tasks. Users had two minutes to look at the first

TABLE 3.1: Tasks performed in phase 1 and 2 of the usability test

Tasks	**Phase 1**		**Phase 2**	
	Concept Map	Spider Map	Bus diagrammatic Map	Bus Spider Map
Look at the map (2 minutes)	Step 1	Step 3	Step 6	Step 8
Perform Usability tasks	Step 2	Step 4	Step 7	Step 9
Open answer questionnaire	Step 5		Step 10	

map (a common concept map) and then they performed a set of usability tasks. Then, they had two more minutes to look to a second map (a mind map) and they performed the same usability tasks for that map. In the last step of phase 1 users answered an open answer questionnaire. In phase 2, which has a similar structure, the objective was to evaluate if spider maps are more effective than diagrammatic maps in communicating transports network information. The overall structure of the test performed by each user is presented in Table 3.1.

In order to gather some subjective user feedback, we performed and open answer questionnaire (steps 5 and 10) with the following questions:

- **Question 1:** *"Which map was easier to learn, and why?"*
- **Question 2:** *"Which map did you find to be more intuitive, and why?"*
- **Question 3:** *"What did you like/dislike in the normal concept map?"*
- **Question 4:** *"What did you like/dislike in the spider concept map?"*
- **Question 5:** *"How do you think those concept maps could be improved?"*
- **Question 6:** *"Overall, which map did you like better?"*

For the design of the experiment, we adopted the approach suggested by Krug [133] and we used several small groups of volunteers instead of a single large group. As the first batch of users gave us valuable feedback, we were able to perform the second batch test with additional knowledge. We used three batches of three/four people, since 11 people volunteered from the test. The users were national and foreigners, with ages ranging from 21 to 33 and different backgrounds. We could have used different people for testing each map type, but we wanted to preserve personal learning and interaction skills to avoid biased results due to different people's skills. Krug's approach is also supported by the evidence gathered from the research work of Jakob Nielsen and Thomas Landauer [134]. They defend that elaborate usability tests are a waste of resources and that the best results come from testing no more than five users (figure 3.8) and running as many small tests as possible.

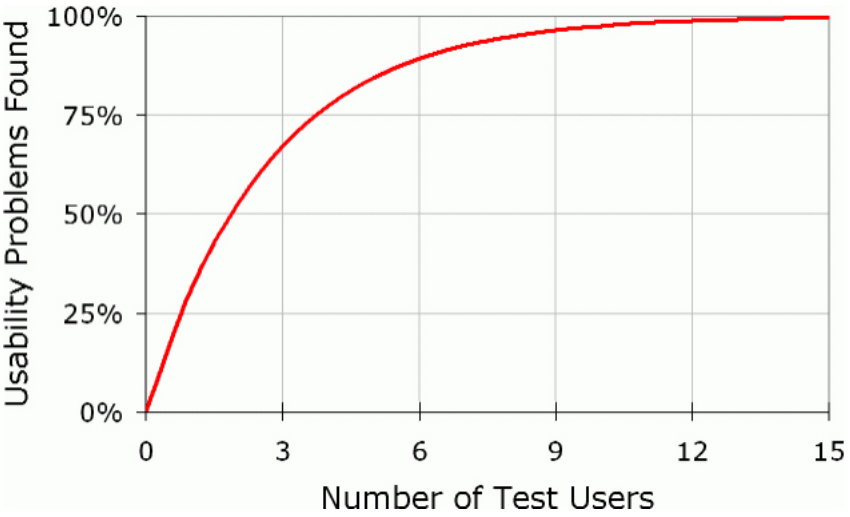

FIGURE 3.8: Diminishing returns for usability testing, as more and more users are tested [41].

Nielsen suggests to run multiple tests because the real goal of usability engineering is to improve the design and not just to document its weaknesses (this was also our goal in this test). After the first batch of people performed the usability test, it is possible to probe deeper into the usability of the fundamental structure of important issues, which are may not be clear at the first test. [41].

3.3.2 Phase 1 - Assessing Conceptual Maps

After looking at each conceptual map for two minutes (steps 1 and 3), the users drew a paper sketch of the analyzed map. Sketches are a complimentary tool of conventional usability testing techniques, and their cost is a fraction of the cost of other methods[135]. User sketches are quick to create, take only a brief time to analyze and provide reactive as well as reflective feedback. Users were told to replicate the maximum number of concepts and relations they remember and to identify the first concept that captured their attention. The sketches were then analyzed with the objective of testing the following parameters:

- **Task 1 - Overall memory recall speed:** the time the user took to draw a sketch of what he could remember from the concept map.
- **Task 2 - Overall memory recall correctness:** the number of correct concepts presented at the drawn map.
- **Task 3 - Overall memory recall correctness of concept relations:** the number of correct relations presented at the drawn map.

- **Task 4 - First looked concept:** the first concept where the user looked at, testing the focus property.
- **Task 5 - Context Learning:** we wanted to assess if the context learning was correct and uniform by the users.

The concept maps used for testing were related to different topics (to avoid the memorization bias effect from map to map), which have the same degree of familiarity to the users. The number of concepts and relations were the same in both maps. Figure 3.9 and Figure 3.10 depict the concept maps used in the first phase.

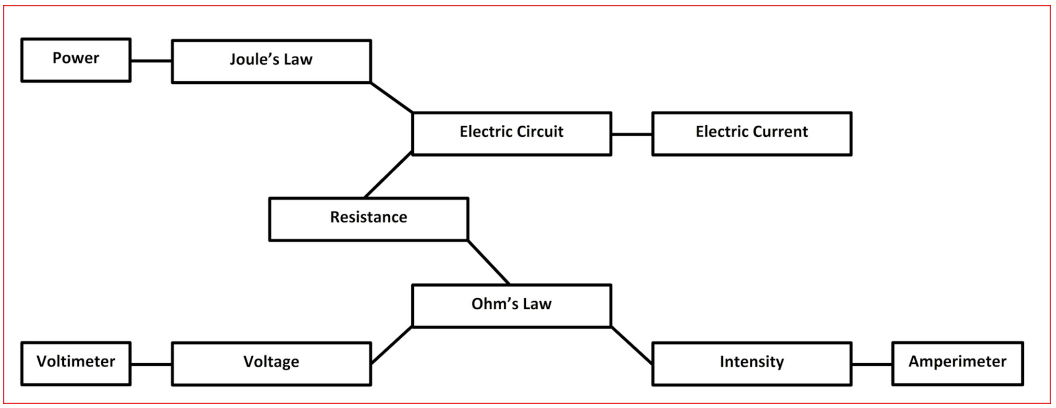

FIGURE 3.9: Concept map used in phase 1 of the test

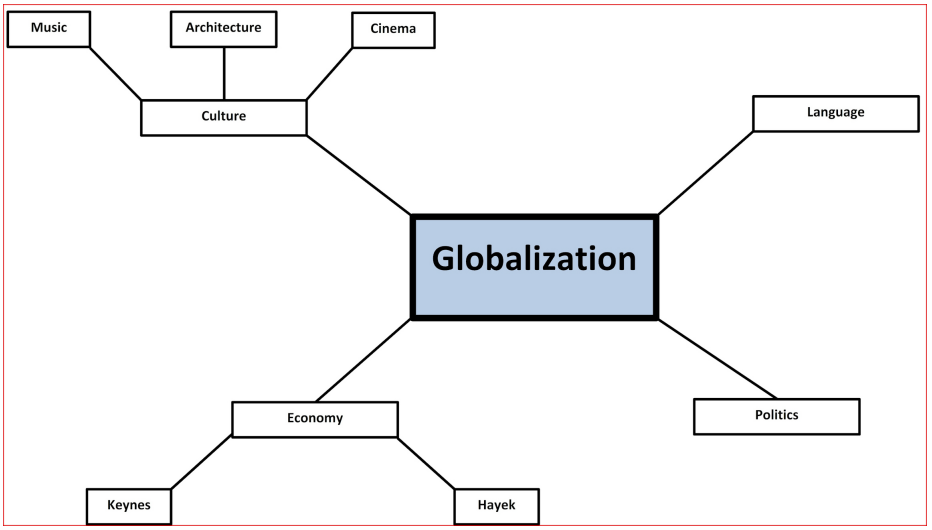

FIGURE 3.10: Spider concept Map used in phase 1 of the test

3.3.3 Phase 2 - Assessing Bus Network Maps

In phase 2, users tested real transportation network maps, which are in use in the city of Lisbon. The first one was a traditional diagrammatic map and the second one was a

spider map of the area where the user was. Both maps are publicly available in some bus shelters in Lisbon. The Lisbon diagrammatic map is presented in figure 3.11. The spider map was produced with our GenX framework and algorithms, with minor manual aesthetic work and is depicted in figure 1.1. The questions regarding location of specific places were different in each of them to avoid the memorization bias effect from map to map, which have the same degree of familiarity to the users. The parameters to be tested were concerned to the four main actions users normally perform with geographic services [122] [106]:

- **Orientation and Localization:** This action relates to locating, and answers to questions like "Where am I","Where is person/object"?
- **Navigation:** This action relates to navigating through space, such as planning a route, and answers to questions like "How do I get to place X?"
- **Search:** This action relates to searching for people/objects/etc. It answers to questions like "Where is the nearest person/object/etc".
- **Identification:** This action is related the identification of people or objects and answers questions like "How many objects are here?"

The users had 2 minutes to look to the first map (traditional diagrammatic map), then performed the usability tasks (step 7). Then they had two minutes again to look at the second map (the spider map) and they performed the same tasks (step 9). Each usability test was made of the following tasks:

- **Task 1 Locating - self:** We asked the user to locate himself on the map and we counted the elapsed time.
- **Task 2 Locating - notable point:** We asked the user to locate a point of higher importance (such as the city airport) on the map and counted the elapsed time.
- **Task 3 Navigation:** We asked the user how would he/she get to the notable point from his/her current location and measured the time it took to answer correctly.
- **Task 4 Search:** We asked the user to search for a random stop on the map. We measured the response time and registered whether the user gave up (which is an undeniable sign of user frustration).
- **Task 5 Identification:** We asked the user to count the stops in the neighborhood of its current location.

Finally, in step 10, the users where invited to answer the same set of open questions, comparing both maps, in order to gather some subjective user feedback.

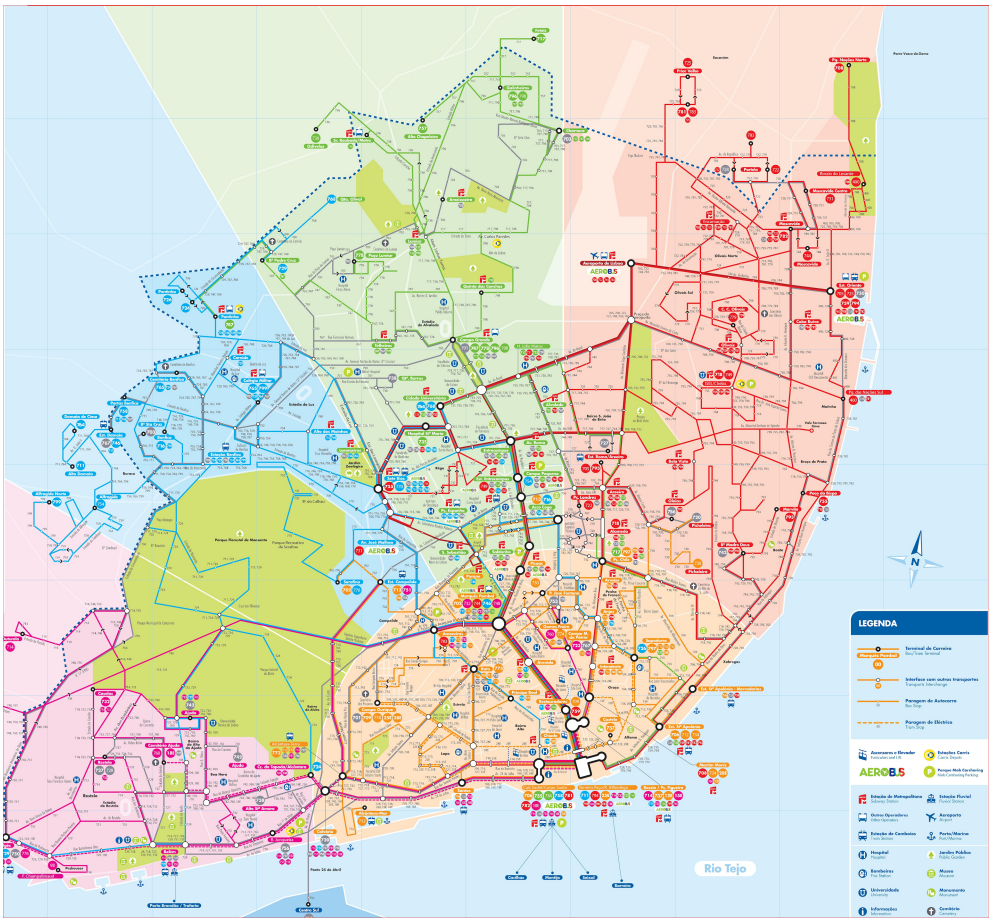

FIGURE 3.11: Diagrammatic Map of the Bus Network of Lisbon

3.3.4 Results

3.3.4.1 Phase 1 - Conceptual Maps Test

In the first task, we measured the time the user took to draw a sketch of each concept map and it showed that, in average, the spider map has a significant better performance, improving it by 25%, as shown in table 3.2.

TABLE 3.2: Average variation of sketch drawing times for mind maps

Normal Map AVG (s)	Spider Map Drawing AVG (s)	Var	% Var
72,63636364	54,54545455	**-18,09090909**	**-25%**

In the second task, we measured the overall memory recall correctness, by counting the number of correct concepts presented in the user drawn sketches. The results show that, in average, the spider map has a slight better performance, improving by 3%, as shown in table 3.3. In the third task, the objective was to measure the overall concept relation memory recall, by counting the number of correct links presented in the user drawn

sketches. The results show that, in average, the spider map has a better performance, improving by 8%, as shown in table 3.3.

TABLE 3.3: Average number of correct concepts and correct relations present at the drawn sketch

	Normal concept map	Spider concept map	Variation
Avg. N. of correct concepts (out of 10)	9,56	9,82	3%
Avg. N. of correct relations (out of 9)	8,27	8,91	8%

In the fourth task, we wanted to identify the first looked concept in each map, the "focus" point of each map. The results show that the common concept map has more "focus" points than the spider map. The user attention spreads over a larger number of points, making it more unpredictable to know where the user will look at. Figure 3.12 shows a comparison between the concepts that firstly attracted user attention in the common concept map (left) and in the spider concept map (right). The semitransparent ball size is proportional to the number of users that looked at that concept. If we compare both concept maps, we can see that the common concept map has four focus points, with the larger point being on the up left side of the map, while the other three focus points are divided around the middle area of the map. We can see that in the spider concept map there is a clear point of focus, which is the central concept. This is consistent with the theory behind the advantages of the spider architecture. At the same time, is seems natural that the center and the left upper corner are the areas that first attract people's attention in a picture. People (in occidental cultures) start reading documents from top left, so that may explain the small circles in the spider concept map and in the common concept map at that area.

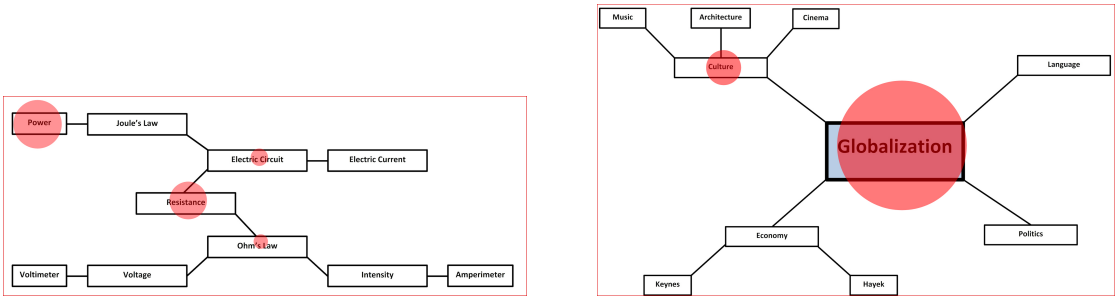

FIGURE 3.12: Comparison of the points that firstly attracted user attention in normal concept map (left) and spider concept map (right). The semitransparent ball size is proportional to the number of users that looked at that point.

In the fifth task, we wanted to assess if context learning was correct and uniform. Regarding the normal concept map, users divided themselves on five contexts (that are very close semantically to each other and all could be considered correct). Regarding

the spider map, users divided themselves in just two contexts, very close semantically to each other, even sharing the same word. We can say that the spider concept map leads to a more unified perception of context. Table 4 shows the concepts identified by the users.

TABLE 3.4: Concepts identified by users in normal concept map and in spider concept map

Normal Concept Map	
Context Identification	N. Users
Electricity	3
Electric circuits	3
Electromagnetism	3
Electric current	1
Physics	1

Spider Concept Map	
Context Identification	N. Users
Globalization	6
Globalization and Culture	3

To analyze the feedback from the open answer questionnaire (step 5), we used tag clouds since they provide a quick feedback about the main topics mentioned by the users. The representation is compact, and draws the eye towards the largest, most important items, and three dimensions are represented simultaneously (the words themselves, their relative importance) [136] without having to transcribe every word the users wrote in their multiple line answers. In the answers to the first question, *"Which map was easier to learn and why"* 10 users (out of 11) chose the spider concept map and one single user chose the common concept map. When asked why, the main words revealed by the tag cloud were "layout", "keyword", and "central" (see Figure 3.13). Therefore, we can say that the spider concept map is easier to learn due to its layout and structure.

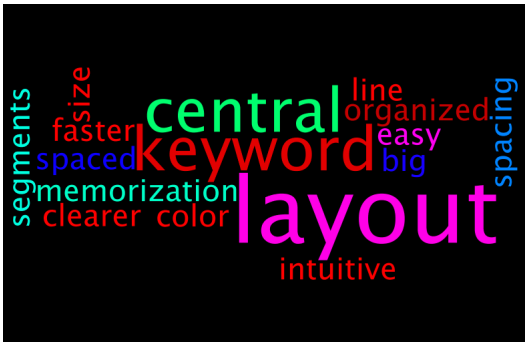

FIGURE 3.13: Tag Cloud showing the user justification about the easiness of learning of the conceptual spider mind map.

Regarding the second question, *"which map did you find more intuitive and why"*, 9 users (out of 11) have chosen the spider concept map, while 2 users chose the common concept map. Analyzing the tag cloud (Figure 3.14), the highlighted words used to justify the choice are "better", "layout", "memorization", "easy/easier","understanding". This seems to indicate that users think the spider concept map is more intuitive due to its structure, as it allows easier memorization and understanding.

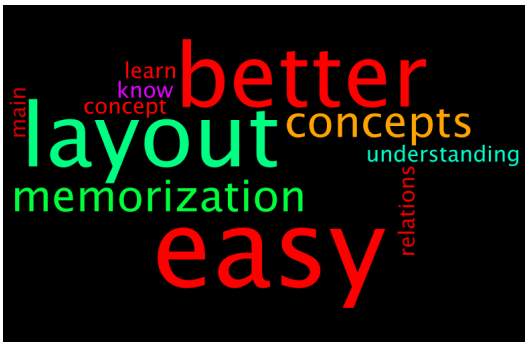

FIGURE 3.14: Tag Cloud showing the user justification about the intuitiveness of the conceptual spider mind map.

Regarding the third question, *"What did you like/dislike in the normal concept map"* a single user said it was well organized and structured, while the others disliked it due to the bad layout and because it was hard to understand. Figure 3.15 presents the corresponding tag clouds.

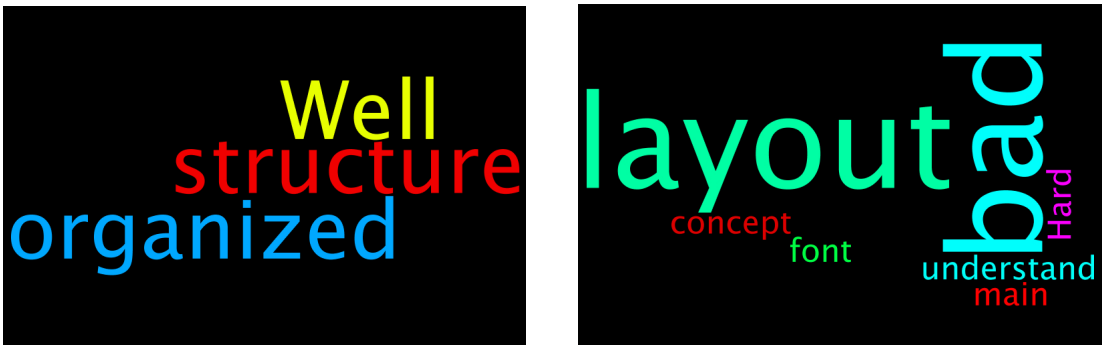

FIGURE 3.15: What users liked (at left) and disliked (at right) about the normal mind map

To the fourth question, *"What did you like/dislike in the spider concept map"* users liked it mostly because of the layout, the highlighted main concept, the easiness, the readability and intuitiveness. When asked what they disliked in the spider concept map, users referred the font and the spacing. Therefore, those aspects can be improved in spider maps. Figure 3.16 presents the corresponding tag clouds.

When asked the fifth question, "How do you think those maps could be improved", users identified a main word for it: "differential". Figure 3.17 shows some of the related concepts: differential styles, lines, colors, sizes, fonts, keyword highlighting, according to the relative importance of the topic.

To the sixth question, "Overall, which map did you like the better and why", all the respondents chose the spider map and the main justifications were the better layout, the central keyword, the easiness of learning and the intuitiveness, as we could see in the respective tag cloud (figure 3.18).

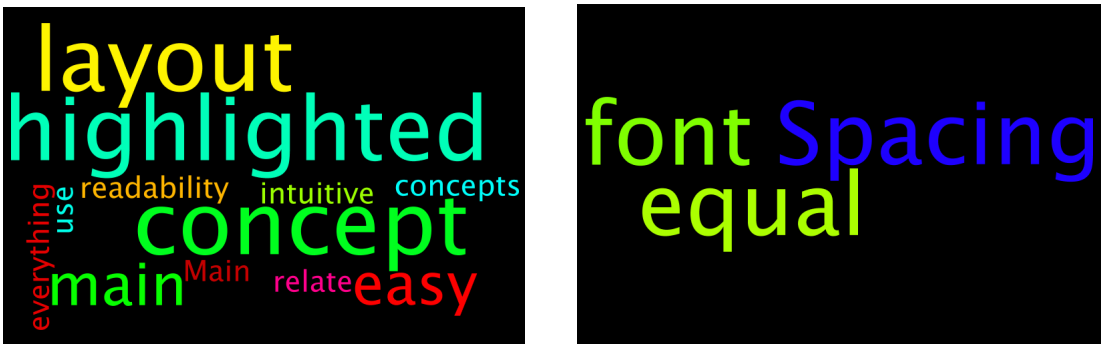

FIGURE 3.16: What users liked (at left) and disliked (at right) about the spider mind map

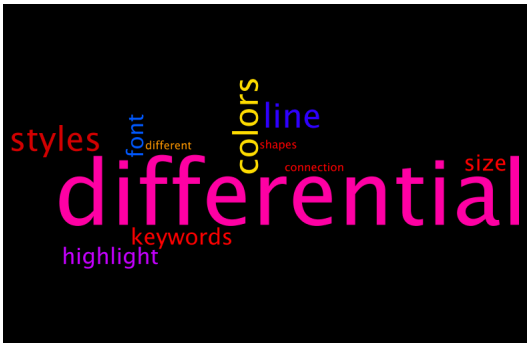

FIGURE 3.17: Tag Cloud showing the user topics on how conceptual maps could be improved.

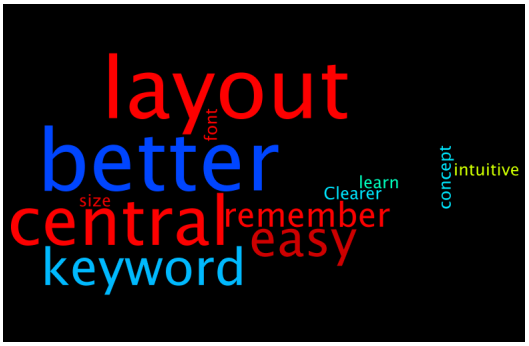

FIGURE 3.18: Tag Cloud showing the user justification about their overall preference on spider mind map.

3.3.4.2 Phase 2 - Transportation Network Maps Test

In the first task, we measured the time the users took to locate themselves on the map. Table 3.5 shows that the spider map presents a significant improvement over traditional diagrammatic maps, decreasing the self-location time by 94%.

TABLE 3.5: Average time of self location on maps

Diagrammatic Map AVG (s)	Spider Map AVG (s)	Variation	% Variation
24,54545455	1,363636364	**-23,18181818**	**-94%**

We believe this is due to the hub focus property, which depicts very clearly the place where the user currently is. Some traditional maps try to overcome this problem by placing a "You are here" tag. In the second task we measured the time the users took to locate a notable point (a point with a higher relevance) on the map, such as an airport, for example. In this task, the spider map showed no advantage or disadvantage compared to the diagrammatic map. Notable points are usually highlighted by different icons, so it is probably due to that reason they can be easily found on either type of map. The third task measured the time the users took to find the bus routes from their current location to the previously identified notable point (navigation task). Table 3.6 shows that the spider map reduces by 84% the average time users took to find their way in the transportation network

TABLE 3.6: Average variation of navigation task on maps

Diagrammatic Map AVG (s)	Spider Map AVG (s)	Variation	% Variation
39,27272727	6,272727273	**-33**	**-84%**

In the fourth task, we measured the time the users took to find the nearest metro station from a random stop of the bus transportation network. This task involved two subtasks: the first one was to search for the mentioned bus stop, and the other one was to search for the nearest metro station. Table 3.7 shows that spider map decreased by 97% the time to complete this of task for those users that completed the task. In fact, eight users (in 11) gave up from the search in the diagrammatic map, seven of them after having spending some time searching for the mentioned bus stop. All users were able to complete the task with the spider map. Besides reducing the searching time, spider maps are able to reduce user frustration. We can see that there is a connection between efficiency and emotions [137]: an efficient design improves user satisfaction.

TABLE 3.7: Average variation of the searching task time on maps

Diagrammatic Map AVG (s)	Spider Map AVG (s)	Variation	% Variation
201,9090909	5,363636364	**-196,5454545**	**-97%**

In the fifth task (identifying) we asked the users to count the stops in the neighborhood of its current location. We measured the time the users spent in counting the stops and the number of counted stops. The results presented in table 3.8 show that spider maps reduce the time counting the neighbor stops by 79%.

TABLE 3.8: Variation of the time spent by the users in counting the stops in the neighborhood

Diagrammatic Map AVG (s)	Spider Map AVG (s)	Variation	% Variation
35,63636364	7,545454545	**-28,09090909**	**-79%**

The most curious aspect of this task was the concept of "neighborhood". This concept was deliberately not explained to the users while performing this identification task nor further details were provided on the radius of the neighborhood area. The results show that while in a traditional diagrammatic map, users have different ideas of neighborhood (identifying on average 7 different stops), in a Spider Map the vast majority of users have the same idea of neighborhood (on average, users identified 2 different stops). This happens because in the spider map, the neighborhood is contained in the hub, so it seems to be an intuitive concept for most of the users. As the hub is bounded by a frame, that frame encloses the area of the neighborhood, so there is a standard precise intuitive definition of neighborhood in the spider map. This does not happen in the traditional diagrammatic maps due to their design limitations.

In what concerns to the subjective user evaluation, to the first question, "Which map was easier to learn and why", all the users considered the spider map as easier to learn. When asked why, the main words revealed by the tag cloud (figure 3.19) are: easier, better, clearer, less, detail, organization, location, reading. Therefore, we can say that the spider map is easier to learn because people perceive it as being clearer, better organized, with less detail and easier to read than the diagrammatic map.

FIGURE 3.19: Tag Cloud showing the user justification about the easiness of learning of the Spider Map.

The answers to the second question, *"Which map did you find more intuitive and why"*, showed that ten out of eleven users chose the spider map and one single user chose the diagrammatic map. When looking at the tag cloud (figure 3.20), some words emerge immediately: easier, clearer, learning, faster, context. This seems to indicate that users think the spider map easier to learn, clearer, with a better organization, faster navigation and location. The single user who considered the traditional map more intuitive justified his choice by saying that diagrammatic maps are more similar to the traditional maps and that he could have a more accurate idea of distance. This seems to be the effect of the distortion caused by differential scale function referred in section 2.2.

FIGURE 3.20: Why users considered traditional diagrammatic map more intuitive (at left) or the Spider map more intuitive (at right)

The results of the answers to the third question, *"What did you like/dislike in the traditional diagrammatic map"*, are presented in figure 3.21. Users highlighted the words "cit","information", "accurate", "visualization" and "topology".

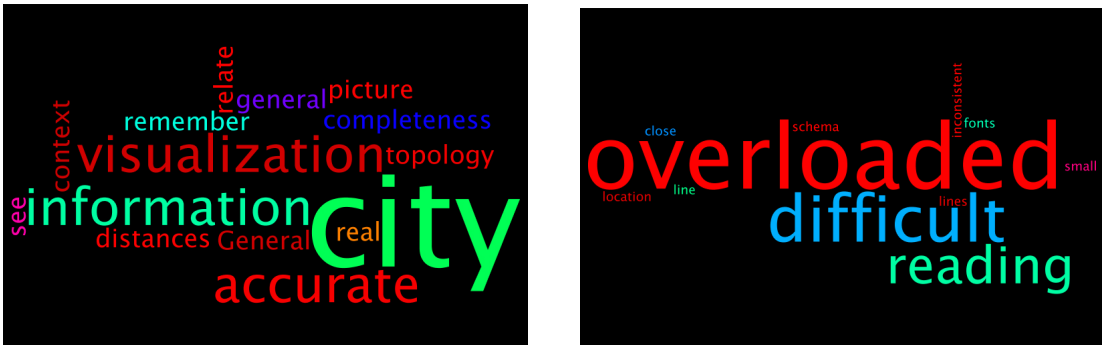

FIGURE 3.21: What users liked (at left) and disliked (at right) about the normal concept map

This may indicate the positive aspects of the traditional diagrammatic maps are that it gives a more accurate visualization of the whole city and its topology, providing more information to the user. However, when users where asked what they disliked in the traditional diagrammatic map, there were three main words: "overloaded", "difficult" and "reading". This means that, although users like to have lots of information, this becomes a two-edged sword, as too much information makes the map overloaded and difficult to read [17]. This may lead to user frustration and refusal to use the map or to use the transport. To the fourth question, "What did you like/dislike in the Spider map", the answers show that the aspects users like in spider maps were "easy", "neighborhood", "simplicity", "clearer", "layout" and "location" (as shown in figure 3.22-left).

This may indicate that users like spider maps because they are easier to learn, give a precise sense of neighborhood, are simpler, clearer, with a better layout, which enhances location tasks. What the users did not like was mostly described by two words: "information" and "lack". This may that users would like to have more information in the

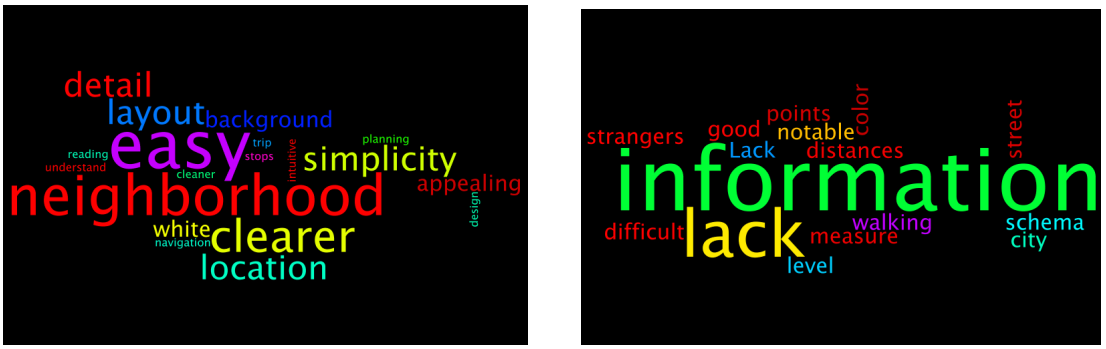

FIGURE 3.22: What users liked (at left) and disliked (at right) about the spider map

spider maps. When asked how those maps could be improved, users suggested the reduction of line crossings, the use of the colorADD[1] schema to allow better line visualization, especially when aggregated. Users also suggested the inclusion of more geographical tags and clues, more notable points that could help people in their location, searching and identifying tasks. The answers to the sixth question, "Overall, which map did you like better, and why", showed that one single user preferred the diagrammatic map while the remaining 10 chose the spider map. Figure 3.23 highlights the main reasons (better layout, the central keyword, easiness of learning and memorization).

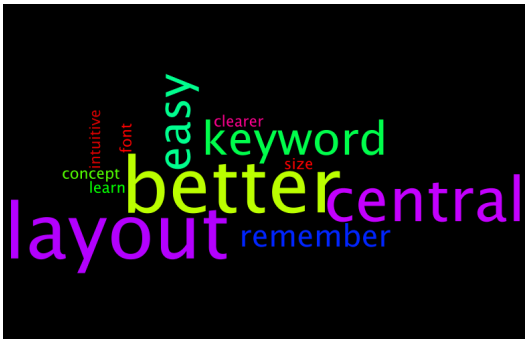

FIGURE 3.23: Tag Cloud showing the user justification about their overall preference on spider mind map.

3.3.5 Test Conclusions

Spider maps are particularly suited to represent transport networks, and they are much more effective in the communication of spatial information than their traditional diagrammatic counterparts. In order to access the effectiveness of spider maps, we designed and performed an experiment with real users. The experiment was composed of two phases. In the first phase, we used simple concept maps to test if the spider architecture improves short-term memory and learning. In the second phase, we used real bus transportation maps to evaluate location and navigation tasks. In both phases we collected

[1] http://www.coloradd.net/

subjective evaluation from the users, in addition to objective measurements. Regarding the concept maps, the main conclusions are that the spider concept maps present better results in short term memory tasks, improving concepts and links memory recall. Spider maps also effectively implement focus-and-context technique. Concerning the subjective evaluation performed by the users, the results showed that users consider that spider concept maps are easier to learn, more intuitive, have a better layout and, ultimately, they showed a clear preference for the spider map over the common concept maps. Regarding the real transportation network maps, the tests have shown that the time to complete the tasks with success was considerably inferior for the spider map than for the diagrammatic map. In this section, we have also illustrated different user-centered design principles and strategies that spider maps use to reduce the extraneous cognitive load[131] so user's available intellectual resources could be devoted to their main actions of using a map. The test users considered spider maps easier to learn and more intuitive than diagrammatic maps, with a better layout, less information overload and improved readability. Spider maps are more attractive to users, and that may also be a reason they work better, as as Norman says *"attractive things work better"*[138]. Nevertheless, users also suggested that spider maps could be further improved by reducing line crossings, by using ColorADD schema to allow better line visualization, and by including more geographical clues.

3.4 Spider Maps as Location Based Services and Public Transportation Improvement

As it what concerns to Location-Based Services, spider maps can be an excellent tool for presenting spatial information on Public Transportation networks. As spider maps improve information quality [54] [39] through the inherent advantages and innovations of their design, this leads to higher user satisfaction and consequently increase the intention to use the service. Dziekan [139] studies mention that one way to create high ridership in public transportation services is to strengthen their attractiveness by improving the quality of service. The author also presents the example of the city of Stockholm cooperation with the public transport authority in order to increase traveler numbers. This objective was also achieved by improving factors such orientation and information. Spider maps can play a fundamental role here in improving those factors, through their use in LBS. Spider maps are, therefore a highly adequate vehicle to communicate transport network information in mobile services due to their higher information quality in comparison with normal or schematic maps.

3.5 Conclusions

In this chapter we presented the Spider Maps, their definitions and their advantages regarding traditional diagrammatic maps. The full potential of spider maps can only be unleashed if they are generated in soft real time, which leads us to the conclusion that it is fundamental that they could be produced automatically. Only the automated production of spider maps can allow them to be agile enough to reflect different contexts, supporting what Steiniger calls *adaptative services*: services that dynamically respond to context [106]. It is easy to understand that it is faster to update a map if we generate it automatically than if we build it manually. This would allow us to produce spider maps to fit specific users, time or spatial contexts, extracting the full potential of the spider map enhancements. It is our conviction that automatically produced spider maps could also be a good solution to improve homing in mobile devices [140].

Chapter 4

Problem Modeling

The automated generation of Spider Maps presents a complex optimization problem that requires efficient data structures while preserving the semantic meaning of map features, such as lines, hub and stops. The production begins with a normal transportation map and ends with the production of a Spider Map. The rich semantics of an transportation network usually needs the use of a semantically rich data structure to store all the relevant map feature relations. Nevertheless, this structure would not be adequate for intensive processing due to its complexity, and consequently, this data model needs to be converted to a graph structure, simple and efficient, but keeping stored apart all the relations between map features. The output of the generation of a Spider Map would require an inverse conversion to a semantically rich structure together with the restoration of the relations between map features. The processing of the simple graph data structure is performed as an optimization process. This chapter presents the modeling of the the automated generation of Spider maps in what concerns to the underlying problem, the needed data structures and the optimization process.

4.1 Problem Description

Given a transportation network, the objective is to automatically generate a spider map for a specific location (with the hub detailing a specific area). The transportation network is comprised of transportation lines (the routes) and stops. Each transportation line is made up of segments, which are paths between two consecutive stops. A stop can be shared among several lines. Each line, segment or stop has a set of properties. These properties can be related to the transportation network (ex: location, sequence number), to its identification(ex: name/label) or to its visual presentation(ex. color). The goal is to generate a Spider Map with the best possible presentation while keeping

topological and semantic "correctness" as needed. We may preserve topological relations by enforcing the compliance of hard constraints (constraints that can not be violated). The final presentation is obtained by the minimization of an objective function built around a set of soft constraints (constraint that may be violated, but that violation yields a score penalty) which models the *visual quality* of the solution. The algorithm starts with a geographically accurate map, where there may be precise geographic accuracy regarding the location of the transportation network elements. The final result is a spider map satisfying all the hard constraints and with maximun quality. This is a complex optimization problem with multiple and conflicting objectives, and it was also proved that an analog problem is an NP-complete problem [87]. A solution for our problem is a Spider map with all the data needed presented visually. This includes all the spatial coordinates and all the relevant features (colors, shapes, labels, etc) of the components that comprise the spider map.

4.2 Spider Map Modeling

In this section we propose a formal model of a spider map. A Spider Map is a structure complying with the following conditions:

- **C1:** $SpiderMap = (P, V, H, E, L, A, G_r)$ (A Spider Map is an ordered 7-tuple of a set of points (P), vertices (V), a hub (H), a set of directed edges (E), a set of Lines (L), a set of angles (A) and a set of geographical Restrictions (G_r).
- **C2:** $\forall p \in P, p = (p_x, p_y, p_z), p_x, p_y, p_z \in \mathbb{R}$. (Each point is formed by an ordered 3-tuple (coordinates x, y, z).
- **C3:** $\mid P \mid \in]0, \infty[$ (The Point Set cannot be empty and must contain a finite number of elements).
- **C4:** $\forall v \in V, v = (p, l), p \in P$. (Each vertex is formed by an ordered 2-tuple of a point and a label which identifies the vertex, (usually the corresponding Stop Name)).
- **C5:** $\mid V \mid \in]0, \infty[$ (The Vertex Set cannot be empty and must contain an a finite number of elements)
- **C6:** $H = (p_1, p_2, P_{hub}$ (The Hub is defined by a top-left point p_1 and a bottom-right point p_2 and P_{hub} is a subset of P, and contains the points located inside the Hub).

- **C7:** $\forall e \in E, e = (v_1, P_e, v_2,), v_1, v_2 \in V, P_e \in P, \mid P_e \mid = [0, \infty[$ (Each point is formed by an ordered 3-tuple of the initial vertex v_1, a set of inflection points P_e and a final vertex v_2. The set of breakpoints P_e may be empty but must contain a finite number of elements).
- **C8:** There is a mapping function $\delta : E \rightarrow V \times V.\delta(e) = (v, w) \wedge (v \neq w)$ (The mapping function returns an ordered pair. There are no loops and therefore a Spider Map is a simple directed graph).
- **C9:** $\mid E \mid \in [0, \infty[$ (The Edge Set must contain an a finite number of elements)
- **C10:** $\forall l \in L$, l is an ordered sequence of edges $e \in E$ (A line is comprised of an ordered sequence of edges: $l = (e_1, e_2, ..., e_k), k = \mid l \mid$).
- **C11:** $\mid A \mid = 8$ (The number of angles of the edge representation is 8, corresponding to the angles 0, 45, 90, 135, 180, 225, 270 and 315 degrees).
- **C12:** $\forall g_r \in G_r$, g_r is a sequence of points $p \in P$, and a sequence of line segments connecting consecutive points, forming a closed polygon.

As an illustrative example, we present the spider map in figure 4.1 which corresponds to the following map model:

- $SpiderMap := (P, V, H, E, L, A, G_r)$
- $P := \{P_1, ..., P_{45}\}$
- $V := \{V_1, ..., V_{24}\}$ (The spider map has 24 Vertices that correspond to line stops on the map), where $V_i = \{P_i, Label_i\}$
- $H := \{P_{38}, P_{39}, \{P_{40}, P_{41}, P_{42}, P_{43}, P_{44}, P_{45}\}\}$
- $E := \{E_1, ..., E_{25}\}$
- $E_8 := (V_7, \{P_8\}, V_9)$ The edge 8 has V_7 as start node and V_9 as end node. It contains also an inflection point P_8. Similar definitions apply for the other Edges of the map.
- $L := \{L1, L2, L3, L4, L5, L6\}$, where $L_k = (E_j), , j \leq |E|$ ex: $L1 := (E_4, E_5, E_6, E_7, E_8, E_9)$
- $A := \{0, 45, 90, 135, 180, 225, 270, 315\}$
- $G_r := \{G_{R_1}\}$
- $G_{R_1} := (P_{29}, P_{30}, P_{31}, P_{32}, P_{33}, P_{34}, P_{35}, P_{36}, P_{37})$

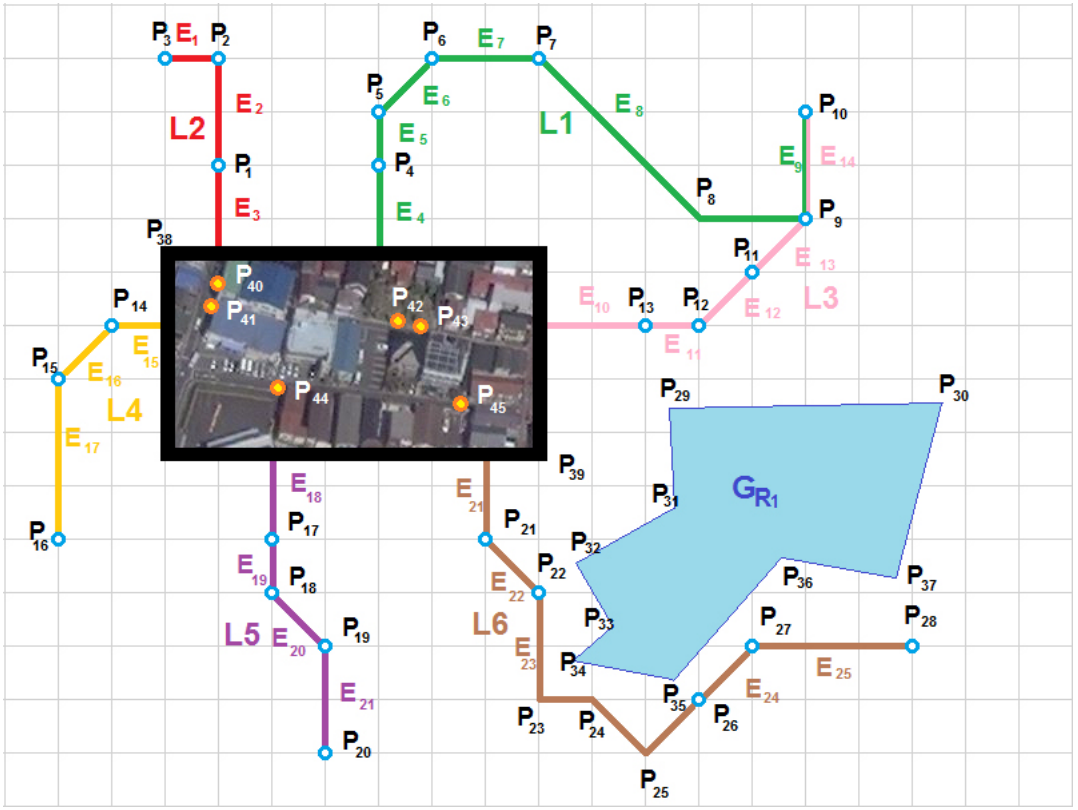

FIGURE 4.1: Spider Map Definition example

This definition is semantically richer than the definition of a simple Graph $G = (V, E)$, which is not able to capture the semantic complexity of a transportation network spider map. To exploit the advantages of having a semantically rich data structure (SRDS) that is powerful enough to describe a real spider map while having fast computational performance, we decided to implement two data structures: one that is semantically richer, reflecting the real spider map analogies and complexity, and a simple graph data structure (SGDS) which will serve as a working data model for our information system. This way we were able to combine all the advantages while not having any disadvantage. This approach is a considerable improvement over the state of the art approaches to this problem. The conversion between these two structures is achieved through two mapping functions: one can convert from the SRDS to the SGDS, and the other one can convert the other way around. Figure 4.2 shows the initial conversion from the semantically rich data structure (SRDS) to the simple graph data structure (SGDS) and the mappings.

4.2.1 Semantically Rich Data Structure

The semantically rich data structure closely follows the real spider map semantics, so the mapping between the reality of a transportation network and our semantically rich spider

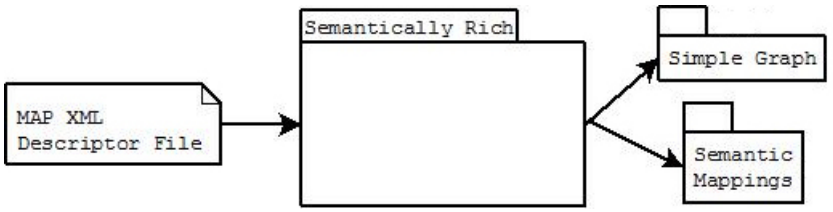

FIGURE 4.2: UML Diagram depicting the initial data structure conversion from the semantically rich data structure to the simple graph data structure and mappings.

data structure is direct. To every feature in a real spider map there is a correspondent piece in that structure, as shown in table 4.1

TABLE 4.1: Mapping Between Real Spider Maps and our Semantically Rich Data Structure

Spider Map	**Semantically Rich Data Structure**
Hub	H
Hub Exit Points	P
Hub Limits	P
Stops in Hub	P
Line	L
Segment	E
Segment/Line Bendings	P
Stop	V
Geographical Restrictions	G_r

The other data sets that comprise the semantically rich data model which are not a direct translation of spider map features, such as the number of the angles and the inner relations between vertices and points are automatically derived.

4.2.2 Simple Graph Data Structure

A spider map can also be modeled through a graph $G = (V, E)$. Using a simple graph to model a spider map we lose most of the semantic meanings (for example, the notion of lines, geographical constraints, hub, etc), nevertheless we gain simplicity and agility for algorithm processing. Therefore, to avoid the downside and extract the full potential, we do not use this simple graph to directly model the spider map. We use this simple graph data structure as a middleware model which can only be mapped from (and to) the semantically rich data structure. To solve the problem of losing the semantic meanings of the map features, besides the simple graph (which will be processed across the algorithm), we keep a mapping table that stores the semantic meanings of the spider map. This way, when the algorithm finishes processing the simple Graph model, we can convert the simple graph data structure to the semantically rich data structure by

recovering the semantic mappings stored in that table. This way, the algorithm can be executed over a light data structure but as we keep the semantic mappings, we can solve real world complexity problems. Table 4.2 shows the relation between the semantically rich data structure and the simple graph data structure.

TABLE 4.2: Relations Between the Semantically Rich Data Structure and the Simple Graph Data Structure

SRDS	**SGDS**
P	Vertices
V	n.a.(stored in semantic mappings table)
H	Vertices
E	Edges
L	n.a. (stored in semantic mappings table)
A	n.a. (used to process Vertex and Edge positioning)
G_R	n.a. Converted to Polygons (looped graph structure)

The mapping table stores the correspondences between the semantically rich and the simple graph structures. This table contains a structure where each vertex is tagged with a unique ID, and contains the information of the line and the segment (edge) it belongs to, and whether it is the start or the end node of the segment. Each Vertex can have several mapping table entries, if it makes part of more than one line. Table 4.3 exemplifies the mappings table for vertices 1 to 10 of the spider map depicted in figure 4.1.

TABLE 4.3: Semantic Mappings Table example for Points 1 to 10 for the spider map depicted in figure 4.1

Mapping #	**Vertex ID**	**Line ID**	**Segment ID**	**isStartNode**
1	1	2	2	True
2	2	2	2	False
3	2	2	1	True
4	3	2	1	False
5	4	1	5	True
6	5	1	5	False
7	5	1	6	True
8	6	1	6	False
9	6	1	7	True
10	7	1	7	False
11	7	1	8	True
12	9	1	8	False
13	9	1	9	True
14	9	3	13	False
15	9	3	14	True
16	10	1	9	False
17	10	3	14	False

4.3 Data Structures

The data structures were designed to directly support the spider map data modeling described in the previous section, as well as their processing. The global view of the data structures organization, and data flow is shown in the UML diagram depicted in figure 4.3.

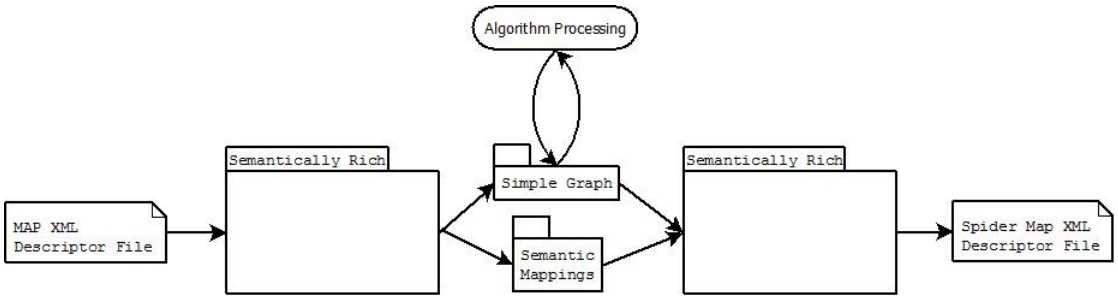

FIGURE 4.3: UML Diagram depicting the data structures organization and data flow.

Our approach to the generation of Spider Maps starts by reading an XML file which describes the transportation network to be transformed in a spider map. It contains the locations of the map features (not schematized) and every semantic information that is inherent to the complexity of a real transportation network. This file (which can be obtained through TCP/UDP[1] (for example, through a web service negotiation), disk reading or through pipelines[2]) is read, de-serialized, parsed to their atomic components (such as stops, lines, etc) and then converted and stored into the semantically rich data structure. This structure is then converted into the simple graph data structure which is agile enough to be processed, while the semantic mappings are kept to avoid the loss of semantic information regarding the transportation network. After processed by the algorithms, the simple graph data structure. This structure is combined with the semantic mappings to obtain the resulting processed semantically rich data structure, which can then be stored into an XML file that can be used to output the map in several physical (ex: paper) or virtual supports (computer screens, 3D projections, mobile device screens, etc). The XML file is usually a SVG file due to its system interoperability, scalability and lossless encoding which allows it to be displayed into lots of media types without losing visual fidelity. This XML file can then be directly presented to end users, sent to a network, or further processed by designers or other professionals.

The UML Class Diagram depicted in figure 4.4 implements the semantically rich data structure. It is worth to mention the detail of the coordinate system used: we keep

[1] TCP and UDP are network communication protocols for digital information exchange

[2] Usually just called "pipes", the pipelines are connections between two computer processes, such that the standard output from one process becomes the standard input of the other process, allowing the transfer of information between them.

the real world coordinates (latitude and longitude), but in what concerns to algorithm processing, we use *paper coordinates*, which are related to the coordinates in the map canvas (either physical or virtual). This class diagram also features some classes (ex. Interface, Area) that are used for tailoring the map to specific purposes such as specific events or themes. The "MapPoints" class map allow us to process the spider map by grouping nearby stops into the so called "mapPoints", considering each mapPoint as a "dense area of stops".

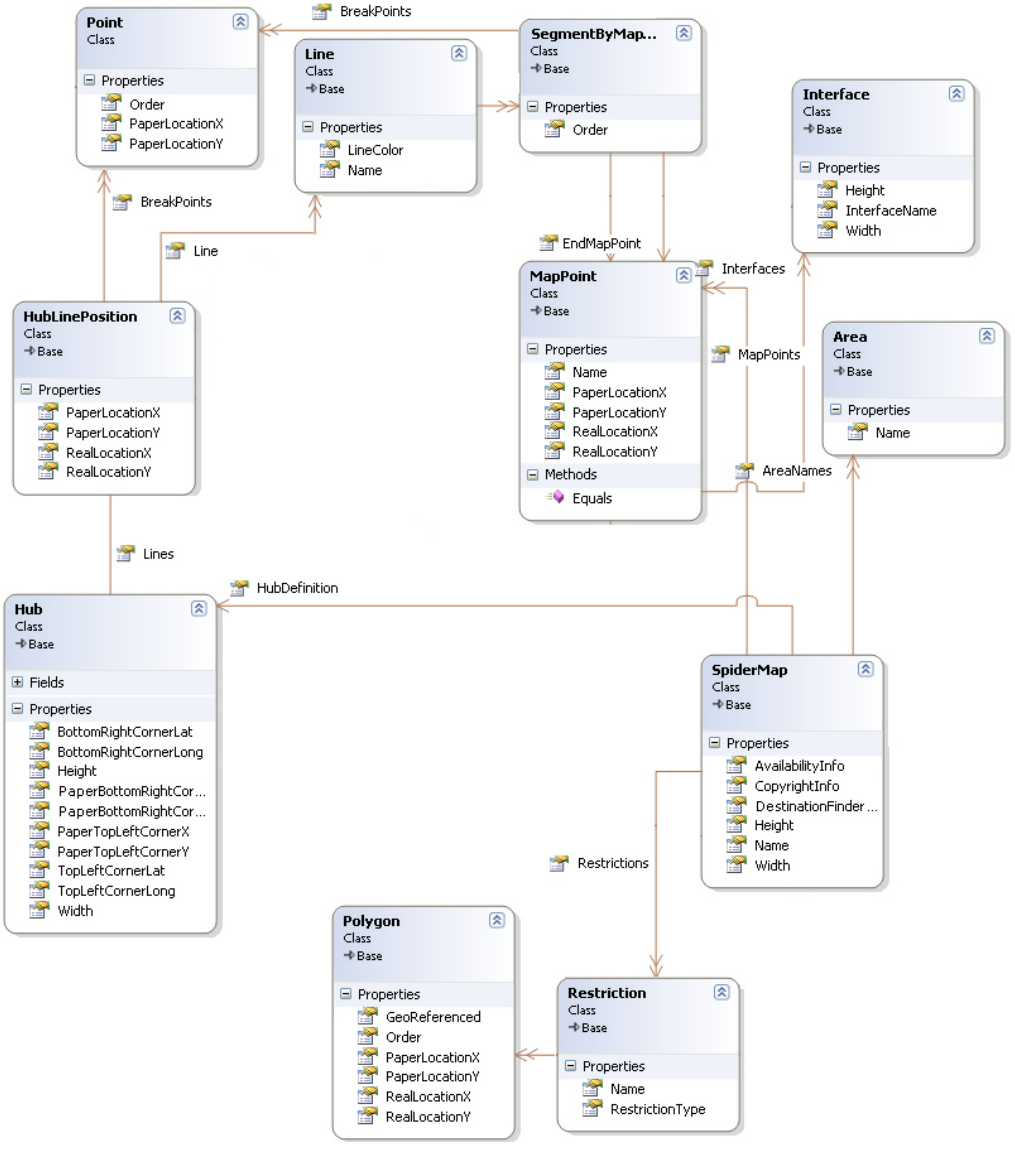

FIGURE 4.4: UML Class Diagram that implements the Semantically Rich Data Structure

The UML Class Diagram depicted in figure 4.5 implements the simple graph data and the semantic mappings structures. The simple graph data structure is implemented by the Graph, Edge and Vertex classes. The SimplePoint class supports the vertex processing.

The information about each restriction is stored in the restriction and vertex class. The classes that store the semantic information mappings are the Mapping class (each Graph contains a "MappingSet" property which is a set of "mapping" objects (that are of the type "Mapping" class). Other relevant classes such as the AngleSet and the Settings store information about the derived angles. The VertexMove class is related to the processing of the simple graph throughout the algorithm.

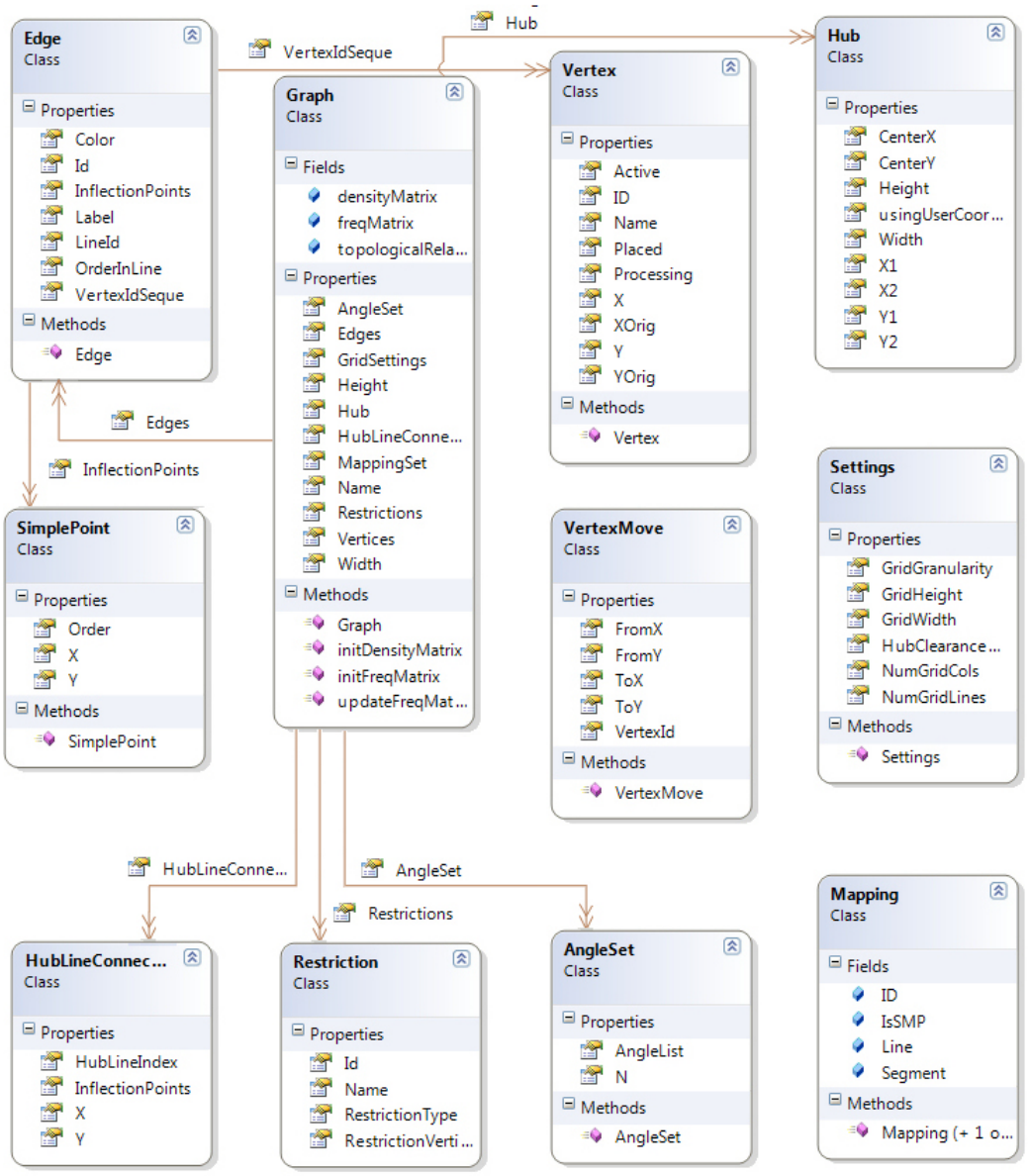

FIGURE 4.5: UML Class Diagram that implements the Simple Graph Data Structure.

The conversion from the semantically rich data model to the simple graph data structure is performed through the following actions:

- The points where the transportation lines connect to the hub that are stored

through instances of "HublinePosition" class are converted to instances of "HubLineConnection" classes

- The original map "Spider Map" class is converted to the Graph class
- The geographic constraints stored through the initial "Restriction" and "Polygon" classes are merged and converted to "Restriction" classes.
- The "MapPoint" class objects, initially used for storing the transportation network stops, are converted to instances of the "Vertex" class
- The "Line" class used to store line information is converted to the "Edge" class.
- The "Simplepoint" class on the simple graph data structure will store points that are inside the hub and other points that may be placed in the edges to achieve better visual quality throughout the map processing

4.4 Modeling the Multicriteria Optimization Problem

We modeled the problem of the automatic generation of spider maps as a multicriteria optimization problem. We have a minimization objective function and a set of constraints we need to respect. The generation of a Spider Map involves the inclusion of many - and often conflicting - set of design guidelines. Those design guidelines were modeled through two sets of constraints: soft and hard constraints. The soft constraints measure and influence the visual *quality* of a spider map and it is desired they are respected as most as possible, while the hard constraints must be respected and enforced in order to generate a spider map: if they are violated, the produced solution is not feasible.

4.4.1 Decision Variables

The decision variables of this problem correspond to the spatial coordinates of each vertex $v \in V$ and point $p \in P$. Starting with precise geographic coordinates, the goal of the objective function will be to position every vertex and point (and consequently, all of the Spider Map structures that rely on them).

4.4.2 The Objective Function

We modeled most of the soft constraints based on an improved version of Stott's [26] work. We improved his model both on the formulation and on its implementation. The

goal is to minimize the objective function, which means that if all the soft constraints are perfectly respected, the optimization function would have a value of zero, which would correspond to the "perfect map". Soft constraints are related to the *quality* of the spider map, and allow us to evaluate the *niceness* of the spider maps [87]. It is worth to mention that although not categorized exactly as "soft constraints", some of these soft constraints have their roots in previous research works [26], [87], [63] and [60]. Soft constraints represent desirable map characteristics, although they are not critical to achieve a spider map, a feasible solution to the problem. At table 4.4 we enlist the set of soft constraints that comprise the optimization function.

TABLE 4.4: Soft Constraints - summary

Constraint	Description
SC1	Adjacent edge angle shall be as wide as possible for each vertex
SC2	Homogeneous inter-vertex spacing
SC3	All consecutive vertices shall dist a certain distance
SC4	Reduce edge crossings to the minimum
SC5	Lines should be as straight as possible
SC6	Benefit horizontal and vertical edges

Constraints SC1 to SC3 use the same modeling as Stott's work, while SC4 to SC6 are a completely different modeling, much improved through innovative algorithms. All constraints were improved in its implementation to improve execution speed, which is one of the downfalls of Stott's work[93]. The modeling of each constraint is as follows:

- **SC1**: Adjacent edge angle shall be as wide as possible for each vertex. This makes all the adjacent edge angles uniform (figure 4.6). Regarding this soft constraint we used the following mathematical formula:

$$SC1_{score} = \sum_{v \in V} \sum_{\{e_1, e_3\} \in E_v} |\frac{2\pi}{\rho(v)} - \theta(e_1, e_3)|$$

  where $\rho(v)$ here is the vertex degree, while $\theta(e_1, e_3)$ is the angle between to adjacent edges e_1 and e_3 incident to v.

- **SC2**: Homogeneous inter-vertex spacing. All vertices should be at equal distance from their predecessor and successor in a transportation line (figure 4.7). Regarding SC2, we used the following mathematical formula:

$$SC2_{score} = \sum_{e \in E} |\frac{|e|}{l * g} - 1|$$

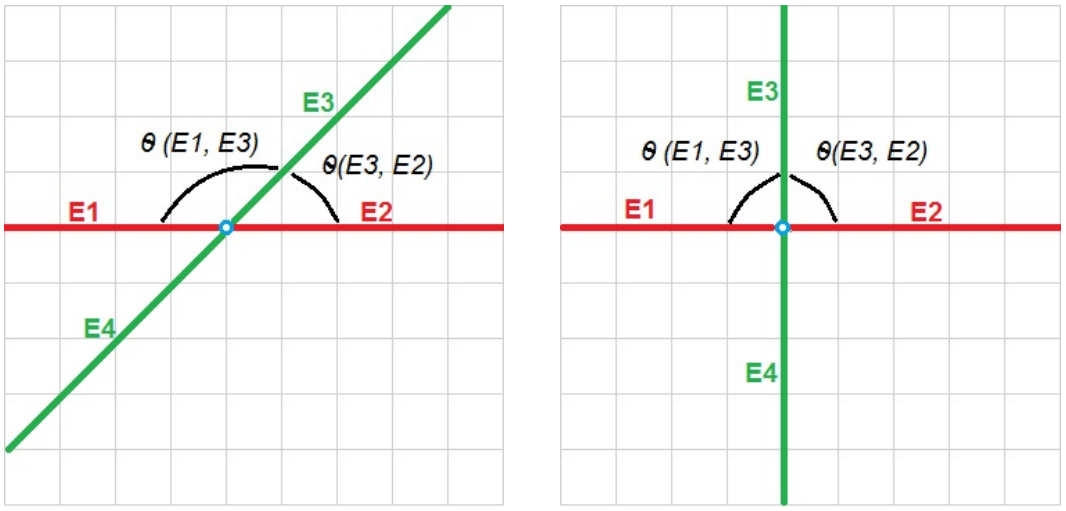

FIGURE 4.6: SC1 Constraint: the map on the left yields a worse score than the one on the right

Where l is the ideal grid length (as user defined parameter) and g is the grid granularity (grid aperture size). The best grid aperture size is automatically calculated by our algorithm, which is a remarkable enhancement regarding to Stott's work, where user needs to try several grid apertures to see which one fits better to each specific map.

- **SC3**: All consecutive vertices shall dist a certain distance (user definable parameter)(figure 4.7). Regarding SC3, we used the following mathematical formula:

$$SC3_{score} = \sum_{v \in V, \rho(v)=2} ||e_1| - |e_2||$$

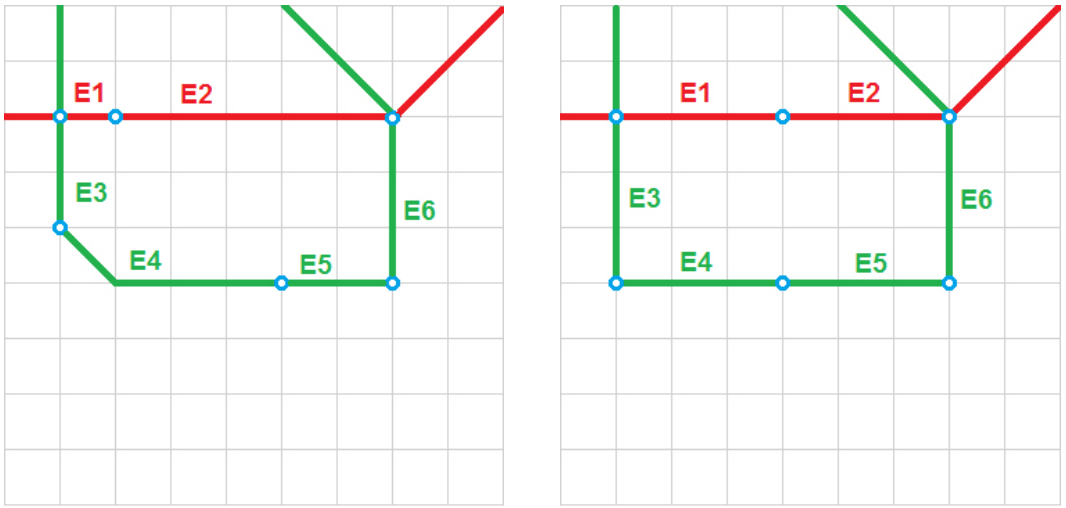

FIGURE 4.7: An example of the combination of the SC2 and SC3 Constraints. The map on the left presents a non-uniform inter-vertex spacing, while the map on the right presents uniform inter-vertex spacing and a distance of 3 grid units between vertices.

- **SC4**: Reduce edge crossings to the minimum (figure 4.8). As an innovative approach in comparison with Stott's work and current literature, we developed an enhanced version of the Bentley-Ottmann algorithm to get the number of edge crossings:

$$SC4_{score} = k$$

where k is the number of edge crossings on the map.

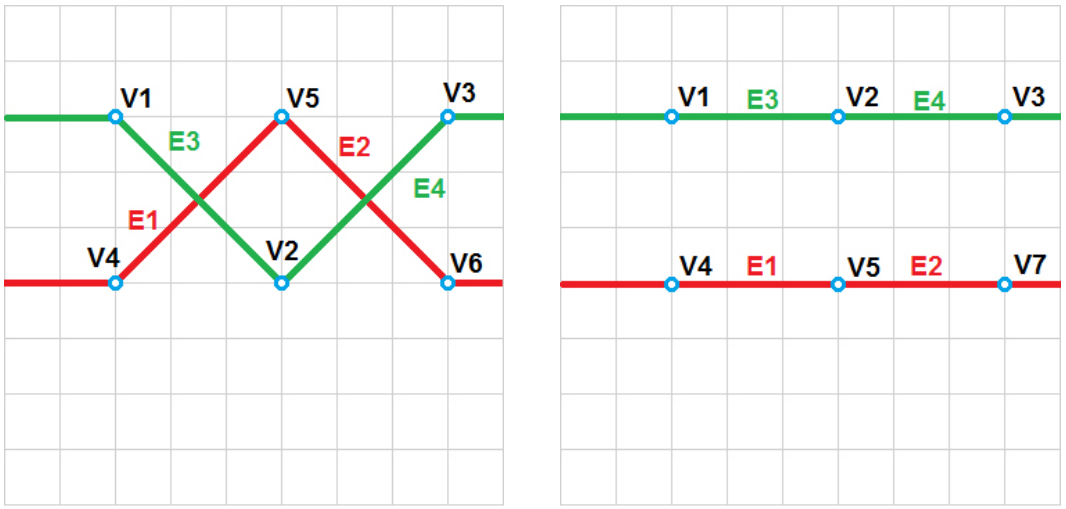

FIGURE 4.8: SC4 Constraint: the map on the left has two segment crossings while the map on the right has zero, yielding a better score at this constraint.

Regarding SC4, we did not use Stott's model as it was based on a brute-force naive approach that tests every pairs of edges for crossings, with $O(e^2)$ [93]. Instead, we developed an improved version of the Bentley-Ottmann algorithm [141] (also known as the "sweepline algorithm") which runs in $O((e + k) \log n)$ time, with e being the number of edges and k the number of edge crossings. The main idea of the Bentley–Ottmann algorithm is to use a sweepline, which is a vertical line L (figure 4.9) moving from left to right across the plane. As the line moves, it intersects the edge segments in sequence[142]. L will always intersect the input line segments in a set of points whose vertical ordering changes only at a finite set of discrete events. Thus, the continuous motion of L can be broken down into a finite sequence of steps, and simulated by an algorithm that runs in a finite amount of time. There are two types of event that may happen during the course of this simulation. When L sweeps across an endpoint of a line segment s, the intersection of L with s is added to or removed from the vertically ordered set of intersection points. These events are easy to predict, as the endpoints are known already from the input to the algorithm. The remaining events occur when L sweeps across a crossing between two line segments s and t. These events may also

be predicted from the fact that, just prior to the event, the points of intersection of L with s and t are adjacent in the vertical ordering of the intersection points. The Bentley–Ottman algorithm itself maintains data structures representing the current vertical ordering of the intersection points of the sweep line with the input line segments, and a collection of potential future events formed by adjacent pairs of intersection points. It processes each event in turn, updating its data structures to represent the new set of intersection points.

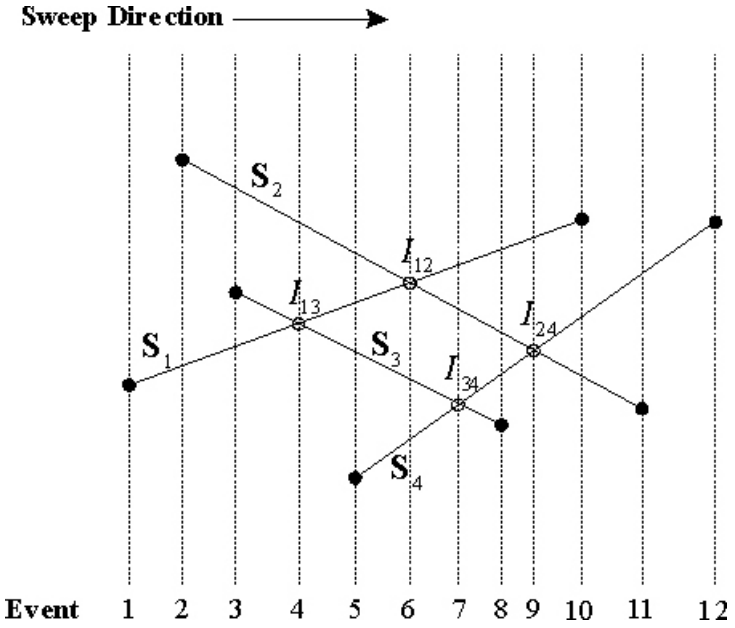

FIGURE 4.9: Visual Example of the Bentley-Ottmann Algorithm[42]. S are Edges

The original Bentley-Ottmann algorithm has some limitations[143][144] and to be successfully executed it assumes that:

1. No two edge endpoints or crossings can have the same x-coordinate
2. No edge endpoint lies upon another line segment
3. No three edges intersect at a single point.

These assumptions are not reasonable for our current problem as it is common to find examples that violate one or more of these assumptions. In those cases we found that this algorithm enters in an infinite looping state, thus not supporting the complexity of transportation networks. Therefore, we developed an improved version of the Bentley-Ottmann algorithm to support complex graphs used in transportation networks. Pseudo-algorithm 1 shows our enhanced version. The enhancements include:

– The consideration that events may consist of the crossing of two or more lines

  - The reversion of the incident segments when an event point is reached (not just swapped as the original algorithm states, as there may be more than two segments)
  - After a crossing is handled, we consider the hypothesis that there may be more than two old event points to be removed or more that two new event points to be inserted.

- **SC5**: Lines should be as straight as possible (Line Straightness) (figure 4.10). It is possible to observe in the figure that the number of inflection points of each edge is reduced and the visual "staircase effect" is eliminated. Schematization through simplification of lines is used to reduce visual complexity and entropy. [27]. Regarding SC5, we used the following following mathematical expression:

$$SC5_{score} = \sum_{v \in V} \left(\sum_{\{e_1, e_2\} \in E_v} \theta(e_1, e_2) \right)$$

Where $\theta(e_1, e_2)$ is the smaller angle between adjacent edges e_1 and e_2.

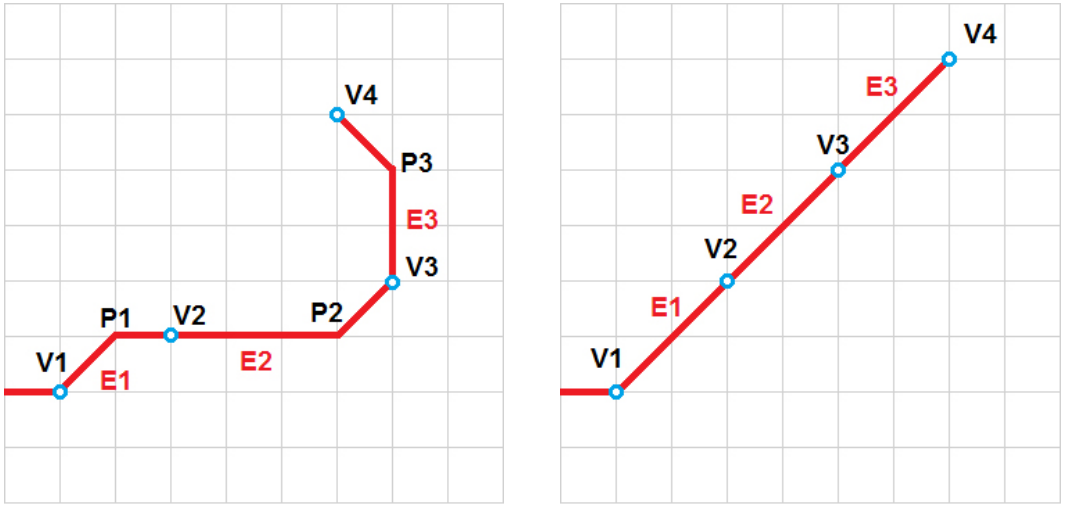

FIGURE 4.10: SC5 Constraint: The map on the right yields a better score at this constraint than the map on the left.

- **SC6**: Benefit horizontal and vertical edges in a higher magnitude, and then favor oblique edges (all angles shall be multiple of 45 degrees) Regarding SC6, we used the following following mathematical formula:

$$SC6_{score} = \sum_{e \in E} \left| \sin 4 \left(\tan^{-1} \frac{y(e_s - y(e_f)}{x(e_s - x(e_f)} \right) \right|$$

Where $\theta(e_1, e_2)$ is the smaller angle between adjacent edges e_1 and e_2.

Algorithm 1 Enhanced Bentley-Ottmann Pseudo-Algorithm

```
1: procedure BentleyOttmanFixed
2:     Event queue EQ ← all edge endpoints/vertices;
3:     Sort EQ by increasing x and y;
4:     Sweep Line SL ← empty; output intersection list IL to be empty;
5:     while EQ is nonempty do
6:         Let E = the next event from EQ;
7:         if E ∉ IL then
8:             if E is a left endpoint then
9:                 Let segE = E's segment; Add segE to SL;
10:                Let segA = the segment Above segE in SL;
11:                Let segB = the segment Below segE in SL;
12:                if I = intersect(segE with segA) exists) then Insert I into EQ;
13:                end if
14:                if I = intersect(segE with segB) exists) then
15:                    Insert I into EQ;
16:                end if
17:            end if
18:            if E is a right endpoint then
19:                Let segE = E's segment; Add segE to SL;
20:                Let segA = the segment Above segE in SL;
21:                Let segB = the segment Below segE in SL;
22:                Delete SegE from SL;
23:                if I = intersect(segE with segA) exists) then
24:                    if I is not in EQ already then Insert I into EQ;
25:                    end if
26:                end if
27:            end if
28:            if E is an intersection Event then
29:                Add E's intersect point to the output list IL;
30:                Let segE1 above segE2 be E's intersecting segments in SL;
31:                Swap their positions so that segE2 is now above segE1;
32:                Let segA = the segment above segE2 in SL;
33:                Let segB = the segment below segE1 in SL;
34:                if I = intersect(segE2 with segA) exists) then
35:                    if I is not in EQ already then Insert I into EQ;
36:                    end if
37:                end if
38:                if I = intersect(segE1 with segB) exists) then
39:                    if I is not in EQ already then
40:                        Insert I into EQ;
41:                    end if
42:                end if
43:            end if
44:        end if
45:        Remove E from EQ;
46:    end while
47: end procedure
```

SC1, SC5 and SC6 constraints can be easily understood: the less line bends, the cleaner and less cluttered the map presentation will be. SC2 and SC3 have the objective to turn the vertex placement and edge length the most homogeneous possible. SC6 constant is related to the findings described by several art studies that show that human brain evaluates horizontal lines as the most pleasant, followed by vertical and oblique lines [69]. All these soft constraint components are part of a weighted sum which makes up the final evaluation function. Therefore, the final objective Function is given by the following expression:

$$Min\ F = \sum_{i=1}^{6} (W_{SC_i} * SCi_{score})$$

Where W_{SCi} is the weight given to each of the soft constraints i. W_{SCi} are user defined parameters.

4.4.3 Constraints

The automated generation of spider maps must comply with a set of constraints that preserve their main properties. We divided those constraints into three sets of constraints: hard constraints, mixed constraints and geographical constraints. Any violation to one of these constraints will produce an unfeasible solution. The process that assures these constraints are respected is explained in the next chapter.

4.4.3.1 Hard Constraints

Hard constraints are related to critical aspects of a spider map. This means that if a hard constraint is violated, the solution produced is not feasible. The set of relevant hard constraints is summarized in table 4.5.

Table 4.5: Hard Constraints - summary

Constraint	Description
HC1	Layout must be octilinear
HC2	Avoid forbidden areas
HC3	Avoid Occlusions
HC4	Maximum vertex displacement range

The constraints are explained as follows:

- **HC1**: All vertices must respect the octilinear embedding. Vertices should be placed on the predefined grid intersections (discretization of space) (figure 4.11).

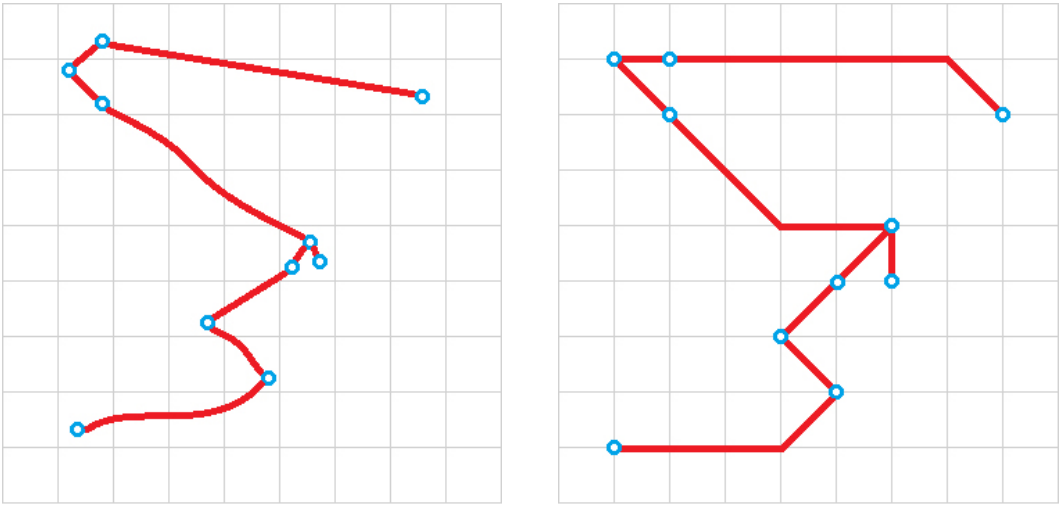

FIGURE 4.11: HC1 Constraint: the figure on the right respects the HC1 constraint while the left one does not.

- **HC2**: Avoid forbidden area: we assure the transportation map elements are not on an area in which they are not supposed to be, i.e: the hub, outside the map canvas limits, or above geographical accidents (figure 4.12).

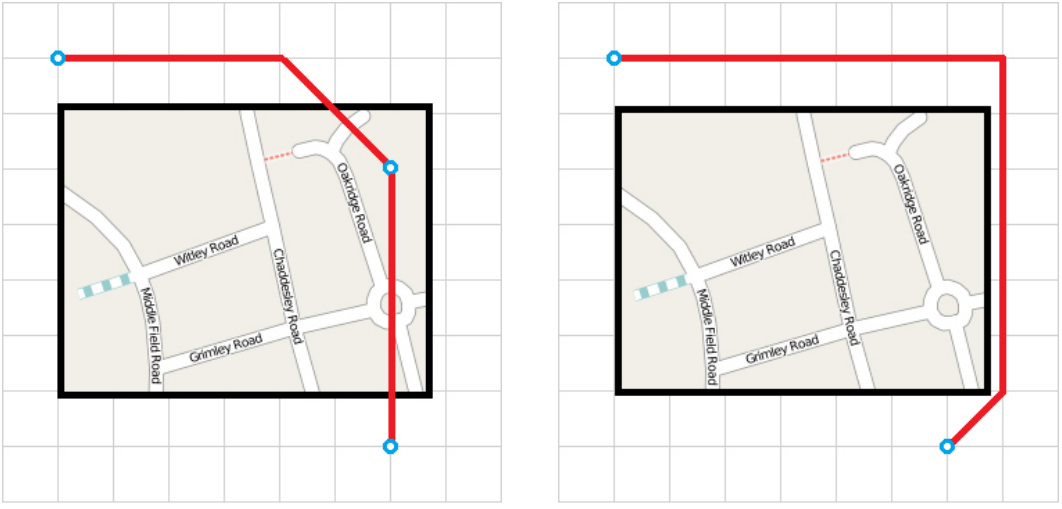

FIGURE 4.12: HC2 Constraint: the figure on the left shows a vertex occluding the hub (thus violating the HC2 constraint). The figure on the right complies with the HC2 constraint.

- **HC3**: Avoid Occlusion: different vertices shall not be put on the same grid intersection (avoid occlusion) (figure 4.13).

- **HC4**: Maximum displacement: all vertices must be within a certain (user definable parameter) range from its original geographic location (figure 4.14).

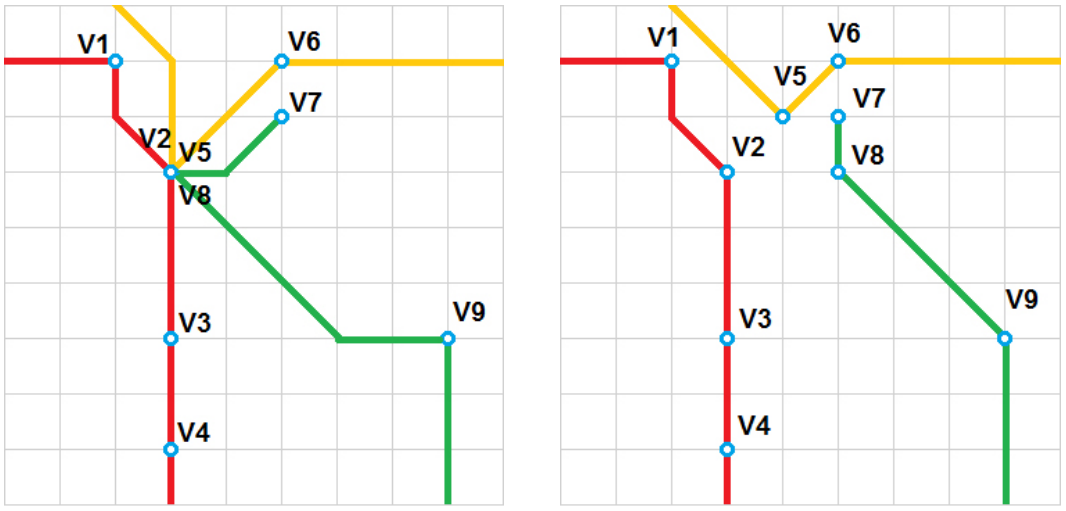

FIGURE 4.13: HC3 Constraint: the figure on the left three vertices (V2, V5 and V8) placed at the same grid intersection, occluding each other and violating the HC3 constraint. The figure at the right complies with this constraint.

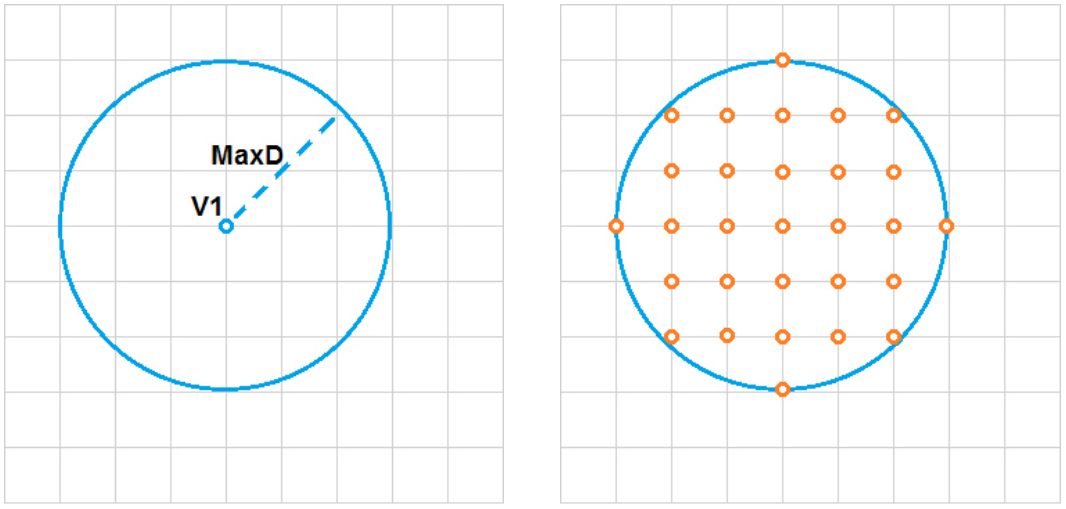

FIGURE 4.14: HC4 Constraint: the user definable parameter *Max_Displacement* defines the area (left figure) to where the vertex can be displaced from its original placement, through all the algorithm (right figure).

4.4.3.2 Hybrid Constraints - Topological Relation Preservation

Not every constraint must be exclusively modeled as a soft constraint in the objective function or as a hard constraint. The example of this is the topological relations preservation. The preservation of topological relations is of fundamental importance [88], and to achieve it the initial relative position of vertices must be kept, i.e. the binary relations "north of" ("No"), "south of" ("So"), "west of" ("Wo"), "east of" ("Eo") of the geographically correct initial transportation map shall be preserved throughout the generation of the spider map. Although Stott's work [26] makes a brief mention to the topological relations, it does not provide any model.

We modeled the topological relations preservation as a hybrid constraint that can either be treated as a "soft" or "hard" constraint. If we enable it as a soft constraint, violations may occur, although this relaxation yields a penalty to the objective function score. The penalty is proportional to the ratio of the topological relations that are violated. This means that treating topological relations as a soft constraint makes them desirable but not obligatory. This is useful in some cases where some degree of topological relation violation is necessary in order to achieve better visual layout, though it can create user disorientation.

If the topological relations are treated as a hard constraint, they cannot be violated, although there is some degree of freedom: they can change if they are not reversed in any of the components. This means that if a vertex is "North of" and "West of" another particular vertex at the original map, a solution that would have it placed just "North of" would still be a feasible solution. The same would not happen if the relation is reversed to "South of" and/or "East of". This example is illustrated at figure 4.15.

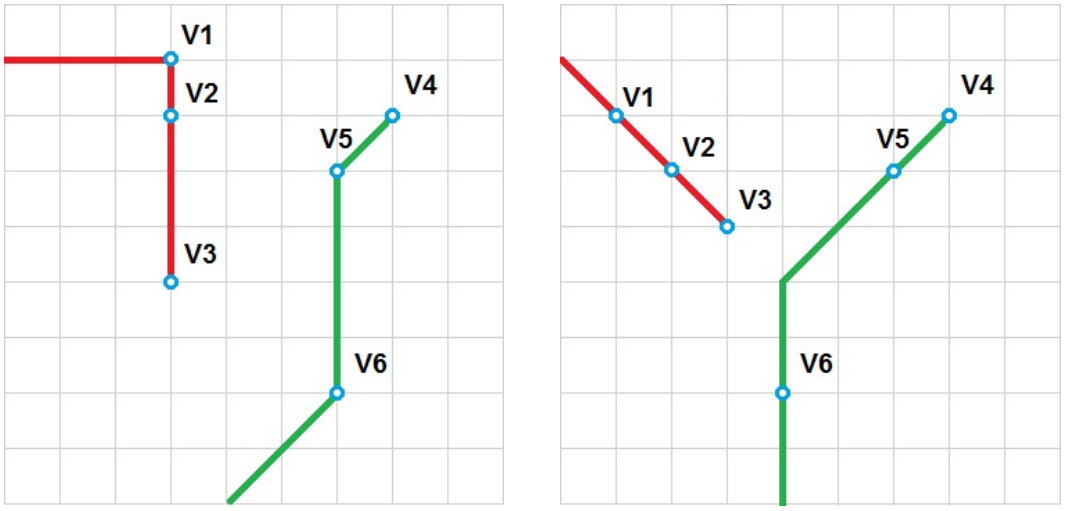

FIGURE 4.15: Topological Relations Preservation Constraint example: the map at the right still respects the hard version, considering the hypothetical original map (left)

As it is possible to see in tables 4.6 and 4.7 the topological relations that changed (shown in italic at table 4.7 do not invert any of the original topological relations thus complying with the hard topological relation preservation constraint.

If there is any inversion of the topological relations, as it is exemplified in figure 4.16, the solution would not considered feasible.

It is possible to observe in tables 4.8 and 4.9 the topological relations that changed. The ones shown in bold at table 4.7 invert the original topological relations.

The hard constraint version can also be modeled to enforce a strict topological relations preservation by not allowing any change on the topological relations throughout the

TABLE 4.6: Topologic Relations of the example map depicted in figure 4.15 (left).

Vertices	**V1**	**V2**	**V3**	**V4**	**V5**	**V6**
V1	-	No	No	NoWo	NoWo	NoWo
V2	So	-	No	Wo	NoWo	NoWo
V3	So	So	-	SoWo	SoWo	NoWo
V4	SoEo	Eo	NoEo	-	NoEo	NoEo
V5	SoEo	SoEo	NoEo	SoWo	-	No
V6	SoEo	SoEo	SoEo	SoWo	So	-

TABLE 4.7: Topologic Relations of the example map depicted in figure 4.15 (right). The relations that changed are in italic.

Vertices	**V1**	**V2**	**V3**	**V4**	**V5**	**V6**
V1	-	*NoWo*	No	*Wo*	NoWo	NoWo
V2	*SoEo*	-	*NoWo*	*SoWo*	*Wo*	NoWo
V3	So	*SoEo*	-	SoWo	SoWo	NoWo
V4	*Eo*	*NoEo*	NoWo	-	NoEo	NoEo
V5	SoEo	*Eo*	NoEo	SoWo	-	*NoEo*
V6	SoEo	SoEo	SoEo	SoWo	*SoWo*	-

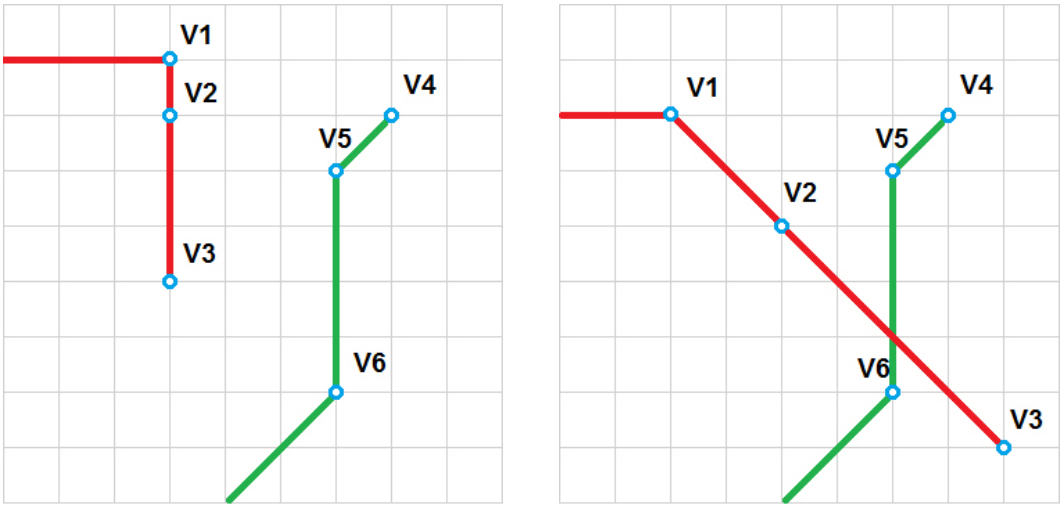

FIGURE 4.16: Topological Relations Preservation Constraint example: the map at the right does not respect the hard version, considering the hypothetical original map (left)

TABLE 4.8: Topologic Relations of the example map depicted in figure 4.16 (left).

Vertices	**V1**	**V2**	**V3**	**V4**	**V5**	**V6**
V1	-	No	No	NoLo	NoLo	NoLo
V2	So	-	No	Lo	NoLo	NoLo
V3	So	So	-	SoLo	SoLo	NoLo
V4	SoRo	Ro	NoRo	-	NoRo	NoRo
V5	SoRo	SoRo	NoRo	SoLo	-	No
V6	SoRo	SoRo	SoRo	SoLo	So	-

TABLE 4.9: Topologic Relations of the example map depicted in figure 4.16 (right). Italic formatting indicates a topological relation violation without inversion of relation while bold formatting indicates a a topological relation violation with inversion, regarding the map in figure 4.16 (right).

Vertices	**V1**	**V2**	**V3**	**V4**	**V5**	**V6**
V1	-	*NoLo*	*NoLo*	*Lo*	NoLo	NoLo
V2	*SoRo*	-	*NoLo*	*SoLo*	SoLo	NoLo
V3	*SoRo*	*SoRo*	-	**SoRo**	**SoRo**	**SoRo**
V4	*Ro*	*NoRo*	**NoLo**	-	NoRo	NoRo
V5	SoRo	NoRo	**NoLo**	SoLo	-	No
V6	SoRo	SoRo	**NoLo**	SoLo	So	-

generation of the spider map (even the ones that do not reverse the original topological relations).

4.4.3.3 Geographical Constraints

Respecting Geographical constraints is a complete innovative breakthrough as they do not exist in literature and past research. They allow us to deal with geographical constraints such as rivers, parks, the ocean, or other features that may condition the visual layout of the map. To our best knowledge, it is the first time they are described and modeled into a schematization optimization algorithm.

- **GC1**: The position and geometry of the geographical accident can not be changed throughout the generation of the spider map. As a consequence geographical accident polygons do not need to comply with the octilinearity criteria. This happens because it is not relevant to the transportation network topology that external features such as geographical constraints have their polygonal chain points on the top of grid intersections. In fact, they would occupy available positions for placing vertices, and, depending on the grid aperture, their shape could be severely distorted, making them hard to identify for the user.

- **GC2**: Graph edges or vertices must not occlude any area of the geographical constraint polygons. Geographical constraints shall be respected, as it would be nonsense to place a vertex on a river or ocean (figure 4.17). Sometimes, complying with this constraint may imply disrespecting any of the hard constraints. To avoid this problem, a differential grid aperture size (with finer grid near the geographical constraint polygon, for higher grid resolution) can be introduced. Figure 4.17 (left) shows a river and a vertex and an edge belonging to the transportation network occluding the river. A vertex is even located on the river. This is not plausible

in real world, and according to the geographical constraint GC2 it would not be a feasible solution. The figure on the right shows a feasible solution where the path between two vertices in opposite sides of the river is drawn by finding a suitable path between the rivers. To achieve this and simultaneously comply with the hard constraints, a higher resolution grid was used to allow the path to cross the river through the available space. The use of different resolution grids (differential grid aperture sizing) is used to overcome limitations on coarser grids whenever needed to comply with CG2. Figure 4.18 shows an example where a differential grid aperture is used.

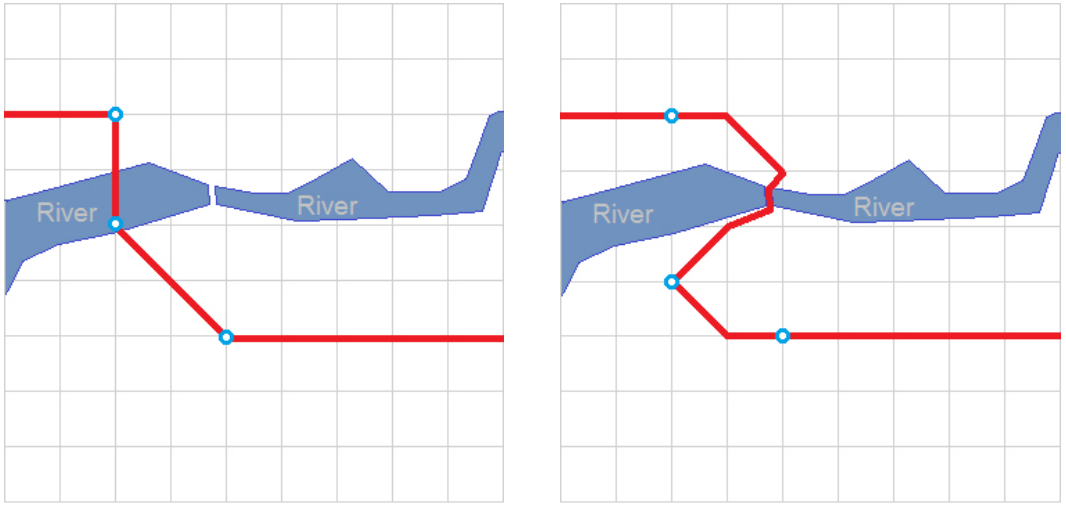

FIGURE 4.17: GC2 Constraint: An example of the violation of this constraint (left) and an example of compliance of this constraint (right).

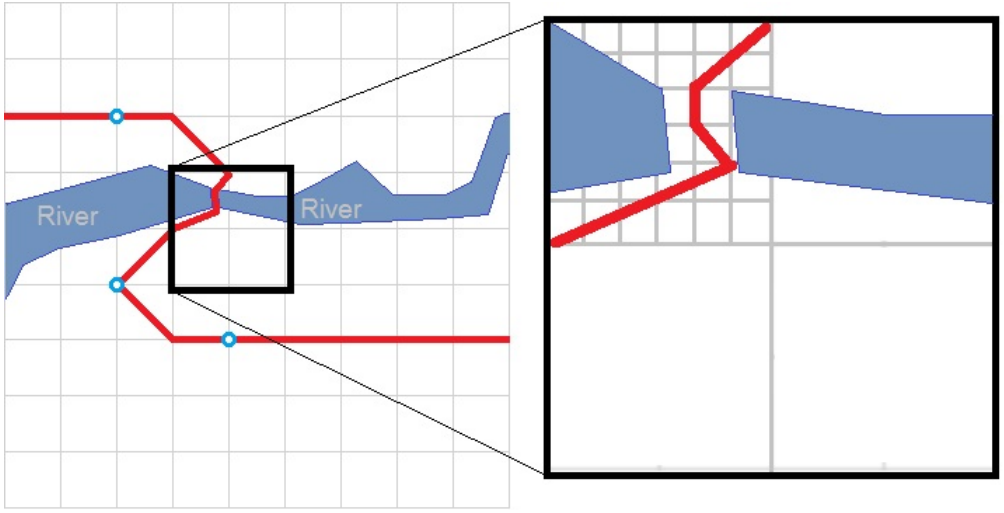

FIGURE 4.18: Example of the use of differential grid aperture - magnification of figure 4.17. The magnification of the river cross shows an increased grid aperture to allow river crossing, while keeping the grid embedding.

4.5 The Tabu Search Algorithm

The automatic generation of Spider maps involves the use of efficient optimization algorithms at the optimization phase to produce the *best* map satisfying all the criteria mentioned in subsection 4.4. Tabu search was chosen as the backbone of our information system as it fits the enunciated model and has the following advantages:

- It is considered to be a fast metaheuristic
- It has excellent local minimum escaping ability due to the use of the tabu lists
- It usually does not start with a random solution, which is quite adequate to our problem
- It always generates a solution
- It is presents an iterative improvement process which can be stopped at any time
- Its structure is well suited for mapping constraints and optimization functions
- The tabu search refinements presented in the literature have the potential to be applied to our problem

Glover published the original Tabu Search heuristic method more than twenty six years ago [145] [146] and since then several researchers have been applying it to solve operations research problems. Tabu Search has become a popular method in finding good solutions in large combinatorial problems. Even if the obtained solutions are not optimal, they could provide good approaches to the optimal solution. Built around the idea of allowing non improving moves while using memory to avoid cycling back to previously visited solutions whenever a local optimum is found or to pursue intensification/diversification strategies, tabu search has found success in many applications either used as a standalone algorithm or in conjunction with other heuristics [147]. Tabu search relies on two main concepts: adaptive memory and responsive exploration [148]. While adaptative memory is concerned with searching the solution space effectively, responsive exploration is related to intelligent enumeration: a bad strategic choice can yield more information than a good random choice (as other methods such as GRASP, genetic algorithms do, for example).

At an abstract high level point of view, Tabu Search starts as an usual local search strategy would, by iterating from one solution to another through a move chosen from the possible neighborhood moves until the defined termination criterion is met. As we take a deeper look into the Tabu Search, the differences become apparent.

An important first level consideration for Tabu Search is the choice of the move that takes us from one solution to another. A candidate list to restrict the possible moves in the neighborhood is chosen to achieve balancing between the quality of the moves and the effort to find it. If we assume we can guess the quality of each move, we can evaluate and pick intelligent moves that fit our problem. Otherwise we can select randomly, if the neighborhood is random.

Another important consideration is the use of memory, either short or long-term based. Short term memory is achieved through the use of a tabu list which stores a list of the recently chose moves, preventing them to be chosen again (tabu moves). Longer term memory is achieved by saving promising solutions or moves. The use of memory means that the neighborhood of a solution is "*not a static, but rather a set that can change according to the history of the search*" [148]. The memory use can be tailored to specific problems, either its short or long term use.

The use of short term memory can also be temporarily bypassed to allow tabu moves that may lead to unvisited solutions that may be attractive in certain situations. This bypass is performed through the aspiration criteria. Aspiration criteria are problem-dependent.

Memory in Tabu Search can be also used as a basis for the intensification/diversification balance: it can be used to know the presence or absence of certain elements in *good* solutions and aspiration criteria can be dynamically changed to reflect that knowledge (intensification). On the other hand, short term memory can be designed such as the neighborhood may include moves that are separated by a certain degree from other solutions visited previously, for example.[149] (diversification). Algorithm 2 summarizes the tabu search procedures.

Algorithm 2 Tabu Search generic algorithm

```
1:  procedure TabuSearch(a, b)
2:      InitialSolution ←solution
3:      while TerminationCondition ≠ true do
4:          CreateCandidateList(solu)
5:          ChooseBestCandidate()
6:          UpdateSolution()
7:          UpdateAspirationCriteria()
8:      end while
9:      return b
10: end procedure
```

From this generic algorithm it is worth to look with more detail to the ChooseBestCandidate() function, described in algorithm 3.

Tabu Search has seen many improvements and refinements to its original specification, many of them concerning with the dynamic change of memory or objective function

Algorithm 3 Choice of the Best Candidate Move in a generic Tabu Search

```
1: procedure ChooseBestCandidate(a, b)
2:     bestMove ← null
3:     while CandidateMoves.Count ≠ 0 do
4:         if IsTheBestTillNow(move) = true then
5:             if IsTabu(move) = false then
6:                 bestMove ← move
7:             else
8:                 if satisfiesAspirationCriteria(move) = true then
9:                     bestMove ← move
10:                else
11:                    discard(move)
12:                end if
13:            end if
14:        else
15:            discard(move)
16:        end if
17:    end while
18:    return BestMove
19: end procedure
```

definitions on run time to avoid local minima and with the intelligence of the search process. Nevertheless, different real world problems may require different approaches and those refinements are not applicable to every problem.

4.6 Conclusions

Throughout this chapter we presented an comprehensive model for the generation of spider maps, including the data structure model, the multicriteria nature of the problem and presented some solutions for improving the state of the art, such as an improved objective model and constraint modeling. The following chapter will explain how this model is implemented in a real information system.

Chapter 5

The Proposed Approach

This chapter describes the adopted process for the automated generation of Spider Maps. Based on the problem modeling described in the previous chapter, we developed an approach based on three phases:

1. **Pre-Processing:** to find a first feasible solution, an initial Spider Map
2. **Tabu Search Optimization:** improve the solution
3. **Post-Processing:** preparing the best obtained solution to be output

This approach is implemented through a a set of algorithms. The input is a XML file containing the description of the transportation network to serve as the base for the generation of the spider map. The XML file is parsed and loaded into a Semantically Rich data Structure (SRDS) that will be converted into a Simple Graph Data Structure (SGDS) and a mappings table. The pre-processing phase is then executed and the graph is aligned to a regular grid to discretize space and achieve an octilinear graph embedding. This step transforms the original map in a way that it becomes compliant with all the hard constraints. Being so, the pre-processing phase transforms the original transportation map into a feasible solution. This solution is then optimized through our tabu search algorithm until the stop conditions are met: the total number of iterations (user defined). The post-processing phase then takes place, by inserting vertices as needed on the graph in order to make it respect the geographic restrictions, and the SGDS, together with the mappings, are ready to be converted to the SRDS to produce the output map, which is an XML and SVG file, that can be published or further manually edited. This process is summarized in figure 5.1 shows this general description of our process of generation of a spider map through an UML activity diagram.

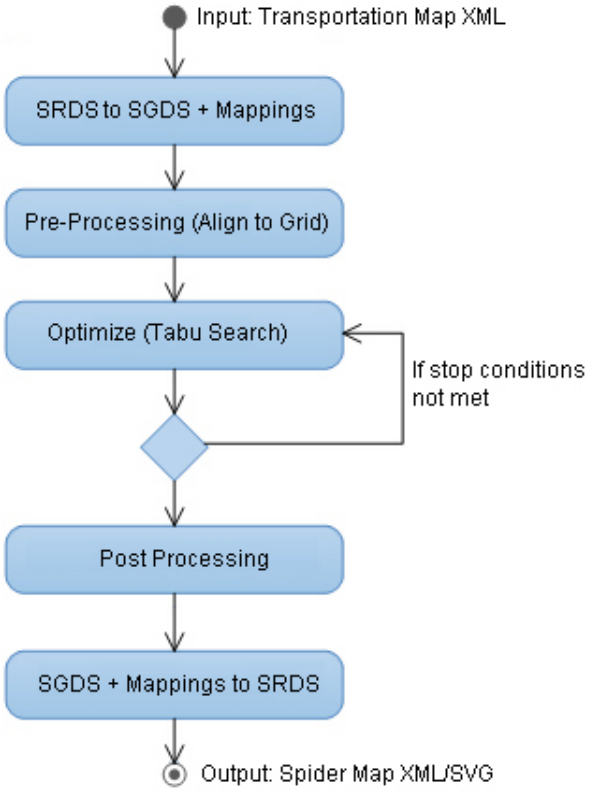

FIGURE 5.1: UML Activity Diagram depicting our process of automatically generating a Spider Map.

5.1 Initialization and Alignment to Grid

As shown in the first part of the pseudo-algorithm 4 the pre-processing phase starts by loading the XML file to a SRDS. This structure is then converted to a SGDS and mappings. The SGDS is the main data structure that supports all the map processing. The map is then rescaled for the Hub dimension defined by the requester of the spider map. With this transformation the map accommodates the Hub according to the input XML file and translates the vertices coordinates if needed, distorting the graph as needed, as shown in figure 5.2. The rescaling occurs by calculating the new position of the transportation network vertices considering the dimensions and position of the center of the hub. The rescaling has a visual effect of "compression" of the map outside the hub, with the vertices that were near the Hub implantation point at the original transportation map are displaced greater distances than the ones that are near the original map limits.

Then, the initial topological relation matrix of the graph (before further processing) is obtained and saved. This matrix will be used to assess, at each iteration in the tabu search optimization phase, the compliance with topological relations. The map is then aligned to a grid in order to discretize space and obtain an octilinear embedding. As the

Algorithm 4 Information System - Pre-processing Initialization and Alignment to Grid

```
1: procedure EXECUTE()
2:     TransportationMap ← XML File Parsed Data
3:     TransportationMap ← MiddleBD.ConvertToSRDS(SpiderMap)
4:     Graph ← TrueGraphTools.ConvertSRDSToGraph(SpiderMap)
5:     SetupGridAndUserHubDimension(Graph)
6:     InitialTopologicalRelationsMatrix ← getTopologicalRelationsMatrix(Graph)
7:     Graph ← alignToGridSmartFit(graph)
8:     Graph ← alignToGridHPPO(graph)
9:     Graph.initializeFreqMatrix();
```

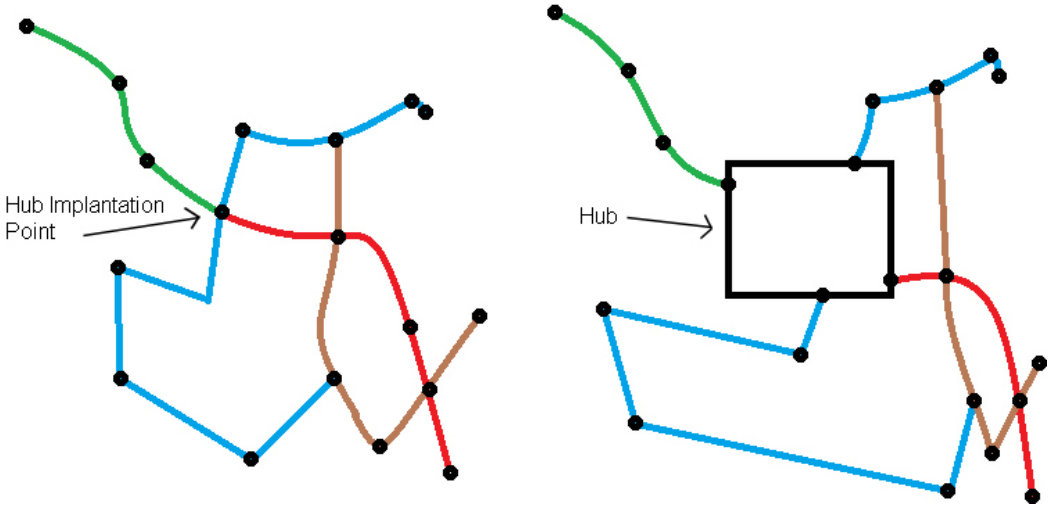

FIGURE 5.2: Before (left) and after (right) the rescaling in a given transportation map to accommodate the Hub.

original map features a graph embedded on an continuous space, the graph embedding on a regular grid limits the possible locations of the vertices. The placement of the vertices on the grid intersections achieved through the SmartFit algorithm, described in the next subsection.

5.1.1 The SmartFit Algorithm

In traditional procedures [99], the grid size is a parameter input by the user, when the graph vertices are placed on the grid intersections, if no occlusions between vertices occur. Otherwise, the user needs to chose a finer grid aperture to allow a higher resolution grid to support the alignment to the grid. In order to avoid this trial and error procedure we developed the SmartFit Algorithm that automatically determines what is the best grid aperture for each specific graph embedding. This algorithm combines a simulation based on the Newton–Raphson root finding method [150] with an algorithm we developed for embedding a graph into a grid avoiding vertex occlusion (collisions) while respecting all topological relations. The lower the resolution grid, the faster all

the subsequent algorithms will be processed (as there is less grid intersections to decide to where to move the vertices and points), the straighter the lines tend to be, but the probability of collisions increases substantially. Being so, it is very important to obtain the best grid aperture automatically. The main idea of the SmartFit algorithm is that the grid must adapt to the vertex density and placement (and not the other way around). Our algorithm automatically reduces the grid aperture to half if there are vertex occlusion problems until a grid with enough high resolution to accommodate all the vertices is found. On the other hand, if there are no problems with vertex occlusion, the algorithm tries to double the grid aperture. The upper bound and lower bound for the size of the grid will guide the process until the best grid aperture value is found. The SmartFit Algorithm is described at pseudo-algorithm 5.

Algorithm 5 SmartFit Algorithm

```
1: procedure SmartFit(Graph, LowerBound, UpperBound, GridGranularity)
2:     gridGranularity ← predefinedValue
3:     UpperBound ← predefinedValue
4:     LowerBound ← predefinedValue
5:     Result ← HPPO(graph, gridGranularity)
6:     if Result == Success then
7:         LowerBound ← GridGranularity
8:         GridGranularity ← gridGranularity * 2
9:         if GridGranularity <= 0.1 * Graph.Width OR GridGranularity >= 0.1 *
   Graph.Height then
10:            UpperBound ← LowerBound
11:        end if
12:    else
13:        UpperBound ← gridGranularity / 2
14:        gridGranularity ← gridGranularity / 2
15:    end if
16:    if LowerBound == UpperBound then
17:        return (Graph, GridGranularity)
18:    else
19:        return SmartFit(Graph, LowerBound, UpperBound, GridGranularity)
20:    end if
21: end procedure
```

The SmartFit algorithm starts with a predefined grid aperture size. If the vertices and points can be successfully placed by the HPPO function, we double the gridGranularity and call the SmartFit recursively with the new GridGranularity bounds. If they can not, then we divide by two the GridGranularity and call the SmartFit recursively with the new GridGranularity bounds. We avoid the GridGranularity to be more than 10% of the maps width or height, as a grid granularity bigger than 10% would cause the graph to be embedded on an disproportionately coarse grid, and this would disturb the overall

map visual presentation. At the end of these simulations, we obtain the best GridGranularity value which offers the best performance possible while being able to place all the vertices and points without occlusions, as exemplified in figure 5.3. This figure shows two examples of this algorithm. On the first example (left) the initial graph is embedded on a grid with grid granularity of 100 through the SmartFit algorithm. It is possible to see that some vertices are not successfully placed as V1 and V2 are contending for the same grid position. Therefore this embedding is not a feasible solution, and the SmartFit runs again, but this time the UpperBound is 50 (half of the GridGranularity value used at the first try), while the lower bound is yet to be found, thus the -1 value. The grid granularity will be 50. This time the graph can be successfully embedded, and consequently the lower bound value is set to 50. As both bounds match, the GridGranularity of 50 is the highest possible value for grid granularity, and the SmartFit algorithm ends. The second example (right) shows a simpler graph. The SmartFit tries to embed it on a grid with GridGranularity of 100, with success. This is the best feasible solution till now and consequently the LowerBound has been found with a value of 100, although the algorithm has not ended as the UpperBound has not yet been found. The SmartFit is executed again, this time with a GridGranularity of 200. The graph embedding is not successful, and so the UpperBound has been found, with a value corresponding to the GridGranularity of the best successful embedding till now, which is 100. As both bounds match, the best GridGranularity value is 100. In all cases the algorithm finishes when the upper and lower bound coincide, and that value is the ideal grid granularity. The process of placing all the vertices and points without occlusions is managed by the HPPO Algorithm, described in the next subsection.

5.1.2 The HPPO Algorithm

The HPPO[1] algorithm [54], is an algorithm responsible for performing the grid alignment and vertex occlusion management that is used by SmartFit to assess if a graph embedding with a particular grid granularity is possible in such a way that the result is a feasible solution (a Spider Map). This is needed by SmartFit in order to determine the best GridGranularity and to produce the first Spider Map that will be then optimized by the tabu search.

The HPPO algorithm manages the placement of the graph vertices on the grid intersections, respecting topological relations among them. It also assures that all conflicts from vertex contention (that arise from the discretization of space by transforming a continuous set of coordinates on a discrete grid graph embedding) are successfully solved through an intelligent interaction analysis between vertex placement and distribution

[1] Heuristic Point Placement Optimization

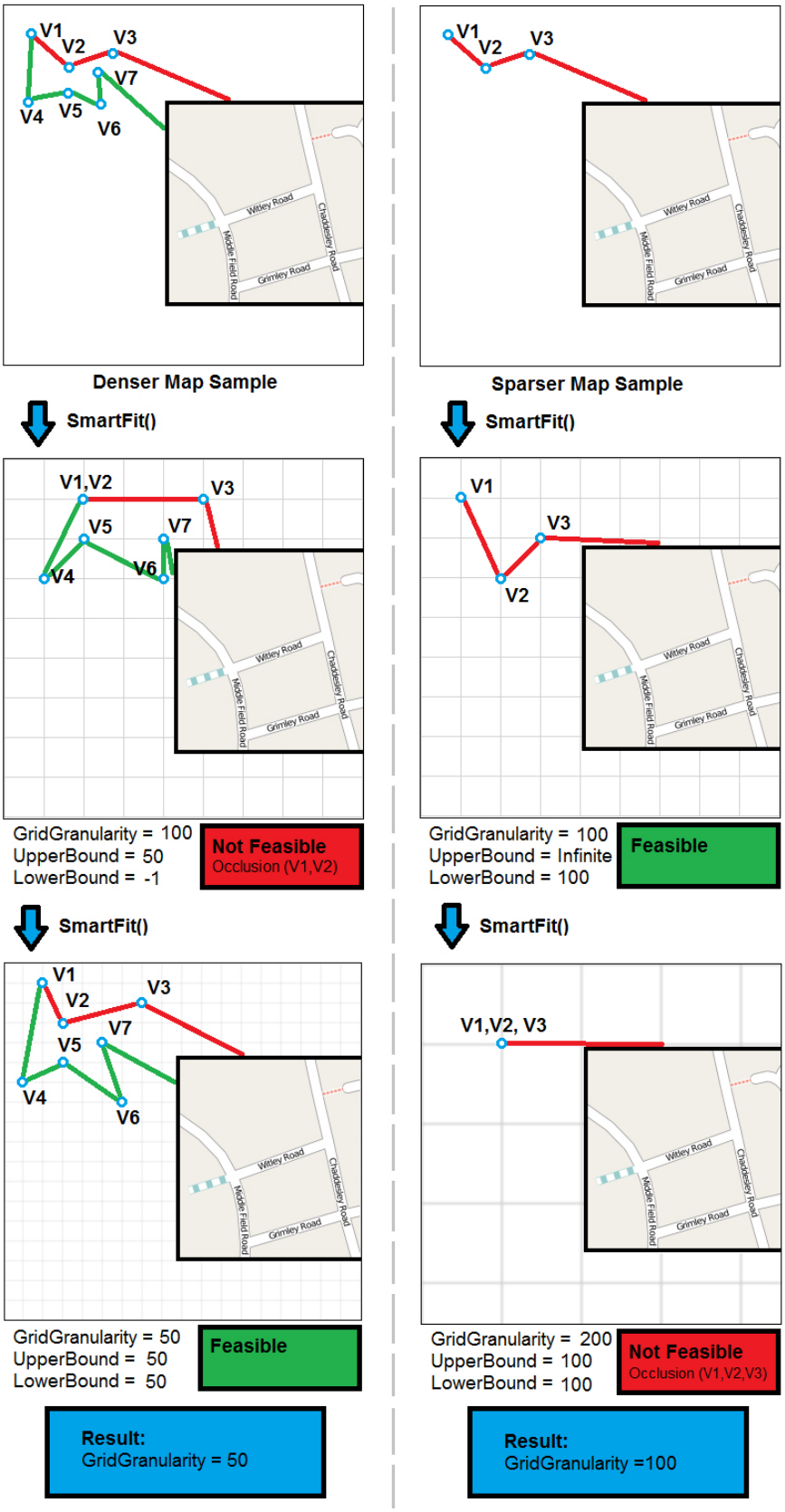

FIGURE 5.3: SmartFit Algorithm Examples

and grid aperture size. Although main HPPO function is to tell the SmartFit algorithm if a graph embedding is possible, internally it performs an intelligent vertex contention management in order to position every graph vertex on the grid in such a way that if it is mathematically possible to perform such positioning without violating any topological relation (eventually by displacing some vertices), it will tell to SmartFit that the embedding is possible. Being so, with SmartFit and HPPO a solution is always possible as the SmartFit algorithm always finds the best grid size for the map, even for very dense maps. The HPPO presents a significant improvement regarding the literature in what concerns to grid alignment [99] in which no contention management strategies are mentioned.

The algorithm, (see algorithm 6) starts by checking the topological relation matrix. Then it analyzes each vertex location (keep in mind that at this point they are not yet aligned to grid as the HPPO is called by the SmartFit algorithm), and calculates what would be the nearest grid intersection where the vertex should be moved to (discretization of space) according to figure 5.4.

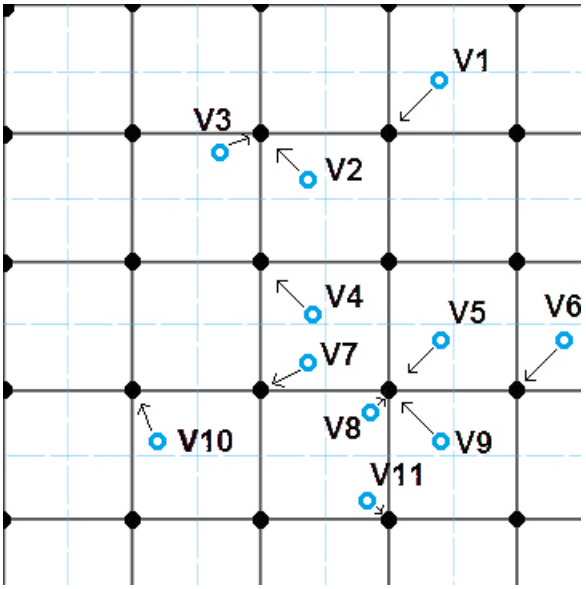

FIGURE 5.4: The grid intersections (black dots) and their area of vertex attraction. The adequate grid intersection for each vertex may cause contentions to arise.

As the figure shows, contentions may arise in denser areas. For example, contentions arise for vertex tuples $(V2, V3)$ and $(V5, V8, V9)$. The challenge is to determine if a set of vertices can be effectively placed on a grid even if contentions arise, by solving them while preserving topological relations. Of course that if we always increase the grid resolution we would ending up by solve this problem but at the expense of much higher processing times at the optimization phase as the potential search space increases. For non-uniform maps where there are high vertex density areas and very sparse areas (this happens on the very majority of maps) this increase of the grid resolution would

cause unnecessary soaring processing times throughout the rest of the automatic generation of spider maps. Therefore, it is fundamental that the HPPO algorithm tries to place all the vertices even if their displacement is needed in order to keep the initial topological relations. The displacement of a vertex a grid intersection is called a *move*. The displacement of a vertex will necessarily affect the displacement of the other vertices, as the topological relations between them must be kept, and if contentions arise, they need to be solved. This "domino effect" may be successfully executed if all the vertices that need displacement can be placed respecting the topological relations with no vertex occlusions, or may not if that is not possible by the current grid granularity and map features (dimensions, hub, geographic restrictions, vertex density). The function responsible for intelligently executing this "domino effect" is the recursive function called "recursionHPPO" (algorithms 7 and 8) that keeps being called whenever a vertex need to be displaced. This function succeeds by returning a list of vertex moves that a particular vertex move can induce, or may return a void move list if the vertex move cannot be executed. If a vertex fails to be placed on one position (either by that move failing by itself or by the chain of moves it causes failing on any point), it backtracks and tries to place the vertex on another position, respecting the topological relations. If it is not possible to backtrack, then the vertex placement has failed and the recursionHPPO returns an empty move list to the HPPO function, and consequently, HPPO returns an error to the SmartFit algorithm which called it, signaling that with that grid aperture size, it is not possible at all to place the vertex.

Algorithm 6 Global HPPO Algorithm

```
1:  procedure HPPO()
2:      CheckGraphMatrixes()
3:      for all v ∈ V do
4:          Move ← empty
5:          Move.vertexID ← v.vertexID
6:          Move.currentPosition ← v.currentPosition
7:          Move.destination ← getNearestGridIntersection(v)
8:          MoveList ← Empty
9:          MoveList ← recursionHPPO(Graph, Move)
10:         if MoveList == Empty then
11:             Throw new Exception("HPPO Failed")
12:         else
13:             for all VertexMove ∈ MoveList do
14:                 UpdateGraph(Graph, VertexMove)
15:             end for
16:         end if
17:     end for
18:     return Graph
19: end procedure
```

Algorithm 7 Recursive HPPO Function - Part 1

```
1: procedure recursionHPPO(Graph, Move)
2:     ResultMoveList ← Empty
3:     if isDestinationGridIntersectionEmpty(Move) then
4:         InPlacementVerticesIDs ← empty
5:         for all v ∈ V do
6:             if v.Placed OR v.Processing OR v.Active then
7:                 InPlacementVerticesIDs.add(v)
8:             end if
9:         end for
10:        if topologicalRelationCheck(Move, inPlacementVerticesIDs) then
11:            ResultMoveList.Add(Move)
12:            Return ResultMoveList
13:        else
14:            ProcessedLocationsMatrix[Move.destination] = true
15:        end if
16:    end if
17:    AlternativeMoveList ← empty
18:    AlternativeMoveList.Add(FindAlternativeMoves(Move))
19:    MovesToRemove ← empty
20:    for all ( doMove in AlternativeMoveList)
21:        if !checkForHardConstraints(Move) OR ProcessedLocationsMa-
   trix[Move.destination] then
22:            MovesToRemove.Add(Move)
23:        end if
24:    end for
25:    for all move in MovesToRemove do
26:        AlternativeMoveList.Remove(move)
27:    end for
28:    if AlternativeMoveList.Count == 0 then
29:        ProcessedLocationsMatrix[Move.destination] == false
30:        Return
31:    end if
32:    for all (Move in AlternativeMoveList) do
33:        if !CauseDisplacement(move) then
34:            RecursionResultMoves ← RecursionHPPO(Graph, Move)
35:            if RecursionResult != null) then
36:                return RecursionResultMoves
37:            else
38:                MovesToRemove.Add(Move)
39:            end if
40:        end if
41:    end for
42:    for all move in MovesToRemove do
43:        AlternativeMoveList.Remove(move)
44:    end for
```

Algorithm 8 Recursive HPPO Function - Part 2

```
45:     for all ( doMove in AlternativeMoveList)
46:         DisplacementDirection ← empty
47:         DisplacementDirection ← CalculateDirectionVector(Move.VertexID, Ver-
    tex[Move.destination])
48:         if (NeedsDisplacement(Move.VertexID, Vertex[Move.destination], Displace-
    mentDirection)) then
49:             RecursionResultMoves ← RecursionHPPO(Graph, Move)
50:         end if
51:         if RecursionResult != null) then
52:             return RecursionResultMoves
53:         end if
54:     end for
55:     return
56: end procedure
```

The HPPOrecursion starts by checking the destination grid intersection to where the vertex referred in the move shall be placed on. If the grid intersection is free (i.e. it has no other vertex on it), the algorithm checks if this move would violate any topological relations, in what concerns to the vertices that are already placed or in placement by other instances of the recursion HPPO function (this function could be able to distribute threads to different cores in multi-core processors to speed up process). Therefore, we must know the list of placed, being processed or active vertices. Having this list set, we can speed up the topological relations assessment as we will only compare the vertices that are necessary against each other, thus saving time. If there is no violation of topological relations, then, the move can be returned (and it will be executed by the HPPO function). If the move's destination is a grid intersection which already has a vertex there or there is any violation of topological relations, then, alternative destinations for the move's vertex are found (within the nearest grid intersections). From those alternative moves we will remove the ones that violate the hard constraints (and thus would not constitute feasible solutions) and those that already have vertices that are definitively placed. An empty list is returned if no alternatives remain. From the remaining alternatives, the algorithm tries first the ones that do not trigger further vertex displacements (corresponding to empty grid intersections), calling recursively the function recursionHPPO. If none of those alternatives is feasible, then the algorithm tries the alternatives that will imply vertex displacements. In this case, the algorithm calculates the displacement vector between the vertex that is originally being moved and the vertex that is at its grid intersection destination. It is important to know this vector to understand in which direction the vertex which is at the destination shall be displaced to allow the original vertex to replace it at its current location. This means we need to know the direction vector of the "domino effect", to avoid further topological relations violations. After knowing the direction vector, the RecursionHPPO

is recursively called to obtain a set of resulting "domino moves" regarding the vertices that need to be displaced, and they will be recursively returned to the HPPO function which will execute them. Each vertex of the map being placed can be in one of the three following situations:

- **No Grid Intersection Contentions or Topological Relation Violations** This is the simplest case to manage: when a vertex needs to be placed on the nearest free grid intersection (figure 5.5 left), and that move does not violate any topological relation, then the vertex can simply be placed there.

FIGURE 5.5: Example of vertex positioning when no contentions nor topological relation violations arise.

- **Grid Intersection Contentions** This happens when a vertex is to be placed on a grid intersection already occupied by another vertex. As figure 5.6 shows, V1 would be placed on the location of V2. The algorithm tries to build a list of alternative moves within a certain range. In this particular case a list of alternative moves $\{C_1, ..., C_{16}\}$ is built. The first alternatives to be tried are the ones that do not cause the displacement of vertices that are already positioned and do not cause topological relation violations. So the alternative destination C11 is chosed as it is the nearest empty location which would not cause a topological relation violation and the vertex V1 is moved there. As no topological relations are violated, there is no need for further vertex displacements.

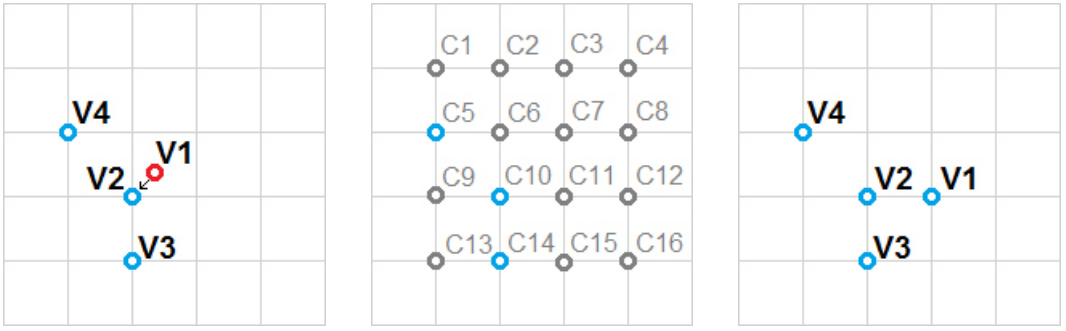

FIGURE 5.6: Example of vertex positioning when Grid Intersection Contentions arise, but there is the chance to solve the contention without violating topological relations.

- **Grid Intersection Contentions with topological relation violations** This case occurs when a contention arises and all the alternative moves imply recursive vertex displacement due to topological relation violations. This is what we call a "domino effect", where vertices need to be displaced. In case it causes any topological relation violation, more vertices may need to be displaced, and if contentions arise, more vertices may need to be displaced. It is important to remember at this point that all the initial topological relations are obtained from original transportation of map, before discretization of space. The displacement recursion occurs until all contentions are solved and all vertices are successfully placed without violating topological relations or until there is no possible vertices placement respecting the topological relations. The displacement of the vertices always analyzes the nearest available grid intersections in the first place if they exist, and the grid intersections that despite not being empty, respect the displacement vector, in the second place if they do not exist. Figure 5.7 illustrates this situation. We can see that V5 would be placed at the grid intersection where vertex V2 is placed. Again, a list of alternative moves $\{C_i, ..., C_{16}\}$ is built. As at this piece of map we have a geographical restriction (a lake), the C_2, C_3, C_4) alternative locations are discarded. From the remaining alternative moves, the ones that do not required apparent vertex displacement (the ones that are on empty grid intersections), are verified to see if they would cause violation of topological constraints. The violation of topological constraint would require a re-placement of some vertices, so, as they all would need re-placement of some vertices (and thus would also cause a"domino effect"), the best option would be to move to the C10 location, where the V2 vertex is. The displacement vector $\vec{d}$ (given by the direction angle between V5 and V2) is calculated and therefore V2 is displaced according to $\vec{d}$, meaning that it would be moved to the next grid intersection that follows the direction of $\vec{d}$. This move, however, will cause a violation of the topological relations. Before the move, V2 was "south of" and "right of" V1, and after the move it is just "south of". Therefore, a movement vector $\vec{d1}$ to fix this problem is calculated for the vertex V1, so V1 will be displaced one grid intersection accordingly. In addition to this, V4 also needs to be displaced for the same reason, with a vector $\vec{d2}$ being calculated. This vector will cause the vertex V4 to be displaced, and this recursive process ends here as there is no need of further vertex displacements as there are no occlusions and all the topological relations were fixed through the compensation moves executed. If those compensation moves caused further occlusions or topological relation violations, then, new compensation moves would be executed till a feasible solution would be found, otherwise the recursionHPPO function would return empty to the main HPPO function, which by its turn would

signal the SmartFit problem that it is not possible to have a feasible solution with this grid resolution.

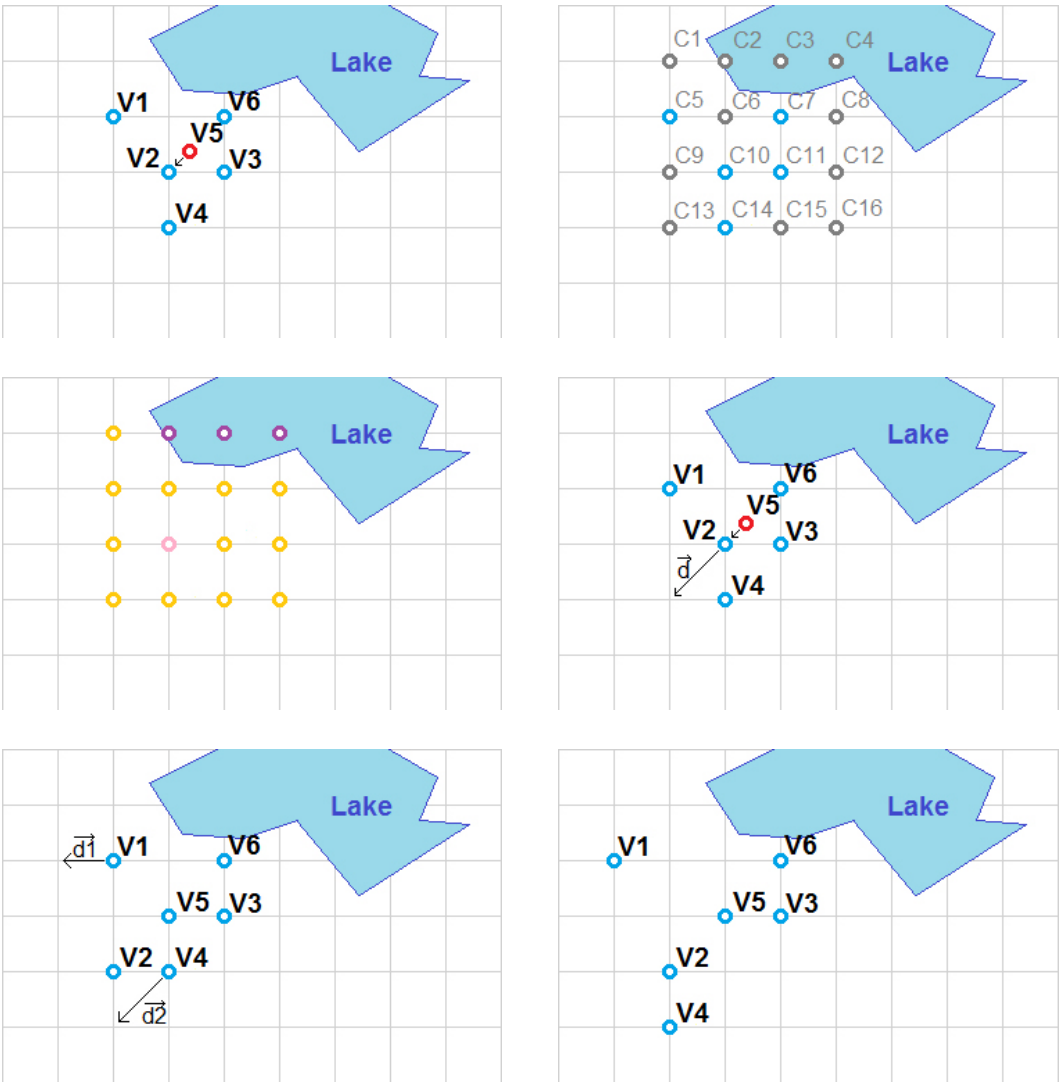

FIGURE 5.7: Example of vertex positioning when Grid Intersection Contentions and topological relation violations arise, causing the "domino effect".

At the end of this phase, we have automatically determined the best grid aperture size, re-scaled the map to include the hub, aligned the vertices to the grid solving the contentions while respecting topological relations so that the whole graph is prepared for the optimization phase (figure 5.8). The map obtained at the end of the execution of the HPPO is a feasible solution for our problem (the first Spider Map).

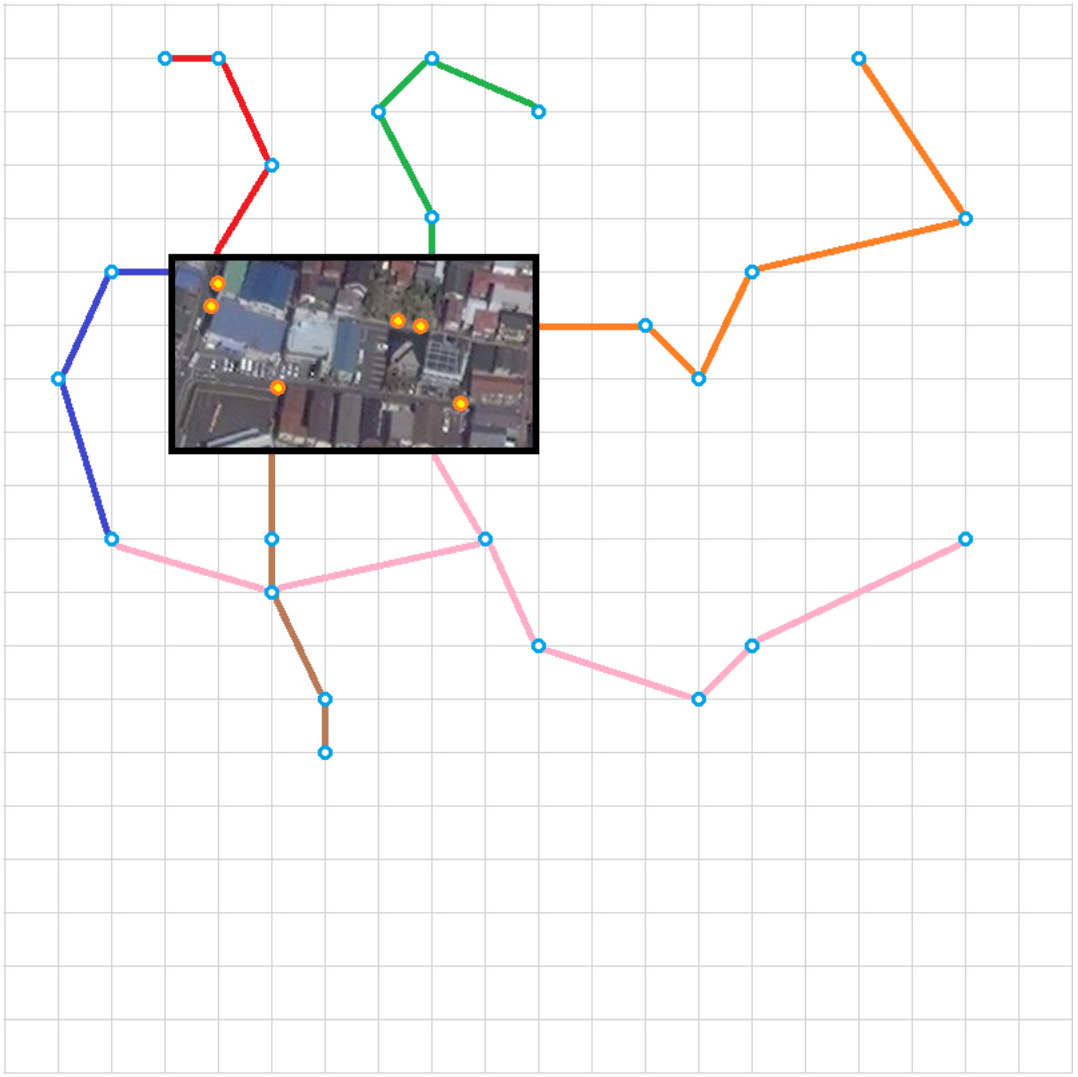

FIGURE 5.8: Example of discretization of vertex coordinates, obtained after the initialization and alignment to grid.

5.2 Tabu Search Optimization Metaheuristic

After having the first feasible solution which is an initial graph embedding that complies with the definition of spider map, we proceed to executing the tabu search optimization phase (algorithm 9). The initial solution in the Tabu Search procedure is the first Spider Map produced as described in the previous section. We begin by initializing the typical tabu search structures, such as the tabu list which stores the moves considered tabu, the history list of the obtained solutions (that a this point contains only the first obtained spider map), and a structure that computes the objective function score. In the tabu search algorithm applied to our problem, a move consists of a reference to the vertex being moved, the origin and destination grid intersections. Therefore a move defines the movement of a vertex to a location on its neighborhood. Each move has a tenure

time that defines the number of iterations it can be kept on the tabu list, meaning that throughout that tenure time the corresponding vertex cannot be moved. We used a tenure time 5. This value has been quoted in the literature [151] [152] [149] as an usual good guess for this type of problem.

Algorithm 9 Information system - General Description - Optimization Phase

```
10:     tabuStructure ← initializeTabuStructures()
11:     ListOfMovesHistory ← empty;
12:     bestGraph ← graph
13:     bestGraphScore ← evaluateGraph(graph)
14:     iterationsWithoutImprovement ← 0
15:     while iteration < maxIterations do
16:         graph.updateFreqMatrix();
17:         candidateMoves ← generateCandidateMoves(graph);
18:         candidateMoves ← checkForHardConstraints(graph)
19:         candidateMoves ← evaluate(candidateMoves)
20:         candidateMoves.orderBy(score)
21:         moveIsTabu ← isTabu(candidateMoves.first)
22:         while moveIsTabu do
23:             if checkAspirationCriteria(candidateMoves.first) then
24:                 break()
25:             end if
26:             if candidateMoves.count == 1 then
27:                 break()
28:             end if
29:             candidateMoves.RemoveFirst()
30:             moveIsTabu ← isTabu(candidateMoves.first)
31:         end while
32:         graph ← update(graph, candidateMoves.first)
33:         graph.updateTopologicalRelationsMatrix()
34:         UpdateTabuStructures()
35:         if candidateMoves.first.score < bestGraphScore then
36:             bestGraphScore ← candidateMoves.first.score
37:             bestGraph ← graph
38:             iterationsWithoutImprovement ← 0
39:         else
40:             iterationsWithoutImprovement++
41:         end if
42:         if (useSpatialDistributionAnalysisType) then
43:             executeSpatialDistributionAnalysis(graph)
44:         end if
45:     end while
```

The optimization phase is based on the vertex movement [153], [93], [91], [26], [94], [92]. At each iteration, a set of possible moves are generated, the moves that lead to unfeasible solutions are discarded, the remaining ones are evaluated and the best move is executed (if it is not in the tabu list, or even if it is but presents a significant increase of the quality of map). This process repeats as many times as the predefined number

of iterations or until a time deadline is met (if we need soft real time map processing), or until a certain quality threshold is obtained.). The algorithm also keeps track of the number of iterations without map score improvement. By the end of the tabu search optimization phase, the best solution found through the optimization phase is retrieved, and the map is ready for post processing.

The algorithm starts by updating the frequency matrix (a frequency matrix is a matrix with the dimension of the spider map grid which tells us how and where the vertices are located on the grid and it is important to speed up several calculations throughout the algorithm, such as as the Hard, Soft and Geographical Constraints. Then the set of possible moves is generated for every vertex that is part of the spider map (the candidate move list). The impact of each move is sequentially tested for each of the hard constraints. If it fails to comply with one of them, it will be eliminated from the candidate move list. The remaining candidate moves are evaluated according to the set of soft constraints and then they are sorted according to the *quality* of the solution they will generate. The best move of the list is chosen and, if it is not a tabu move, it can be applied. Still, if it is a tabu move but satisfies the aspiration criteria, it is also applied. The aspiration criterion here is the evaluation of the improvement of the map over the best map obtained until the moment. If it does not satisfy the aspiration criterion, but there are not any moves left and this is the last move, it is executed to avoid not generating a solution, as a last resource. The optimization phase iterations are repeated until the finishing condition is met.

5.2.1 Getting the Best Move

From the initial feasible solution obtained by the HPPO algorithm, a candidate list of possible vertex moves is generated for each vertex of the graph, within a certain range as shown in figure 5.9: for each vertex, several hypothetical moves are generated within a range defined by the user as an algorithm parameter called *max_displacement* (in the depicted case the range is one). For the vertices A, B, C and D, there is a set of possible moves $\{A_{ij}, B_{ij}, C_{ij}, D_{ij}\}\forall i, j \in 1, 3$ that can lead to the displacement of a vertex from its current position to a destination defined by each move.

The *max_displacement* parameter sets the grid cell range to where each vertex can be displaced. The higher the value of this parameter, the bigger will be the search space, as shown in figure 5.10. This comes with some advantages (faster map quality increase, better results in few iterations) but also with also with some disadvantages (slower algorithm processing per iteration, premature convergence, decreased ability to escape local minima).

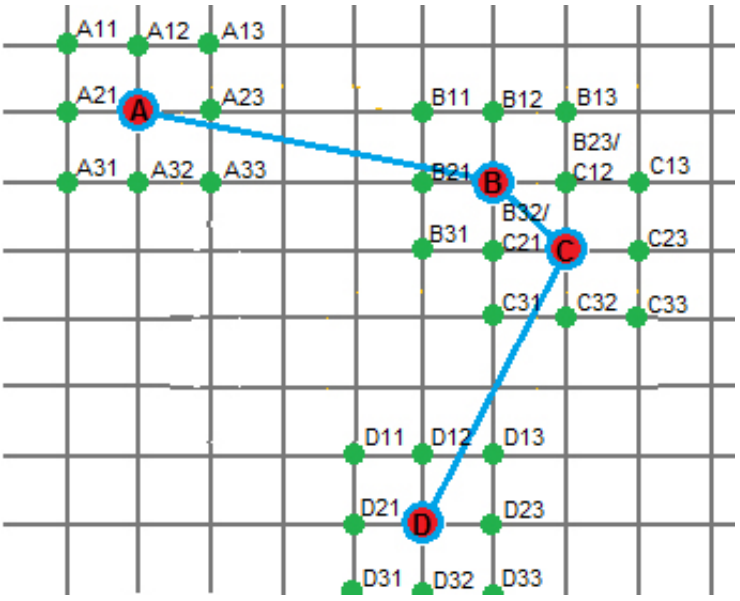

FIGURE 5.9: Generation of candidate moves example.

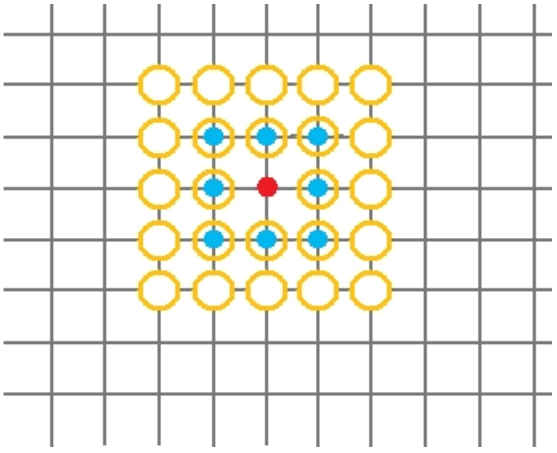

FIGURE 5.10: The *max_displacement* parameter in the generation of the point candidate list: smaller circles show the generated candidate points if *max_displacement*=1, bigger outer circles show the generated candidate points if *max_displacement*=2

Each move will be tested regarding the satisfaction of the hard constraints. If a move produces an unfeasible solution, it is removed from the candidate list. The moves that pass the hard constraint test correspond to a set of moves that will still produce a feasible solution for the spider map generation problem. The next step is to to choose the best possible move. To do this, each of the possible moves is evaluated using the objective function that encompasses the evaluation of our soft constraints. After obtaining the score for each move, they are listed and ordered from the best to the worst. The algorithm selects the first (best) move and checks if it is a tabu move. If it is, it checks if it satisfies the aspiration criteria that allow it to be selected as the move to be executed at the current tabu search iteration. If if does not, the move is discarded. If the move

is not a tabu move, it is selected to be executed at the current tabu search iteration. After having found the best move, it is executed, the graph embedding is updated, the move is added to the tabu list and the next iteration begins. The objective function minimizes the global sum of each of its components, each component being related to an individual soft constraint. Each soft constraint has a relative importance which can be changed through user parameters, according to each type of map.

5.2.2 Spatial Distribution Analysis

Throughout the execution of the tabu search in the optimization phase, there is the possibility that it gets stuck in local minima. This may happen when there are several vertices in consecutive grid intersections (vertices clusters), for example. To solve this problem, we introduced a feedback mechanism to our tabu search implementation which allows us to both eliminate clusters of vertexes and to improve map distribution. Instead of a cluster detection and movement algorithm such as Stott's [26], we developed a spatial density analysis algorithm. As described in pseudoalgorithm 9, it may run at the end of each iteration of the tabu search phase, in order to increase variability in the solution and allow further optimization, allowing to escape local minima. Being so, the Spacial Distribution Analysis algorithm is a significant diversification strategy to this problem. But its advantages go beyond that: it is also a visual map balancing algorithm. Cognitive psychology postulates that it is easier for the human brain to understand regular patterns and balanced presentation of visual information [71] [68][69][87]. Therefore it is desirable that the map produced is visually balanced, without very dense areas with many stops and lines contrasting with sparse areas with almost no lines or stops. This algorithm measures the quality of the spatial distribution of the elements of the map (figure 5.11) as follows:

1. A density matrix with the same dimension of the frequency matrix containing the number of vertices in each cell of the grid.

2. Each cell of the density matrix will store a value which is the number of adjacent cells that also have vertices (fig. 5.12).

3. Calculate the average value of all cells of the matrix (fig. 5.13)

4. Scan every line of the density matrix, calculate the average value, compute the "spike" score (the differences between spikes and the average score value) according to the following formula:

$$SpatialDistribution_Score = \sum_{i,j}^{c,l} \left(\frac{1 - \frac{|M - a_{i,j}|}{Max_{Spike}}}{c \times l} \right)$$

Where c is the number of columns, l the number of lines, a the value of each cell of the density matrix, Max_{Spike} the value of the deviation of the biggest spike regarding the average score per cell, M is the average score per cell.

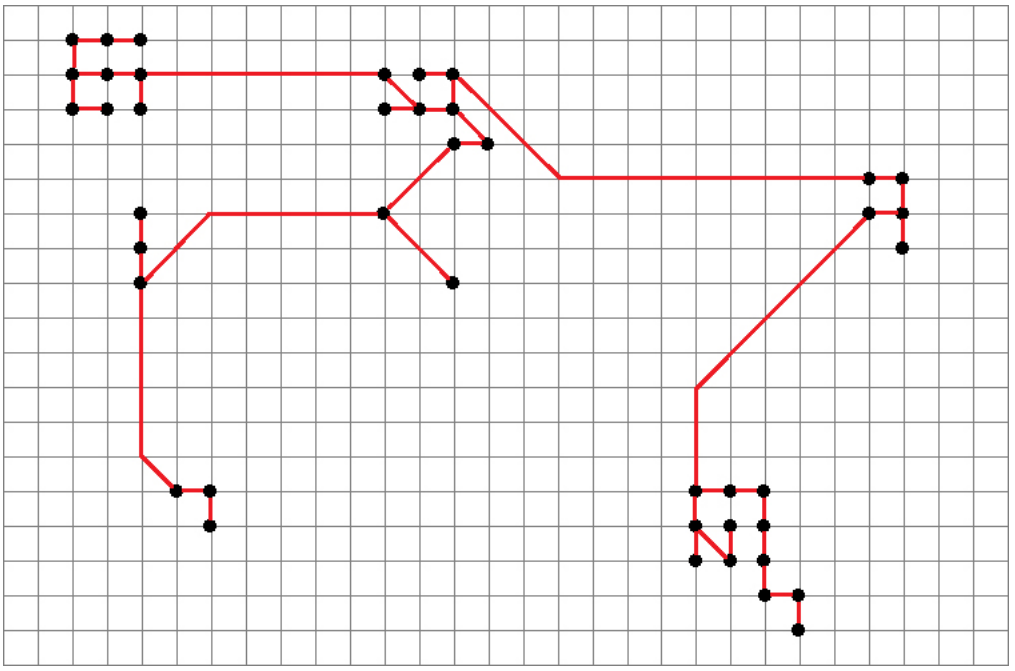

FIGURE 5.11: Spatial Distribution Analysis Algorithm: an example map with a 29x19 grid.

This algorithm allows a simple and efficient analysis of the visual balance of the map. The processing time is linear with the dimension of the matrix. The Spatial Distribution Analysis allows the tabu search algorithm to escape the local minima as it temporarily optimizes only the visual balance of the map, not considering the optimization function of the tabu search phase. It reorganizes the vertices positions in such a way it becomes better balanced (introducing variability on the tabu search solution) thus allowing tabu search to continue on other location of the solution search space. The Spatial Distribution Analysis stops after optimizing the visual distribution of the map, and then the tabu search optimization phase can continue normally.

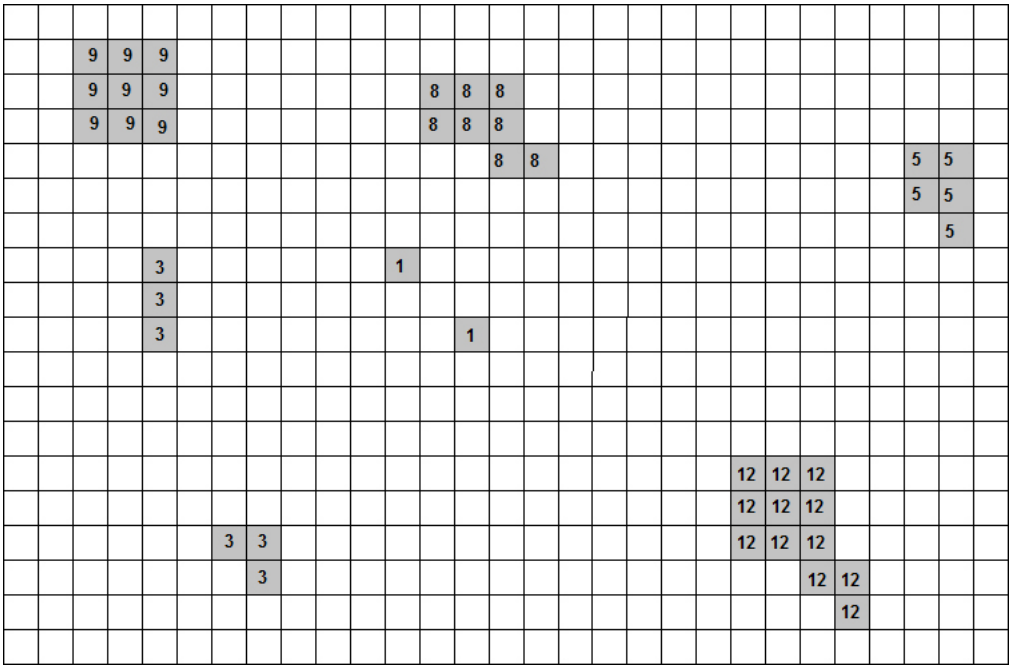

FIGURE 5.12: Spatial Distribution Analysis Algorithm: the corresponding density matrix to the map depicted in figure 5.11. The score on each cell is computed after recursive counting on adjacent stops.

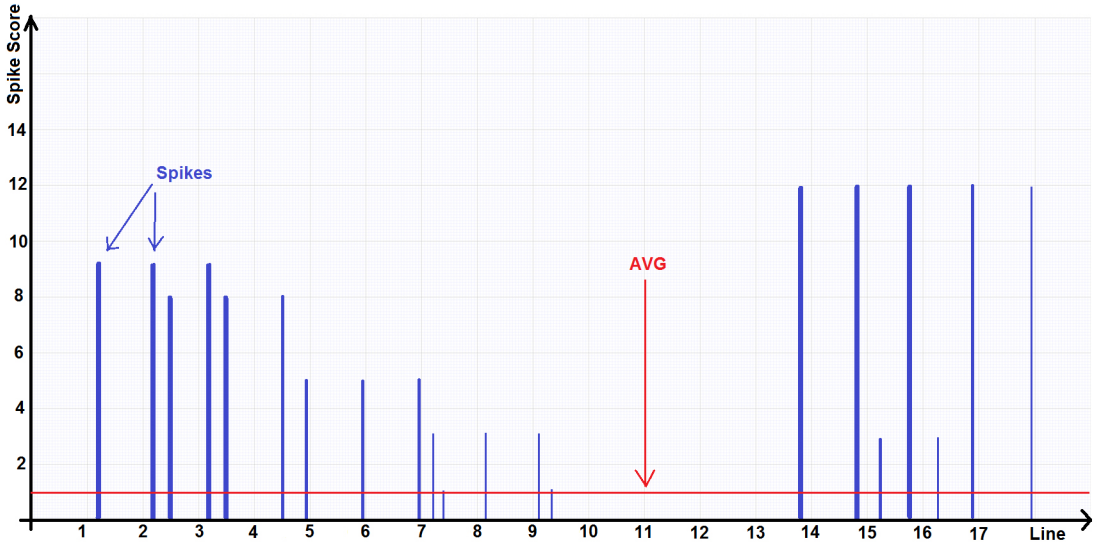

FIGURE 5.13: Spatial Distribution Analysis Algorithm: the graph analysis of the map shows that there are very significant differences between spike values and the average score per cell. This means the map is not well balanced.

5.3 Post Processing and Spider Map Output

At the end of the execution of the algorithm, the best solution found is post processed as described in algorithm 10. Post processing involves two phases:

- **Dealing with the geographical constraints** (if they exist at the map) Dealing with the geographical constraints implies that no vertex or edge is placed on a geographic restriction polygon (geographical constraint GC2). The fist part was achieved throughout the algorithm through the vertex placement, by respecting the hard constraints. The second condition needs to be achieved through post processing.
- **Introducing breakpoints** (or also called inflection points) at strategic edge locations to make them to comply with the 0, 45 and 90 degree schema. Therefore we assure by this way that every spider map can be designed around the geographical constraints and the graph embedding follows the desired orientation schema.

Algorithm 10 Information system - General Description - Post Processing

```
46:     generateInflectionPoints(graph)
47:     Graph ← TrueGraphTools.ConvertGraphToSRDS(graph)
48:     SpiderMap ← TrueGraphTools.ConvertSRDSToSpider(SpiderMap)
49:     return SpiderMap
50: end procedure
```

For both phases we decided to use the same algorithm, the A* algorithm. The A* is a computer algorithm that is commonly used in pathfinding and graph traversal, the process of finding the shortest path between points. It is a high performance and accurate algorithm, with widespread use [154]. This algorithm was first described in 1968 [155] and it is a very adequate algorithm also for avoiding obstacles and returning optimal paths. Some researchers have found A* algorithm to be superior to other approaches [156] such as Dijkra's algorithm in what concerns to speed. A* uses a best-first search and finds a least-cost path from a given initial node to one goal node (out of one or more possible goals). As the algorithm traverses the graph, it follows a path of the lowest expected total cost or distance, keeping a sorted priority queue of alternate path segments along the way. It uses a cost function of point x (denoted $f(x)$) to determine the order in which the search visits nodes in the tree.

As $f(x) = g(x) + h(x)$, with

- $g(x)$ being the known distance from the starting node to the current node x (past path-cost function).

- $h(x)$ being the "heuristic distance" from x to the the destination (future path-cost function). This is an estimate cost of the path from the actual point to the destination point (estimated distance). Several heuristics can be used here, the simplest being the calculation the straight-line distance to the goal, since that is physically the smallest possible distance between any two points or nodes. This is usually called "heuristic" as we do not really know what is the actual distance until we find the path, as many obstacles can be in the way.

While A* is generally considered to be the best pathfinding algorithm[43], similar algorithms such as Dijsktra's can be used. Dijkstra algorithm is essentially the same to A*, except there is no heuristic ($h(x)$ is always zero), and that causes the algorithm to search the space evenly in every direction. Therefore, Dijkstra's algorithm ends up by searching much more search space before the destination is found (thus usually making it much slower that A*). When we know our destination's location (as it happens in our particular case), the A* outperforms by large the Dijkstra's algorithm, and therefore we decided to use the A* algorithm.

Figure 5.14 exemplifies this algorithm in a normal situation, when we need to find path between two points and there is an obstacle in between. In each cell, the top left value is $f(x)$, $g(x)$ is printed in the bottom left and $h(x)$ is printed in the bottom right. The direction from the predecessor node is printed at the middle of each cell. The obstacles are marked as "unwalkable" as the path shall not be over them. We begin by enlisting the adjacent paths to our direction and chose the point which has less $f(x)$ cost, and that point will be part of the path to the destination. This process loops till we find the destination.

Our implementation of the A* algorithm features some enhancements to improve both the quality and speed of the results:

- **Smoother paths:** while A* gives the shortest path (lowest cost), it automatically will not give the smoothest looking path (as shown in figure 5.14). We use an enhanced heuristic function $h(x)$ that calculates the total cost also in diagonal ways to avoid the "staircase effect") and an automatic differential grid aperture (for avoiding geographical constraint polygons) for better looking results, too.
- **Map pre-processing:** before the algorithm runs, we pre process our map to exclude areas that are forbidden (the path shall not go there) or inaccessible. This considerably decreases the search time, thus improving efficiency. Differential grid aperture also helps on achieving speed, specially when no geographical constraints exist.

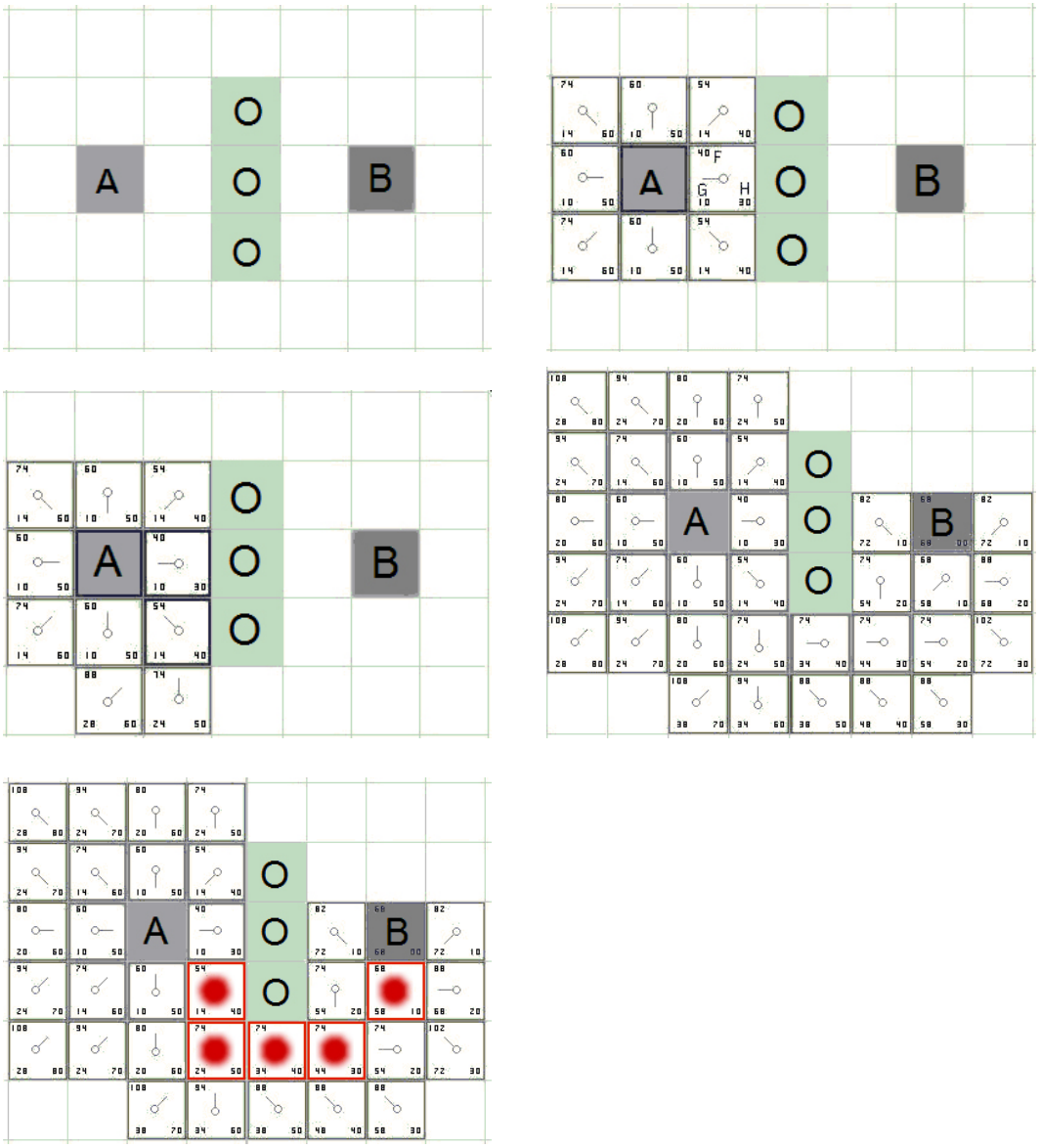

FIGURE 5.14: A simple example of the A* algorithm, finding the shortest path from A to B. [43] (adapted)

- **Keeping and ordered cost list:** we maintain a sorted list and simple grab the first item off the list every time we need the lowest $f(x)$ cost point. This avoids searching for the lowest value at every iteration.

5.3.1 Inserting Inflection Points

As part of the post processing, edges are scanned and inflection points are introduced to allow them to comply with the octilinear degree schema, as seen in figure 5.15, for the edges that still do not comply with it (that should be few or none, after the optimization phase has been completed). This is achieved through an A* algorithm version, which finds the most direct path on the grid schema from one vertex to another, creating

the breakpoints as needed. This procedure must not violate any of the hard constraints (otherwise we would not have a feasible solution) and therefore our A* algorithm observes all the hard constraints.

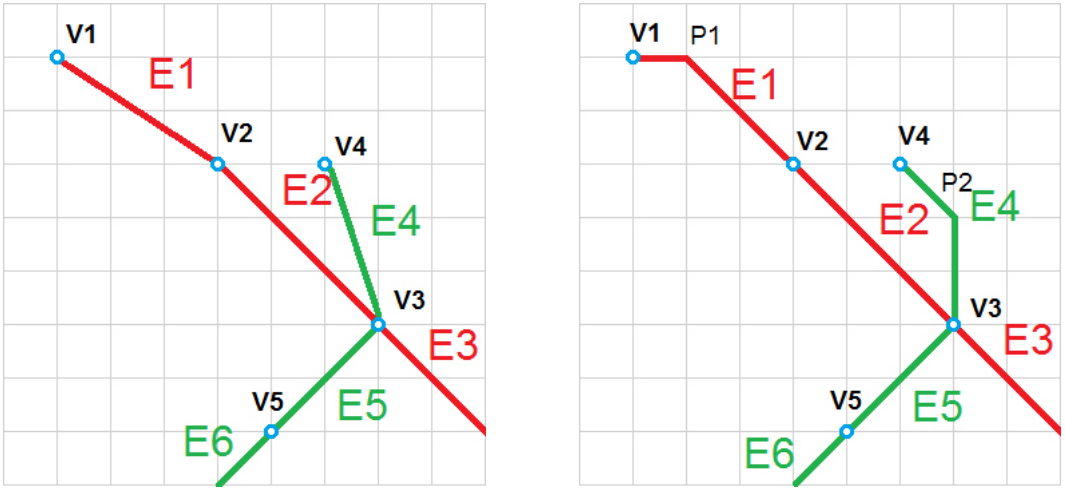

FIGURE 5.15: Example of the generation of inflection points: before(left) and after(right).

5.3.2 Dealing with Geographic Restrictions

It may happen, however, that some edges may occlude the geographic restriction polygons. In this case we use the A* algorithm to find the smoother way around the polygons. If the polygons are very irregular our algorithm may decide to differentially increase the grid resolution to allow the edges to go through the available spaces, not disturbing the other map elements. Through automatic differential grid aperture integrated with A*, we manage to avoid the geographical constraints. When needed, the algorithm increases the grid resolution into the relevant grid cells by a factor that optimizes the performance and result of the A* algorithm. If the grid resolution is too high, it will face performance issues, and therefore we use a mechanism similar to the SmartFit algorithm previously used to obtain the best grid aperture as needed. Figure 5.16 shows a detailed example of differential grid aperture near a bridge.

At the end of this last step, the final solution is a map that is ready to be exported to an SVG file or to a serialized file and it is ready to use. All the algorithm steps were have been carefully planned to allow it to be implemented to power location-based services [54].

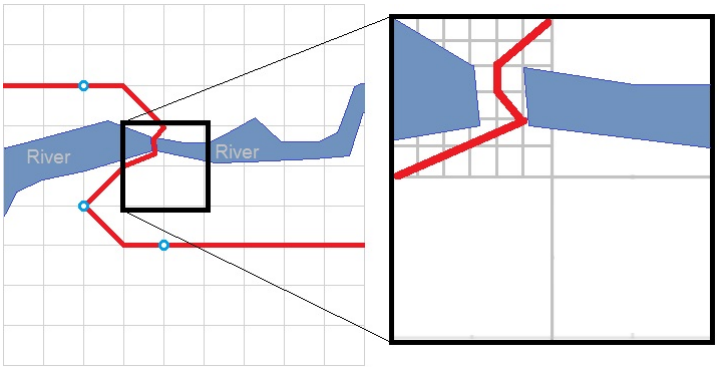

FIGURE 5.16: Intelligent Differential Grid Resolution.

Chapter 6

Testing in Real World

In this chapter we present the tests conducted on real map instances, together with the description of the test environment and software framework used to perform the tests. The results are discussed and at the end of chapter some maps that are used in real world are shown.

6.1 The GenX Framework

The developed algorithms were implemented using C# programming language and were tested through a software framework developed through a collaboration research performed by a team involving collaborators from FEUP [1], OPT [2], STCP [3] , FWT [4], INEGI [5]. Our information system is already being used to generate spider maps which are already being used in Porto, Lisbon and Santo Tirso cities.

The software framework which was developed through a joint effort with OPT in order to support our approach for the generation of spider maps provides the map XML file input. The framework connects to the company proprietary databases, where the transportation network raw information is stored in (the stop locations, the names of the stops, lines and all the meta data and semantic richness that describe a network). It also contains a designer-side GUI which allows user to select the hub zone and the transportation lines the final spider map shall feature. The business logic handles the

[1] Faculty of Engineering of University of Porto *http://www.fe.up.pt*

[2] OPT is an company based in Porto which develops IT infrastructures for Transportation Services. *http://www.opt.pt*

[3] STCP is a public transportation company operating in Porto. *http://www.stcp.pt*

[4] FWT is a company based in London which produces maps for transportation networks. *http://www.fwt.co.uk*

[5] INEGI is a research institute based in Porto

GUI requests and sends them to the database, returning all the relevant raw data for that map, and building an XML file which is the input of our information system, as shown on figure 6.1

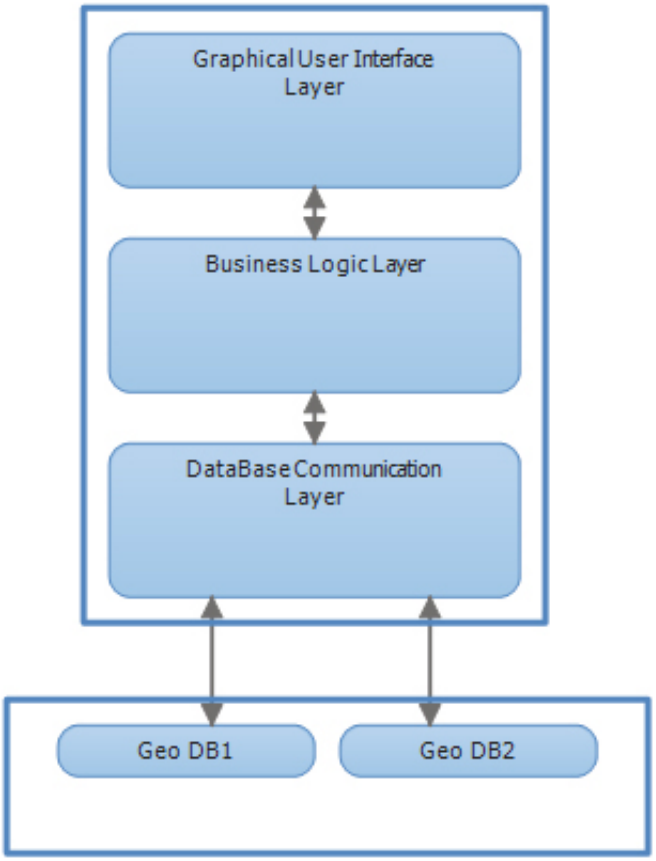

FIGURE 6.1: UML Layer Diagram providing an overview of the OPT Framework

Once our the set of algorithm finishes map processing and outputs the resulting map, the framework gets the resulting file for further automatic adaptation to public use (executing automatic label positioning and visual line arrangement, or further manual processing.

6.2 Tests and Results

We began the tests to assess some performance measures, such as Execution Time (raw performance execution time, in seconds), perceived quality *versus* iterations and execution time. Other assessment we have made was the evaluation of explicit searching versus implicit searching regarding Execution Time and quality. We also tested the isolated effect of each soft constraint to understand how to tune the weights for the soft criteria for some maps. The topological relations hard/soft approaches were also subjected to

test, as well as the results of the A-Star algorithm and the spatial distribution analysis algorithm. All the maps subjected to test can be found in appendices A to F, in three versions: raw version (geographically accurate), after the pre-processing phase and after being fully processed by our approach.

6.2.1 Test Environment and Description

All the tests performed involved real data with real world complexity, on a laptop computer, with an AMD N830 CPU with 4GB RAM and a typical low grade SATA hard drive using Microsoft Windows 7 64-bit and Microsoft Visual Studio to run the framework and the algorithm. Although it may be much slower, the algorithm was compiled on debug mode for testing and demonstration purposes. As the release mode packs several optimizations in terms of code and hardware architectures, it is expected that the live use of this algorithm can see an increase of up to five times in raw performance, depending on specific hardware optimizations. Nevertheless, hardware environment optimizations or considerations are out of the scope of this research.

We used 6 different maps from Porto bus transportation network, whose properties are presented in table 6.1. Each of the six maps considers a variation where geographical constraints are included. The 6 maps are presented in appendices A to F.

The map in figure 6.2 (map 3) was generated with our algorithm, with some minor manual aesthetic work performed such as label inclusion. The algorithm was capable of generating an understandable spider map in soft real time, with no conflict points and respecting the topological relationships between stops.

TABLE 6.1: The maps subjected to test and their features.

Map Id	Zone	Vertices	Edges	GeoCons	XML Size (KB)
1A	Av. Republica	36	45	Yes	295
1B	Av. Republica	36	45	No	269
2A	Castelo do Queijo	47	58	Yes	307
2B	Castelo do Queijo	47	58	No	282
3A	Polo Universitario	22	20	Yes	149
3B	Polo Universitario	22	20	No	124
4A	Rotunda da Boavista	104	155	Yes	885
4B	Rotunda da Boavista	104	155	No	859
5A	S. Joao	114	156	Yes	866
5B	S. Joao	114	156	No	841
6A	Paranhos	26	25	Yes	165
6B	Paranhos	26	25	No	140

We executed a set of 8 tests:

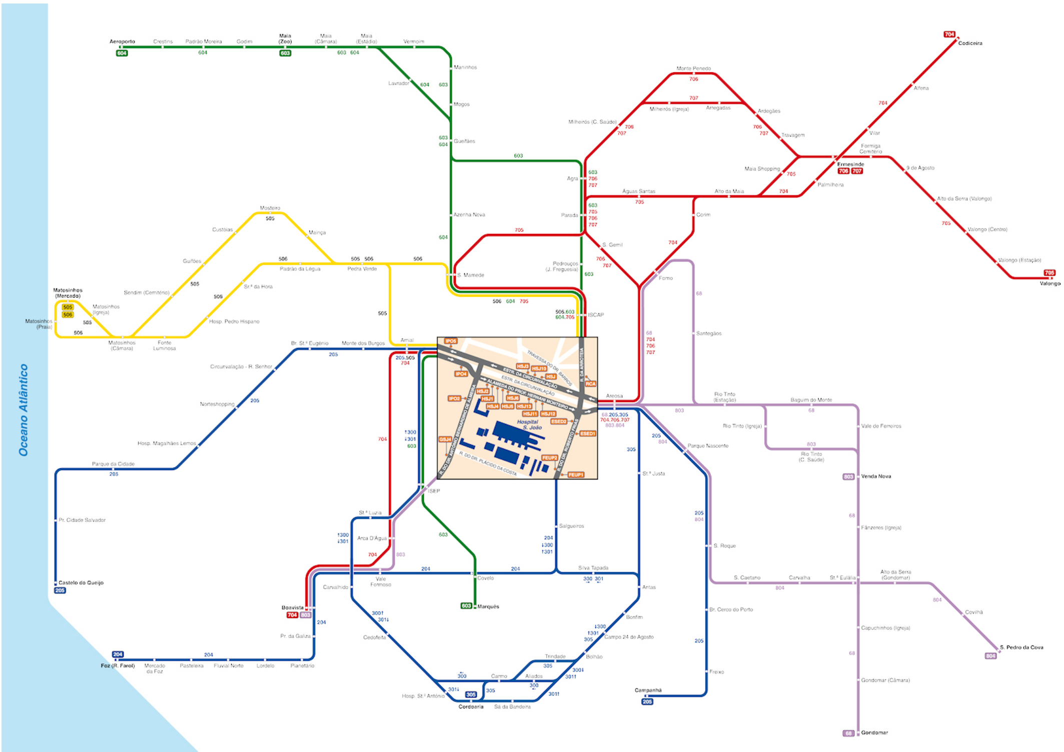

FIGURE 6.2: Spider Map of Hospital de Sao Joao area of the city of Porto, generated through our enhanced Tabu Search algorithm

- **Test 1: Execution Time** - Execution time for each map with standard settings[6].
- **Test 2: Quality *versus* Iterations Number** - The graphical evolution of the solution score (quality) over the execution of the algorithm in 10000 iterations.
- **Test 3: Maximum Vertex Displacement parameter influence on result** - Quality *versus* Execution Time regarding the variation of this parameter. The Maximum Vertex Displacement corresponds to the *max_displacement* parameter which sets the grid cell range to where each vertex can be displaced, as shown in figure 5.10.
- **Test 4: Candidate Move Generation Ratio parameter influence on result** - Quality *versus* Execution Time regarding the variation of this parameter. The Candidate Move Generation Ratio is a parameter that controls explicit or implicit search in tabu search: a ratio of 100% means that all possible moves are analized by the algorithm.
- **Test 5: Soft Constraint Isolated Effect of Parameters in map visual presentation** - We measure the visual isolated effect of each of the soft constraints that comprise the evaluation function.

[6]The standard settings *may* not be the optimal settings. They are the default settings shown at the parametrization window as seen in figure 6.3

- **Test 6: Hard *versus* Soft topological relations** - The effect of treating topological relation enforcement as a soft or hard constraint in final map visual quality and execution time.
- **Test 7: A* Pathfinding Overhead** - Measurement of overhead time of the A* pathfinding algorithm when finding paths around geographical accidents.
- **Test 8: Spatial Distribution Smart/Blind Algorithm Evaluation** - For every map, test the effect of the Spatial Distribution Algorithm distribution with either by running it automatically when the tabu search solution quality is not improving after a number of iterations (Smart) or by running it periodically along the tabu search (Blind).

With these tests we intend to analyze the most important parameters of the algorithm, their influence on the common performance indicators and on visual quality of the maps. Except for the parameter under analysis, all the other parameters are set to their default values. The purpose is to to analyze how the algorithm behaves regarding the variation in each parameter and to extract conclusions that can be applied to generate effective and high quality spider maps.

6.2.2 Parametrization

When the GenX framework is launched, a controller window appears. This window allows the parameter values to be set by the user for specific maps and to test the algorithm parameters sensitivity and performance (figure 6.3).

This dialog groups the parameters per categories. The algorithm related parameters include:

- **Grid Granularity** is the grid aperture size. This parameter is automated through HPPO, but the user can change its initial guessing value. Default unit type is milimiter.
- **Hub Clearance Multiplier** is a parameter that defines on how big the grid cell range clearance area will be. The clearance area is an area that surrounds the hub which will not contain any vertex. It is used to improve readability. The larger this parameter, the larger will be the empty area surrounding the hub.
- **Number of Iterations** defines the number of iterations of the algorithm
- **Maximum Vertex Displacement (iteration)** defines the range of the move candidate list for each vertex at each iteration, as shown in figure 5.9.

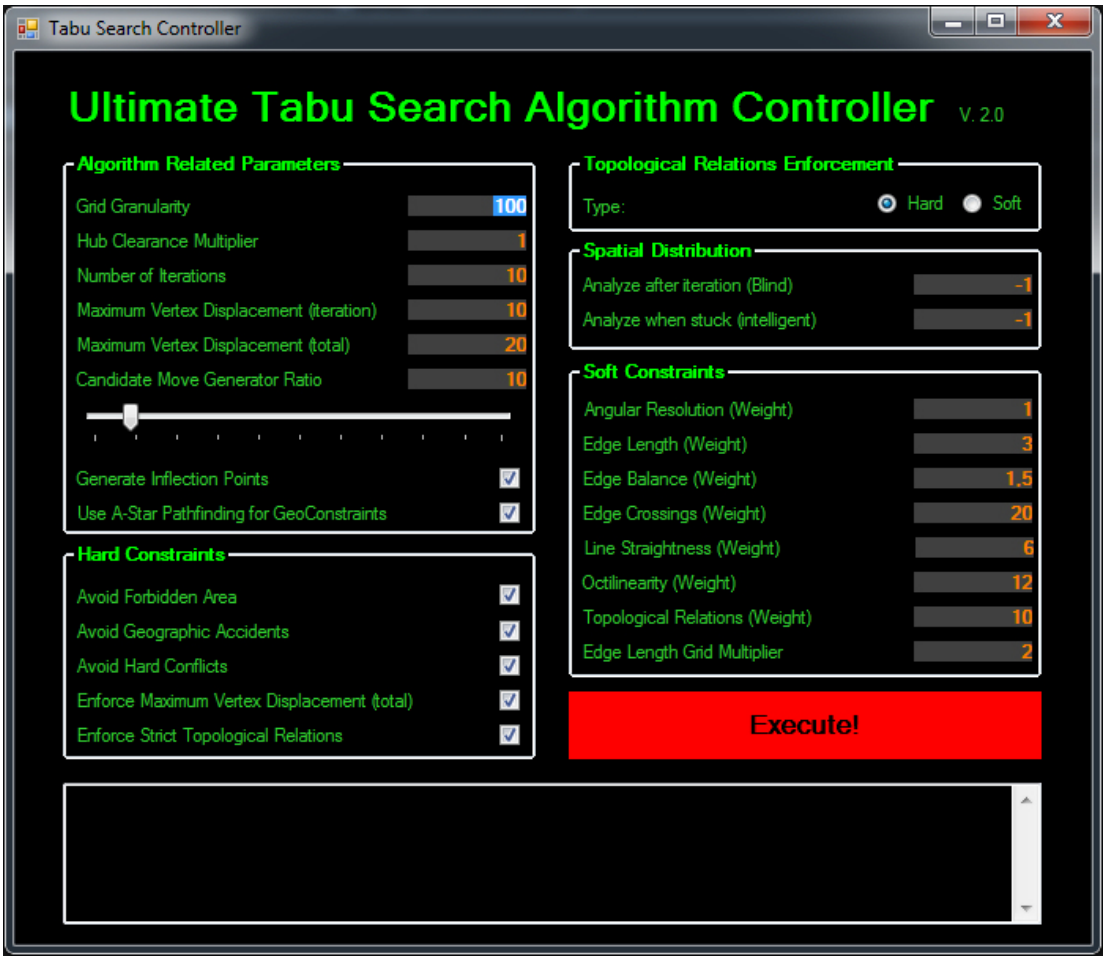

FIGURE 6.3: Parametrization Window, showing the standard parameter values.

- **Maximum Vertex Displacement (total)** defines the maximum range of displacement for each vertex regarding its original (geographically accurate) position.
- **Candidate Move Generator Ratio** defines the percentage of the the possible move candidate list elements to be effectively generated at each iteration (the percentage of all possible vertex moves at each iteration that will be analyzed by the tabu search optimization). This affects the balance between implicit vs explicit search.
- **Generate Inflection Points** toggles the generation of inflection points in post processing.
- **Use A-Star Pathfinding for GeoConstraints** switches the A-Star pathfinding in post processing.

The Hard Constraints related parameters include:

- **Avoid Forbidden Area** guarantees that edges and vertices are not be placed on forbidden areas, like the hub, the outside of the canvas area and the hub clearance area.
- **Avoid Geographic Constraints** guarantees that the vertices are not placed placement of vertices on the areas defined by the geographic accident polygons.
- **Avoid Hard Conflicts** guarantees that the vertices are not placed on top of other vertices.
- **Enforce Maximum Vertex Displacement (total)** ensures that every vertex must be within a certain pre-defined range from its original (geographically accurate) position. The value of this parameter is the range.
- **Enforce Strict Topological Relations** ensures the enforcement of strict topological relations.

The parameters related to the soft constraints include the weight of each soft constraint in the optimization function. This parametrization can be used to tailor the execution of the algorithm for specific maps, to assess the effect each soft constraint has in the final result and as a sensitivity analysis, useful for normalization purposes and quality testing.

The Spatial Distribution related parameters control the execution of the spatial distribution analysis. The Spatial Distribution Analysis Algorithm can be used in two ways:

- **Analyze after iteration (Blind)** switches the execution of the spatial distribution analysis algorithm periodically after a number of tabu search iterations (defined by the user). To be disabled, it shall be set to -1.
- **Analyze when stuck (Intelligent)** switches execution of the spatial distribution analysis algorithm whenever the tabu search algorithm is not able to improve the current solution for a number of iterations (defined by the user). To be disabled, it shall be set to -1.

The topological relations enforcement can be “hard” or “soft”, if we want it to be mandatory or desirable, respectively. For specific maps we may need to consider it part of the optimization function, not being a mandatory feature but a desirable feature. This option adds even more flexibility to the algorithm.

For normalization purposes, we propose a set of predefined values for the weight of each soft constraint. These values were obtained through intensive testing and after Stott's research work [26]. Nevertheless, different maps may require different weight relations.

6.2.3 Results and Analysis

For each test performed the results were summarized in tabular form, and discussed, extracting conclusions on the data obtained. By default, all the algorithm parameters keep their standard values as shown in figure 6.3, unless stated otherwise.

6.2.3.1 Test 1 - Execution time

Table 6.2 shows the result metrics for Test 1, where we wanted to assess the execution time for each map with standard settings. The execution time is measured in seconds. The Geo Var shows the variation of the Execution Time regarding the equivalent map version without geographical restrictions. The Geo Execution Time Delta indicates the decrease in the Execution Time of the maps without geographical restrictions in comparison with the same map with geographic restrictions. Complexity is a measurement of the map XML file size (the larger the file, the more complex the graph is and more amount of information it contains) in relation to map 1A, considered as a reference for comparison purposes (100%).

Table 6.2: Results of Test 1 - (ET = Execution Time, Cpx = Complexity)

Map ID	ET	Geo Var	Geo ET Delta	Cpx	Cpx Index
1A	2,56	100%	-	295	100%
1B	1,93	75%	-25%	269	91%
2A	4,16	100%	-	307	104%
2B	3,03	73%	-27%	282	96%
3A	1,61	100%	-	149	51%
3B	1,15	71%	-29%	124	42%
4A	12,91	100%	-	885	300%
4B	12,29	95%	-5%	859	291%
5A	13,92	100%	-	866	294%
5B	12,56	90%	-10%	841	285%
6A	1,78	100%	-	165	56%
6B	1,31	74%	-26%	140	47%
		Geo Bias Avg	-20%		

As we can see, the algorithm is in average 20% faster (in average) for maps without geographical restrictions, regarding their versions with geographical restrictions. However, this difference shrinks for higher complexity maps.

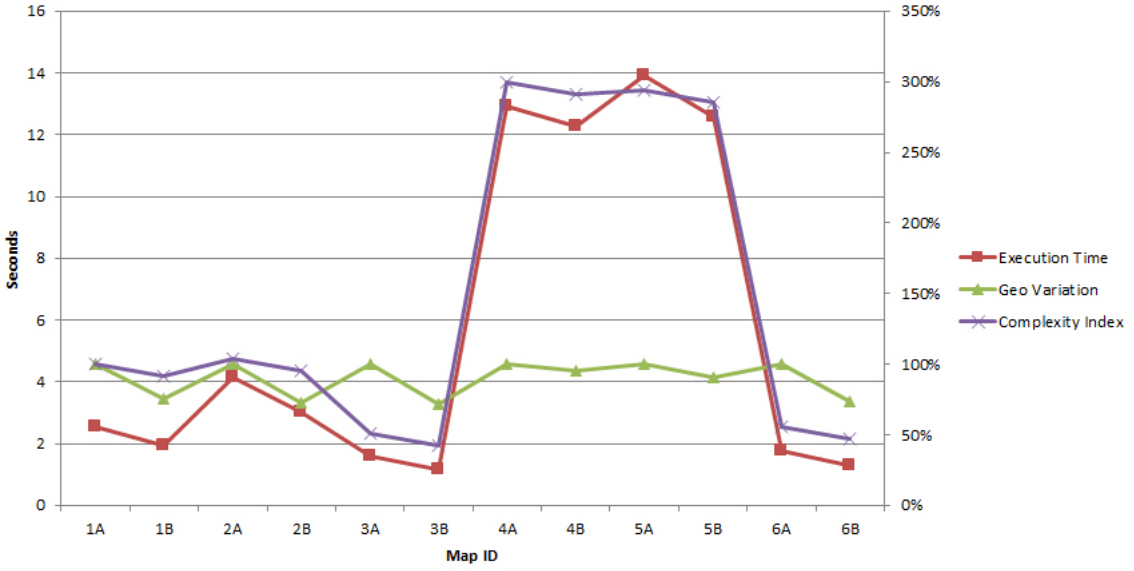

FIGURE 6.4: Relation between the Execution Time and the Geo Variation parameter and the complexity index for each map.

Figure 6.4 shows that the Execution Time varies with the map complexity. For example, the complexity of maps 4A,4B,5A and 5B is about almost three times the complexity of map 1, but the Execution Times increase about six times. We observe that map complexity has a strong effect on Execution Time, while the existence of geographical constraints has not such a drastic effect.

6.2.3.2 Test 2 - Quality versus Iterations Number

Regarding Test 2, we wanted to assess the graphical evolution of the execution of the algorithm concerning the score solution over 10000 iterations for each map. Figure 6.5 shows an example of the execution of the algorithm for the map 1. For practical purposes, all the execution graphs can be found in appendix G.

The figure shows that the first hundred iterations provide a continuous and steady score improvement. After that the improvement rate decreases (although the score continues to improve). After about 1000 iterations the score improvement is quite slower as the algorithm continues to explore the search space. Occasionally, better solutions are found and the score improves. The results also show that solution score variability throughout the algorithm execution decreases as map complexity increases. A possible explanation is that highly complex maps have less score variability as they have more map points, and each move has less impact in global map score. In high complexity maps, increasing variability and diversification would be a good improvement. Another improvement

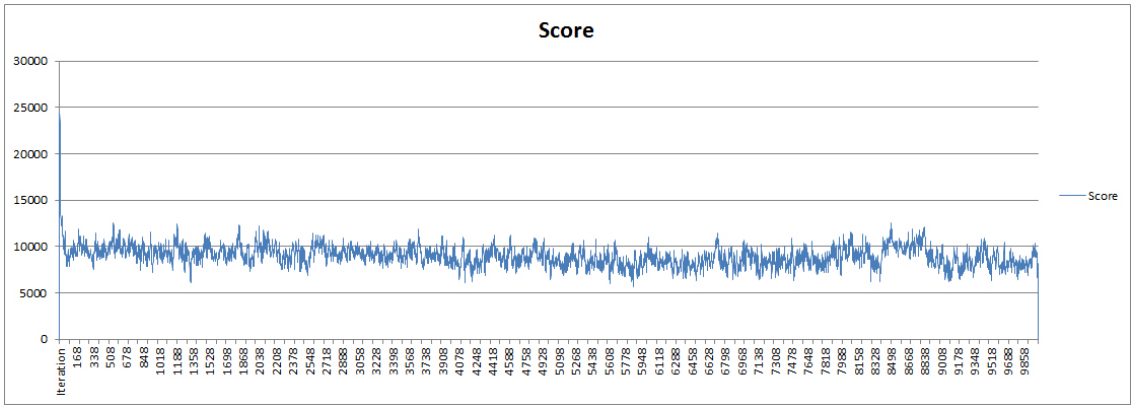

FIGURE 6.5: Test 2 result for map 1A. The y-axis measures the solution score (value of the objective function) while the x-axis shows the number of iterations.

could be to increase the weights of the most significant soft constraints for the particular map.

Table 6.3 summarizes the results for every map at test 2. The pound sign stands for "iteration", for abbreviation purposes.

TABLE 6.3: Results of Test 2 - ID = Map ID, ET = Execution Time for 10000 iterations, Init Sc = Initial Score, Sc BS = Best Score, it# = iteration number, SC = Score, Impr = Improvement from the initial score to the Best Score, Impr 2 = Improvement from the Best Score relatively to the score obtained at iteration 100)

ID	ET	Init Sc	BS	Impr	BS it#	Sc at #1000	Impr 2
1A	533	24341,32	5635,405	430%	5861	7245,83	29%
1B	284	24341,32	6553,408	371%	4871	7605,55	16%
2A	1414	30673,16	9818,673	312%	4200	10327,41	5%
2B	968	30673,16	9006,87	340%	8261	10016,56	11%
3A	325	22705,26	3239,692	700%	8681	5925,07	83%
3B	153	22106,58	3206,224	689%	6008	5166,96	61%
4A	7955	62571,59	30848,1	202%	9991	31727,23	3%
4B	6456	62571,59	30677,12	203%	6244	32787,27	7%
5A	8072	70863,90	37723,98	187%	9750	38807,96	3%
5B	6914	70863,90	35074,35	202%	9612	37094,49	6%
6A	337	14256,78	2776,376	513%	298	2776,37	0%
6B	158	14256,78	1869,578	762%	7749	2725,79	46%
Avg:				409%			22%

Table 6.3 shows that with 10000 iterations, the average improvement on map score is 409%, a very significant improvement. If we compare the best scores with the best scores obtained at iteration 1000 for each map, we see that the average improvement is much more reduced (22%). This means that running the algorithm for 1000 iterations yields a much better map quality *versus* execution time relation and shows that this algorithm can produce *good* quality solutions very quickly. These results also reveal

that the algorithm yields better scores for map versions without geographic restrictions. This was expected, as the solution search space is more reduced, as the geographic restrictions take space that otherwise could be available for map point moves. The maps with geographical restrictions also take longer to process (in simpler maps this difference is bigger), as more constraints are introduced.

Otherwise, if no time constraints arise and/or we run this for live execution in "release mode" (which may be up to 5 times faster) and we have good hardware, we can choose 10000 (or more) iterations to get the best possible map.

6.2.3.3 Test 3 - Maximum Vertex Displacement Parameter Influence

In Test 3, we evaluated the influence of the Maximum Vertex Displacement (total - throughout all the algorithm) parameter influence on result. We performed a test with two different values: 20 and 50, for 10000 iterations This means that for this test the vertices must not be placed outside the range of 20 (in one case) or 50 (in the other case) grid intersections far from their initial location. Tests with values higher than 50 would not bring an analysis advantage as for big values of this parameter many of the possible vertex displacements would be blocked by the topological relations enforcement in the algorithm. Figures 6.6 and 6.7 show the test results for with maps A1 and 5B, and the results for all maps are described in table 6.4.

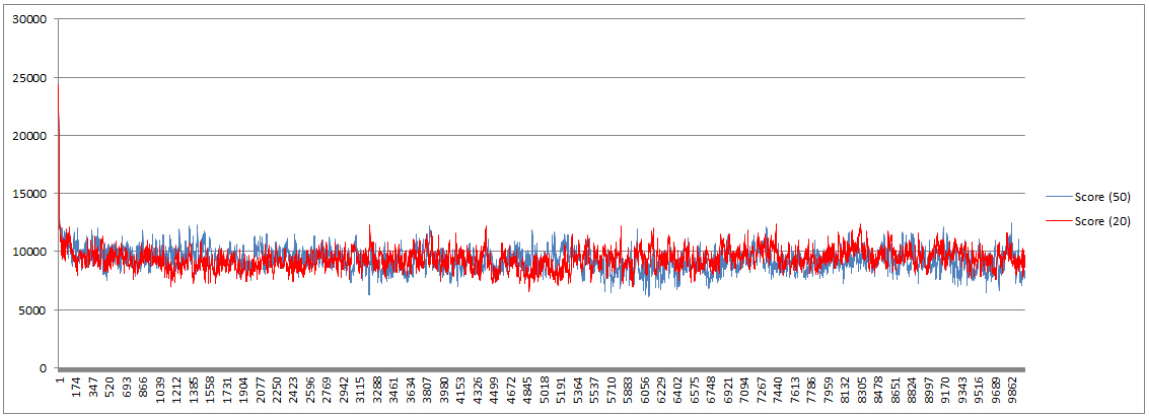

FIGURE 6.6: Test 3 result for map 1A

This test shows that higher values of the Maximum Vertex Displacement parameter yield slightly better scores, without any significant performance decrease. The higher this parameter is set, the more freedom the map points have to be placed in the map, and that is the reason this yields better map score. Nevertheless, care needs to be taken when setting high values of this parameter in conjunction with soft topological relations, as they may cause poor usability and user disorientation (due to potential high

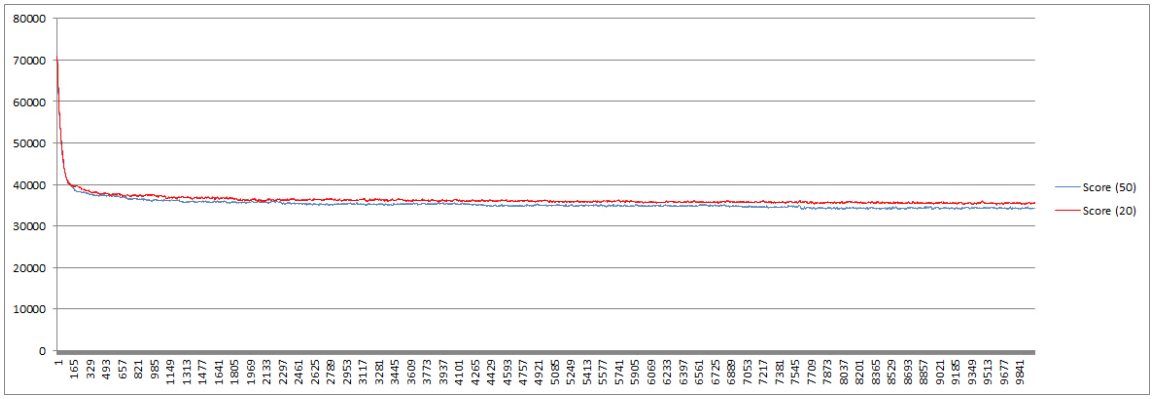

FIGURE 6.7: Test 3 result for map 5B

TABLE 6.4: Results of Test 3. MVD = MaxVertexDisplacement, ET = Execution time, ET Var = Execution Time Variation in comparison with the same map with MVD=20, SC = Score, SC Var = Score Variation in comparison in comparison with the same map with MVD=20.

Map ID	**MVD**	**ET**	**ET Var**	**SC**	**SC Var**
1A	20	533	100%	5635	100%
1A	50	527	99%	5351	95%
1B	20	284	100%	6553	100%
1B	50	279	98%	6304	96%
2A	20	1414	100%	9818	100%
2A	50	1392	98%	9762	99%
2B	20	968	100%	9006	100%
2B	50	955	99%	8911	99%
3A	20	325	100%	3239	100%
3A	50	324	100%	2907	90%
3B	20	153	100%	3206	100%
3B	50	147	96%	2938	92%
4A	20	7955	100%	30848	100%
4A	50	7892	99%	30287	98%
4B	20	6456	100%	30677	100%
4B	50	6302	98%	29842	97%
5A	20	8072	100%	37723	100%
5A	50	7955	99%	35221	93%
5B	20	6914	100%	35074	100%
5B	50	6741	97%	33999	97%
6A	20	337	100%	2776	100%
6A	50	326	97%	2537	91%
6B	20	158	100%	1869	100%
6B	50	157	99%	1752	94%

distance vertex movements that may violate topological relations), which is obviously not desirable.

6.2.3.4 Test 4 - Candidate Move Generation Ratio Parameter

At Test 4 we tested the Candidate Move Generation Ratio parameter influence on result. We used a ratio of 10% (only one in ten possible candidates moves in the neighborhood of a vertex is evaluated) and 100% (all possible candidate moves are evaluated), for 10000 iterations, with all maps. This test evaluates the implicit *versus* explicit search impact in execution time and solution quality. Figures 6.8 and 6.9 show the test results for maps 1A and 5B, and table 6.5 shows summarizes the results for all maps.

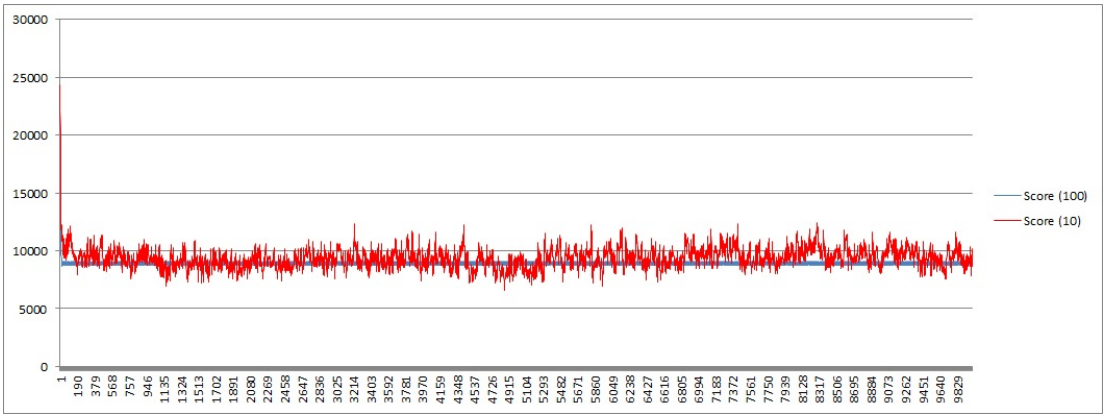

FIGURE 6.8: Test 4 result for map 1A

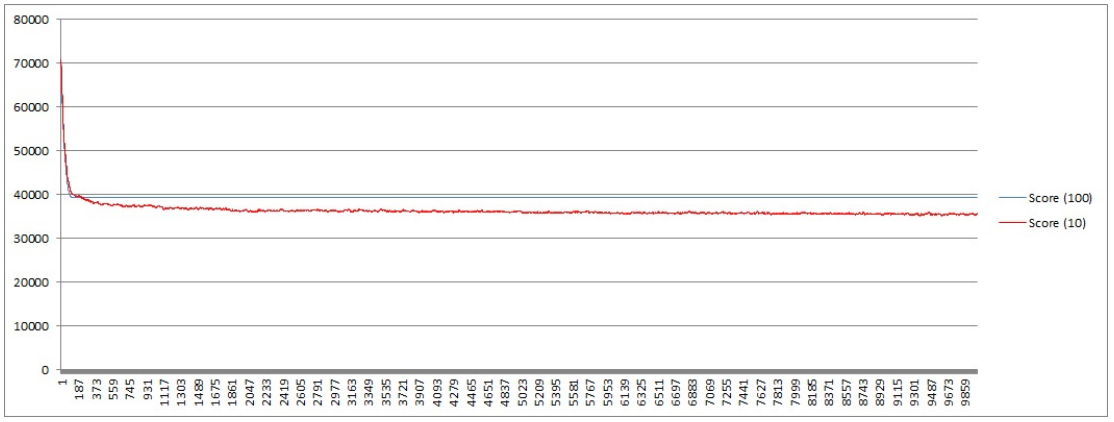

FIGURE 6.9: Test 4 result for map 5B

We can see that evaluating all possible candidate moves causes premature convergence of solution, decrease in variability, incapacity to escape local minima and therefore, worse scores. Lower values of this parameter bring faster results, better scores and higher variability.

TABLE 6.5: Results of Test 4 - CMGRatio = Candidate Move Generation Ratio Value, ET = Execution time, ET Var = Execution Time Variation in comparison with the same map with CMGRatio=10, SC = Score, SC Var = Score Variation in comparison in comparison with the same map with CMGRatio=10.

Map ID	CMGRatio	ET	ET Var	SC	Sc Var
1A	10	533	100%	5635	100%
1A	100	4583	860%	8684	154%
1B	10	284	100%	6553	100%
1B	100	2032	715%	8966	137%
2A	10	1414	100%	9818	100%
2A	100	1327	94%	12282	125%
2B	10	968	100%	9006	100%
2B	100	8445	872%	10698	119%
3A	10	325	100%	3239	100%
3A	100	2651	816%	4256	131%
3B	10	153	100%	3206	100%
3B	100	798	522%	4133	129%
4A	10	7955	100%	30848	100%
4A	100	80028	1006%	36681	119%
4B	10	6456	100%	30677	100%
4B	100	61123	947%	37542	122%
5A	10	8072	100%	37723	100%
5A	100	81252	1007%	39201	104%
5B	10	6914	100%	35074	100%
5B	100	65384	946%	39200	112%
6A	10	337	100%	2776	100%
6A	100	2749	816%	3232	116%
6B	10	158	100%	1869	100%
6B	100	981	621%	2324	124%

Table 6.5 shows that for a Candidate Move Generation ratio of 100%, the Execution Time increases 8-9 times in comparison with a value of 10%, with no score improvement. On the contrary, we can see it actually yields worse scores, so explicit search is not a good option for this problem. It is important to note, however, that very low values of the Candidate Move Generation Ratio parameter may deny the generation of any candidate points, specially if the ones that are to be generated are filtered by the hard constraint enforcement.

6.2.3.5 Test 5 - Soft Constraint Isolated Effect of Parameters in Map Visual Presentation

At Test 5 we measured the visual isolated effect of each of the soft constraints that comprise the evaluation function. We selected map 1B for the test, without inflection

point generation nor A* pathfinding algorithm enabled. These tests where performed with default settings, so the hard topological relation enforcement is used.

Map 1B presents the initial feasible solution (with grid alignment, just before entering the optimization phase) is presented at figure 6.10.

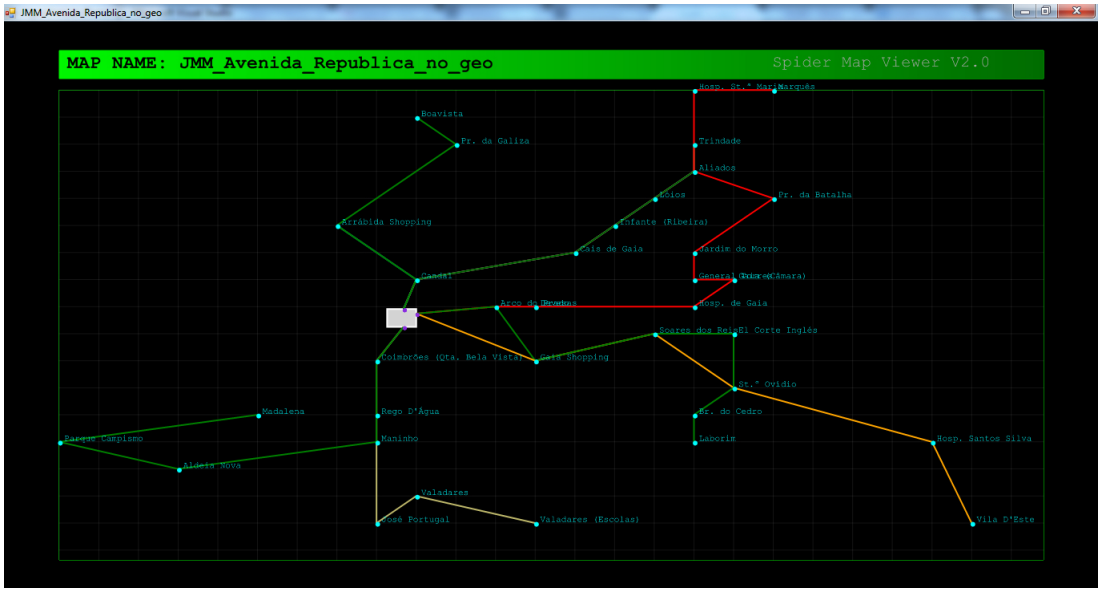

FIGURE 6.10: Test 5 - Initial embedding of Map 1B (non processed)

We can see the results in figures 6.11 to 6.16.

Figure 6.11 show that SC1 constraint causes the adjacent edge angles to be as wide as possible to improve reading and map elements distribution.

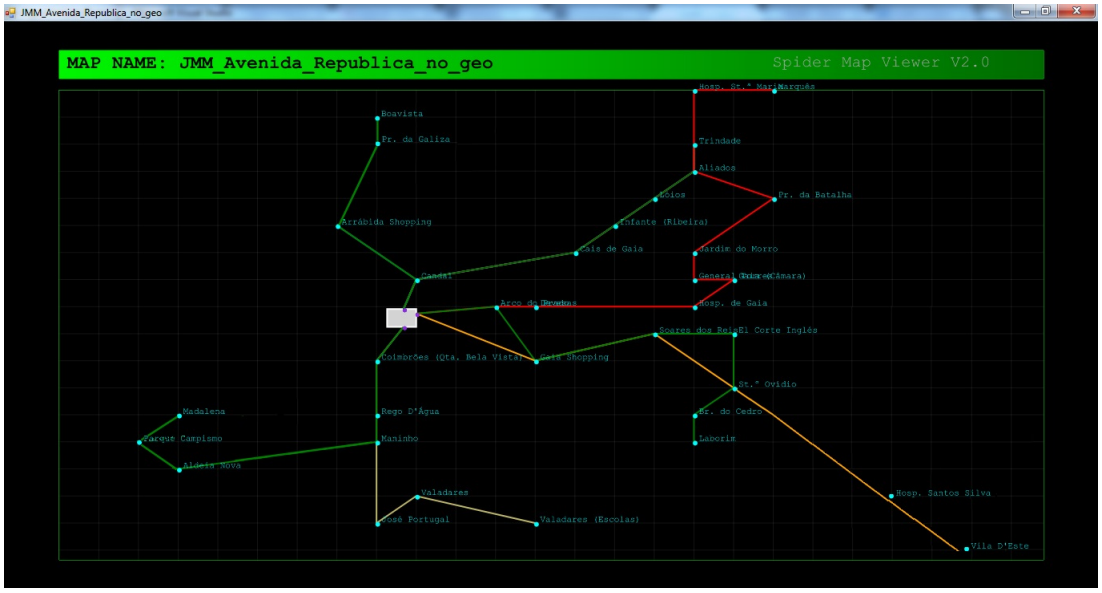

FIGURE 6.11: Test 5 - Effect of Angular Resolution Soft Criteria

Figure 6.12 shows that SC2 constraint tries to put all vertexes with the same distance from its predecessor and successor in a transportation line.

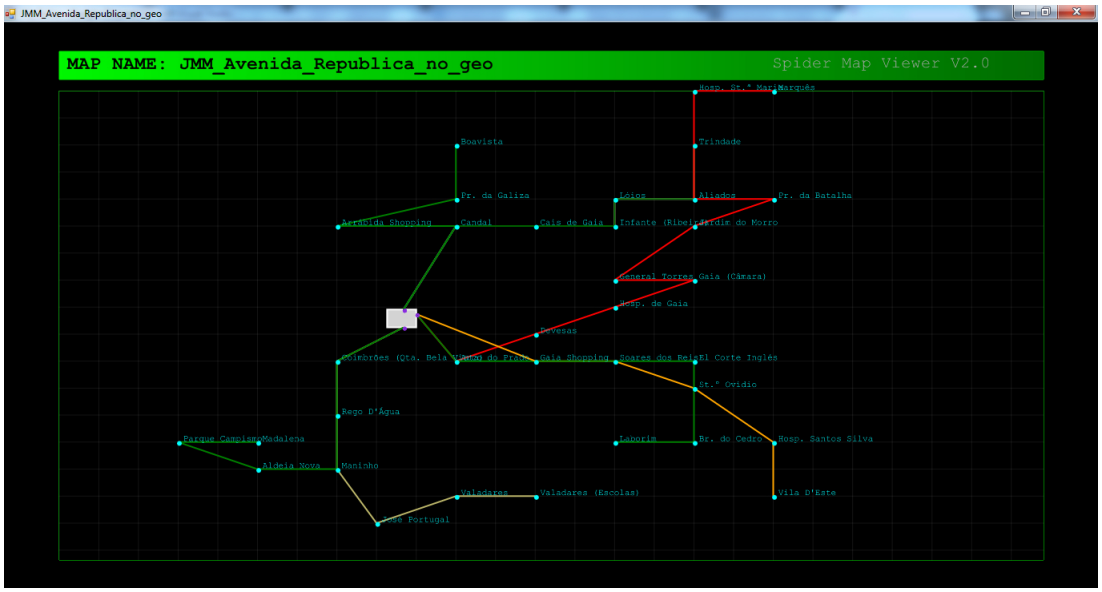

FIGURE 6.12: Test 5 - Effect of Edge Length Soft Criteria

Figure 6.13 shows that SC3 constraint tries to force consecutive vertexes to be at a certain distance (user definable parameter).

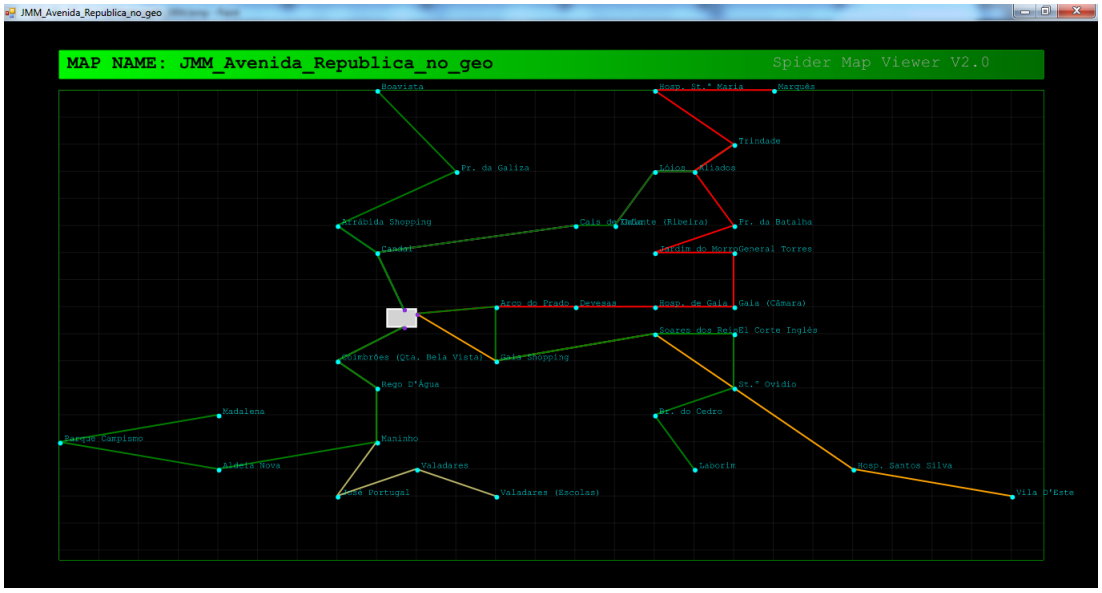

FIGURE 6.13: Test 5 - Effect of Edge Balance Soft Criteria

Figure 6.14 shows that SC4 constraint tries reduce as much as possible the edge crossings.

Figure 6.15 shows that SC5 constraint tries to put lines as straight as possible (avoiding line bendings).

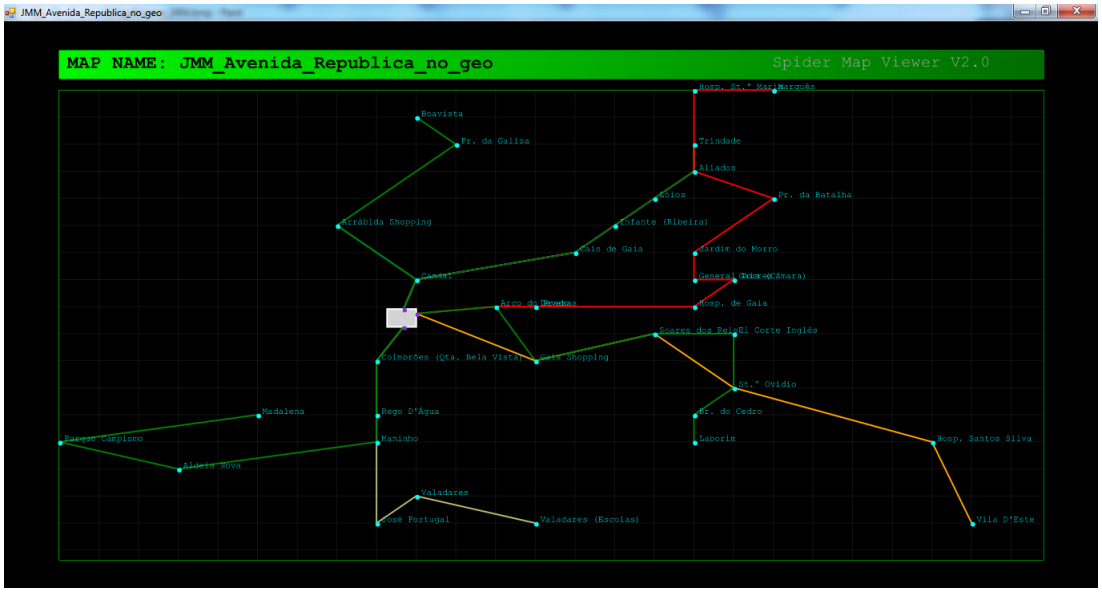

FIGURE 6.14: Test 5 - Effect of Edge Crossings Soft Criteria

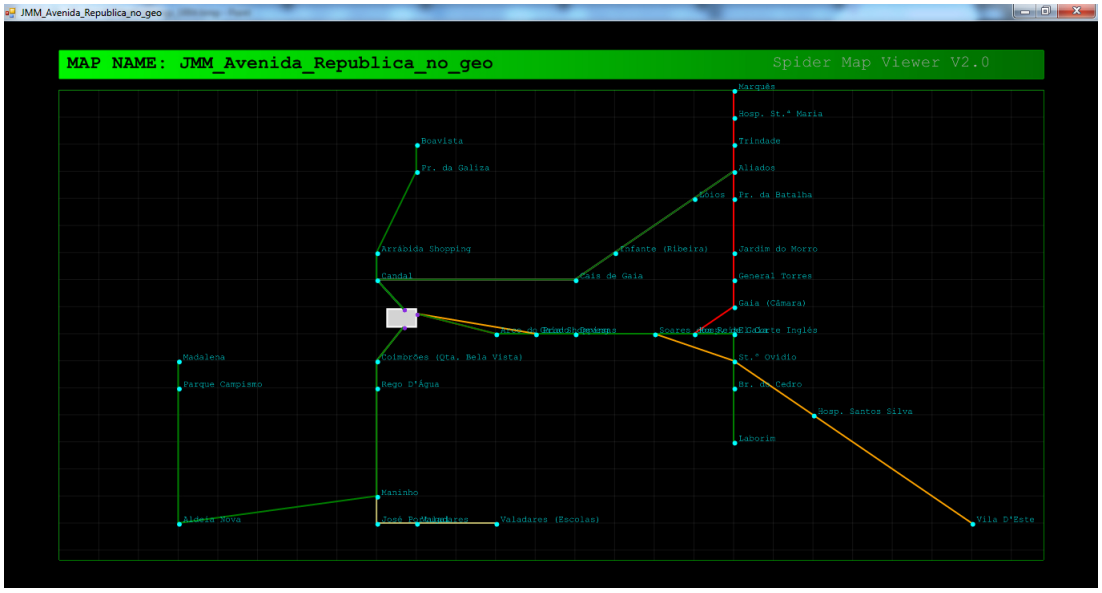

FIGURE 6.15: Test 5 - Effect of Line Straightness Soft Criteria

Figure 6.16 shows that SC6 constraint tries to put edges as much as possible to comply with a 45 multiple degree schema.

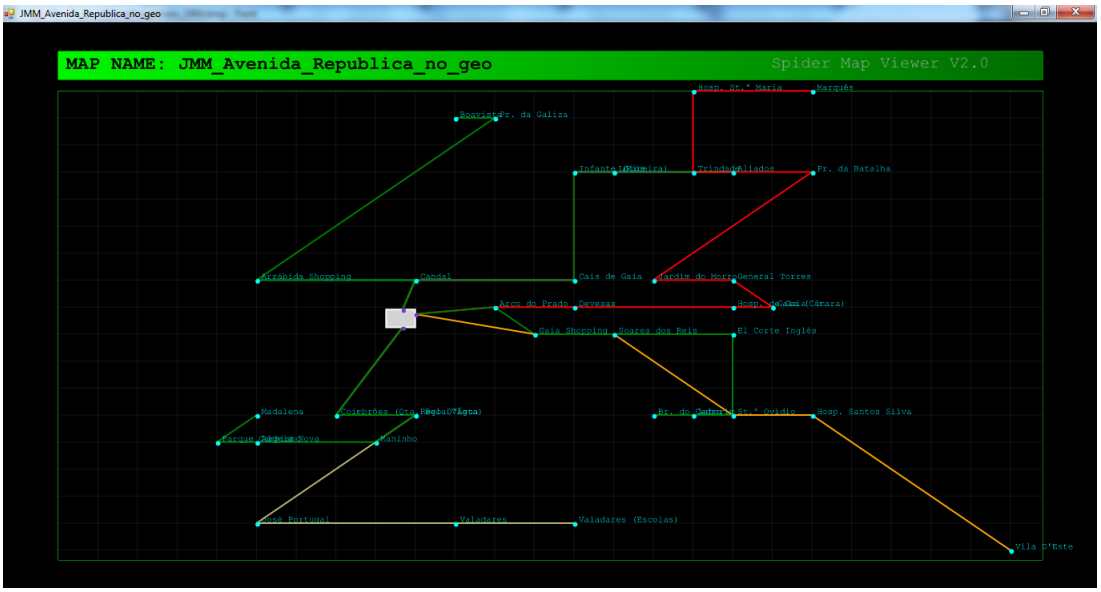

FIGURE 6.16: Test 5 - Effect of Octilinearity Soft Criteria

Balancing the weights of the soft constraints is useful for tuning the algorithm for particular maps or purposes, or to highlight specific map features.

6.2.3.6 Test 6 - Hard versus Soft Topological Relation Enforcement

At Test 6 we tested how treating the topological relations as a soft or hard restriction impacts the execution time and map visual quality. We have toggled the topological relations switch to soft or hard at the parametrization window and tested with default settings. Figures 6.17 and 6.18 show the obtained maps after 10000 iterations. It is possible to see that the topological relations are not kept in map 6.18, as they are compromised to obtain the best visual possible layout (even if it decreases usability and map learnability). In our algorithm the soft constraint weight is also adjustable, so reducing it can bring benefits. If we use the topological soft constraint option in conjunction with SC5 and SC6 weights, it is possible to obtain really visually beautiful maps (but probably with usability and user disorientation issues, depending the purpose of the map).

Table 6.6 compares the performance indicators for this test.

Table 6.6 shows that the soft topological relations option causes a big increase in Execution Time (about 30 times) due to increased degree of vertex movement freedom that hard topological relations enforcement do not provide, so less candidate moves are

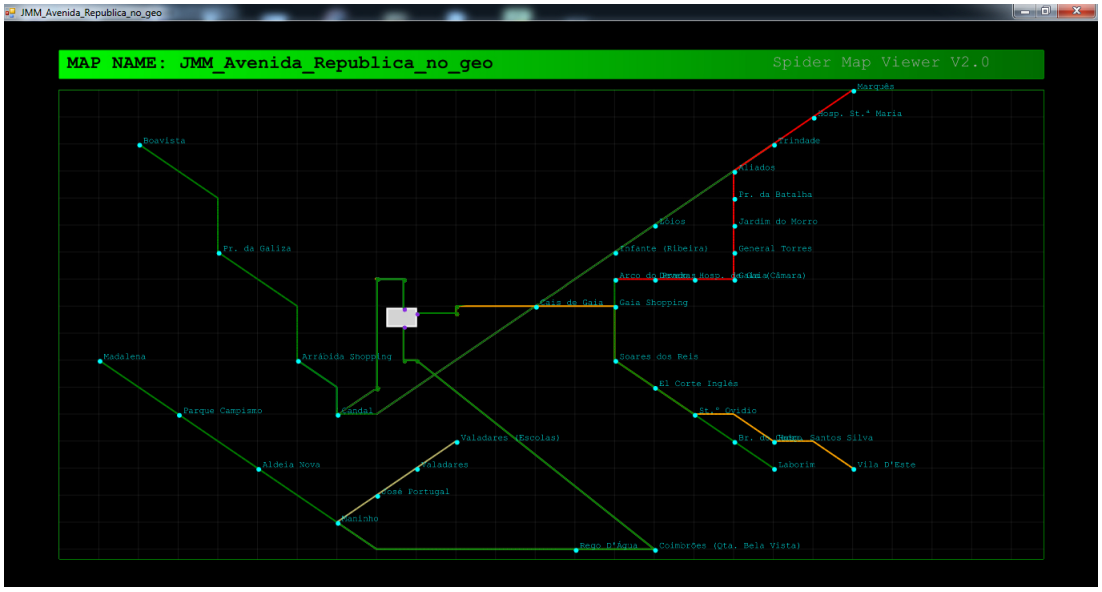

FIGURE 6.17: Test 6: Effect topological relations enforcement as a hard constraint

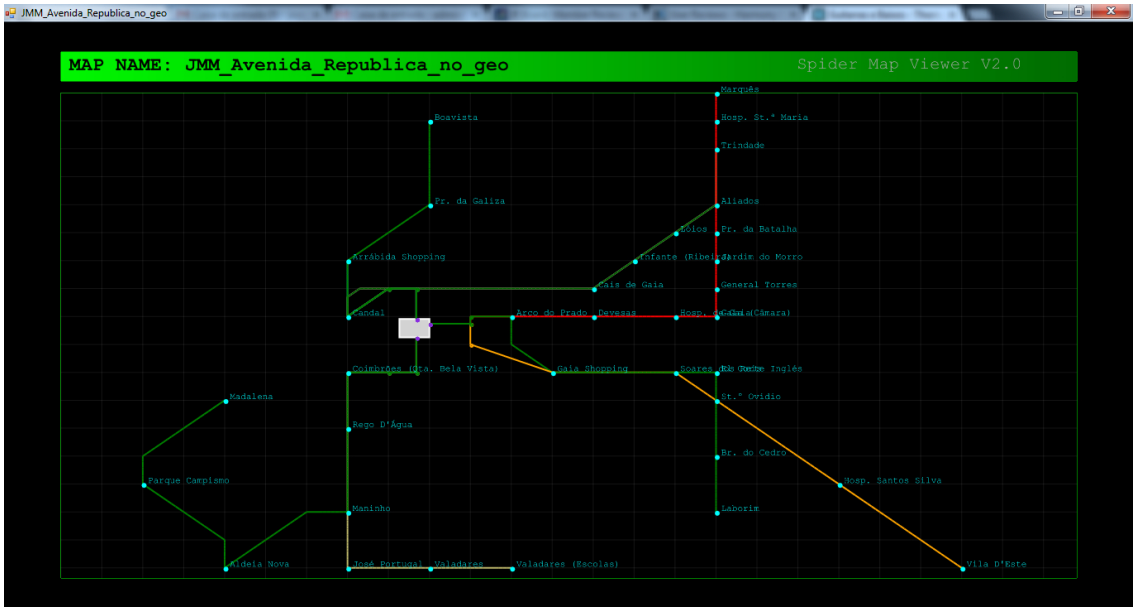

FIGURE 6.18: Test 6: Effect topological relations enforcement as a soft constraint

TABLE 6.6: Results of Test 6. ET - Execution Time, ET Var - Execution Time Variation, SC Var - Score Variation

Map ID	Topological Relations	ET	ET Var	Score	Sc Var
1A	Hard	284	100%	6071,701	100%
1A	Soft	9621	3388%	3968,649	65%

discarded before evaluation. We have to note that even if the score may be better, in practice, violating the topological relations can cause usability problems and user disorientation, event if the map may look better aesthetically. One suggestion would be to decrease the value of the Maximum Vertex Displacement parameter in conjunction with soft topological relations to prevent those issues.

6.2.3.7 Test 7 - A* Pathfinding Overhead

At Test 7 we assessed the A-Star pathfinding execution time overhead when finding paths around geographical accidents. Table 6.7 shows the variation the execution time of the post processing phase for each map with A* activated or deactivated.

TABLE 6.7: Results of Test 7

Map ID	A* Pathfinding Enabled	Post Processing ET
1A	Yes	6
1A	No	6
1B	Yes	4
1B	No	4
2A	Yes	15
2A	No	15
2B	Yes	11
2B	No	11
3A	Yes	3
3A	No	3
3B	Yes	2
3B	No	2
4A	Yes	88
4A	No	84
4B	Yes	76
4B	No	78
5A	Yes	98
5A	No	93
5B	Yes	85
5B	No	84
6A	Yes	4
6A	No	4
6B	Yes	2
6B	No	2

It is possible to conclude that A* Algorithm pathfinding execution time overhead is neglectable. This happens because A* is a trim, effective and highly efficient algorithm, used in many performance-critical applications.

6.2.3.8 Test 8 - Spatial Distribution Smart/Blind Algorithm Evaluation

At the final test we assessed the effect of the spatial distribution algorithm with the intelligent/blind parameters in map quality and execution time, for 10000 iterations for every map. The results are stated in table 6.8. For abbreviations purposes, "No SDA" means "No Spatial Distribution Analysis", "Sc" is the score, "Sc Var" is the score percentage (regarding the corresponding score with no Spatial Distribution Analysis), "ET" is the Execution Time and "ET Var" is the relative Execution Time regarding the "No SDA" Execution Time. Best values for Execution Time and Score are highlighted in bold for each map. The data show that in average, the intelligent approach can achieve relativescores of 87%, much better than without the SDA, while the increase in Execution Time is only about 1%. The blind approach can achieve similar scores (slightly worse, but with a more significant increase in the Execution Time (9%). Comparing intelligent and blind approaches we can see their main difference is the Execution Time: while the blind approach executes the Spatial Distribution Algorithm every n iterations, the intelligent approach only executes the Spatial Distribution Algorithm when no score improvement is detected, so it runs only when needed. This explains the much better Execution Time. This means that the Spatial Distribution Algorithm it is an algorithm that can effectively introduce diversification and variability. The graph in figure 6.19 confirms that SDA can yield better scores for every map, and if we use the intelligent version, there is no significant performance loss in what concerns to Execution Time.

TABLE 6.8: Results of test 8 (ID = Map ID)

	No SDA		**Intelligent**				**Blind**			
ID	**Sc**	**ET**	**Sc**	**Sc Var**	**RT**	**RT Var**	**Sc**	**Sc Var**	**RT**	**RT Var**
1	7245	533	**5632**	78%	**522**	98%	6237	86%	581	111%
2	6553	**284**	**5191**	79%	310	109%	5921	90%	576	186%
3	**9211**	**1414**	9318	101%	1527	108%	9295	101%	1443	94%
4	10016	968	9510	95%	936	97%	**8939**	89%	**899**	96%
5	5925	325	3684	62%	**323**	99%	**2801**	47%	336	104%
6	5166	**153**	**3620**	70%	177	116%	4451	86%	169	95%
7	31727	7955	29538	93%	**7865**	99%	**28763**	91%	8307	106%
8	32787	**6456**	29263	89%	7052	109%	**27875**	85%	7771	110%
9	38807	8072	**35464**	91%	**7938**	98%	36434	94%	8318	105%
10	37094	6914	36447	98%	**4264**	62%	**34275**	92%	4476	105%
11	2776	**337**	**2283**	82%	361	107%	2809	101%	356	99%
12	2725	158	2716	100%	175	111%	**2680**	98%	172	98%
			AVG	87%	**AVG**	101%	**AVG**	88%	**AVG**	109%

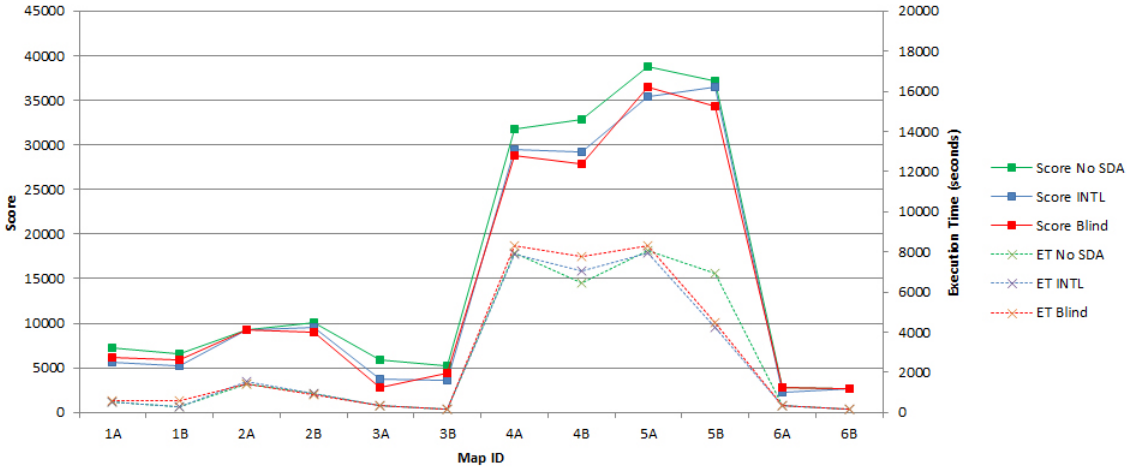

FIGURE 6.19: Test 8: Graphical comparison of Spatial Distribution Algorithm performance variation

6.3 Real World Use

The maps generated by the GenX framework are already being used in real world. In Porto, they have been published, for example, at the Hospital de São João area, with the spider map depicted in figure 6.20.

This map is used to depict the bus transportation lines around the Hospital de São João Hub. Figure 6.21 shows the map being used in reality.

.

Our algorithm is also being used to create spider maps for Lisbon (figure 6.22), for Rossio, Marquês de Pombal and Belém zones. Spider maps can also be found at Santo Tirso (figure 6.23).

All of these projects were developed in cooperation with OPT, FWT and public transportation companies. It was the first time that this complex network representation scheme was published in Portugal [44] being designed through a system that creates spider maps in an automatic way.

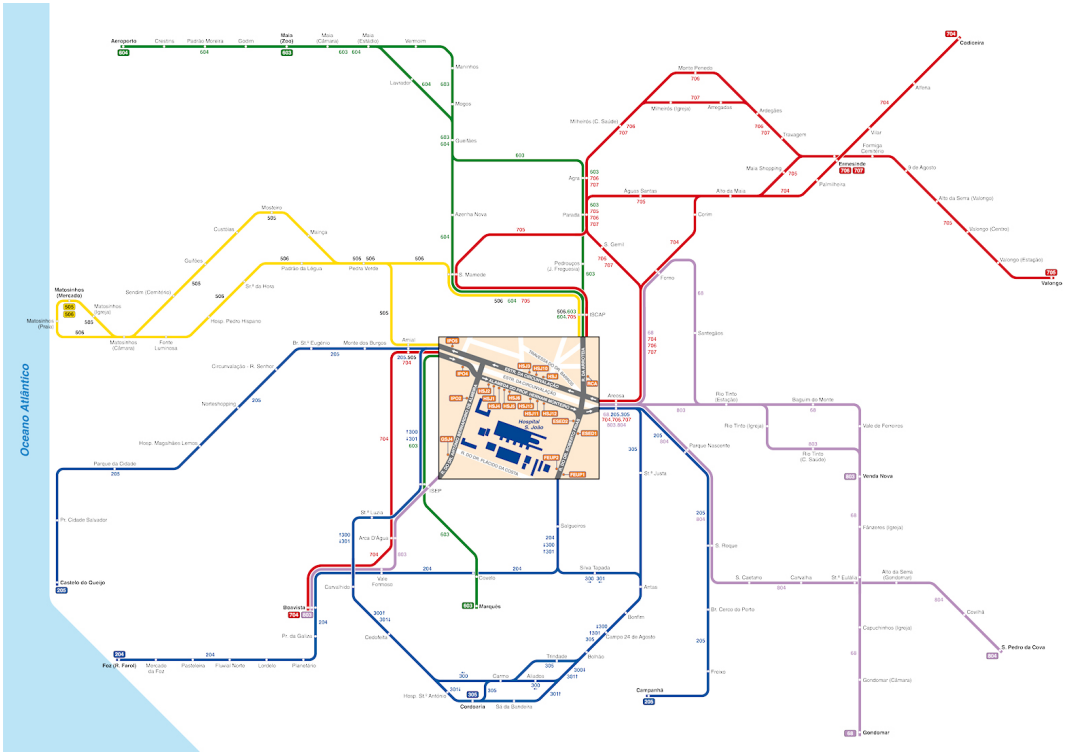

FIGURE 6.20: Example of a Spider Map generated through the GenX framework. The hub corresponds to the downtown of the City of Porto. (Published by STCP, scaled down on a 1:7 proportion)

FIGURE 6.21: The map depicted in figure 6.20 being used in Porto [44].

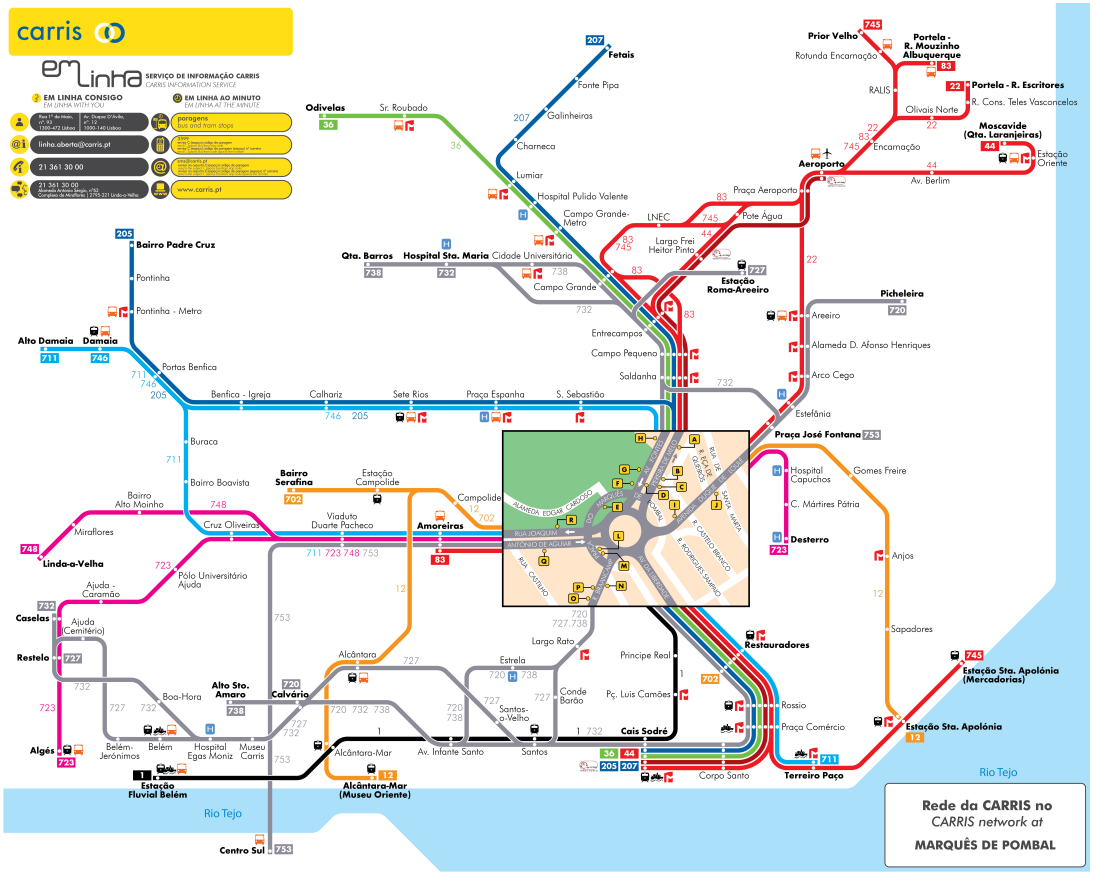

FIGURE 6.22: Example of a published Spider Map of Lisbon (Marquês de Pombal area) generated through the GenX framework scaled down on a 1:7 proportion

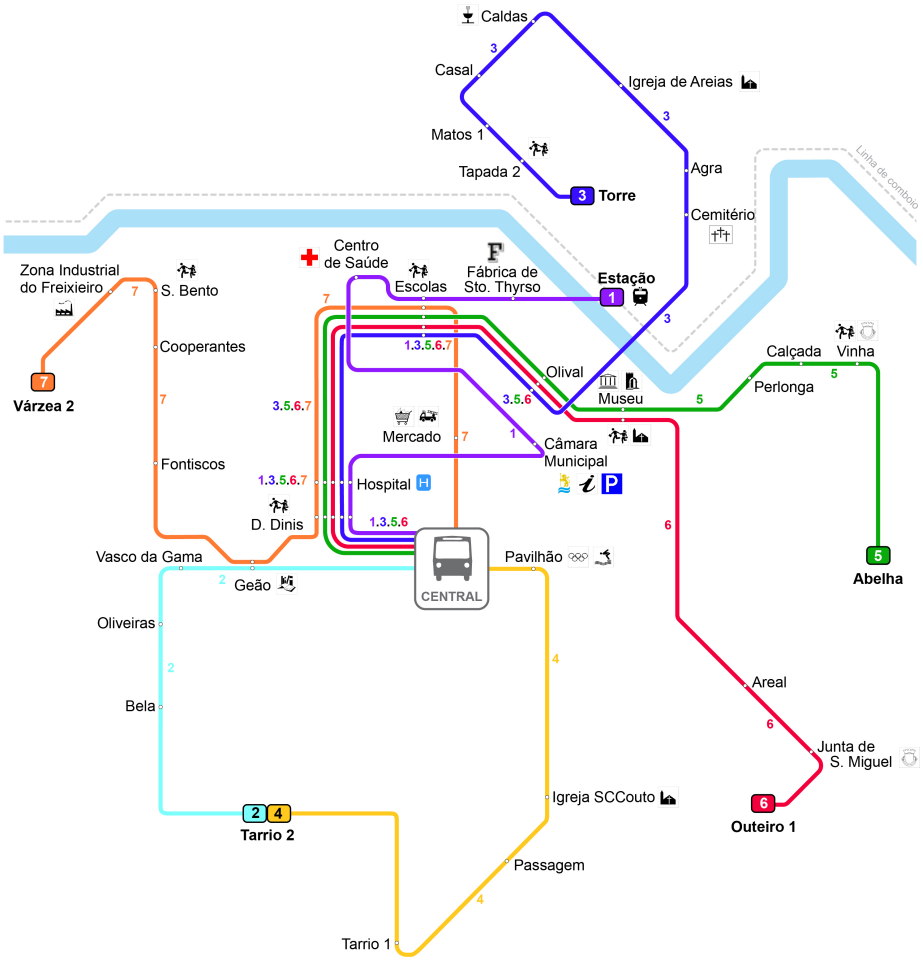

FIGURE 6.23: Example of a published Spider Map of Santo Tirso generated through the GenX framework scaled down scaled down on a 1:8 proportion

Chapter 7

Conclusions

This chapter provides a summary of the main contributions of this thesis and raises some questions for further research.

7.1 Summary

This research work defines the concept of spider maps and provides evidence of their effectiveness in what concerns to context awareness regarding traditional maps, through theoretical research and an empirical study supported by mixed user testing involving closed and open questions, following the best practices found on the literature. We showed through user testing that spider maps are more effective for self location, route planning and navigation tasks and that they are preferred by users over traditional diagrammatic maps as they find them to be simpler and clearer.

This research work also proposes an approach for the automatic generation of spider maps, and it can be used to power agile Location-Based Services (besides traditional map visualization applications).

Our approach is based on a tabu search procedure where at each iteration the transportation nodes are moved, through a combination of hard constraints and an evaluation function comprised of soft constraints, and an innovative Spatial Analysis Distribution algorithm. This approach comprises three phases: the Initialization and Alignment to Grid, the Tabu Search optimization metaheuristic and the Post Processing phase. At the Initialization and Alignment to Grid, the transportation map to be processed is read, and the adequate data structures are created. The map is prepared to include the Hub according to the user specification. The graph structure of the transportation map is then embedded in a regular grid through the SmartFit and HPPO algorithms, and at the

end of this phase, the first feasible solution is obtained. At the Tabu Search Optimization Meta Heuristic phase, this solution is improved iteratively according to a model of the problem that contains a set of hard constraints and an optimization function that comprises a set of soft constraints. Geographical constraints assure that geographical elements (such as rivers, parks, oceans, etc) can be successfully accommodated in the spider map layout. Variability may be introduced at this phase automatically as needed through the Spatial Distribution Analysis de-clustering algorithm. The map is then subjected to the Post Processing Phase, where the edges are routed around the geographic constraints through an implementation of the A* algorithm and inflection points are introduced.

The effectiveness of our approach was shown by a comprehensive set of tests with real world data with and without geographical restrictions (supported by the evidence that it is already being used in real world in cities like Lisbon, Porto and Santo Tirso) for the generation of spider maps). It was also shown that maps with geographical restrictions do not impose significantly higher processing times. One key point of our approach is that due to its automatic grid alignment and post processing algorithm combinations, it always generates a solution that complies with the definition of spider map. It is also worth to mention that the breakthrough algorithms we created and all the research work can be applied not only to the automatic generation of spider maps, but also to traditional schematic maps. The innovative grid alignment algorithms, de-clustering and the combination of the A* algorithm with differential grid aperture can be highly valuable tools for solving other type of problems such as VLSI and network planning.

7.2 Limitations of this Work

Although this was a comprehensive research work on the problem of the automatic generation of spider maps, (either in broadness and deepness), it presents three limitations:

- This research work does not focuses on node labeling. Node labeling strategies are found in the literature using both multicriteria optimization [26] or MIP approaches [87] [91] [92] and they can also be applied to our approach.
- The line separation is not modeled in our approach.
- The weighting of the soft constraints was not subject of a deep analysis. We used Stott's predefined values for most of the constraints [26] as good weight values, although different weightings could provide improved quality and/or speed for specific maps.

- Parametrization by the user is still necessary: more sophisticated methods such as neuronal networks could be applied on our approach to extract the best weightings for each soft constraint and to understand what parameters values worked best for specific type of maps.

7.3 Main Contributions of this Thesis

We can summarize the main contributions of this thesis in three areas:

- Development of the Spider Map Concept
- Automated Spider Map Generation
- Value to Society

7.3.1 Spider Map Concept

This thesis has provided several contributions concerning map production, schematic maps, context enhancement,

- **Literature Review:** a comprehensive perspective of the map production history and state of the art.
- **Definition and Proof of Concept:** a comprehensive definition was provided and their roots and origins were traced. Their properties were also elicited and summarized. The demonstration of their superiority regarding traditional diagrammatic maps was well supported by empirical and theoretical evidence, as well as their potential use in powering Location Based Services.
- **Enhancement:** besides the literature review, we included context enhancement strategies to spider maps. By applying communication, learning and context enhancement strategies, it was possible to build more effective maps in what concerns to communication of space, user learning and interaction. Context enhancement (a topic to which this thesis contributes) is the key for better space communication, and our user tests and theoretical framework support this evidence.

7.3.2 Automated Spider Map Generation

The generation of Spider Maps through the use of a tabu search algorithm contributed to:

- **State of the Art:** Our research effectively contributed to the state of the art in metaheuristics for schematic map generation process improvement, as advancements were made in all phases of the automated generation of spider maps.
- **Data Modeling:** The use of both SRDS and SGDS allows faster map processing while retaining all the semantics behind a transportation map.
- **Algorithm Enhancements** The improvements over existent literature begin with the the Alignment to Grid phase, where the SmartFit and HPPO algorithms presenting enhancements in what concerns to graph grid embedding and improving user interaction (by saving time to the user to decide what would be the best grid aperture). The use of optimized algorithms such as the Bentley-Ottmann algorithm to improve the line crossings and the development from root of the Spatial Distribution Analysis algorithm to improve variability on the tabu search phase are significant improvements to the Optimization phase. The use of the A* algorithm for pathfinding and inflection point creation are improvements to the post-processing phase.

7.3.3 Value to Society

The implications of this work for the society can be traced in several domains such as as public transportation and economy and sustainability:

- **Real World use:** Our approach is already producing real world maps, available and used in the major Portuguese cities.
- **Economy:** the reduction on map production costs and decrease in lead times bring benefits for both the transportation companies and end users.
- **Public Transportation Use and Sustainability:** in a world where about 80% of the people live in urban areas [157], public transportation systems are fundamental to support efficient mobility and high quality of urban life [139]. Spider Maps were shown at this research that they reduce information overload which improved user experience and increase information quality, which by its turn leads to improvement of Public Transportation Ridership[54], contributing to a more sustainable world.

- **Accessibility:** Our approach can be used to produce Spider Maps tailored for specific user groups (for example for kids, colorblind people, etc), as different maps can be generated for different purposes in minutes, improving public transportation accessibility.
- **Transportation Network Bottleneck Management:** benefiting from the quick generation of schematic maps, bottleneck management could be achieved by producing different map accordingly to the transportation network load. By presenting only the best options for the users, it is possible to achieve managed load balancing in networks.

7.4 New Insights for Future Research

Future research on the topic of this thesis could be made at several areas, to address its limitations:

- **Spider Map Concept**: The Spider Map is not the perfect map, it still is a "static map" as after produced it cannot change (another instance needs to be created if conditions change). Therefore, an entirely electronic and dynamic Spider Map concept would be a very desirable evolution of the spider map concept. Electronic spider maps that can change dynamically with respect to the user current location would create new challenges in what concerns to context enhancement strategies and visual presentation, that would lean more towards a user-centered perspective.
- **New Media Types:** With the massification of mobile and wearable devices, there is room for evolution in what concerns to dynamic spider maps: context enhancement in a hard real time use would face new opportunities such as pervasive learning of user preferences. Some techniques could be studied such as the use of Interactive Compression (IC) techniques [158] which could allow users to select the right amount of information on the spider map in mobile devices, or user modeling techniques that could provide automated adaptation of the spider map to each specific user, making use of the MPEG-21 Digital Item Adaptation (device adaptation strategies) while assuring intellectual property management and protection. Besides that, Location-Based Services can take advantage of Spider Maps to improve user locations through mobile devices or even with augmented reality glasses, or any other device that can provide visual description of map.
- **Algorithm Improvements**: future processing of spider maps would benefit from allowing them to be generated at user devices, so future algorithms need to take

in account the nature of these devices. Algorithms tailored for multicore architectures, simpler and lighter in such a way they can produce visual and incremental results in hard real time. Machine learning algorithms could be used to allow maps to be generated without parameter insertion by users. The algorithms will probably need to be designed in such a way they can blend reality with map visualization. This means that a shift from static, "full solution" traditional algorithms towards dynamic, simpler, incremental solution and real time compliant algorithm paradigms is something that will be needed in the future.

Being men and women bounded by a infinitely small space and time frame makes us humble enough to know that the true wisdom is in knowing we know nothing, as the Greek philosopher Socrates said. With these broad topics open, there is infinite space for further research!

Appendix A

Map 1 - Avenida da República

This appendix illustrates the raw map 1 (figure A.1), pre-processed with (figure A.2) or without (figure A.4) geographical restrictions and processed by our approach with (figure A.3) or without (figure A.5) geographical restrictions.

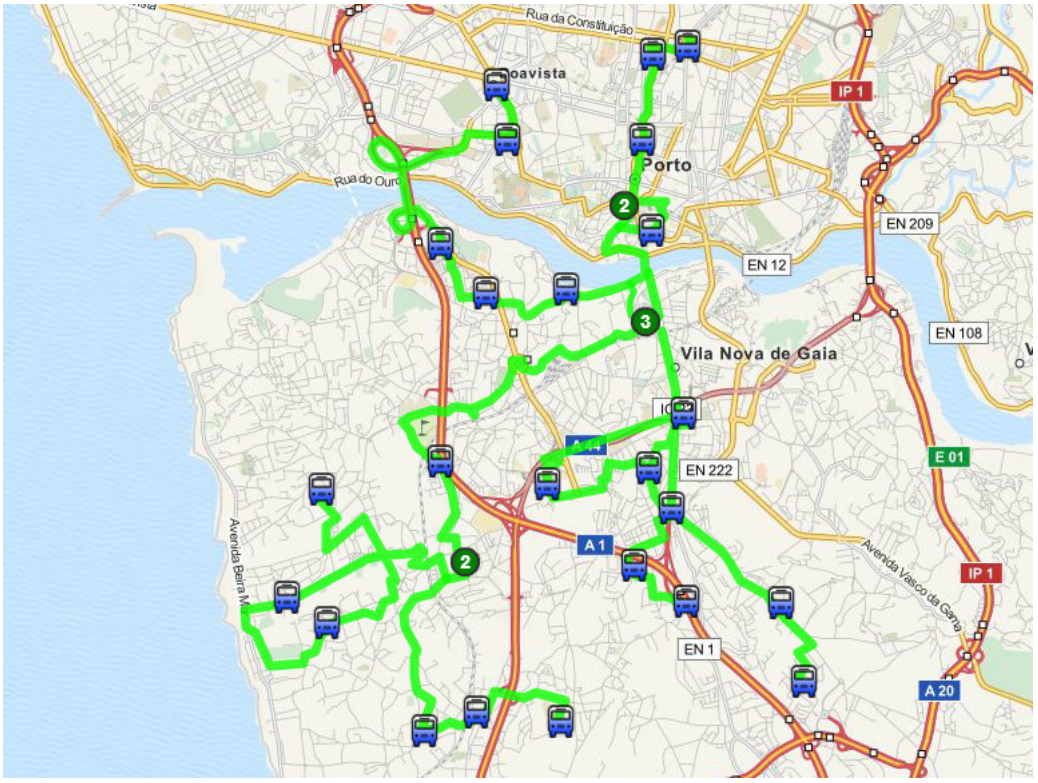

FIGURE A.1: Raw Map 1 - Avenida da República

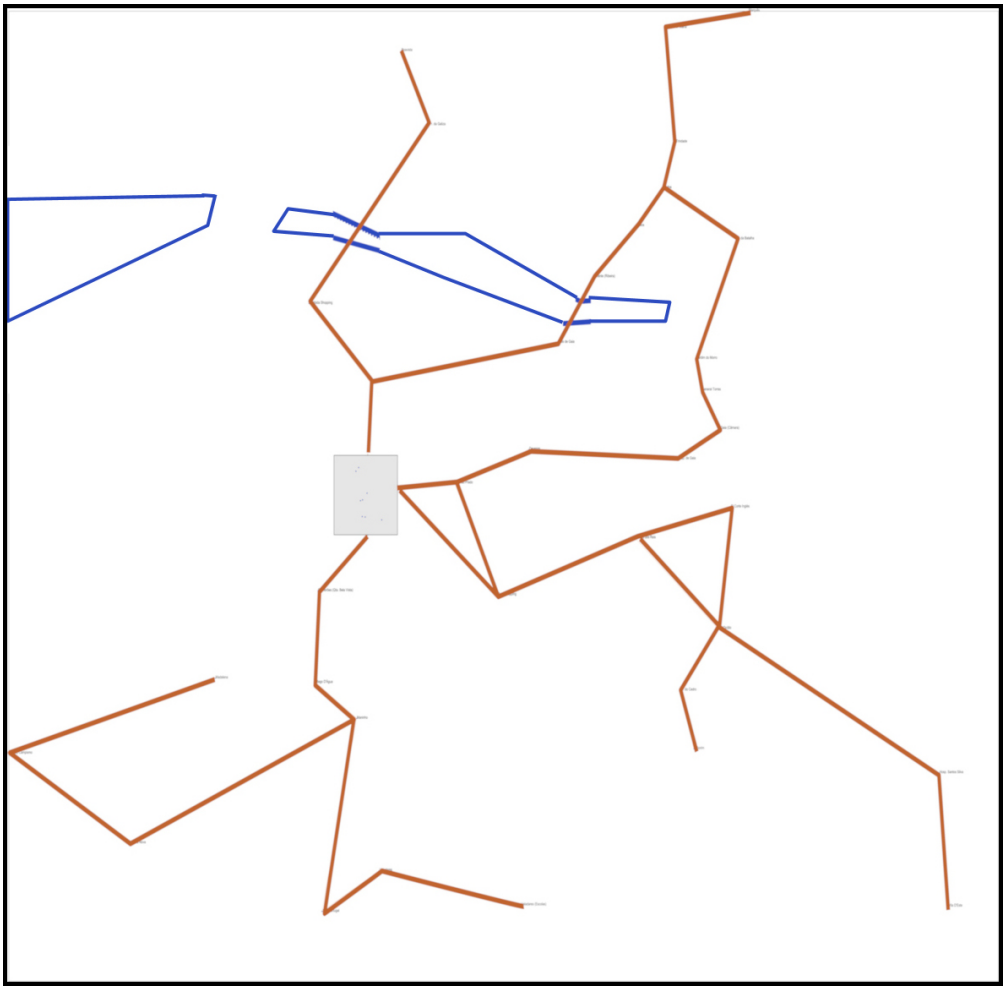

FIGURE A.2: Pre-Processed Map 1A - Avenida da República with geographical restrictions

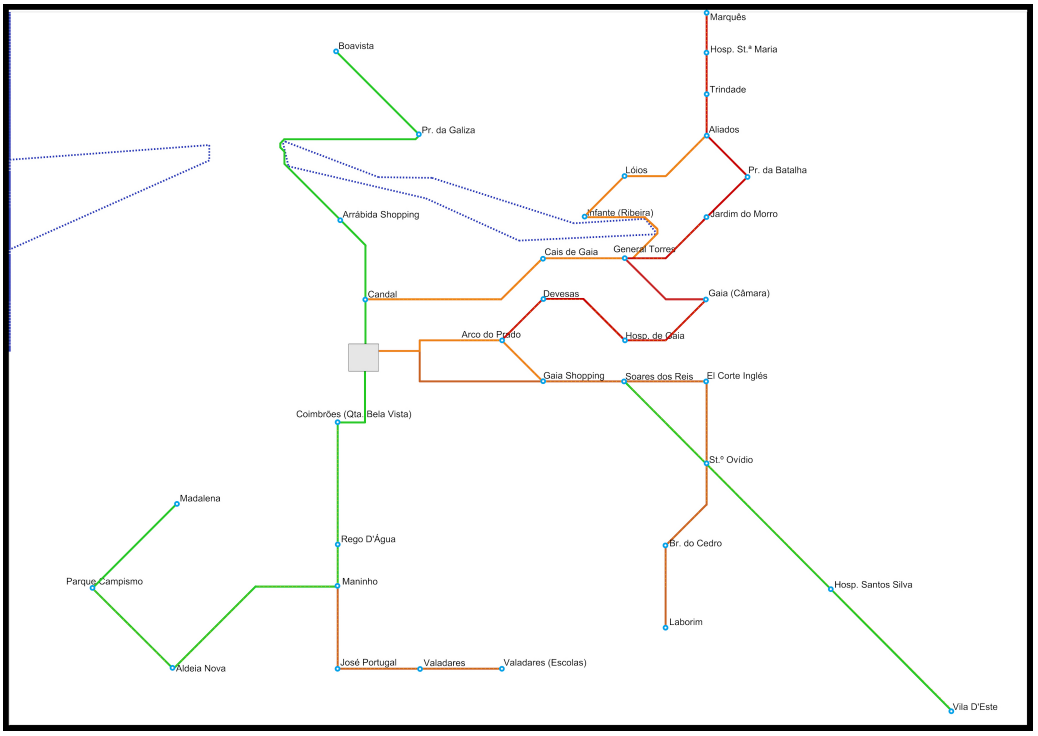

FIGURE A.3: Processed Map 1A - Avenida da República with geographical restrictions

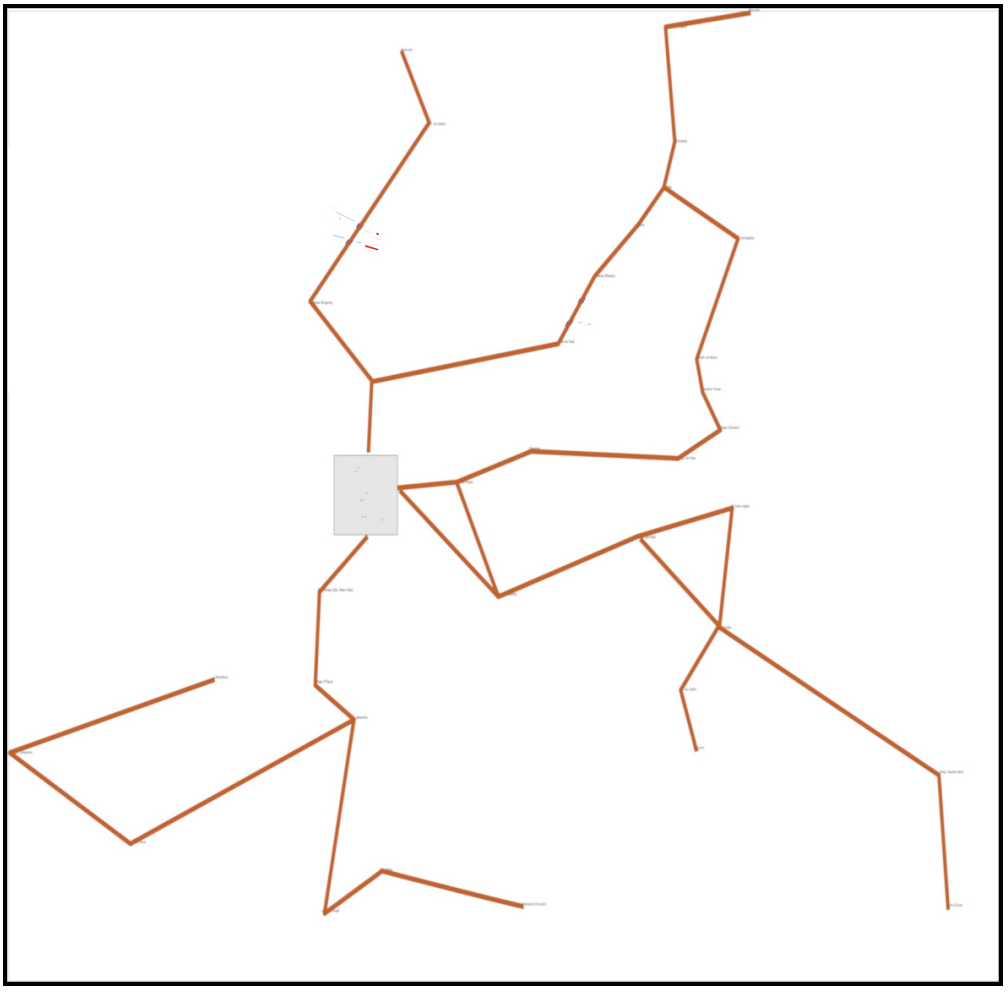

FIGURE A.4: Pre-Processed Map 1B - Avenida da República without geographical restrictions

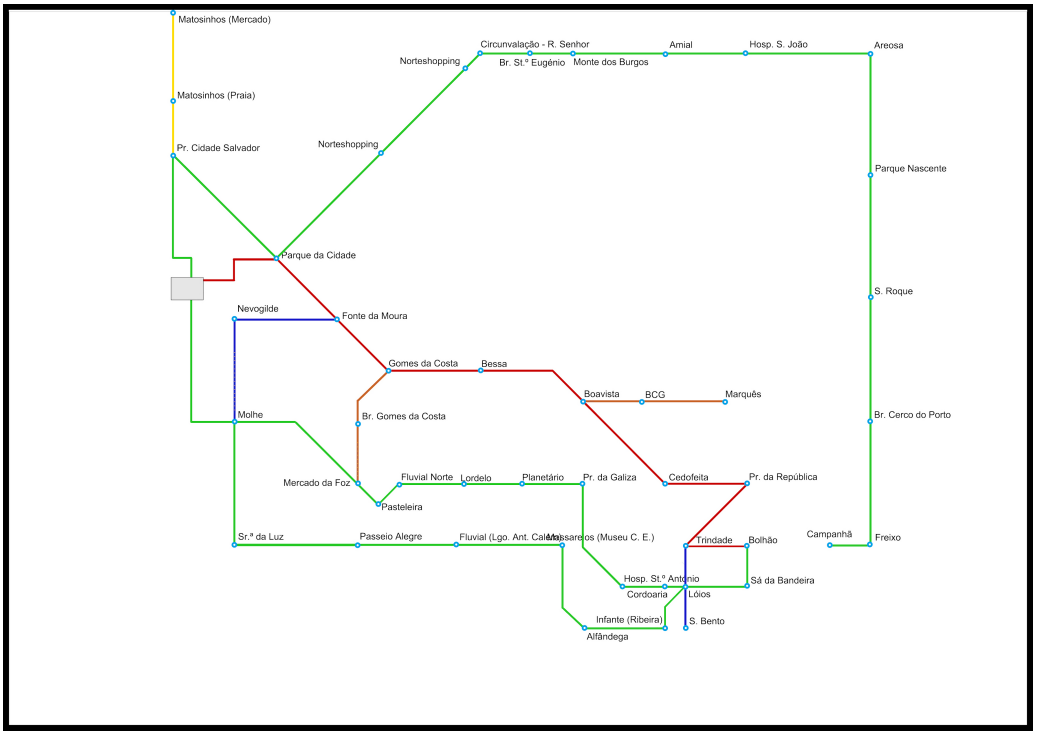

FIGURE A.5: Processed Map 1B - Avenida da República without geographical restrictions

Appendix B

Map 2 - Castelo do Queijo

This appendix illustrates the raw map 2 (figure B.1), pre-processed with (figure B.2) or without (figure B.4) geographical restrictions and processed by our approach with (figure B.3) or without (figure B.5) geographical restrictions.

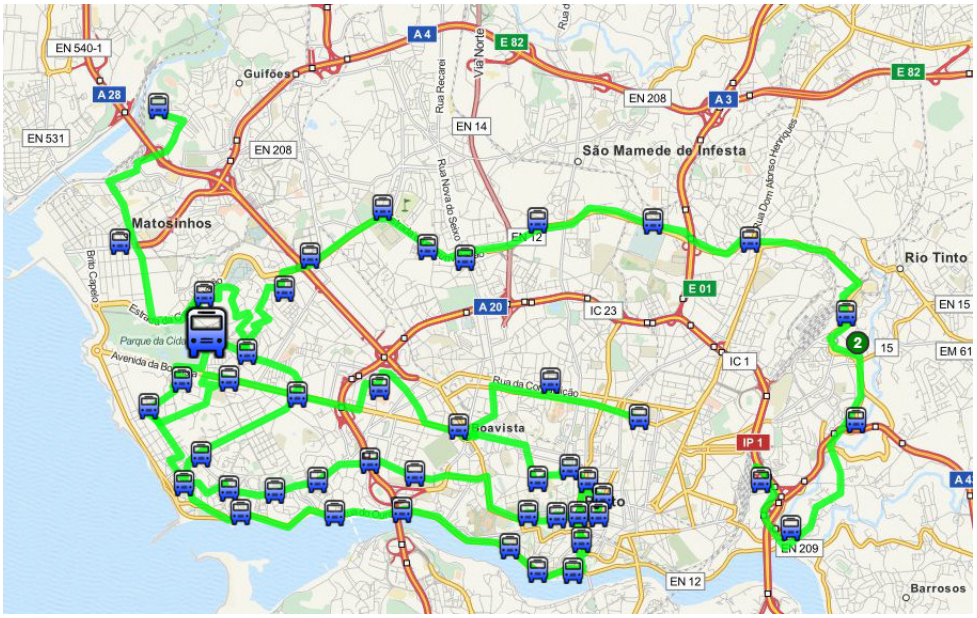

FIGURE B.1: Raw Map 2 - Castelo do Queijo

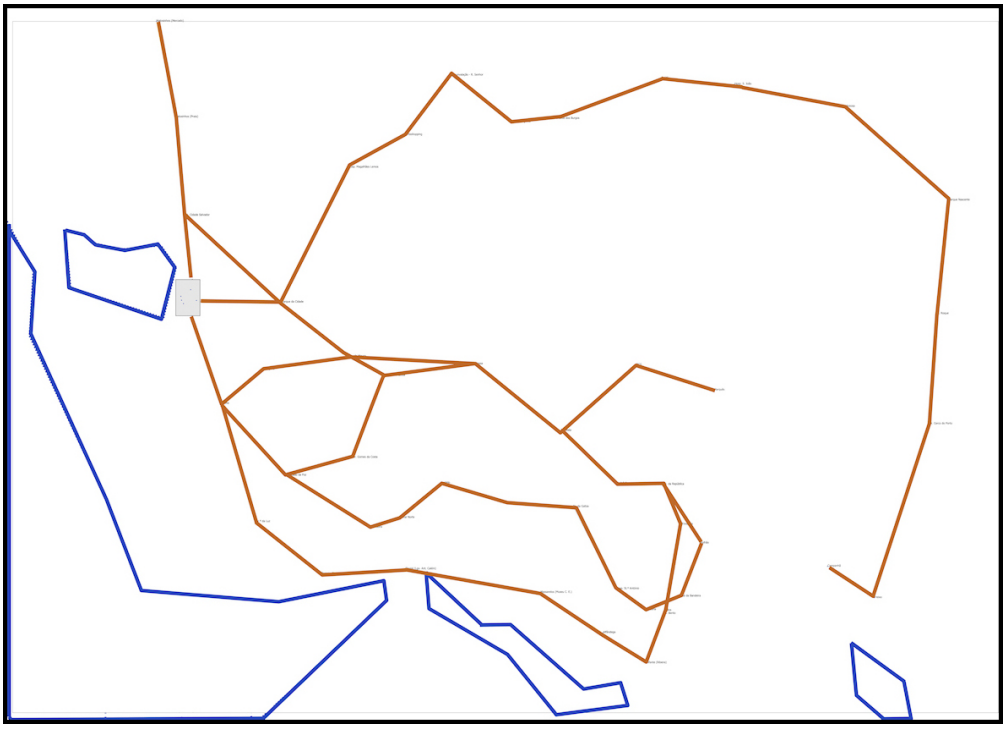

FIGURE B.2: Pre-Processed Map 2A - Castelo do Queijo with geographical restrictions

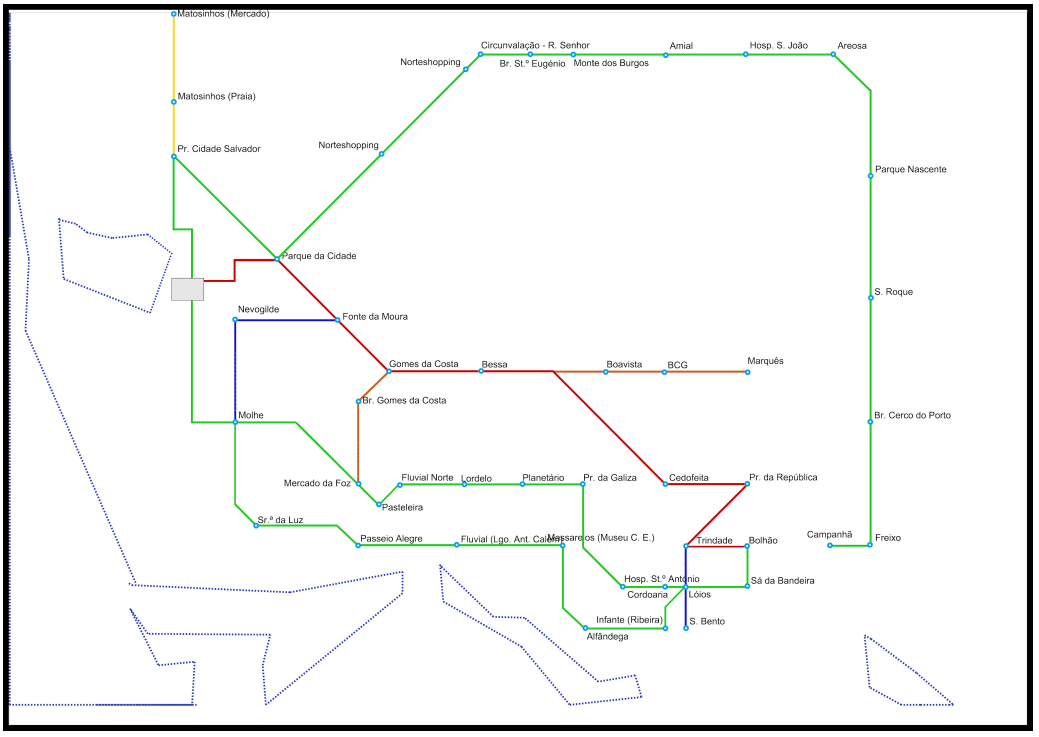

FIGURE B.3: Processed Map 2A - Castelo do Queijo with geographical restrictions

FIGURE B.4: Pre-Processed Map 2B - Castelo do Queijo without geographical restrictions

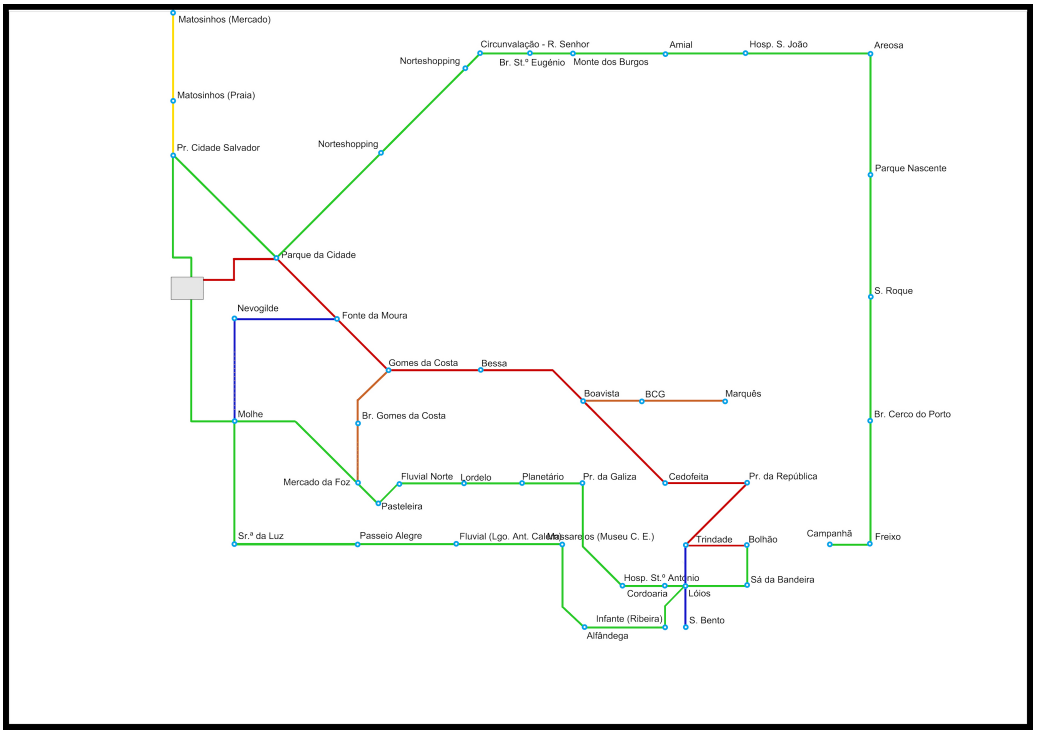

FIGURE B.5: Processed Map 2B - Castelo do Queijo without geographical restrictions

Appendix C

Map 3 - Pólo Universitário

This appendix illustrates the raw map 3 (figure C.1), pre-processed with (figure C.2) or without (figure C.4)geographical restrictions and processed by our approach with (figure C.3) or without (figure C.5) geographical restrictions.

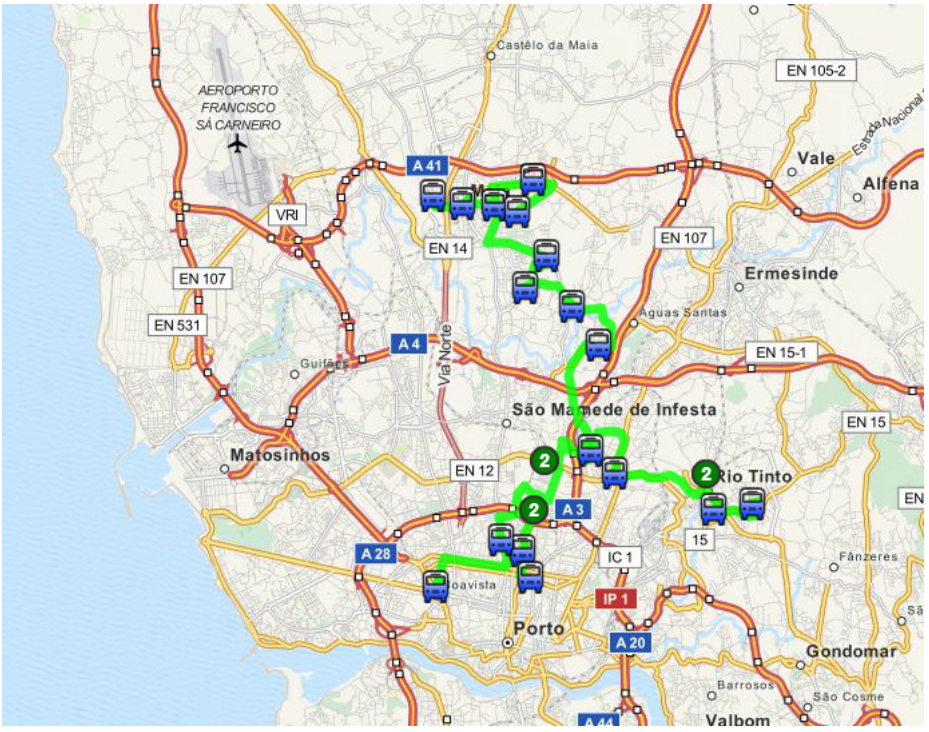

FIGURE C.1: Raw Map 3 - Pólo Universitário

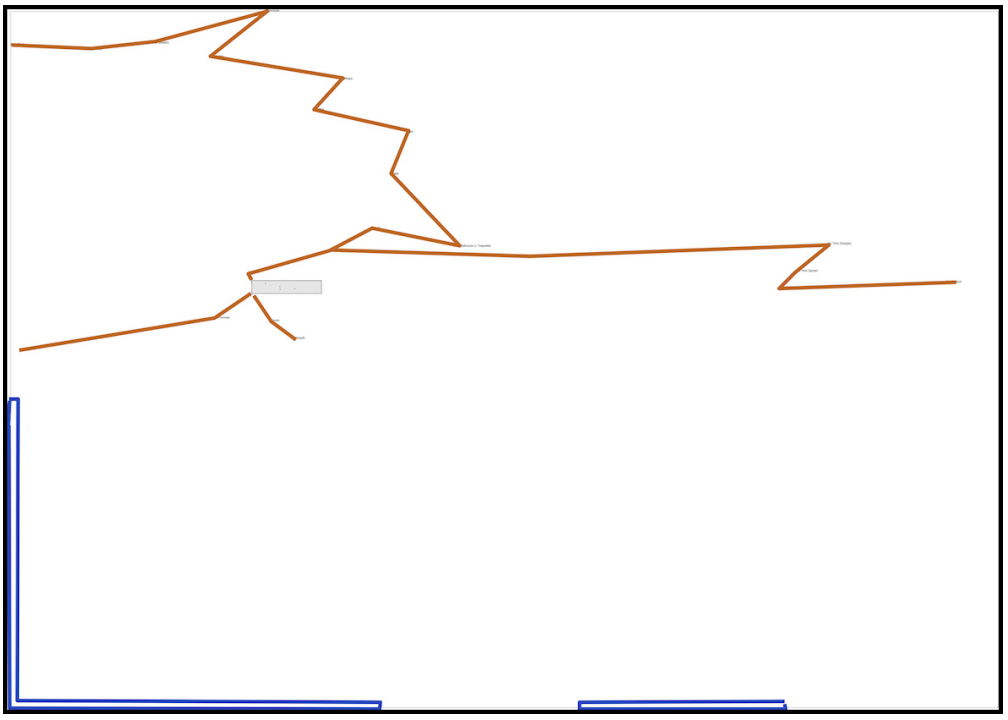

FIGURE C.2: Pre-Processed Map 3A - Pólo Universitário with geographical restrictions

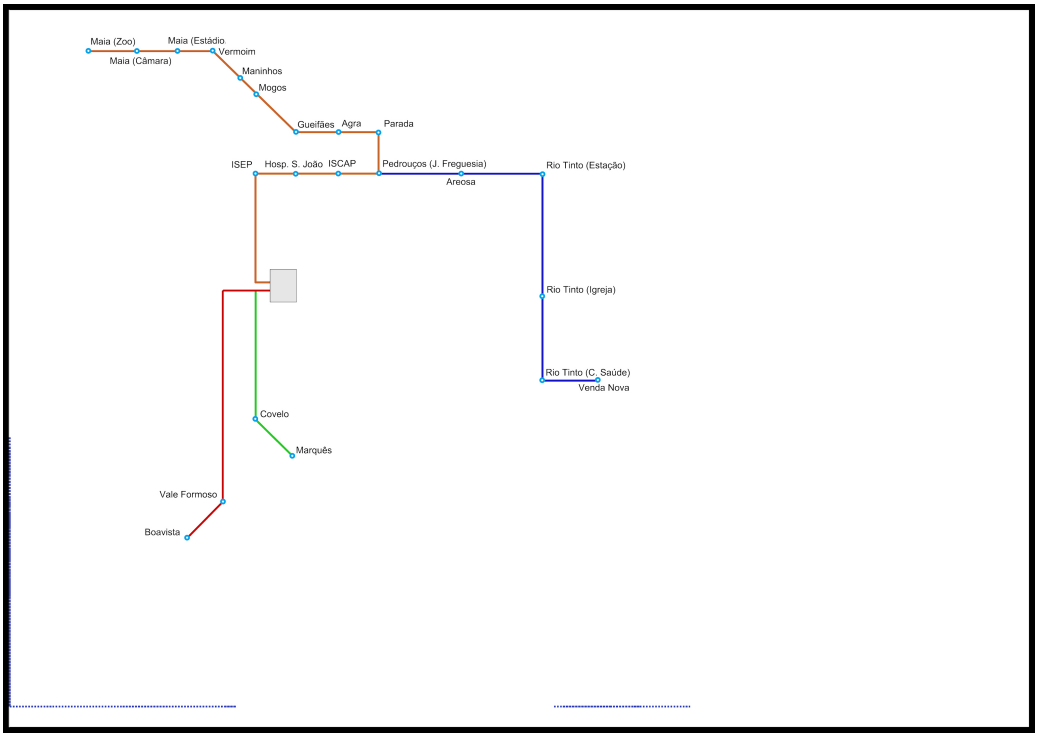

FIGURE C.3: Processed Map 3A - Pólo Universitário with geographical restrictions

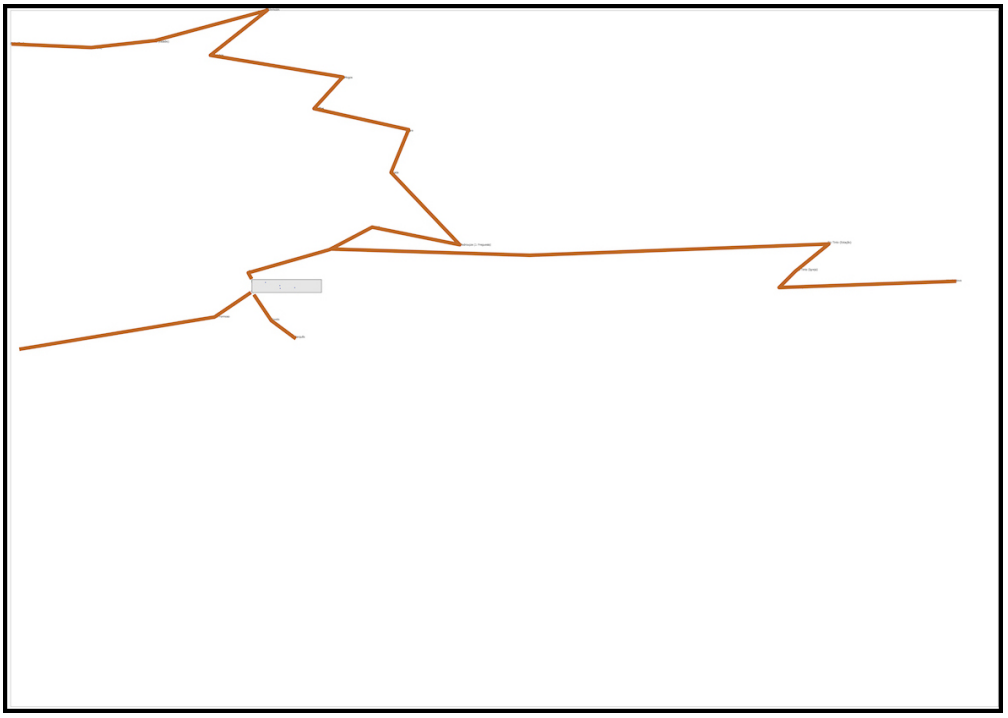

FIGURE C.4: Pre-Processed Map 3B - Pólo Universitário without geographical restrictions

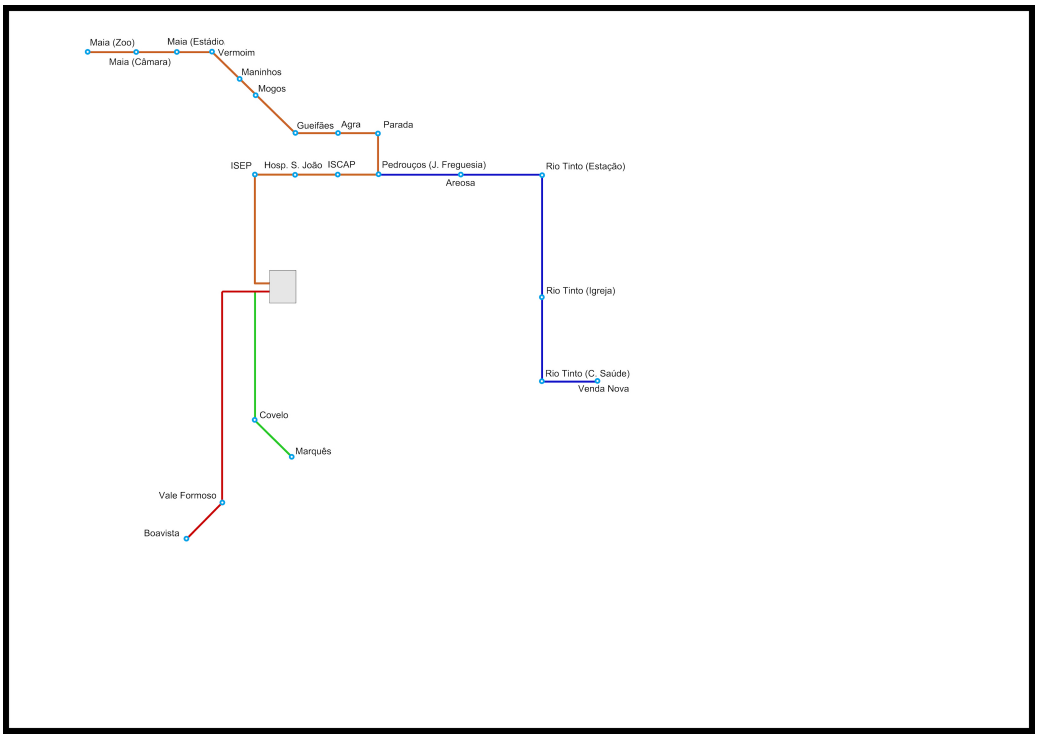

FIGURE C.5: Processed Map 3B - Pólo Universitário without geographical restrictions

Appendix D

Map 4 - Rotunda da Boavista

This appendix illustrates the raw map 4 (figure D.1), pre-processed with (figure D.2) or without (figure D.4) geographical restrictions and processed by our approach with (figure D.3) or without (figure D.5) geographical restrictions.

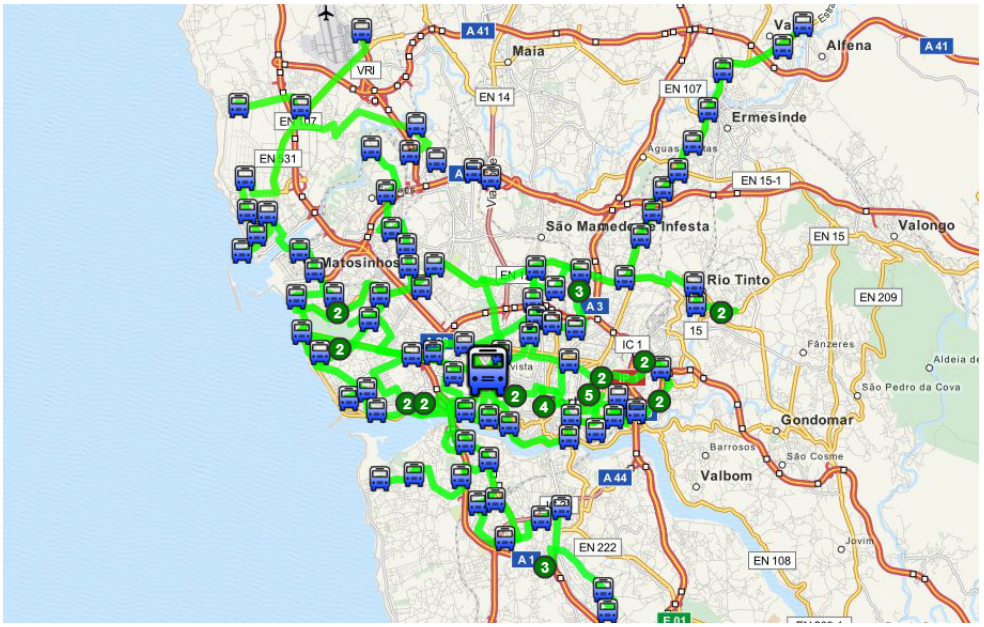

FIGURE D.1: Raw Map 4 - Rotunda da Boavista

FIGURE D.2: Pre-Processed Map 4A - Rotunda da Boavista with geographical restrictions

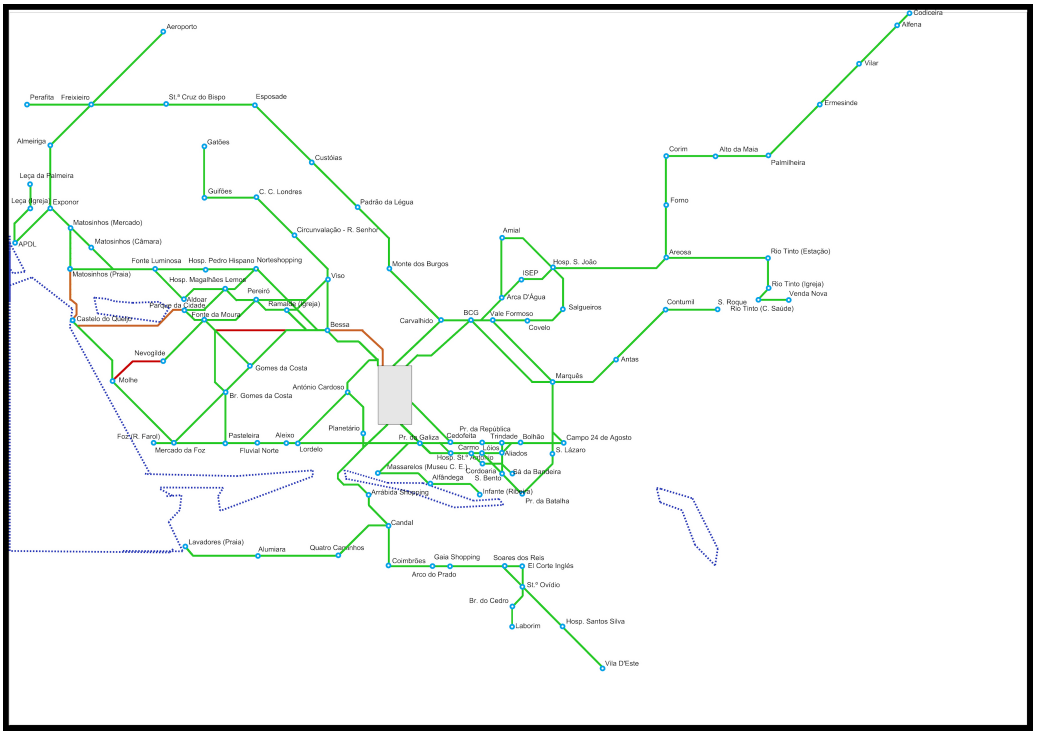

FIGURE D.3: Processed Map 4A - Rotunda da Boavista with geographical restrictions

FIGURE D.4: Pre-Processed Map 4B - Rotunda da Boavista without geographical restrictions

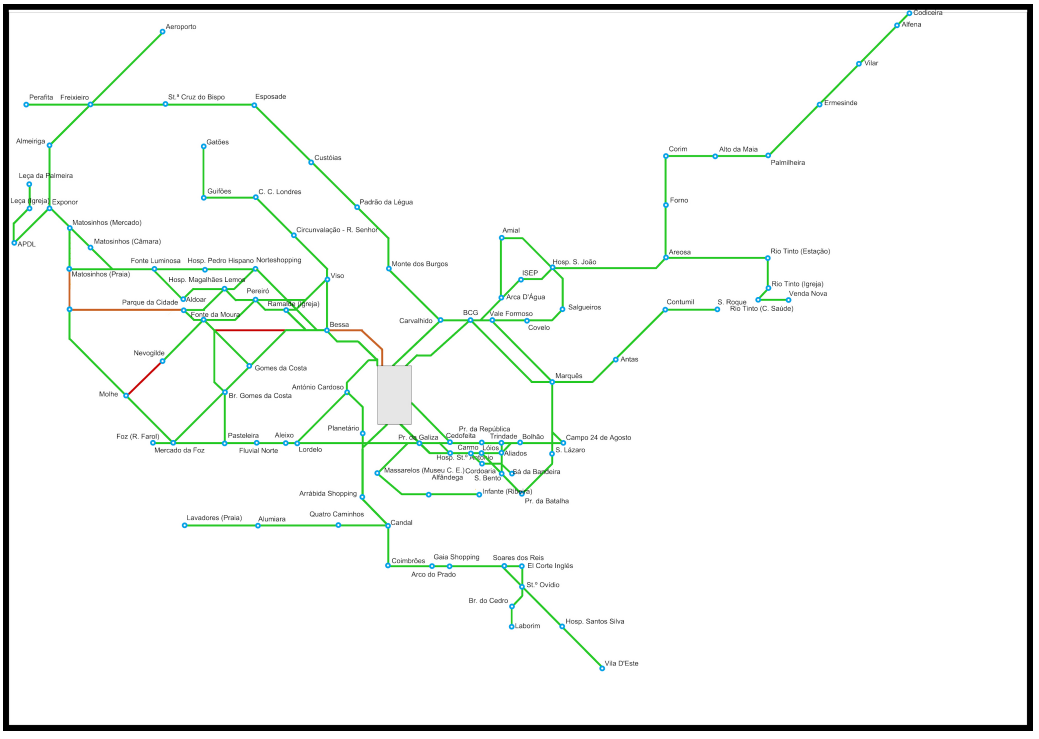

FIGURE D.5: Processed Map 4B - Rotunda da Boavista without geographical restrictions

Appendix E

Map 5 - S. João

This appendix illustrates the raw map 5 (figure E.1), pre-processed with (figure E.2) or without (figure E.4) geographical restrictions and processed by our approach with (figure E.3) or without (figure E.5) geographical restrictions.

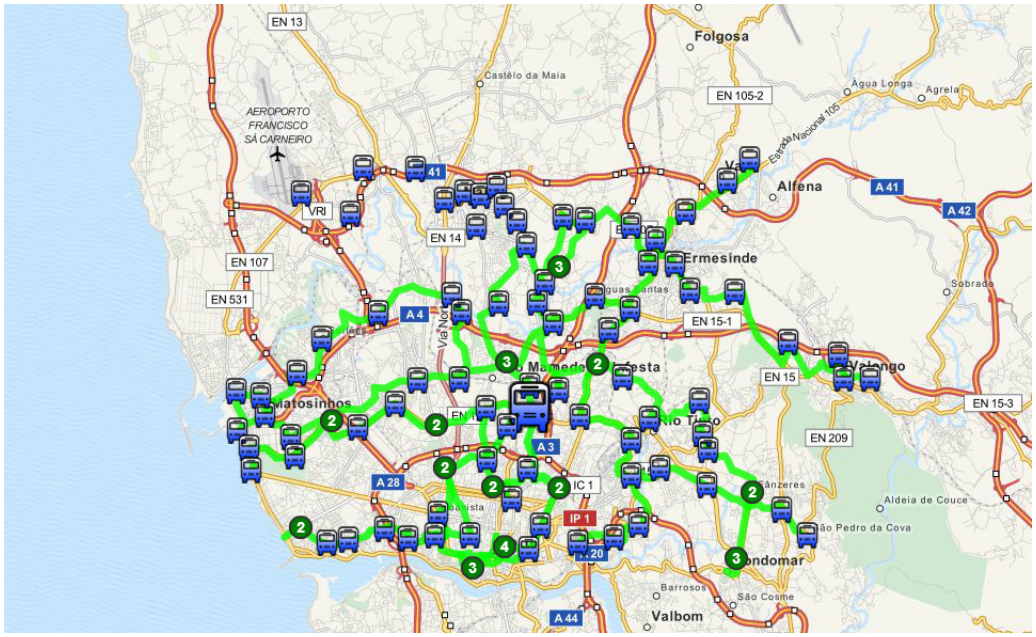

FIGURE E.1: Raw Map 5 - S. João

FIGURE E.2: Pre-Processed Map 5 - S. João with geographical restrictions

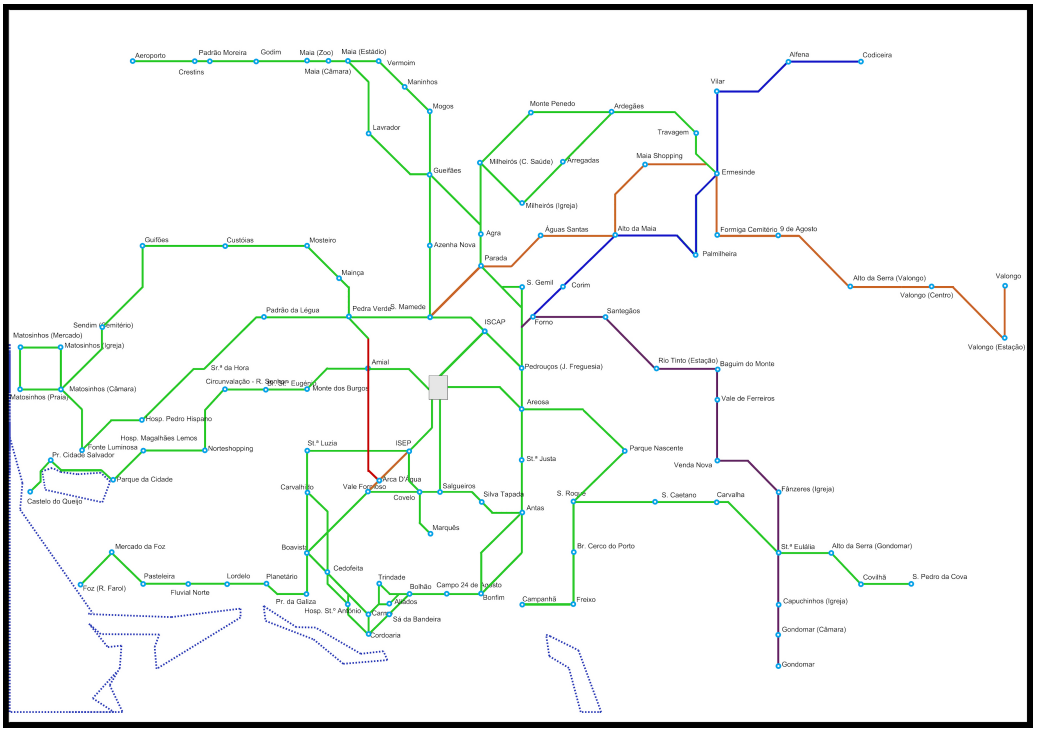

FIGURE E.3: Processed Map 5 - S. João with geographical restrictions

FIGURE E.4: Pre-Processed Map 5 - S. João without geographical restrictions

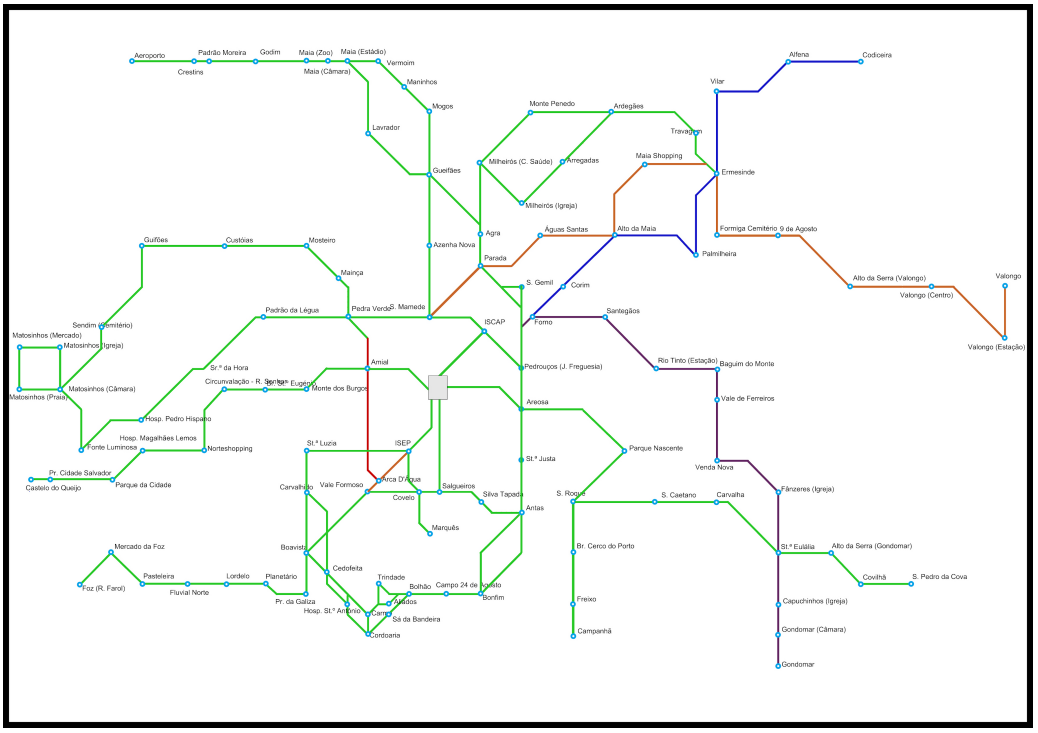

FIGURE E.5: Processed Map 5 - S. João without geographical restrictions

Appendix F

Map 6 - Paranhos

This appendix illustrates the raw map 6 (figure F.1), pre-processed with (figure F.2) or without (figure F.4) geographical restrictions and processed by our approach with (figure F.3) or without (figure F.5) geographical restrictions.

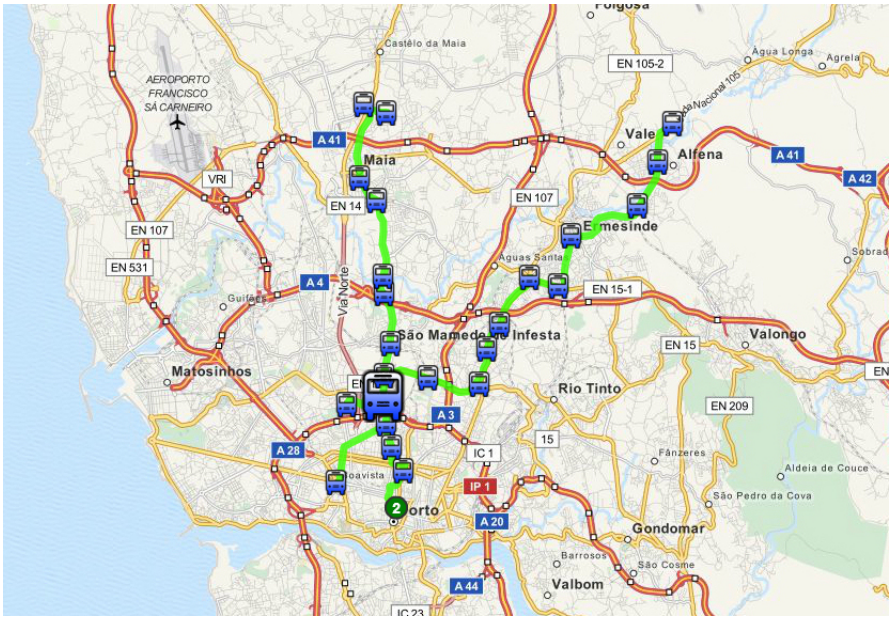

FIGURE F.1: Raw Map 6 - Paranhos

FIGURE F.2: Pre-Processed Map 6A - Paranhos with geographical restrictions

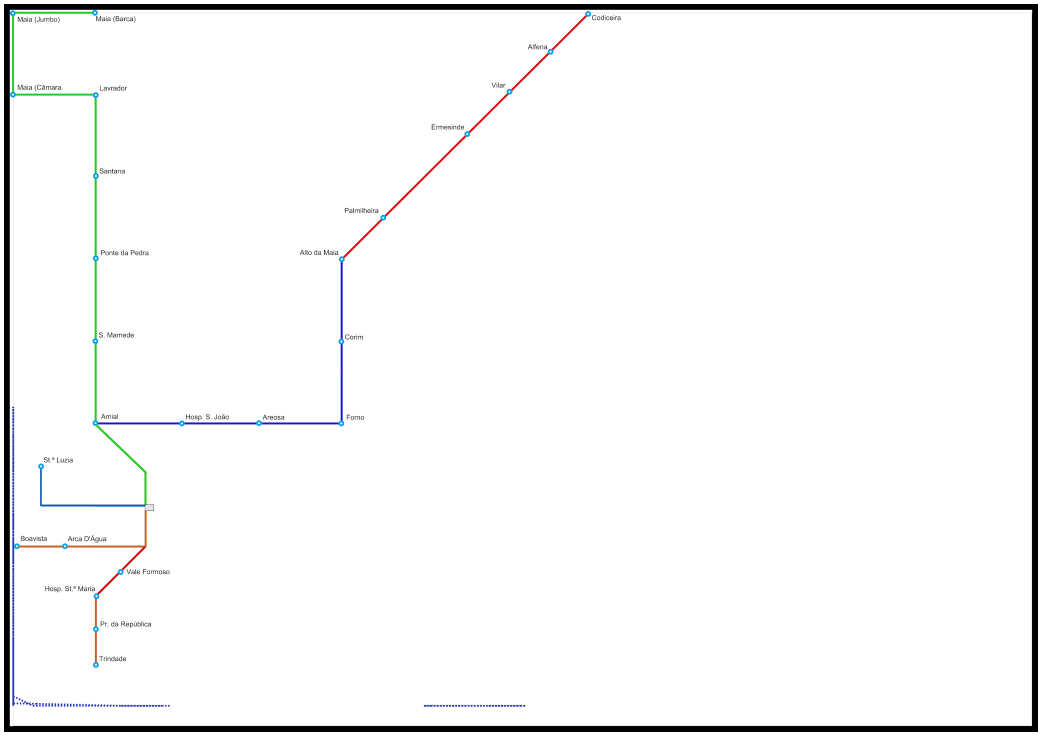

FIGURE F.3: Processed Map 6A - Paranhos with geographical restrictions

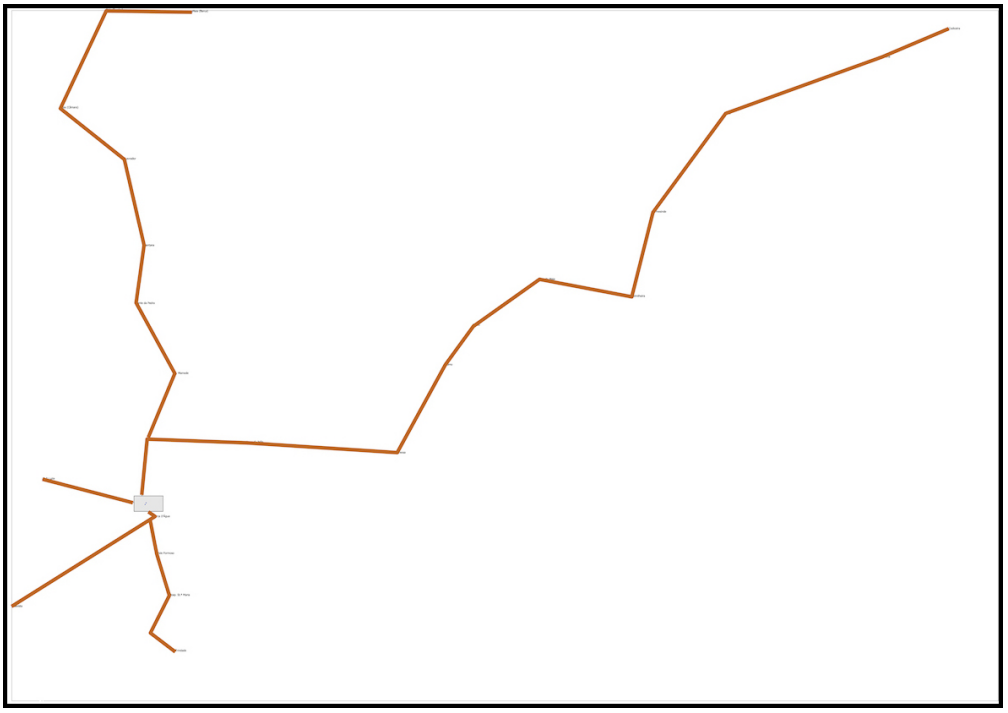

FIGURE F.4: Pre-Processed Map 6B - Paranhos without geographical restrictions

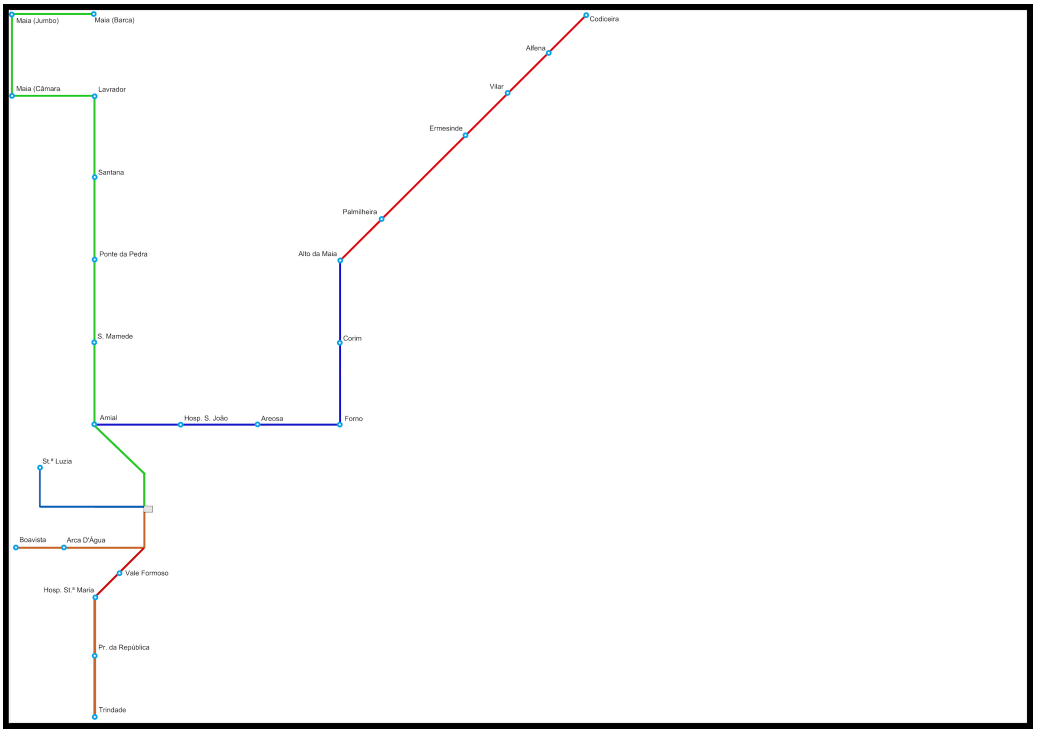

FIGURE F.5: Processed Map 6B - Paranhos without geographical restrictions

Appendix G

Algorithm Execution Graphs for Test 2

The following graphs show the quality evolution throughout the execution of 10000 iterations of the tabu search algorithm. The y-axis measures the solution score (value of the objective function) while the x-axis shows the number of iterations.

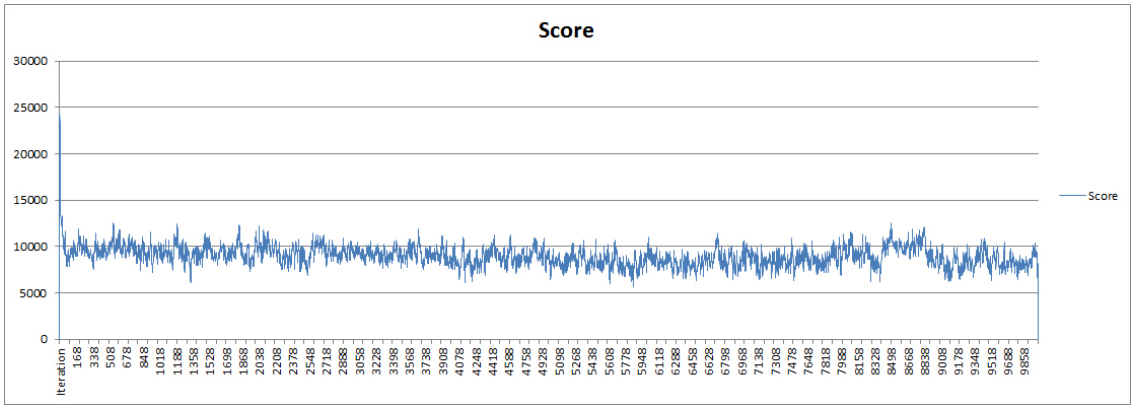

FIGURE G.1: Test 2 result for map 1A

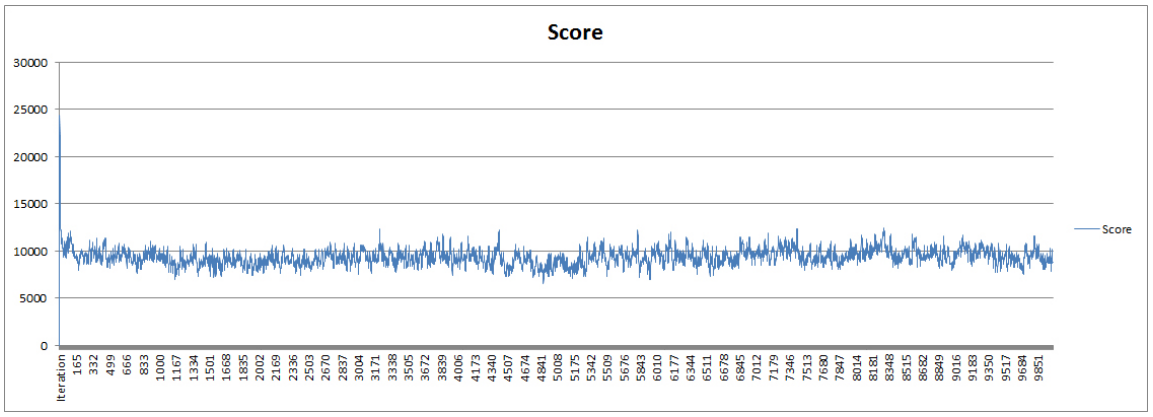

FIGURE G.2: Test 2 result for map 1B

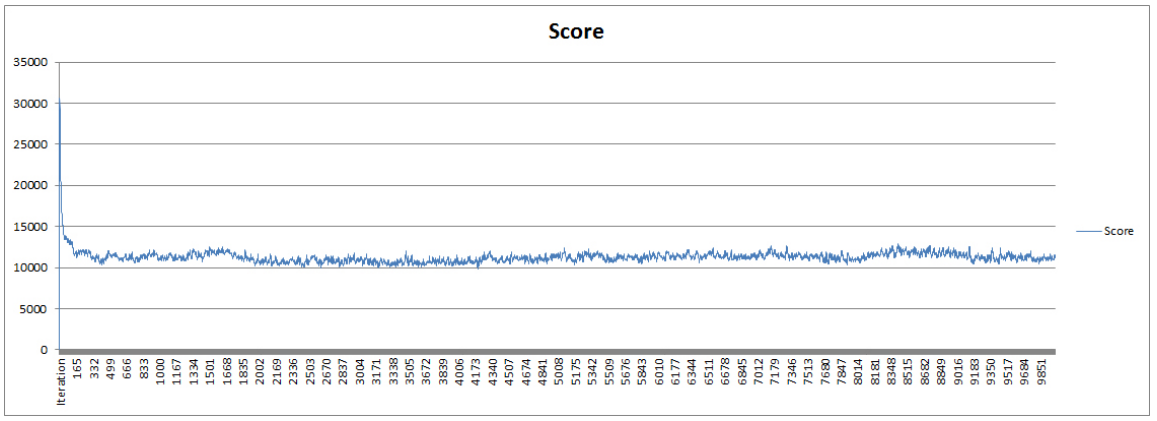

FIGURE G.3: Test 2 result for map 2A

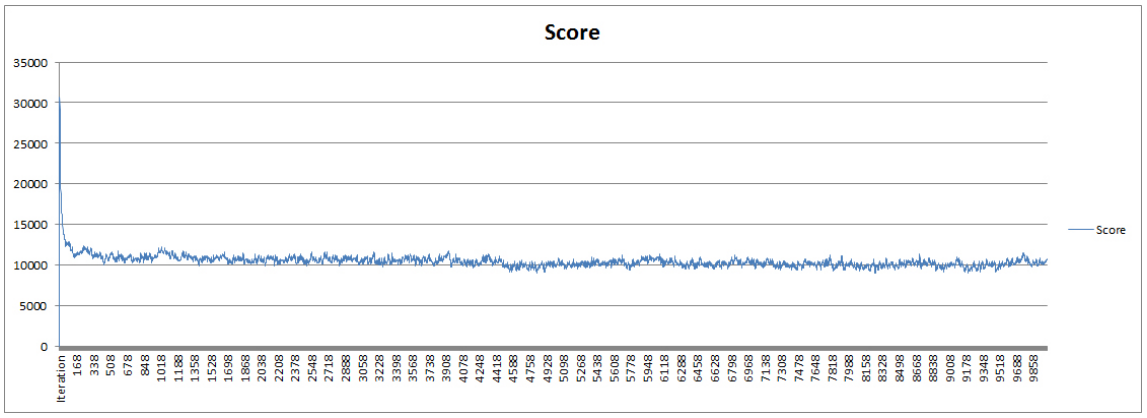

FIGURE G.4: Test 2 result for map 2B

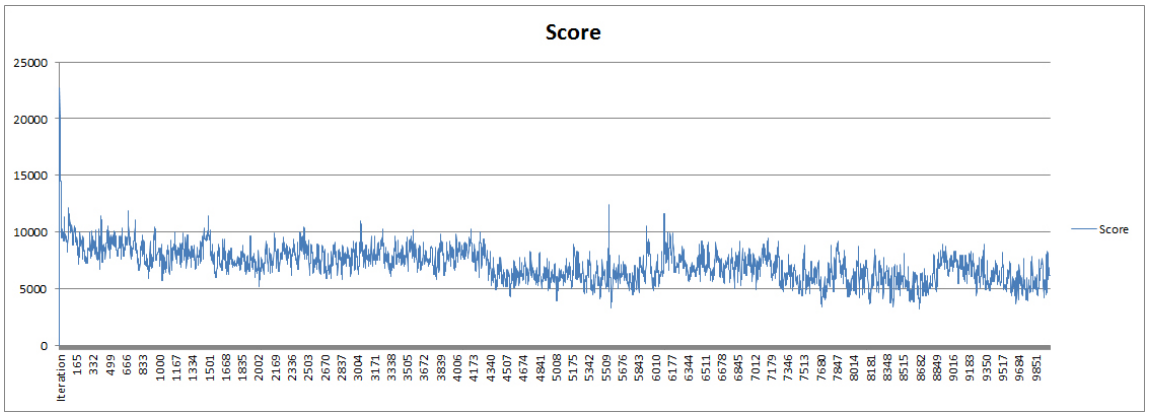

FIGURE G.5: Test 2 result for map 3A

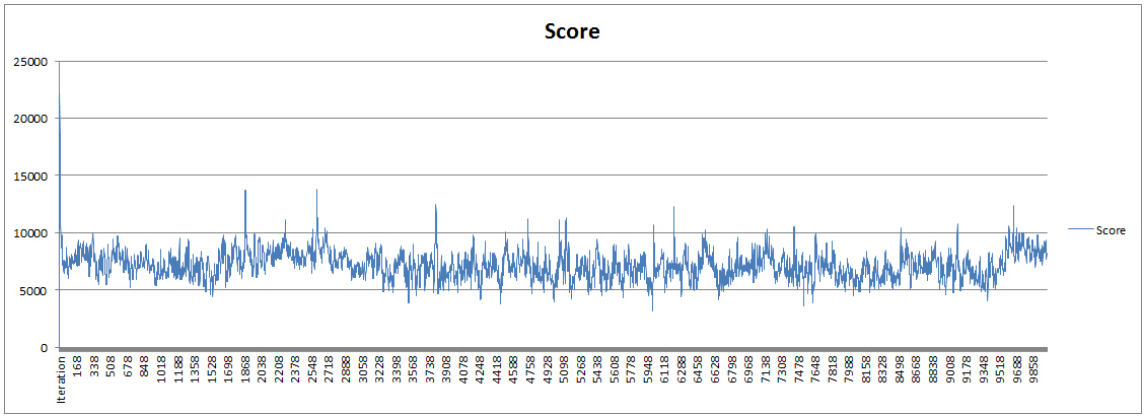

FIGURE G.6: Test 2 result for map 3B

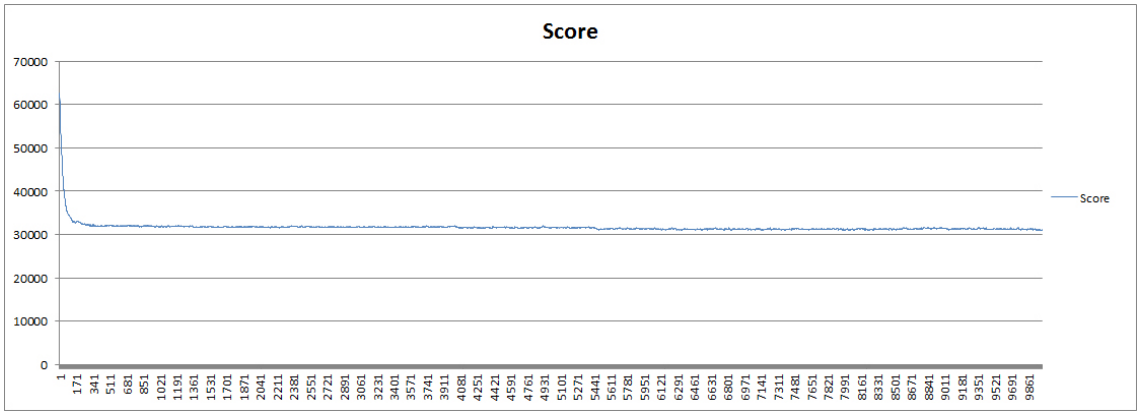

FIGURE G.7: Test 2 result for map 4A

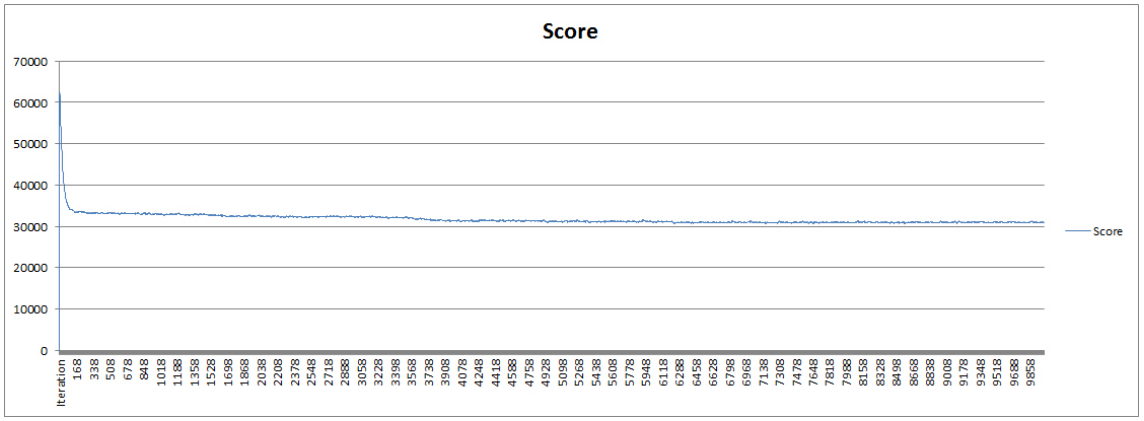

FIGURE G.8: Test 2 result for map 4B

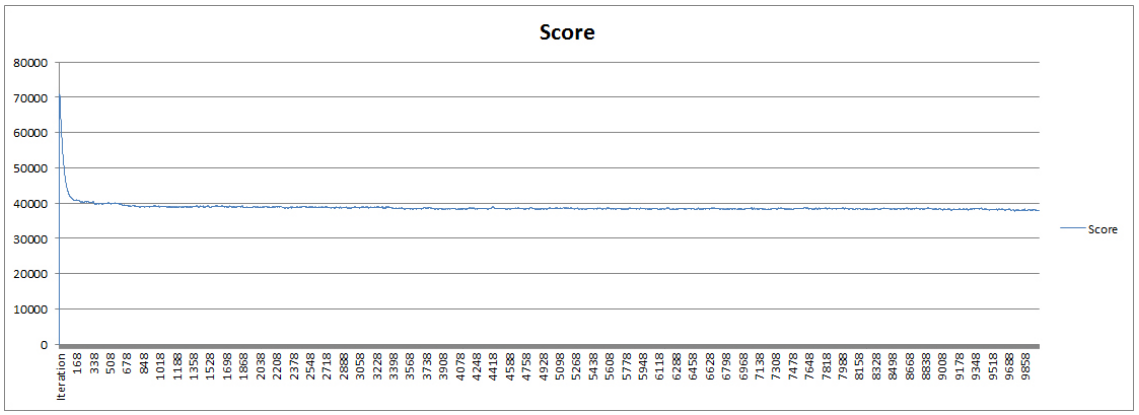

FIGURE G.9: Test 2 result for map 5A

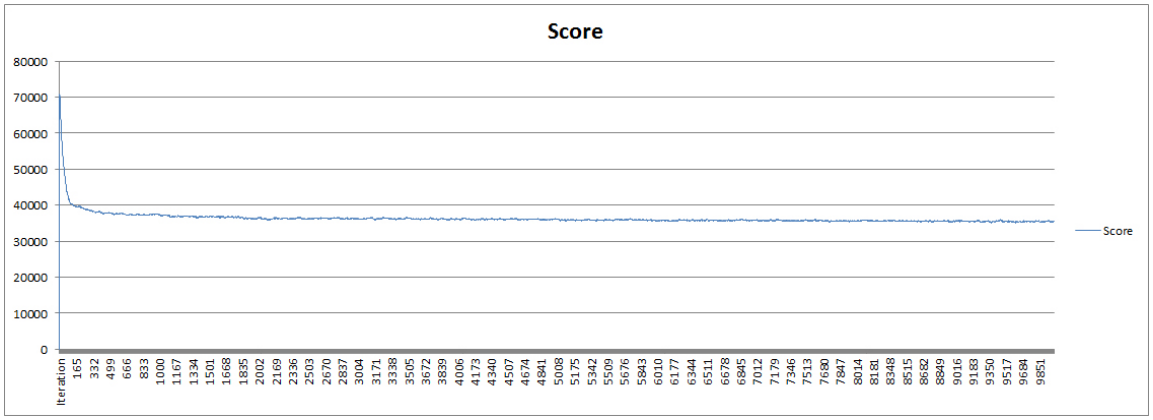

FIGURE G.10: Test 2 result for map 5B

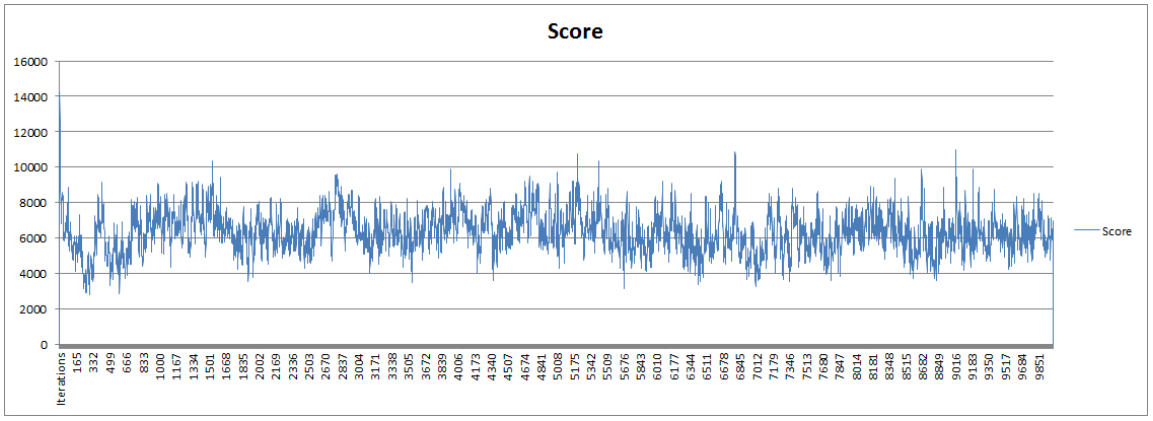

FIGURE G.11: Test 2 result for map 6A

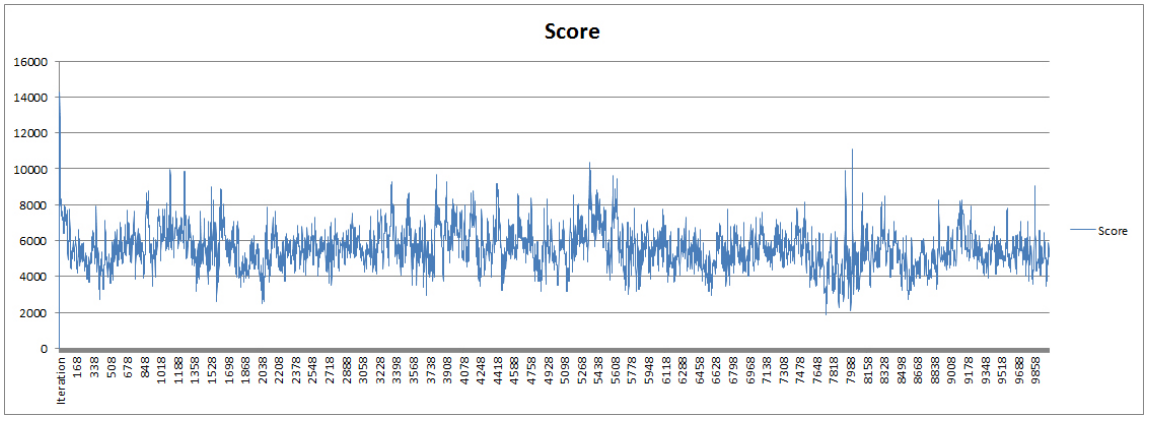

FIGURE G.12: Test 2 result for map 6B

Bibliography

[1] B. W. Boehm. A spiral model of software development and enhancement. *Computer*, 21(5):61–72, May 2010. ISSN 0018-9162. doi: 10.1109/2.59. URL http://dl.acm.org/citation.cfm?id=45797.45801.

[2] Ian Sommerville. *Software Engineering 7*, volume 2003. Cambridge University Press, 2010.

[3] Babylonian Maps. Imago Mundi, 2010. URL http://upload.wikimedia.org/wikipedia/commons/a/a5/Baylonianmaps.JPG.

[4] Hecataeus of Miletus. Hecataeus World Map, 2010. URL http://en.wikipedia.org/wiki/File:Hecataeus_world_map-en.svg.

[5] Charles Knight. The Penny Magazine of the Society for the Diffusion of Useful Knowledge. 2010.

[6] Ptolemy. Ptolemy World Map, 2010. URL http://upload.wikimedia.org/wikipedia/commons/2/23/PtolemyWorldMap.jpg.

[7] Jona Lendering. Peutinger map, 2010. URL http://www.livius.org/pen-pg/peutinger/map.html.

[8] Al-Idrisi. Tabula Rogeriana, 2010. URL http://upload.wikimedia.org/wikipedia/commons/thumb/d/d3/TabulaRogeriana.jpg/1280px-TabulaRogeriana.jpg.

[9] Da Ming Hun Yi Tu. Da Ming Hun Yi Tu World Map, 2010. URL http://en.wikipedia.org/wiki/File:Da-ming-hun-yi-tu.jpg.

[10] De Virga. De Virga World Map, 2010. URL http://upload.wikimedia.org/wikipedia/commons/7/7b/DeVirgaDetail.jpg.

[11] Andrea Bianco. Bianco World Map, 2010. URL http://en.wikipedia.org/wiki/File:Biancomap.jpg.

[12] Unknown. Genoese World Map, 2010. URL http://upload.wikimedia.org/wikipedia/commons/d/d0/Genoese_map.jpg.

[13] Fra Mauro. Fra Mauro World Map, 2010. URL http://upload.wikimedia.org/wikipedia/commons/2/26/FraMauroMap.jpg.

[14] Diogo Ribeiro. Diogo Ribeiro World Map, 2010. URL http://upload.wikimedia.org/wikipedia/commons/f/f9/Worldmap_1529-Ribero.jpeg.

[15] Leventhal Map Center and Leventhal Map Center. Charting an Empire: The Atlantic Neptune. *Exhibitions*, 2010. URL http://www.bpl.org/exhibitions/past-exhibitions/charting-an-empire-the-atlantic-neptune/.

[16] David Hale. Pigot & Co.'s New Map Of England & Wales 1840., 2010. URL http://mapco.net/pigot1840/pigot08.htm.

[17] Society for All British Sabre and Irish Road Enthusiasts. Roader's Digest: The SABRE Wiki, 2010. URL http://www.sabre-roads.org.uk/wiki/index.php?title=Main_Page.

[18] The Leikr Team. Leikr: The new Danish designed GPS sports watch, 2010. URL https://www.kickstarter.com/projects/903141699/leikr-the-new-danish-designed-gps-sports-watch.

[19] Google. Google Maps, 2010. URL https://www.google.pt/maps.

[20] Mapsof.net. India Topographic Map, 2010. URL http://mapsof.net/map/india-topographic-map.

[21] Sean Briggs. Walmarts in the United States, 2009, 2010. URL www.wallmart.com.

[22] Tour de France Organization. Le Parcours 2011, 2010. URL http://2.bp.blogspot.com/_y7XDYcXb1Mw/TL38uMdV_WI/AAAAAAAACEo/GhTo4XRpcs8/s1600/Tour-de-France-2011-route.jpg.

[23] Phillip Muehrcke, Juliana Muehrcke, Aileen Buckley, and Kimerling A. *Map Use - Reading and Analysis*. ESRI Press, Redlands, California, 6th editio edition, 2010.

[24] Doug Cho. Cheam Bus Route Map, 2010. URL http://www.mappery.com/Cheam-Bus-Route-Map.

[25] J Britton. Beck's London Underground Map, 2010. URL http://britton.disted.camosun.bc.ca/beckmap.htm.

[26] J Stott. *Automatic Layout of Metro Maps Using Multicriteria Optimisation*. PhD thesis, University of Kent, 2010.

[27] D. Douglas and T. Peucker. Algorithms for the Reduction of the Number of Points Required to Represent a Digitezed Line or its Caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 10(2): 112–122, October 2010. ISSN 0317-7173. doi: 10.3138/FM57-6770-U75U-7727.

[28] Manojit Sarkar and Marc H. Brown. Graphical fisheye views of graphs. *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '92*, pages 83–91, 2010. doi: 10.1145/142750.142763.

[29] Bernhard Jenny. Geometric distortion of schematic network maps. *Bull. Soc. of Cartographers*, 40:15–18, 2010.

[30] L Latecki and R Lakämper. Polygon evolution by vertex deletion. In Mads Nielsen, Peter Johansen, Ole Fogh Olsern, and Joachim Wickert, editors, *SCALE-SPACE '99 Proceedings of the Second International Conference on Scale-Space Theories in Computer Vision*, pages 398–409, London, 2010. Springer-Verlag.

[31] ITS Training Services. iStudy for Success!, 2010. URL http://tutorials.istudy.psu.edu/.

[32] SDCOE. Spider Map, 2010. URL http://www.sdcoe.net/score/actbank/tspider.htm/.

[33] University York. Fisheye view of central Washington, D.C., 2010. URL http://euclid.psych.yorku.ca/SCS/Gallery/images/fisheye-map.gif.

[34] John Lamping, Ramana Rao, and Peter Pirolli. A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '95*, pages 401–408, 2010. doi: 10.1145/223904.223956.

[35] Yu-Shuen Wang and Ming-Te Chi. Focus+Context Metro Maps, 2010. ISSN 10772626.

[36] Microsoft. Bing Maps, 2006. URL http://www.bing.com/maps/.

[37] Dominik P Käser, Maneesh Agrawala, and Mark Pauly. FingerGlass : Efficient Multiscale Interaction on Multitouch Screens. In *Interfaces*, volume 2011 of *CHI '11*, pages 1601–1610. ACM Press, 2010. ISBN 9781450302289. doi: 10.1145/1978942.1979175.

[38] A.J. Brimicombe. GIS - Where are the frontiers now? In *Proceedings GIS*, pages 33–45, Bahrain, 2010.

[39] Minhee Chae, Jinwoo Kim, Hoyoung Kim, and Hosung Ryu. Information Quality for Mobile Internet Services: A Theoretical Model with Empirical Validation. *Electronic Markets*, 12(1):38–46, 2010. ISSN 10196781. doi: 10.1080/101967802753433254.

[40] OPT. Spider Maps - Mapas Esquemáticos de Zona, 2010. URL http://www.opt.pt/produto.asp?codProduto=8.

[41] Jakob Nielsen. Why You Only Need to Test with 5 Users, 2010. URL http://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/.

[42] Dan Sunday. Geometry Algorithms Home, 2010. URL http://geomalgorithms.com/Pic-sweepline.gif.

[43] Patrick Lester. A* Pathfinding for Beginners, 2010. URL http://www.policyalmanac.org/games/aStarTutorial.htm.

[44] OPT. Spider Maps - Mapas Esquemáticos de Zona. URL http://opt.pt/produto.asp?codProduto=8.

[45] Noah Bierman. MBTA upgrades maps, some 40 years old, 2010. URL http://www.boston.com/news/local/breaking_news/2009/09/mbta_upgrades_m.html.

[46] Ian Sommerville. *Software Engineering*, volume ESTIA lect of *International computer science series*. Addison-Wesley, 2010. ISBN 0321313798. doi: 10.1109/MS.2007.18.

[47] ICA. Menno-Jan Kraak Re-elected Vice-President of ICA, 2010. URL http://www.itc.nl/Pub/News/in2011/July/Menno-Jan-Kraak-Re-elected-Vice-President-of-ICA.html.

[48] Menno-Jan Kraak and Ferjan Ormeling. *Cartography: visualization of geospatial data*, volume 2. Pearson Education, 2010. ISBN 0130888907.

[49] C Board. Report of the working group on cartographic definitions. *Cartographic Journal*, 29:65–69, 2010.

[50] John Campbell. *Map Use & Analysis*. McGraw-Hill Science/Engineering/Math, 2010. ISBN 0073037486. URL http://www.amazon.com/Map-Use-Analysis-John-Campbell/dp/0073037486.

[51] Jorge Luis Borges and Andrew Hurley. *Collected fictions*. 2010. URL http://www.worldcat.org/isbn/0140286802.

[52] J Harley and D Woodward. *The History of Cartography, Volume 1: Cartography in Prehistoric, Ancient, and Medieval Europe and the Mediterranean*, volume Vol. 1. Chicago University Press, 2010. ISBN 0226316335.

[53] Matthew Edney. The History of Cartography Project Web Page, 2010. URL http://www.geography.wisc.edu/histcart/index.html.

[54] Joao Mourinho, Teresa Galvao, and Joao Falcao. Spider Maps for Location-Based Services Improvement. In Jean-Henry Snene, Mehdi and Ralyté, Jolita and Morin, editor, *Second International Conference on Exploring Services Sciences*, pages 16–29. Springer Berlin Heidelberg, 2010. doi: 10.1007/978-3-642-21547-6_2.

[55] R. F. Tomlinson. A Geographic Information System for Regional Planning. *Journal of Geography (Chigaku Zasshi)*, 78(1):45–48, 2010. ISSN 0022-135X. URL http://joi.jlc.jst.go.jp/JST.Journalarchive/jgeography1889/78.45?from=CrossRef.

[56] Robert Laurini, Sylvie Servigne, and Guillaume Noel. Soft real-time GIS for disaster monitoring. pages 465–480. Springer-Verlag, 2010. ISBN 978354024988.

[57] Thomas Porathe. User-centered map design. In *Usability Professionals' Association Conference*, Austin, Texas, USA, 2010.

[58] Daniel Delling. Engineering and Augmenting Route Planning Algorithms. *PhD Thesis*, page 161, 2010. URL papers2://publication/uuid/7F722816-1187-4A7A-B2EF-249C3DDFCE9D.

[59] Timo Pietilä. *Alternatives to map-based pedestrian navigation with location-aware mobile devices.* Media programme thesis, TAMK University of Applied Sciences, 2010.

[60] Silvania Avelar. *Schematic Maps on Demand: Design, Modeling and Visualization.* PhD thesis, Swiss Federal Institute of Technology Zurich, 2010.

[61] M Agrawala and C Stolte. Rendering effective route maps: improving usability through generalization. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 1:241–249, 2010. ISSN 00978930. doi: 10.1145/383259.383286.

[62] A M MacEachren. *How Maps Work: Representation, Visualization and Design*, volume 81. 2010. ISBN 157230040X. URL http://books.google.com/books?hl=en&lr=&id=xhAvN3BOCkUC&oi=fnd&pg=PA1&dq=How+maps+work:+representation,+visualization,+and+design.&ots=Or_UpI0ZTY&sig=v1gCTIpoamwjwylUxSbDOqWq7g4.

[63] Thomas Barkowsky, L Latecki, and Kai-florian Richter. *Schematizing maps: Simplification of geographic shape by discrete curve evolution*, volume 8. Springer, Berlin, 2010.

[64] Michael Denis. The description of routes : A cognitive approach to the production of spatial discourse. *Cahiers de psychologie cognitive*, 16(4):409–458, 2010. ISSN 0249-9185. URL http://cat.inist.fr/?aModele=afficheN&cpsidt=2773586.

[65] Barbara Tversky and PU Lee. Pictorial and verbal tools for conveying routes. *... theory. Cognitive and computational foundations of ...*, 1661:51–64, 2010. doi: 10.1007/3-540-48384-5_4. URL http://www.springerlink.com/index/n7hpph8428d4y2jn.pdf$\delimiter"026E30F$nhttp://link.springer.com/chapter/10.1007/3-540-48384-5_4.

[66] Katharine S. Willis, Christoph Hölscher, Gregor Wilbertz, and Chao Li. A comparison of spatial knowledge acquisition with maps and mobile maps. *Computers, Environment and Urban Systems*, 33(2):100–110, March 2010. ISSN 01989715. doi: 10.1016/j.compenvurbsys.2009.01.004.

[67] Bia Kim, S Lee, and Jaesik Lee. Gender differences in spatial navigation. *World Academy of Science, Engineering and ...*, pages 297–300, 2010.

[68] Silvania Avelar and Lorenz Hurni. On the Design of Schematic Transport Maps. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 41(3):217–228, September 2010. ISSN 0317-7173. doi: 10.3138/A477-3202-7876-N514.

[69] R Latto. Do we like what we see? In Grant Malcom, editor, *Multidisciplinary Approaches to Visual Representations and Interpretations, Vol 2*, volume 1, pages 343–356. Elsevier B.V., Amsterdam, March 2010. ISBN 978-0-444-51463-9.

[70] AM MacEachren and GB Johnson. The evolution, application and implications of strip format travel maps. *The Cartographic Journal*, 2010. URL http://www.maneyonline.com/doi/abs/10.1179/caj.1987.24.2.147.

[71] H. Casakin, T. Barkowsky, A. Klippel, and Christian Freksa. Schematic maps as wayfinding aids. *Spatial Cognition II*, pages 54–71, 2010.

[72] Christian Freksa. Spatial aspects of task-specific wayfinding maps. *Visual and spatial reasoning in design*, 2010. URL http://www.informatik.uni-hamburg.de/WSV/SPP_onlines/ProjektQ/Freksa1999.pdf.

[73] Bettina Berendt, Thomas Barkowsky, Christian Freksa, and Stephanie Kelter. Spatial representation with aspect maps. *Spatial Cognition*, (1998), 2010. URL http://link.springer.com/chapter/10.1007/3-540-69342-4_15.

[74] Alastair Morrison. Public Transport Maps in Western European Cities. *Cartographic Journal, The*, 33(2):93–110, December 2010. ISSN 00087041. doi: 10.1179/000870496787757230. URL http://openurl.ingenta.com/content/xref?genre=article&issn=0008-7041&volume=33&issue=2&spage=93.

[75] William Wyckoff. How to lie with maps, 2010. ISSN 02779390.

[76] Robert Weibel. Map Generalization in the Context of Digital Systems. *Cartography and Geographic Information Science*, 22(4):259–263, October 2010. ISSN 15230406. doi: 10.1559/152304095782540285. URL http://www.tandfonline.com/doi/abs/10.1559/152304095782540285http://openurl.ingenta.com/content/xref?genre=article&issn=1523-0406&volume=22&issue=4&spage=259.

[77] Robert Weibel and Christopher B. Jones. Computational Perspectives on Map Generalization. *GeoInformatica*, 2(4):307–314, December 2010. ISSN 1573-7624. doi: 10.1023/A:1009748903798. URL http://link.springer.com/article/10.1023/A:1009748903798.

[78] J Mark Ware, George E Taylor, and Nathan Thomas. Automated Production of Schematic Maps for Mobile Applications. *Transactions in GIS*, 10(1):25– 42, 2010.

[79] Joao Mourinho, Teresa Galvao, João e Cunha, Fernando Vieira, and Jose Pacheco. A Software Framework for the Automated Production of Schematic Maps. In Selmin Nurcan, editor, *IS Olympics: Information Systems in a Diverse World*, volume 107 of *Lecture Notes in Business Information Processing*, pages 64–78. Springer Berlin Heidelberg, 2010. ISBN 978-3-642-29748-9. doi: 10.1007/978-3-642-29749-6_5.

[80] Susan P. Waldorf. *Schematic Navigational Map Design*. 2010. URL http://books.google.pt/books/about/Schematic_Navigational_Map_Design.html?id=jVx-NwAACAAJ&pgis=1.

[81] D Elroi. Designing a Network Linemap Schematization Software Enhancement Package. In *Proceedings of the Eighth Annual ESRI User Conference*, Redlands, California, 2010.

[82] D Elroi. GIS and schematic maps: A new symbiotic relationship. http://www.elroi.com/papers%20GIS%20LIS%2088/GISLIS88.html, 2010.

[83] KURT E. BRASSEL and ROBERT WEIBEL. A review and conceptual framework of automated map generalization. *International journal of geographical information systems*, 2(3):229–244, January 2010. ISSN 0269-3798. doi: 10.1080/02693798808927898. URL http://dx.doi.org/10.1080/02693798808927898.

[84] Longin Jan Latecki and Rolf Lakämper. Convexity Rule for Shape Decomposition Based on Discrete Contour Evolution. *Computer Vision and Image Understanding*, 73(3):441–454, March 2010. ISSN 10773142. doi: 10.1006/cviu.1998.0738.

[85] Gabriele Neyer. Line simplification with restricted orientations. *Algorithms and Data Structures*, (21957):13–24, 2010.

[86] S Har-Peled. New similarity measures between polylines with applications to morphing and polygon sweeping. *Discrete & Computational Geometry*, 2010. URL http://link.springer.com/article/10.1007/s00454-002-2886-1.

[87] M. Nöllenburg. *Automated drawing of metro maps*. PhD thesis, Universitat Karlsruhe, 2010.

[88] Silvania Avelar and SAM Müller. Generating topologically correct schematic maps. *In Proc. 9th Int. Symp. on Spatial Data Handling*, 2010. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.17.2741.

[89] Sergio Cabello and M De Berg. Schematization of road networks. *Proceedings of the ...*, 2010. URL http://dl.acm.org/citation.cfm?id=378609.

[90] Sergio Cabello and Marc Kreveld. Approximation Algorithms for Aligning Points. *Algorithmica*, 37(3):211–232, August 2010. ISSN 0178-4617. doi: 10.1007/s00453-003-1033-6. URL http://link.springer.com/10.1007/s00453-003-1033-6.

[91] Martin Nöllenburg and Alexander Wolff. A Mixed-Integer Program for Drawing High-Quality Metro Maps. In Patrickj Healy and Nikola Nikolov, editors, *GD'05 Proceedings of the 13th international conference on Graph Drawing*, volume 3843, pages 321–333. Springer-Verlag Berlin, 2010. doi: 10.1007/11618058_29.

[92] Martin Nöllenburg and Alexander Wolff. Drawing and Labeling High-Quality Metro Maps by Mixed-Integer Programming. *IEEE transactions on visualization and computer graphics*, 17(15):626–641, May 2010. ISSN 1941-0506. doi: 10.1109/TVCG.2010.81.

[93] J M Stott and Peter Rodgers. Automatic Metro Map Design Techniques. *Metro*, 16(July):11–16, 2010.

[94] Jonathan Stott, Peter Rodgers, Juan Carlos Martínez-Ovando, and Stephen G Walker. Automatic metro map layout using multicriteria optimization. *IEEE Transactions on Visualization and Computer Graphics*, 17(1):101–114, 2010.

[95] J. Mark Ware, Christopher B. Jones, and Nathan Thomas. Automated map generalization with multiple operators: a simulated annealing approach. *International Journal of Geographical Information Science*, 17(8):743–769, December 2010. ISSN 1365-8816. doi: 10.1080/13658810310001596085. URL http://www.tandfonline.com/doi/abs/10.1080/13658810310001596085.

[96] J.M. Ware, I.D. Wilson, and J.A. Ware. A knowledge based genetic algorithm approach to automating cartographic generalisation. *Knowledge-Based Systems*, 16(5-6):295–303, July 2010. ISSN 09507051. doi: 10.1016/S0950-7051(03)00031-5. URL http://www.sciencedirect.com/science/article/pii/S0950705103000315.

[97] J. Mark Ware, Ian D. Wilson, J. Andrew Ware, and Christopher B. Jones. A tabu search approach to automated map generalisation. In *Proceedings of the tenth ACM international symposium on Advances in geographic information systems - GIS '02*, page 101, New York, New York, USA, November 2010. ACM Press. ISBN 1581135912. doi: 10.1145/585147.585169. URL http://dl.acm.org/citation.cfm?id=585147.585169.

[98] Jerry Swan, Suchith Anand, Mark Ware, and Mike Jackson. Automated schematization using memetic algorithms. *Director*, 44(0):77–82, 2010.

[99] Weihua Dong, Qingsheng Guo, and Jiping Liu. Schematic road network map progressive generalization based on multiple constraints. *Geo-spatial Information Science*, 11(3):215–220, January 2010. ISSN 1009-5020. doi: 10.1007/s11806-008-0090-z.

[100] Johannes Kopf, M Agrawala, D Bargeron, David Salesin, and Michael Cohen. Automatic generation of destination maps. *ACM Transactions on Graphics TOG*, 29(6):158, 2010. ISSN 07300301. doi: 10.1145/1866158.1866184.

[101] Bernard Comrie. Subject and Object Control: Syntax, Semantics, Pragmatics, September 2010. URL http://elanguage.net/journals/bls/article/view/2339.

[102] A P Martinich. *The Philosophy of Language*. Oxford University Press, 2010.

[103] Eckard Rolf, editor. *Pragmatik*. VS Verlag für Sozialwissenschaften, Wiesbaden, 2010. ISBN 978-3-531-13105-4. doi: 10.1007/978-3-663-11116-0. URL http://link.springer.com/10.1007/978-3-663-11116-0.

[104] Anind K Dey. Understanding and Using Context. *Personal and Ubiquitous Computing*, 5(1):4–7, February 2010. ISSN 1617-4909. doi: 10.1007/s007790170019.

[105] AM Nivala and LT Sarjakoski. An approach to intelligent maps: context awareness. In Keith Cheverst and Barbara Schmidt-Belz, editors, *The 2nd Workshop on'HCI in Mobile Guides*, number September, Udine, 2010.

[106] Stefan Steiniger, Moritz Neun, and Alistair Edwardes. Foundations of location based services. *Lecture Notes on LBS*, pages 1–28, 2010.

[107] M. Schouwstra. Mental representation, communication and the transition from animal to human. January 2010. URL http://www.researchgate.net/publication/46710593_Mental_representation_communication_and_the_transition_from_animal_to_human.

[108] MA Ruiz-Primo and RJ Shavelson. Problems and issues in the use of concept maps in science assessment. *Journal of research in science ...*, 33(6):569–600, 2010. URL http://www.stanford.edu/dept/SUSE/SEAL/Reports_Papers/ProblemsIssuesCMScience.pdf.

[109] FilipJ.R.C. Dochy. Assessment of Domain-Specific and Domain-Transcending Prior Knowledge: Entry Assessment and the Use of Profile Analysis. In Menucha Birenbaum and FilipJ.R.C. Dochy, editors, *Alternatives in Assessment of Achievements, Learning Processes and Prior Knowledge SE - 9*, volume 42 of *Evaluation in Education and Human Services*, pages 227–264. Springer Netherlands, 2010. ISBN 978-94-010-4287-1. doi: 10.1007/978-94-011-0657-3_9. URL http://dx.doi.org/10.1007/978-94-011-0657-3_9.

[110] John R McClure, Brian Sonak, and Hoi K Suen. Concept map assessment of classroom learning: Reliability, validity, and logistical practicality. *Journal of Research in Science Teaching*, 36(4):475–492, 2010. ISSN 00224308. doi: 10.1002/(SICI)1098-2736(199904)36:4⟨475::AID-TEA5⟩3.0.CO;2-O.

[111] L. Lagerwerf, L. Cornelis, J. de Geus, and P. Jansen. Advance Organizers in Advisory Reports: Selective Reading, Recall, and Perception. *Written Communication*, 25(1):53–75, January 2010. ISSN 0741-0883. doi: 10.1177/0741088307309043.

[112] Christo Dichev, D Dichva, and Lora Aroyo. Using topic maps for e-learning. In *IASTED International Conference ...*, 2010.

[113] Matthias Baldauf, S Dustdar, and F Rosenberg. A survey on context-aware systems. *International Journal of Ad Hoc and Ubiquitous Computing*, 2(4): 263–277, 2010. ISSN 17438225. doi: 10.1504/IJAHUC.2007.014070. URL http://www.inderscience.com/link.php?id=14070.

[114] Alfred Kobsa. Generic user modeling systems. *User modeling and user-adapted interaction*, pages 49–63, 2010.

[115] J.A. Bangham, S.E. Gibson, and Richard Harvey. The art of scale-space. *Proc. British Machine Vision ...*, pages 569–578, 2010.

[116] GW Furnas. The FISHEYE view: A new look at structured files. *Readings in information visualization: using vision to ...*, 2010. URL http://herakles.zcu.cz/seminars/docs/infovis/papers/Furnas_fisheye_86.pdf.

[117] Stéphane Huot and Eric Lecolinet. Focus+Context Visualization Techniques for Displaying Large Lists with Multiple Points of Interest on Small Tactile Screens. *Context*, pages 219–233, 2010. doi: 10.1007/978-3-540-74800-7_18.

[118] Robert Kosara, Silvia Miksch, and Helwig Hauser. Focus+Context Taken Literally. *IEEE Computer Graphics and Applications*, 22(1):22–29, 2010. ISSN 02721716. doi: 10.1109/38.974515.

[119] K Virrantaus, J Markkula, A Garmash, YV Terziyan, J Veijalainen, A Katanosov, and H Tirri. Developing GIS-supported location-based services. In *Proc of WGIS*, pages 423–432, 2010.

[120] Open Geospatial Consortium (OGC). Open Location Services. Technical report, 2010.

[121] Narushige Shiode, Chao Li, Michael Batty, Paul Longley, and David Maguire. The impact and penetration of location-based services. *Telegeoinformatics locationbased computing and services*, 44(0):349–366, 2010.

[122] Tumasch Reichenbacher. *Mobile Cartography – Adaptive Visualisation of Geographic Information on Mobile Devices.* PhD thesis, Technischen Universitat Munchen, 2010.

[123] Petra Hocova and Joao Cunha. A Service Science and Engineering Approach to Public Information Services in Exceptional Situations - Examples from Transport. In *Lecture Notes in Business Information Processing*, pages 65–81, 2010.

[124] T Koivumäki, A Ristola, and M Kesti. The effects of information quality of mobile information services on user satisfaction and service acceptance–empirical evidence from Finland. *Behaviour & Information ...*, 27(5):375–385, 2010. ISSN 0144929X. doi: 10.1080/01449290601177003.

[125] E K R E Huizingh. The content and design of Web sites: an empirical study. *INFORMATION MANAGEMENT*, 37(3):123–134, 2010. ISSN 03787206.

[126] B Schilit, N Adams, and R Want. Context-aware computing applications. In Luis-Felipe Cabrera and Mahadev Satyanarayanan, editors, *Workshop on Mobile Computing Systems and Applications*, volume Santa Cruz of *Proceedings. Workshop on*

Mobile Computing Systems and Applications (Cat. No.94TH06734), pages 85–90. Columbia Univ, New York, NY, USA, IEEE Comput. Soc. Press, 2010. ISBN 0818663456. doi: 10.1109/MCSA.1994.512740.

[127] T P Novak, D L Hoffman, and Yiu-Fai Yung. Measuring the Customer Experience in Online Environments: A Structural Modeling Approach. *Marketing Science*, 19 (1):22–42, 2010. ISSN 07322399. doi: 10.1287/mksc.19.1.22.15184.

[128] Young Eun Lee and Izak Benbasat. A Framework for the Study of Customer Interface Design for Mobile Commerce. *International Journal of Electronic Commerce*, 8(3):79–102, 2010. ISSN 10864415.

[129] D Gehrke and E Turban. Determinants of successful Website design: relative importance and recommendations for effectiveness. *Proceedings of the 32nd Annual Hawaii International Conference on Systems Sciences 1999 HICSS32 Abstracts and CDROM of Full Papers*, 00(c):8, 2010. doi: 10.1109/HICSS.1999.772943.

[130] Gerald Haubl and Valerie Trifts. Consumer Decision Making in Online Shopping Environments: The Effects of Interactive Decision Aids. *Marketing Science*, 19 (1):4–21, 2010. ISSN 07322399. doi: 10.1287/mksc.19.1.4.15178. URL http://mktsci.journal.informs.org/cgi/doi/10.1287/mksc.19.1.4.15178.

[131] Sharon Oviatt. Human-centered design meets cognitive load theory. In *Proceedings of the 14th annual ACM international conference on Multimedia - MULTIMEDIA '06*, page 871, New York, New York, USA, 2010. ACM Press. ISBN 1595934472. doi: 10.1145/1180639.1180831.

[132] Joachim Böttger, Ulrik Brandes, Oliver Deussen, and Hendrik Ziezold. Map warping for the annotation of metro maps. *IEEE computer graphics and applications*, 28(5):56–65, 2010. ISSN 0272-1716. doi: 10.1109/MCG.2008.99.

[133] Steve Krug. *Don't Make Me Think: A Common Sense Approach to the Web (2nd Edition)*, volume 24 of *Circle.com library*. New Riders Publishing, Thousand Oaks, 2 edition, June 2010. ISBN 0321344758.

[134] Jakob Nielsen and Thomas K. Landauer. A mathematical model of the finding of usability problems. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '93*, pages 206–213, New York, New York, USA, May 2010. ACM Press. ISBN 0897915755. doi: 10.1145/169059.169166. URL http://dl.acm.org/citation.cfm?id=169059.169166.

[135] Maryam Tohidi, William Buxton, Ronald Baecker, and Abigail Sellen. User sketches. In *Proceedings of the 4th Nordic conference on Human-computer interaction changing roles - NordiCHI '06*, number October, pages 105–114, New York,

New York, USA, 2010. ACM Press. ISBN 1595933255. doi: 10.1145/1182475.1182487.

[136] Marti a. Hearst and Daniela Rosner. Tag Clouds: Data Analysis Tool or Social Signaller? In *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008)*, pages 160–160. IEEE, January 2010. ISBN 0-7695-3075-8. doi: 10.1109/HICSS.2008.422.

[137] Leysia Palen and Susanne Bø dker. Don't Get Emotional. In Christian Peter and Russel Beale, editors, *Affect and Emotion in Human-Computer Interaction*, chapter 2, pages 12–22. Springer-Verlag Berlin, 2010. doi: 10.1007/978-3-540-85099-1_2.

[138] Donald Norman. *Emotional Design: Why We Love (or Hate) Everyday Things*. Basic Books, New York, USA, 2010.

[139] Katrin Dziekan. *Ease-of-use in Public Transportation: A User Perspective on Information and Orientation Aspects*. PhD thesis, Royal Institute of Technology, 2010.

[140] Niels Olof Bouvin, Christina Brodersen, Susanne Bø dker, Allan Hansen, and Clemens Nylandsted Klokmose. A comparative study of map use. In *CHI '06 extended abstracts on Human factors in computing systems - CHI EA '06*, page 592, New York, New York, USA, 2010. ACM Press. ISBN 1595932984. doi: 10.1145/1125451.1125575.

[141] J O N L Bentley and Thomas A Ottmann. Algorithms for Reporting and Counting Geometric Intersections. *IEEE Transactions on Computers*, C-28(9):643–647, September 2010. ISSN 0018-9340. doi: 10.1109/TC.1979.1675432. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1675432.

[142] Mark Berg, Marc Van Kreveld, Mark Overmars, and Otfried Schwarzkopf. Chapter 2: Line segment intersection. In *Computational Geometry*, pages 19–44. Springer-Verlag, 2010. ISBN 9783540656203.

[143] Michiel Smid. Computing intersections in a set of line segments: the Bentley-Ottmann Aalgorithm. 2010.

[144] Wikipedia. Bentley–Ottmann algorithm, 2010. URL http://en.wikipedia.org/wiki/Bentley\OT1\textendashOttmann_algorithm.

[145] F Glover, C McMillan, and B Novick. Interactive decision software and computer graphics for architectural and space planning. *Annals of Operations Research*, 5

(3):557–573, 2010. ISSN 02545330. URL http://www.springerlink.com/index/h44121747q7430t4.pdf.

[146] Fred Glover. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*, 13(5):533–549, 2010. ISSN 03050548. doi: 10.1016/0305-0548(86)90048-1. URL http://linkinghub.elsevier.com/retrieve/pii/0305054886900481.

[147] Michel Gendreau. An introduction to tabu search. *Handbook of metaheuristics*, pages 37–54, 2010.

[148] Fred Glover and Manuel Laguna. Tabu search. volume 16, December 2010. doi: 10.1089/cmb.2007.0211.

[149] Fred Glover and Eric Taillard. A user's guide to tabu search. *Annals of Operations Research*, 41(1):1–28, March 2010. ISSN 0254-5330. doi: 10.1007/BF02078647. URL http://link.springer.com/10.1007/BF02078647.

[150] Allan J. Macleod. A generalization of Newton-Raphson, 1984. ISSN 0020-739X.

[151] F Glover. Tabu search-part I. *ORSA Journal on Computing*, 1(3):190–206, 2010. ISSN 10919856. doi: 10.1287/ijoc.1.3.190. URL http://www.smartframe.de/studentsatwork/literatur/sel_glover_a.pdf.

[152] Fred Glover. Tabu search: A tutorial, 2010. ISSN 00922102.

[153] S Cabello, M Deberg, and M Vankreveld. Schematization of networks. *Computational Geometry*, 30(3):223–238, March 2010. ISSN 09257721. doi: 10.1016/j.comgeo.2004.11.002.

[154] Daniel Delling, Peter Sanders, Dominik Schultes, and Dorothea Wagner. Engineering route planning algorithms. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5515 LNCS, pages 117–139, 2010.

[155] P E Hart, N J Nilsson, and B Raphael. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *Ieee Transactions On Systems Science And Cybernetics*, 4(2):100–107, 2010. ISSN 05361567. doi: 10.1109/TSSC.1968.300136. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4082128.

[156] W. Zeng and R. L. Church. Finding shortest paths on real road networks: the case for A*. *International Journal of Geographical Information Science*, 23(4):531–543, April 2010. ISSN 1365-8816. doi: 10.1080/13658810801949850. URL http://dl.acm.org/citation.cfm?id=1552426.1552452.

[157] UITP. Strategic Research Agenda for urban, suburban and regional public transport and urban mobility in the European Union. Technical report, Brussels, 2010.

[158] Simon Tucker and Steve Whittaker. Have A Say Over What You See: Evaluating Interactive Compression Techniques. In *Proceedingsc of the 13th international conference on Intelligent user interfaces - IUI '09*, page 37, New York, New York, USA, February 2010. ACM Press. ISBN 9781605581682. doi: 10.1145/1502650.1502659.