District Design for a Parcel Delivery and Pick up Problem

Title	District Design for a Parcel Delivery and Pick up Problem
Issue Date	01/06/2009
Abstract	In this dissertation, a mathematical formulation and a solution method are proposed for a logistics districting problem. Experimental results in comparison with CPLEX 11.1 are presented in order to test the performance of the proposed solution procedures. The testing was performed using several types of generated instances as well as using data provided by a parcel company. The procedures strive to optimize two criteria: compact shape of the districts and balance of workload content among districts. The problem in which a geographical area is portioned into districts with the aim of optimize some criteria under consideration has been shown to be NP- Complete by Altman, 1997. The difficulty of the problem makes employing an exact procedure impractical and motivated us to propose heuristics to solve the problem. The procedure proposed is a hybrid heuristic that combines some elements of metaheuristics such as Tabu Search and GRASP.
Discipline	Ingeniería y Ciencias Aplicadas / Engineering & Applied Sciences
Item type	Tesis de Doctorado
???pdf.cover.sheet.thesis.degree.name???	Programa de Gradurados en Arquitectura e Ingeniería
???pdf.cover.sheet.dc.contributor.advisor???	Dr. Neale R. Smith

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

CAMPUS MONTERREY

ARQUITECTURE AND ENGINEERING DIVISION

ENGINEERING GRADUATE PROGRAM

"District design for a parcel delivery and pick up problem"

PRESENTED AS PARTIAL FULFILLMENT FOR THE DEGREE OF:

DOCTOR OF PHILOSOPHY (PhD) IN INDUSTRIAL ENGINEERING

PRESENTED BY:

ROSA GUADALUPE GONZALEZ RAMIREZ

MONTERREY, N.L. JUNE 2009

INSTITUTO TECNOLÓGICO Y DE ESTUDIOS SUPERIORES DE MONTERREY

CAMPUS MONTERREY

ARQUITECTURE AND ENGINEERING DIVISION
ENGINEERING GRADUATE PROGRAM

Committee members recommend that the research project presented by Rosa Guadalupe Gonzalez Ramirez be accepted as a partial requirement to obtain the academic degree:

Ph.D in Engineering Sciences, Major in Industrial Engineering

Thesis committee

Dr. Neale R. Smith
Advisor

Dr. Ronald G. Askin
Advisor

Dr. José Luis González Velarde
Committee member

Dr. Vyacheslav Kalashnikov
Committee member

Dr. José Manuel Sanchez
Committee member

Dr. Joaquín Acevedo Mascarúa

Director de Investigación y Posgrado.
Escuela de Ingeniería
Junio, 2009.

District design for a parcel delivery and pick up problem

Advisor: Dr. Neale R. Smith Cornejo

Advisor: Dr. Ronald G. Askin

Committee members:

Dr. José Luis González Velarde

Dr. Vyacheslav Kalashnikov

Dr. José Manuel Sánchez

District design for a parcel delivery and pick up problem

By

Rosa Guadalupe González Ramírez

Submitted to the School of Engineering and Architecture on June 2009,
in partial fulfillment of the requirements for the degree of Doctor of
Philosophy in Engineering Sciences

Abstract

In this dissertation, a mathematical formulation and a solution method are proposed for a logistics districting problem. Experimental results in comparison with CPLEX 11.1 are presented in order to test the performance of the proposed solution procedures. The testing was performed using several types of generated instances as well as using data provided by a parcel company. The procedures strive to optimize two criteria: compact shape of the districts and balance of workload content among districts. The problem in which a geographical area is portioned into districts with the aim of optimize some criteria under consideration has been shown to be NP-Complete by Altman, 1997. The difficulty of the problem makes employing an exact procedure impractical and motivated us to propose heuristics to solve the problem. The procedure proposed is a hybrid heuristic that combines some elements of metaheuristics such as Tabu Search and GRASP.

Keywords: Districting, compactness, workload balance, hybrid heuristics, hyperheuristic, Tabu Search, GRASP.

Acknowledgments

I thank God for giving me the opportunity to complete this important stage in my life. Studying a PhD is a hard decision. It implies sacrifice and devotion. It is a decision that not everybody understands nor agrees with. I had many doubts at the beginning, but now I am sure it was the best decision I could have made.

I want to express my thanks to all the people who have contributed to this achievement. I will do this in chronological order. I want to express my gratitude to my professors from Campus Toluca. Thanks to Dr. Rodolfo Torres, Dr. Ileana Castillo, Dr. Juan Gaytan, Dr. Luciano Chirinos, Dr. Manuel Robles, Dr. Glicina Merino, Dr. Federico Trigos, and Dr. Agustin Pichardo among others. Thanks for all your support and for encouraging me to study the PhD. As I have previously mentioned, it was a hard decision, but thanks for all your advice and recommendations.

I also want to thank all the professors from Campus Monterrey, that supported me during my studies, thanks to Dr. Dagoberto Garza, Dr. Garrigoux, Dr. Jorge Limon, Dr. Francisco Angel Bello, Dr. Eduardo Uresti, among others. Thanks for all your support and contributing to my formation during my PhD studies. I also thank the professors at Arizona State University (ASU), Dr. Ahmet Keha, Dr. Colbourn, and Dr. Rene Villalobos, among others. Thanks for all your time and support during my studies at ASU. I want to especially thank Dr. Ahmet Keha for his support to my studies and research.

I want to thank Lic. Adrian Alvarez and the personnel of the Computer Laboratory, for their support with the equipment that I used to perform the numerical experimentation of this research. Thanks also to all my friends, from Zamora, Morelia, Toluca, Monterrey, and Arizona. Thanks for making my life more enjoyable. I also thank to my parents Everildo and Celina for their love and support.

Given that the last requirement to conclude my studies is the Degree Examination, I want to end this section by expressing my gratitude to all the members of my committee and my advisors. Thanks to Dr. Jose Manuel Sanchez for your comments and accepting to be part of my committee. Thanks to Dr. Slava Kalashnikov for being part of my committee, my professor at Campus Monterrey, as well as a coauthor of one of my first papers submitted to a journal. Thanks to Dr. Jose Luis González for several reasons. First, for accepting me as a member of the Supply-Chain Research Group and for supporting my studies both at Campus Monterrey and ASU, and also for all his advice that contributed significantly to my work. I also thank him for the support with the equipment to perform the numerical experimentation of this research.

Thanks to Dr. Askin for being both my professor at ASU and my advisor. Thanks for all your ideas and contributions to my research, including my inventory and pricing paper, the aerospace project, directed jointly with Dr. Rene Villalobos, and this dissertation. Thanks for all your comments and significant ideas. I enjoyed the time I spent at ASU, it was an enjoyable and enriching experience.

I want to conclude expressing my gratitude to my advisor Dr. Neale Smith. I do it at the end because it is with him that I end up this important stage in my life. Thanks for all his guidance and training to do research, and especially all his patience and attention that he devoted to me. I have been very fortunate for having him as my advisor. I have been also fortunate of being his first PhD student. Thanks Dr. Neale for making my days in the PhD truly enjoyable and satisfactory. I am forever grateful. And I expect having you as my advisor forever.

Dedication

To my beloved parents and sisters Celina and Virginia.

Thanks for your comprehension and support.

Table of contents

Chapter 3 Problem formulation

Chapter 4 Methodology description

Appendixes

List of tables

List of figures

Preface

This thesis is concerned with a logistics districting problem faced by a parcel company. The operations consist of the delivery and pickup of packages within a region. To perform the operations, the region is divided into “districts” (zones), each of which is served by a single vehicle that departs from a central depot where packages are received. The districting process consists of partitioning the region into smaller areas while optimizing some criteria under consideration. Specifically, the parcel company is interested in two criteria: districts of compact shape and a balance of the workload content among the districts.

In this dissertation we propose a mathematical model for the logistics districting problem previously described as well as a solution methodology that consists on a heuristic algorithm. The use of a heuristic is justified by the difficulty of the problem. The definition of the boundaries of district is a complex task that depends on different factors such as traffic conditions, the urban structure of the region which includes obstacles, bridges, close streets, as well as the density of the points to be served over the region. This problem has been shown to be NP-Complete in (Altman, 1997), reason for which an exact method may be difficult to implement and not practical, since the difficulty of the problem increases exponentially with respect to the size of the instance. Furthermore, the mathematical formulation proposed for this problem requires that the capacity of the districts with respect to the number of pickups and deliveries to be served should not be exceeded making that even finding a feasible solution results very difficult.

Since many combinatorial optimization problems of practical interest, such as the problem addressed here, are computationally intractable, heuristic and metaheuristic techniques have been widely used in the literature. In recent years, it has been shown that a skilled combination of concepts from different optimization techniques can provide better results and more flexibility to solve real-world and large-scale problems. These findings motivated us to propose hybrid heuristic procedures which combine elements of some metaheuristics such as Tabu search and GRASP, and also include simple hyperheuristics.

The methodology proposed in this research may be part of a decision system that will support the process of redesigning the districting configuration of a parcel company with the aim of optimizing the criteria previously mentioned. To test the performance of the procedure, several types of instances were generated, including one that was created with data provided by a parcel company that operates in the metropolitan area of Monterrey. Experimental results in comparison to CPLEX solutions are presented in which we observe good solutions and low computation times.

Chapter 1

Introduction

Districting or territory design is a geographical problem that involves the partitioning of a region into smaller areas in order to optimize the operations for some criterion under consideration (Muyldermans et al., 2003). A district design is used for a relatively long period of time because of the big effort that a redistricting process requires. For this reason, districting is considered a strategic activity allowing a good performance of the operations performed in the region and robustness to small changes in a variety of factors.

Districting emerges in different contexts such as politics, health care, sales territory alignment, emergency centers such as fireman and police stations, logistics or routing applications, and schools. In each case, districting serves a different purpose and can be economically motivated or have a demographic background (Kalcsics et al. 2005). For instance, in a political districting problem, the region under consideration is partitioned into smaller regions from which electoral candidates are selected. On the other hand, logistics districting is mainly associated with the routing activities of a company, having a strong impact in their performance, (Van Oudheusden et al., 1998). The most common contexts of the logistics districting problem are: distribution/collection services, emergency services, medical, fire and police, (Moonen, 2004).

In this thesis, a logistics districting problem for the distribution and collection of packages within a service region is addressed, motivated by a real-world application. We present a mathematical model with some variants as well as a solution methodology based on a hybrid heuristic that combines some elements of metaheuristics such as Tabu search and GRASP. A heuristic is proposed because of the difficulty of the problem that has been shown by Altman (1997) to be NP-Complete.

The aim of the methodology proposed is to be part of a decision system that may support in the redesign of the districting configuration of a service region such that demand variations may be considered for the districting configuration. It was not possible to implement the proposed procedures in the parcel company under consideration, but the company provided GPS data with the location of the customers served according to their currently defined districts. Based on this data, some instances were generated and tested as part of the numerical experiments performed. Numerical results show a good performance of the methodology proposed, and we solve relatively large instances in reasonable computational times. The heuristic proposed is able to find solutions for which CPLEX could not even find an integer solution.

1.1 Problem description

A parcel company needs to design the districting configuration of a service region with the aim of optimizing the performance of its operations. The service activities consist of picking up and delivering packages within the service region. The company has *m* vehicles, each serving a single district. The locations of most of the customers vary each day but there are also a small number of fixed customers. Demand is seasonal, tending to increase during the Christmas season. The company has a central depot from which the vehicles depart to perform the services required in each district before returning to the depot. There are two shifts: a.m. and p.m., each of 4 hours.

The company currently performs its districting procedure manually, requiring approximately three weeks to be completed. Districts are redesigned every year and a half and the routing is assumed to be defined dynamically during the daily operations. The exact location of the customers and the daily volume of demand are stochastic. However, the districting configuration managers think that a reasonable approach is to use the data of a high workload day which is representative of the current and growing demand. Using the data of the representative day, the locations of the customers are identified on a map and the districts are defined following what they call a *spiral-sweep procedure* in which they attempt to balance the workload of the districts.

The company has defined some metrics to measure the efficiency of its district designs. The most important of them is the *Stops Per On-Route Hour* (SPORH), which is computed by dividing the number of stops in a route by the number of hours on the route. Stops in a route include the number of deliveries, pickups, empty and special stops. Empty stops refer to the cases in which the customer was not at the address in which a package should be delivered and special stops include emergency stops (which are not common). The number of hours on the route includes the traveling time required for the vehicle departing from the depot until it returns. The company considers that a high value of this metric reflects an efficient and correct performance of the operations under a districting configuration, which is also affected by other factors of human nature not only by the districting design but allows the company to estimate the performance. The company desires that the districts may have a relatively equal value of this metric, because this estimates a balanced workload content among the districts.

Another two metrics defined by the company are based on the line haul distance from the depot to a point in a district: the *distance from the depot to the last point visited in the route* (STEM IN) and the *distance from the first point visited in the route to the depot* (STEM OUT). The company is interested in designing districts such that the SPORH metric is maximized which implies that the workload content among the districts should be balanced. They are also interested in creating non-overlapping districts of compact shape. A districting configuration in which the districts overlap would not have any meaning for the company, but it depends on the urban structure of the region. In this problem, leaving the district to travel from one point to another point in the district is permitted.

1.2 Research problem

It is required to cluster a set of points on a geographical area for pickup and delivery service operations. The location of the points is assumed to be known (from a representative working day that usually is a high workload day in order to estimate the demand growth). Each point represents a customer that requests a service: pickups and/or deliveries. There exists a single depot from which *m* vehicles depart to perform the activities requested in their corresponding district and then come back to the depot. Districts should be defined such that the workload content among the districts is balanced the districts are of compact shape considering the urban structure of the service region. This research is only concerned with the districting design and the actual route that the vehicles follow within a district is assumed to be defined dynamically during the day's operations. We also must point out that given that the methodology relies on a representative working day, there could be areas of the region in which no service is required by any customer but which may have required service on a different day. For this reason, the solution obtained by this methodology is part of a decision support system in which the final definition of the districts borders will be done manually.

1.3 Research questions and objectives

1. Is it possible to model the logistics districting problem as a deterministic mixed integer program?
2. How can the different objectives and constraints of the parcel company may be included in the mathematical formulation of the problem?
3. Is the heuristic solution methodology proposed in this research an efficient and computationally tractable approach?
4. How can the urban road structure be modeled?
5. For the parcel company under consideration, compared with the current districting configuration of the company, is the solution found with the proposed methodology a better districting design?

The general objectives of this research are:

a) Propose a mathematical formulation of the districting problem faced by a parcel company that includes the requirements and considerations of the company.
b) Propose a solution methodology to efficiently solve the problem.
c) Generate a set of data with different characteristics, that include instances formed by data from a parcel company, and also that resembles the infrastructure of an urban area, with the aim of evaluating the performance of the heuristics proposed.

1.4 Research Methodology

The methodology followed in this thesis is described below:

1. Obtain relevant information of the problem,
2. Review of the literature related to the problem,
3. Present some mathematical models of the problem,
4. Propose a heuristic procedure to solve the problem,
5. Design a set of test instances of different types, including instances that resemble the infrastructure of a urban area.
6. Prepare an instance with real data provided by the parcel company.
7. Solve the set of instances with the different algorithms proposed.
8. For comparison purposes, model with AMPL-CPLEX the mathematical formulation proposed and solve the test instances in which CPLEX is able reports an integer solution.

1.5 Thesis Structure

This document is organized as follows: a literature review of related work is presented in Chapter 2. In Chapter 3 a mathematical formulation of the research problem is presented as well as a short description of the complexity of the problem. Chapter 4 presents the solution methodology proposed to solve the problem and experimental results in comparison with CPLEX are presented in Chapter 5. Finally some conclusions and recommendations for future research are discussed in Chapter 6.

Chapter 2

Literature review/ State of the Art

In this chapter we survey the literature related to the districting problem with an emphasis on the logistics districting problem for the collection and distribution of services. We also include some literature that does not address a districting process but allows to assess the quality of a given district design. It is not intended to be an exhaustive survey of the field and for a more general literature review on the districting problem please refer to Moonen (2004), who presents the state of the art of the districting problem for the urban police context but also includes a survey for the different contexts of the problem. Also, Kalcsics et al. (2005) present an extensive review of territory design problems, identifying common features to many territory design problems with special attention to the politics and sales territory alignment applications.

Section 2.1 presents a description of related work done for logistics districting applications as well as the differences with respect to the problem and approach proposed in this research. In Section 2.2 a brief review of related districting problems in other applications is done. Section 2.3 presents a classification of the districting problem with respect to the solution method approach.

2.1 Main contributions on the logistics districting problem

One of the earliest works done in this area is presented by Keeney (1972). He address the problem of districting an area for the case in which there is a set of service facilities with known location so that each district is assigned to a single facility. The objective function consists of a single criterion, which is the minimization of the distance traveled to serve the customers. The author proposes a graphical solution method. His work mainly differs to this research in that the author considers a single criterion and assumes a facility per district while here two criteria are optimized and there exists a central depot from which all the vehicles depart to service the region. Deckro (1977) presents a general heuristic to partition a region into districts considering multiple and divergent criteria. He proposes an algorithm based on a clustering technique, forming districts that fall between certain ranges of the criteria in a lexicographic order. His works differs in that he is not addressing a specific problem, and he proposes a general methodology which can be extended to the specific conditions of a districting problem.

Hardy (1980) compares the method for vehicle routing proposed by Clarke and Wright (1964) with a methodology based on a districting approach. He analyzes the urban delivery network of a wholesale milk processor and shows that both procedures give similar results but the districting approach designs districts with a more balanced workload content. The problem addressed by the author differs to the problem studied here in that there is a facility per district while in this research there is a central depot from which the vehicles depart. The methodology proposed to solve the problem is also different because the author proposes an algorithm based on the transportation problem while our approach is based on local search and we model the problem as a graph.

Wong et al. (1984) consider a problem very similar to districting analysis known as the *Vehicle Routing using Fixed Delivery Areas* (VRFDA), in which a service area is divided into fixed sub areas in which the daily route followed may change from day to day. The authors propose a methodology in which the distance traveled is minimized, differing from the present work, in which two objectives are optimized.

The authors include the workload content of the districts as a constraint that should not exceed a defined limit and no attempt is made to balance workloads among districts. Another difference is that the authors assume that location of the customers is known and only the demand is variable. A special case of the VRFDA is the *Fixed Routes Problem* (FRP) studied by Beasley (1984) in which the service region is divided into sub areas in which the route is fixed from day to day.

Daganzo (1984a) proposes an approximation method for the design of multiple-vehicle delivery tours assuming that the underlying network is equivalent to a Euclidean metric and that the objective is to find tours of minimum total length, but the number of tours is not necessarily large. The objective of his work is to explore the impact that the zone shape has on the expected length of each route. He also shows how the travel time or distance depends on the parameters of the system such as the size of the region, the density of the points and the capacity of the vehicle. He presents a very simple strategy for building good traveling salesman tours in zones of irregular shapes without the help of a computer. This paper is related to the problem addressed here but the author does not specifically deal with a districting problem. Daganzo (1984b) presents a methodology in which the region is partitioned into zones of nearly rectangular shape elongated toward the source. In his work, the number of points is large compared to the capacity of the vehicles. This is different from our problem, in which this is not an issue because of the type of activities and the type of products managed by a parcel company.

Newell and Daganzo (1986a) analyze the districting of a region in which the underlying network of roads is a dense ring-radial network. Their analysis is similar to that of Daganzo (1984a,b), and propose an approximation method for the design of multiple-vehicle delivery tours in which the aim is to minimize total distance, using a ring-radial network in a large region that contains many routes. They consider the situation in which the origin is located at the center of the region and the density of demand varies according to the distance from the origin. Their work differs from the present work in that a ring-radial network metric is assumed and a single criterion is optimized. Han and Daganzo (1986) investigate the design of delivery zones for distributing perishable freight without transshipment, with no consideration of the capacity of the vehicle.

Their work differs in that it is applied to perishable items and the aim of the methodology proposed is to minimized total costs. Langevin and Soumis (1989) study the problem of designing multiple vehicle delivery tours satisfying time constraints for the letter and parcel pick-up and delivery problem with a continuous approximation model, which is not employed in the present work. The authors proposed a methodology that involves partitioning the region into approximately rectangular delivery zones that are arranged into concentric rings around the depot, assuming a circular area with the depot located in its center and a random location of the points with point density as a function of the radius using a ring-radial grid. The present work differs in that the ring-radial representation is not employed.

Rosenfield et al. (1992) study the problem of planning service districts with a time constraint and derive analytical expressions to determine the optimal number of service districts for the U.S. postal system. Their work differs in that they consider that each district contains a service facility at its center from which deliveries are made to the customers uniformly distributed within the district and analyze the tradeoff between the variable cost of delivery and the fixed cost of the facilities.

Novaes and Graciolli (1999) present a methodology to design multi-delivery tours associated with the servicing of an urban region of irregular shape, where the density of servicing points and the amount of cargo vary over the served area. They propose a cluster first – route second strategy and consider customer demands and service times as random variables. Each route is assigned to a vehicle, being both restricted by time and capacity constraints in a stochastic way. The proposed methodology was applied to a parcel delivery problem in the city of Säo Paulo, Brazil. Their work differs in that the authors assume a rectangular grid structure and they propose a methodology based on a sweep approach. Novaes et al. (2000) present a methodology for solving the same problem but use a continuous approach to represent the region under analysis. Galvao et al. (2006) extend the previous work, introducing some improvements to the ring-radial model.

Muyldermans et al. (2002, 2003) address the problem of districting for salt spreading operations on roads. They utilize a graph model of the problem, which was not found in previous reviewed work, but it differs from this research in that demand occurs at the edges of the road network rather than at the nodes. They propose heuristics methodologies to solve the problem, but their main objective is based on the minimization of the deadheading distances and the number of vehicles required to service the region. Another difference is that a facility to serve each district is assumed, while in this research there is only a central depot.

Haugland et al. (2005) consider the problem of designing districts for vehicle routing problems with stochastic demands. They present a tabu search and a multistart heuristic to solve the problem. Their work differs in that they present a two-stage stochastic problem with recourse that aims to minimize the expected travel time for each district while in this work the mathematical formulation is a minimax objective function that aims to balance the workload content of the districts and form districts of compact shape. The solution approach proposed by the authors is similar to the methodology proposed here, but it differs in that we are proposing a hybrid heuristic that combines two metaheuristics (GRASP and Tabu Search), while the authors propose two different heuristics, one based on a multistart approach and the other on a Tabu Search.

Galvao et al. (2006) extended the model presented by Novaes et al. (2000).They present a special case of Voronoi diagram: the Multiplicatively-Weighted (MV)-Voronoi diagram, combined with an iterative computational procedure that differs from a geometrically-shaped districting pattern such as a ring-radial structure. The problem is modeled as a continuous approximation that differs from the present work in which a graph model of the problem is utilized. Novaes et al. (2008) develop two continuous location-districting models applied to transportation and logistics problems combining a Voronoi diagram with an optimization algorithm, which also differs from the present work in that a continuous approximation model is assumed.

Tavares-Pereira et al. (2007a) consider the districting problem with multiple criteria. They propose a method to approximate the Pareto front based on an evolutionary algorithm with local search. They apply this methodology to the Paris public transportation system with the aim to analyze its current pricing system. Their work differs in that they measure the workload content of the districts by the number of points assigned to each district, with no inclusion of the line haul distance from the depot to the districts. Furthermore, they do not differentiate between the service activities as is done in this research. Tavares-Pereira et al. (2007b) propose some metrics to compare partitions obtained in a districting configuration, specifically for the case of a connected, undirected, and planar graph representation of the service region.

2.2 Contributions on related districting problems

In this section we present a brief review of related districting problems in the rest of the applications in which districting problems arise. Section 2.2.1 describes the Political districting applications and the main contributions in this application. Section 2.2.2 describes the Sales territory alignment application with a brief review of the work done in this context. Section 2.2.3 presents a review on the School districting. Section 2.2.4 presents a review on the Emergency sites and Health care systems applications.

2.2.1 Political Districting

According to Morril (1981), the problem of determining political districts consists of dividing a governmental area such as a city or a state into subareas from which political candidates are elected. This problem is important for democratic countries such as New Zealand, Canada and most of the states of U.S. and Germany, in which each territory elects a single member to a parliamentary assembly. Several authors have worked in this problem, among we can mention Hess et al. (1965) who present a location-allocation heuristic under population equality, compactness and contiguity considerations. Garfinkel and Nemhauser (1970) present an exact algorithm to solve this problem under contiguity, compactness and limited population deviation.

Hojati (1996) proposes a three-stage approach in which he first determines the district centers, then an assignment problem for allocating population units to districts and then a sequence of capacitated transportation problems are solved for dealing with splits. A column generation based on branch an price approach is proposed by Mehrotra et al. (1998) to solve a political redistricting problem. A review on political redistricting is presented by Williams (1995).

2.2.2 Sales Territory Alignment

According to Moonen (2004) sales territory alignment is one of the main decisions that has to be taken for the sales force deployment. It consists of dividing a region into smaller regions to which salesman are assigned. An important concept in this problem is the Sales Coverage Units (SCU), which are interpreted as the atomic regions that are added together to constitute sales territories, (Howick and Pidd, 1990). The design of sales territories has different purposes, but in general the most common criteria optimized are contiguity, compactness and workload balance. Workload is measured according to the specific characteristics of the problem, and it may include the sales potential of the territory, the number of customers, number of activities to be performed, etc.

This problem has been studied with numerous and different assumptions and techniques. In (Zoltners, 1979) and (Zoltners and Sinha, 1983) is presented the first review of sales territory design models. In this work, sales units are assigned entirely to SCU. In (Fleishman and Paraschis, 1988) the authors study a sales territory alignment for a German company for consumer goods and develop a procedure based on a location-allocation approach in which sales units are assigned to SCU for a certain proportion of time. This problem has been studied also simultaneously with some other elements of the sales deployment problem such as sales force sizing, salesman location and sales resource allocation, as it is done by Drexl and Haase (1999).

More recently, a commercial territory design was introduced by Rios-Mercado and Fernandez (2009) that differs with respect to the sales territory design in that rather than placing salesmen in territories the authors are interested in locating centers. It can be also interpreted as providing customers with a commercial service by the firm. They propose a GRASP approach that incorporates reactivity and filtering.

The authors study the case of a beverage distribution firm in the city of Monterrey, Mexico. Other variants and extensions of this problem are presented in (Caballero-Hernandez et al., 2007), (Segura-Ramiro et al., 2007) and (Fernandez et al., forthcoming). The authors propose a similar approach as it is done here, but differs in that they are addressing a different problem with different criteria to be optimized. Also the authors do not propose a hybrid algorithm as it is done here.

2.2.3 School Districting

According to Caro et al. (2004), School redistricting is the process of adjusting the boundaries of schools within a given school system. This process is done in response to different factors, such as overcrowded classrooms, projected growth and decline of enrolments, school capacities, etc. School districting aims to form districts in which students are assigned to their closest schools as possible. Some other criteria considered are capacity of the schools, grade levels, hazards for the students in their way to the school, racial balance and the urban structure. Compactness is also a criteria to be optimized as well as contiguity, which avoid that children from the same neighborhood be assigned to different schools.

Several authors have paid attention to this problem. Diamond and Wright (1987) also study this problem but they consider the case in which only a limited number of schools are allowed. Elizondo et al. (1997) presents a model in which individual students are assigned to schools and the objective is to minimized distances of the students to the schools. Later, Church and Murray (1993) extend previous work presenting a multi-objective model. For a review of the most relevant work in this problem refer to Caro et al. (2004).

2.2.4 Health care systems

Districting for health care systems is the process of aggregating administrative units into territories for social facilities like hospitals, such that for every inhabitant is determined to which facility he should go in order to be serviced. According to Pierskalla and Brailer (1994) the regionalization of health services is the partitioning of an area into districts with the required number of hospitals to provide the medical services to the population within the district. The aim of this process is to lower the cost and improve quality of the service provided to the inhabitants. Blais et al. (2003) study a districting problem for a local community health clinic optimizing visiting personnel mobility and workload equilibrium combined into a single objective function, and propose a Tabu Search approach to solve the problem.

2.2.5 Emergency sites

The districting problem in this context consists of portioning a region into response areas for public services such as police, fire and medical emergency services. The servers such as ambulances are mobile and travel to the problem sites to perform the services required. This type of problem is labor intensive and is characterized by urgency and the most important criterion is response time. Police differs from medical and fire in that the servers are possibly mobile when a call is received, while for medical and fire servers are stationed in a fixed place from where they are dispatched according to the received calls.

Several works exist in this context, from which we can mention Carter et al. (1972) who study the districting problem for emergency sites in the plane and Bernand and Larson (1985) who study the problem on a network. Baker et al. (1989) study the redistricting design of primary response areas for county ambulance services. Yang et al. (2004) present a case study for a fire districting using simulation. Regarding the police context, we can mention several works that are found in the literatures, such as Bodily (1978) that designs patrol sectors by a multi attribute utility theory approach to include preferences of the interest groups, which is not a common approach found in other applications of the districting problem.

More recently, D'Amico et al. (2002) present a simulated annealing algorithm for the redistricting police command boundaries, formulating the problem as a constrained graph-partitioning problem, considering factors such as contiguity, compactness and convexity. For a more extensive review of this application of the districting problem, Moonen (2004) presents a more detailed review.

2.3 Classification of the solution methods

Tavares-Pereira et al. (2007a) comment that districting problems can be classified in terms of two factors: the number of criteria and the solution method (exact and non-exact algorithms). However, since it has been shown that this problem is NP-Complete, all the reviewed works propose heuristic methods except for Keeny (1972). In this research a heuristic approach is also proposed.

According to Grilli di Cortone et al. (1999), there are two main techniques for districting problems: division and agglomeration. In the former, the service region is considered as a whole and divided into pieces. The latter consider a region that is already split in small areas that are aggregated to build the districts. Table 4.2 presents a classification of the different districting techniques proposed by Moonen (2004).

Table 2.1: Districting techniques classification, Moonen (2004).

DIVISION	AGGLOMERATION
a) Voronoi	a) Set Partitioning
b) Sweep algorithms.	b) Graph Partitioning
	c) Clustering
	d) Improvement and local search
	e) Allocation
	i. Multi-Kernel growth
	ii. Transportation problem
	iii. Integer programming

Appendix I presents a table in which summarizes the main contributions to the logistics districting problem discussed in section 2.1, with a classification according to the solution approaches proposed and also distinguishes the main differences with respect to the present work both in terms of the problem addressed and the solution methodology proposed.

Chapter 3

Problem formulation

In this chapter a mathematical formulation of the problem is presented. Section 3.1 presents the mathematical formulation proposed. Section 3.2 presents some variants of the formulation considering different metrics of the workload content. Section 3.3 presents a brief discussion of the complexity of the problem Section 3.4 presents some ideas for a lower bound computation.

3.1 Mathematical formulation of the problem

Consider a connected, undirected graph *G*(*V*,*E*) where *V* is the vertex set and *E* the edge set. The graph is generally not complete. We assume that all the edges $e_{rs} = (v_r, v_s)$ have a positive length and represent a real road between adjacent points v_r and v_s. Distances between points are edge lengths for those points that are connected in the graph and shortest path distances for other pairs of points. A district is defined as a subset of the points. Each vertex may require either a pickup or a delivery. The aim of the districting procedure is to optimize two criteria: balance of the workload content among the districts and compactness of district shapes. The mathematical model proposed for this problem consists of a single objective model in which the weighted sum of both criteria is minimized.

Compactness is not defined precisely for all the districting problems in the literature and it is generally defined according to the application context. For this problem we define it as the distance between the two furthest apart points in a district and we proposed a minimax objective in which we attempt to obtain compact districts when the maximum compactness metric is minimized.

The workload content of a district is defined to be the time required to perform all required pickups and deliveries and the time needed to drive from the depot to the farthest point in the district. In order to balance the workload content among districts, we propose to minimize the maximum workload allocated to a district. We also attempt to obtain districts with balanced workload content by minimizing the dispersion of the workload assigned to each district, which is represented by the sum of the absolute value of the differences between the workload content of each district with respect to the average workload.

We propose a hierarchical mathematical formulation in which the first model is a linear programming problem in which the weighted sum of the compactness metric and the maximum workload content assigned to a district is minimized, each of them normalized.

Then a second optimization model is solved in which the objective function aims to minimize the dispersion of the workload content among the districts, but respecting the value of the weighted sum of the compactness and balance workload metrics found when the first optimization problem was solved.

The following notation is defined:

α= Maximum number of pickups for each district,

β= Maximum number of deliveries for each district,

J= District set, $J=\{1,\ldots,m\}$,

wp_i= Number of pickups (1 or 0) requested by demand point $i, i \in V$,

wd_i= Number of deliveries (1 or 0) requested by demand point $i, i \in V$,

Std= Stopping time per delivery in each demand point,

Stp= Stopping time per pick up in each demand point,

d_{ik}= Distance from point i to point k, $i, k \in V$,

λ= Scale factor, $0 \leq \lambda \leq 1$,

d_{0i}= Distance from the depot to the point i $i \in V$,

Sp= average speed,

Nz= normalization parameter for the compactness metric,

Nw= normalization parameter for the workload metric. Computation of both normalization parameters is described in section 3.1.3.

OF1= Objective function value of the first optimization lineal model.

OF2= Objective function value of the second optimization quadratic model.

The following decision variables are defined:

$$X_{ij} = \begin{cases} 1 & \text{if customer } i \text{ is assigned to district } j \\ 0 & \text{otherwise} \end{cases}$$

And the following auxiliary variables are defined:

W= Continuous variable that represent the maximum workload content assigned to a district,

Z= Continuous variable that measure the compactness as the maximum travel time between the furthest apart points of a district,

D_j=Continuous variable that takes the value of the traveling time from the depot to the farthest point of district *j*,

M_j= Continuous variable that takes the value of the traveling time between the two furthest apart points of district *j*,

Y_{ij}= Auxiliary binary variables used to restrict that D_j takes the value of time to travel from the depot to a point in a district.

$dispersion_j$= Auxiliary continuous variables that take the absolute value of the difference between the workload content of each district with respect to the average workload assigned to the districts.

3.1.1 First Optimization Model

$$Min \quad OF1 = \frac{\lambda W}{Nw} + \frac{(1-\lambda)Z}{Nz} \tag{1}$$

subject to

$$\sum_{j \in J} X_{ij} = 1 \qquad \forall i \in V \tag{2}$$

$$\sum_{i \in V} wp_i X_{ij} \leq \alpha \qquad \forall j \in J \tag{3}$$

$$\sum_{i \in V} wd_i X_{ij} \leq \beta \qquad \forall j \in J \tag{4}$$

$$M_j \geq \frac{d_{ik}(X_{ij} + X_{kj} - 1)}{Sp} \qquad \forall j \in J, i \in I, k \neq i \in V \tag{5}$$

$$Z \geq M_j \qquad \forall j \in J \tag{6}$$

$$D_j \geq d_{i0} X_{ij} \qquad \forall i \in V \ \ \forall j \in J \tag{7}$$

$$W \geq Std \sum_{i \in V} wd_i X_{ij} + Stp \sum_{i \in V} wp_i X_{ij} + D_j / Sp \qquad \forall j \in J \tag{8}$$

$$X_{ij} \in \{1,0\} \qquad \forall i \in V, \ \forall j \in J \tag{9}$$

Equation (1) is the objective function that minimizes a weighted average of the maximum workload and maximum compactness metrics. The objectives are normalized and the relative weighting is given by λ. Constraints (2) guarantee that each demand point is assigned to only one district. Constraints (3) and (4) guarantee that each district has a maximum of α pickups and β deliveries, respectively. These constraints help to balance the number of pickups and deliveries allocated to a district so that the capacity of the vehicles is not exceeded. Constraints (5) guarantee that M_j takes the value of the maximum travel time between the points assigned to each district in time units. Constraints (6) guarantee that Z takes the maximum value over M_j. Constraints (7) guarantee that D_j takes the value of the time from the depot to the farthest point of each district j. Constraints (8) guarantee that W takes the maximum amount of workload of a district. Constraints (9) are the binary requirements.

3.1.2 Second Optimization Model

$$Min \quad \sum_{j \in J} dispersion_j \tag{10}$$

subject to

$$\sum_{j \in J} X_{ij} = 1 \qquad \forall i \in V, \tag{11}$$

$$\sum_{i \in V} wp_i X_{ij} \le \alpha \qquad \forall j \in J, \tag{12}$$

$$\sum_{i \in V} wd_i X_{ij} \le \beta \qquad \forall j \in J, \tag{13}$$

$$M_j \ge \frac{d_{ik}(X_{ij} + X_{kj} - 1)}{Sp} \qquad \forall j \in J, i \in I, k \ne i \in V, \tag{14}$$

$$Z \ge M_j \qquad \forall j \in J, \tag{15}$$

$$D_j \ge d_{i0} X_{ij} \qquad \forall i \in V, \forall j \in J, \tag{16}$$

$$D_j = \sum_{i \in V} d_{i0} \left(X_{ij} + Y_{ij} - 1 \right) \qquad \forall j \in J, \tag{17}$$

$$\sum_{i \in V} \left(X_{ij} + Y_{ij} - 1 \right) = 1 \qquad \forall j \in J, \tag{18}$$

$$X_{ij} + Y_{ij} \ge 1 \qquad \forall i \in V, \forall j \in J, \tag{19}$$

$$W \ge Std \sum_{i \in V} wd_i X_{ij} + Stp \sum_{i \in V} wp_i X_{ij} + D_j / Sp \qquad \forall j \in J, \tag{20}$$

$$dispersion_j \ge Std \sum_{i \in V} wd_i X_{ij} + Stp \sum_{i \in V} wp_i X_{ij} + D_j / Sp \quad - \frac{\sum_{k \in J} Std \sum_{i \in V} wd_i X_{ik} + Stp \sum_{i \in V} wp_i X_{ik} + D_k / Sp}{|J|} \qquad \forall j \in J, \tag{21}$$

$$dispersion_j \ge \frac{\sum_{k \in J} Std \sum_{i \in V} wd_i X_{ik} + Stp \sum_{i \in V} wp_i X_{ik} + D_k / Sp}{|J|} \quad - Std \sum_{i \in V} wd_i X_{ij} + Stp \sum_{i \in V} wp_i X_{ij} + D_j / Sp \qquad \forall j \in J, \tag{22}$$

$$\frac{\lambda W}{Nw} + \frac{(1-\lambda) Z}{Nz} \le OF1 + \varepsilon \tag{23}$$

$$X_{ij}, Y_{ij} \in \{1, 0\} \qquad \forall i \in V, \ \forall j \in J \tag{24}$$

Constraints (10) corresponds to the objective function of the second optimization model, that minimizes the sum of the absolute value of the dispersion of the workload content of each district with respect to the average workload. Constraints (11) to (16) are the same constraints as equations (2) to (7) of the first optimization model. Constraints (17) to (19) are auxiliary constraints to guarantee that the variable D_j takes the exact value of the time to travel from the depot to the farthest point of each district. These set of constraints are included in this model because in the first optimization model, constraint (7) allows D_j to take any value greater or equal to the time to travel from the depot to the farthest point of the district, which for the second optimization model is not enough to model the real workload content of each district. Without the inclusion of these constraints, the model would assign to D_j the required value to make that all the districts have the same workload content as the district with the maximum workload assigned, which would make that the dispersion of the workload among districts seems to have the value of zero. Constraints (20) are the same as constraints (8). Constraints (21) and (22) guarantee that for each district *j*, the difference of its workload content with respect to the average workload content takes a positive value. Constraint (23) is included in the model in order to guarantee that the objective function value of the first optimization model may be equal or less to the value obtained when this model was solved, plus an epsilon to avoid conflicts with numerical precision. Constraints (24) are the binary requirements.

3.1.3 Normalization Parameters

Nw is determined by estimating an average workload per district when it is approximately balanced.:

$$Nw = \frac{spick \cdot Stp + sdel \cdot Std + sfar / Sp}{m}, \tag{25}$$

where *spick* and *sdel* refers to the total number of pickups and deliveries requested in the service region respectively, and *sfar* is used to estimate the distance to the farthest point from the depot of each district, by summing up the distance of the *m* farthest points to the depot.

Nz is estimated by the length of the diameter (in time units) of a district, assuming that the service region is equally divided into *m* districts, which is defined by equation (26). For this, it is assumed that the area of the service region is of circular shape with the depot at the center. Also for the districts, it is estimated that are of circular shape and all of them have equal area.

$$\frac{\pi r^2}{m} = \pi (Nz/2)^2 \rightarrow Nz = \frac{2r\sqrt{\frac{1}{m}}}{Sp} \quad (26)$$

3.2 Variants of the workload content metric

We propose two variants for the workload content metric that differ in how the line haul distance from the depot to the district is measured. These variants will not be considered as part of the numerical experimentation but are proposed for further research.

Instead of considering the time to travel from the depot to the farthest point of a district, first variant considers the time to travel to the closest point of each district and the second variant considers the time to travel to the centroid of each district. The centroid is defined as the point that minimizes the sum of distances from the rest of the points assigned to the district.

3.2.1 Closest point optimization model

The following decision variable is introduced:
C_j=Travel time from the depot to the closest point of district *j*.

The first optimization model for the closest point is as follows:

$$Min \quad OF1 = \frac{\lambda W}{Nw} + \frac{(1-\lambda)Z}{Nz} \tag{27}$$

subject to

$$\sum_{j \in J} X_{ij} = 1 \qquad \forall i \in V, \tag{28}$$

$$\sum_{i \in V} wp_i X_{ij} \le \alpha \qquad \forall j \in J, \tag{29}$$

$$\sum_{i \in V} wd_i X_{ij} \le \beta \qquad \forall j \in J, \tag{30}$$

$$M_j \ge \frac{d_{ik}(X_{ij} + X_{kj} - 1)}{Sp} \qquad \forall j \in J, i \in I, k \ne i \in V, \tag{31}$$

$$Z \ge M_j \qquad \forall j \in J, \tag{32}$$

$$C_j \le d_{i0} X_{ij} \qquad \forall i \in V \;\; \forall j \in J, \tag{33}$$

$$W \ge Std \sum_{i \in V} wd_i X_{ij} + Stp \sum_{i \in V} wp_i X_{ij} + C_j / Sp \qquad \forall j \in J, \tag{34}$$

$$X_{ij}, Y_{ij} \in \{1,0\} \qquad \forall i \in V, \; \forall j \in J, \tag{35}$$

Constraint (27) corresponds to the objective function of the first optimization model that is the same as constraint (1). Constraints (28) to (32) are the same constraints as (2) to (6). Constraint (33) are the analogous constraints to (7) but they guarantee that C_j takes the value of the time to travel from the depot to the closest point of the depot instead of the farthest point considered in the previous model. Constraints (34) are the same constraints as (8). Constraints (35) are the binary requirements.

The second optimization model based on the closest point is as follows:

$$Min \quad \sum_{j \in J} dispersion_j \tag{36}$$

subject to

$$\sum_{j \in J} X_{ij} = 1 \qquad \forall i \in V, \tag{37}$$

$$\sum_{i \in V} wp_i X_{ij} \leq \alpha \qquad \forall j \in J, \tag{38}$$

$$\sum_{i \in V} wd_i X_{ij} \leq \beta \qquad \forall j \in J, \tag{39}$$

$$M_j \geq \frac{d_{ik}(X_{ij} + X_{kj} - 1)}{Sp} \qquad \forall j \in J, i \in I, k \neq i \in V, \tag{40}$$

$$Z \geq M_j \qquad \forall j \in J, \tag{41}$$

$$C_j \leq d_{i0} X_{ij} \qquad \forall i \in V, \forall j \in J, \tag{42}$$

$$C_j = \sum_{i \in V} d_{i0}\left(X_{ij} + Y_{ij} - 1\right) \qquad \forall j \in J, \tag{43}$$

$$\sum_{i \in V}\left(X_{ij} + Y_{ij} - 1\right) = 1 \qquad \forall j \in J, \tag{44}$$

$$X_{ij} + Y_{ij} \geq 1 \qquad \forall i \in V, \forall j \in J, \tag{45}$$

$$W \geq Std \sum_{i \in V} wd_i X_{ij} + Stp \sum_{i \in V} wp_i X_{ij} + D_j / Sp \qquad \forall j \in J, \tag{46}$$

$$dispersion_j \geq Std \sum_{i \in V} wd_i X_{ij} + Stp \sum_{i \in V} wp_i X_{ij} + D_j / Sp \; - \frac{\sum_{k \in J} Std \sum_{i \in V} wd_i X_{ik} + Stp \sum_{i \in V} wp_i X_{ik} + D_k / Sp}{|J|} \qquad \forall j \in J, \tag{48}$$

$$dispersion_j \geq \frac{\sum_{k \in J} Std \sum_{i \in V} wd_i X_{ik} + Stp \sum_{i \in V} wp_i X_{ik} + D_k / Sp}{|J|} \; - Std \sum_{i \in V} wd_i X_{ij} + Stp \sum_{i \in V} wp_i X_{ij} + D_j / Sp \qquad \forall j \in J, \tag{49}$$

$$\frac{\lambda W}{Nw} + \frac{(1-\lambda)Z}{Nz} \leq OF1 + \varepsilon \tag{50}$$

$$X_{ij}, Y_{ij} \in \{1,0\} \qquad \forall i \in V, \; \forall j \in J \tag{51}$$

Constraint (36) is the objective function of the second optimization model, that is the same as equation (13). Constraints (37) to (41) are the same as constraints (1) to (6) and constraints (42) to (45) are the same as constraints (16) to (19) but they guarantee that C_j takes the value of the time to travel from the depot to the closest point instead of to the farthest point. Constraints (46) to (51) are the same constraints as constraints (20) to (24).

3.2.1 Centroid optimization model

Consider the following definitions:

$$P_j = \{v_i \mid X_{ij} = 1 \quad i \in V\} \qquad \forall j \in J \tag{52}$$

$$A_j = \{e_{ik} \in E / X_{ij} = X_{kj} = 1 \quad i,k \in V\} \qquad \forall j \in J \tag{53}$$

$$centroid_j = \underset{i,k \in P_j}{\operatorname{argmin}} \left\{ \sum_{k \in P_j} d_{ik} \right\} \tag{54}$$

$$M_{P_j,A_j}(i) = \begin{cases} 1 & \text{if } v_i \text{ is a } \textit{centroid} \text{ for district } j \\ 0 & \text{otherwise} \end{cases} \tag{55}$$

The mathematical model for the centroid model is as follows:

$$Min \quad OF1 = \frac{\lambda W}{Nw} + \frac{(1-\lambda)Z}{Nz} \tag{56}$$

subject to

$$\sum_{j \in J} X_{ij} = 1 \qquad \forall i \in V, \tag{57}$$

$$\sum_{i \in V} wp_i X_{ij} \le \alpha \qquad \forall j \in J, \tag{58}$$

$$\sum_{i \in V} wd_i X_{ij} \le \beta \qquad \forall j \in J, \tag{59}$$

$$M_j \ge \frac{d_{ik}(X_{ij} + X_{kj} - 1)}{Sp} \qquad \forall j \in J, i \in I, k \ne i \in V, \tag{60}$$

$$Z \ge M_j \qquad \forall j \in J, \tag{61}$$

$$W \ge Std \sum_{i \in V} wd_i X_{ij} + Stp \sum_{i \in V} wp_i X_{ij} + 1/Sp \sum_{i \in V} d_{i0} M_{P_j, A_j}(i) \qquad \forall j \in J, \tag{62}$$

$$X_{ij} \in \{1,0\} \qquad \forall i \in V, \ \forall j \in J, \tag{63}$$

Constraint (56) corresponds to the objective function of the first optimization model that is the same as constraint (1). Constraints (57) to (61) are the same constraints as (2) to (6). Constraints (62) guarantee that *W* takes the value of the maximum workload content of a district which is accounted by the sum of the total time to perform the pickups and deliveries as well as the time to travel from the depot to the centroid of the district. Constraints (63) are the binary requirements.

The second optimization model based on the centroid is as follows:

$$Min \quad \sum_{j\in J} dispersion_j \tag{64}$$

subject to

$$\sum_{j\in J} X_{ij} = 1 \qquad \forall i \in V, \tag{65}$$

$$\sum_{i\in V} wp_i X_{ij} \le \alpha \qquad \forall j \in J, \tag{66}$$

$$\sum_{i\in V} wd_i X_{ij} \le \beta \qquad \forall j \in J, \tag{67}$$

$$M_j \ge \frac{d_{ik}(X_{ij} + X_{kj} - 1)}{Sp} \qquad \forall j \in J, i \in I, k \ne i \in V, \tag{68}$$

$$Z \ge M_j \qquad \forall j \in J, \tag{69}$$

$$W \ge Std\sum_{i\in V} wd_i X_{ij} + Stp\sum_{i\in V} wp_i X_{ij} + 1/Sp\sum_{i\in V} d_{i0} M_{P_j,A_j}(i) \qquad \forall j \in J, \tag{70}$$

$$dispersion_j \ge Std\sum_{i\in V} wd_i X_{ij} + Stp\sum_{i\in V} wp_i X_{ij} + D_j / Sp \; - \frac{\sum_{k\in J} Std\sum_{i\in V} wd_i X_{ik} + Stp\sum_{i\in V} wp_i X_{ik} + D_k / Sp}{|J|} \qquad \forall j \in J, \tag{71}$$

$$dispersion_j \ge \frac{\sum_{k\in J} Std\sum_{i\in V} wd_i X_{ik} + Stp\sum_{i\in V} wp_i X_{ik} + D_k / Sp}{|J|} \; - \; Std\sum_{i\in V} wd_i X_{ij} + Stp\sum_{i\in V} wp_i X_{ij} + D_j / Sp \qquad \forall j \in J, \tag{72}$$

$$\frac{\lambda W}{Nw} + \frac{(1-\lambda)Z}{Nz} \le OF1 + \varepsilon \tag{73}$$

$$X_{ij} \in \{1,0\} \qquad \forall i \in V, \; \forall j \in J \tag{74}$$

Constraint (64) is the objective function of the second optimization model, that is the same as equation (13). Constraints (65) to (70) are the same as constraints (57) to (62) and constraints (71) to (73) are the same as constraints (24) to (26). Constraints (74) are the binary requirements.

3.3 Complexity of the problem.

The redistricting problem has been shown to be NP-Complete by Altman (1997). Moreover, since constraints (3) and (4) impose a limit in the amount of pickups and deliveries assigned to a district, it turns out that even finding a feasible solution is a difficult task.

3.4 A lower bound.

We need to estimate a lower bound on the maximum workload in a district (W) and the maximum compactness metric estimated by the maximum time to travel between the two furthest apart points on a district (Z). For the lower bound on the maximum workload of a district we propose to compute it by taking the total pickup and delivery workload and divide it by the number of districts m and then add the mth shortest distance from a point to the depot. For the lower bound on the compactness metric we propose to estimated by the mth shortest distance between two points excluding the depot. Finally a lower bound on the objective function is computed as using equation (1) using the lower bounds on W and Z . We compared the lower bound to optimal values of instances in which we know the optimal solution and it resulted very weak. For this reason, in the numerical experimentation section, we do not compare the solutions obtained with respect to a lower bound.

Chapter 4

Methodology description

In this chapter we describe the five heuristic solution procedures proposed for the logistics districting problem of a parcel company. The difficulty of the problem as described in section 3.2 motivated us to propose a heuristic since it is well know that exact methods become impractical when applied to NP hard problems. Section 4.1 presents the general details of the methodology. Section 4.2 describes the procedure of the hybrid heuristic proposed. Section 4.3 presents the complexity analysis of the heuristics.

4.1 General details

The five heuristics consists of a hybrid multi-start heuristic algorithm that combines elements of metaheuristics as well as some hyperheuristics in the different steps of the procedure. All of them have in common the procedure to construct the initial solutions and differ with respect to the local search procedure.

A metaheuristic is a heuristic method that solves an optimization problem by combining heuristics in a hopefully efficient way and is usually equipped with some way of escaping local optima. Metaheuristics are classified as point based and population based. Point based metaheuristics maintain only one solution at a time, while a population of solutions are maintained in the latter category. Among the point based metaheuristics we can mention Tabu search (TS), Simulated Annealing, Greedy randomized adaptive search procedures (GRASP), iterated local search, guided local search, variable neighborhood search and other perturbation methods. The five heuristics proposed in this research hybridizes two metaheuristics: TS and GRASP.

TS is an adaptive memory based technique proposed in 1977 by Glover (1977), that enhances the performance of a local search procedure through the use of memory structures to aid escaping from local optima by accepting even non-improving moves. To prevent getting cycled back to previously visited solutions, last moves are labeled as *"tabu-active"* during a predetermined number of iterations. However, good quality solutions that are currently *tabu active* may be visited under some criteria that are referred to as *"aspiration criteria"*.

Comprehensive tutorials on Tabu Search are found in Glover and Laguna (1997) and (2002). GRASP is a multi-start constructive metaheuristic proposed by Feo and Resende (1989) in which an iteration consists of two phases: construction of an initial solution from scratch and then improvement of the solution by a local search approach. The construction phase includes a greedy function but is randomized by the definition of a list with the best candidates, from which one is selected randomly. Among all the solutions created, the best solution is reported as the final step of the algorithm. For a detailed description of GRASP, see Resende and Ribeiro (2002), in which the authors present details of different solution construction mechanisms, techniques to speed up the search, strategies for the implementation of memory, hybridization with other metaheuristics, and some applications.

As it was previously mentioned, the proposed methodology also involves hyperheuristics. A hyperheuristic is a heuristic that selects heuristics or metaheuristics and it has the advantage of being more general and of higher level of abstraction than a metaheuristic itself. The inclusion of some hyperheuristics in the procedure was motivated either by the wide variety of alternatives to evaluate a move and also as a way to combine local search algorithms.

4.2 Hybrid districting heuristics (HDH) description

We propose a multi-start heuristic algorithm that hybridizes a GRASP with a TS metaheuristic including also some hyperheuristics. Given that we propose five different variants of the heuristic proposed for one of its main phases, we will refer to each of them as a different heuristic. The general procedure for all the heuristics proposed consists of two phases as it is typical of a GRASP approach: construction of a feasible initial solution and improvement by local search. A solution is considered to be feasible if all the points are allocated to a district and the capacity limits with respect to both services (pickups and deliveries) are respected for all the districts, as it is established by equations (2), (3) and (4).

Among all the solutions created and improved, the best of them is reported as the final solution for a given instance. Given that we have two metrics for the balance of workload content among districts, a solution will be evaluated according to equation (1) which is based on the primary balance workload metric. In case of ties, solutions will be evaluated with equation (10) and the solution that provides the lowest dispersion value for the workload content among districts will be selected.

For the second phase of the algorithm, we propose three local search algorithms that may be applied independently or may be combined through the use of hyperheuristics. In all, five heuristics are proposed. All five attempt to improve the same initial solution constructed during each iteration of the procedure but may end up with a different final solution depending on the results of the local search phase.

Let's introduce some definitions before describing each of the five heuristics. First, we say that a point is allocated to a district when we are constructing the initial solution. Once a solution is constructed, it may be feasible or infeasible with respect to the capacity limits of the districts. A *move* refers to the process of reallocating a point from a district to an adjacent district. An *exchange* refers to the process of reallocating a pair of points from adjacent districts.

A key concept is the *adjacency* among points and districts, which for all the heuristics is a condition that should be updated when a point is assigned or moved to a district. This requirement is imposed as part of the procedure with the aim of constructing districts of compact shape. The definition of the adjacency structure has a big impact on the difficulty of an instance. For example, two points that are located on a plane may, at first sight, appear to be adjacent. However, a natural or man-made barrier may separate them, preventing direct travel between the two points and making the travel distance between them much greater than the direct distance between them. Such situations are commonly present in urban settings. Figure 4.1 illustrates this situation:

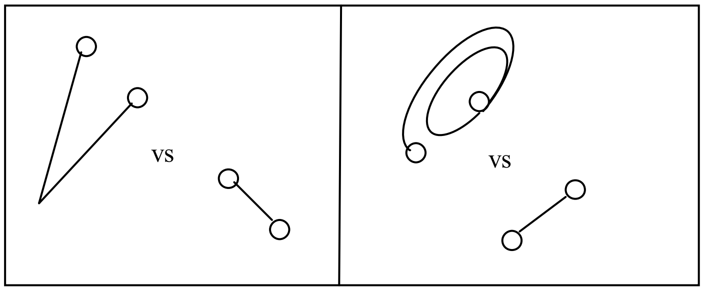


Fig. 4.1 Illustration of the impact of the edge set definition.

A point is considered *adjacent* to a district if there exists at least an edge connecting the point with one of the points already allocated to the district. A point can be allocated to a district only if it is adjacent to that district. Two districts are adjacent if there exists at least one edge connecting points from both districts. Those points are considered to be at the border of the districts and are potential points to be moved from their current district when searching for a better solution. The points are referred to as *border points*.

Knowledge of the adjacency helps to avoid unnecessary evaluations that may result in long computational times and also enhance compactness of the solution constructed.

Each time that a point is assigned to a district, adjacency among districts needs to be updated. A flow chart of the general procedure is shown in Figure 4.2 and details are explained in sections 4.2.1 and 4.2.2.

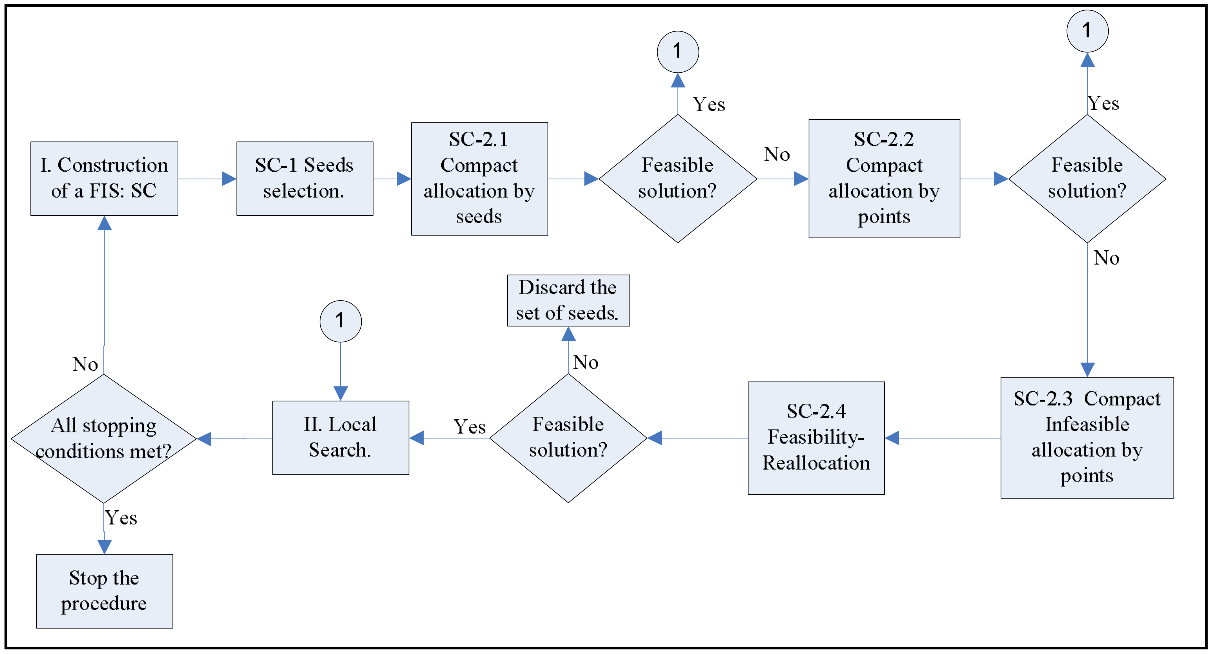

I. Construction of a FIS: SC
SC-1 Seeds selection.
SC-2.1 Compact allocation by seeds
Feasible solution?
Yes
1
No
SC-2.2 Compact allocation by points
Feasible solution?
Yes
1
No
SC-2.3 Compact Infeasible allocation by points
SC-2.4 Feasibility-Reallocation
Feasible solution?
No
Discard the set of seeds.
Yes
II. Local Search.
1
All stopping conditions met?
No
Yes
Stop the procedure

Fig. 4.2: HDH flow diagram

4.2.1 Phase I: Initial Feasible Solution Construction (IFSC)

As it was mentioned in section 3.2, even finding a feasible solution is a hard task, but given that we propose a multi start approach a determined number of initial solutions are attempted to be constructed based on a greedy randomized approach.

We propose two main steps to construct the initial feasible solution: Selection of a set of *m* seeds and allocation of points to the districts formed by a seed. During all the procedure, every time that a point is assigned to a district, the adjacency among points and districts should be updated. To enhance compactness, points are attempted to be assigned to the closest seed as long as adjacency conditions are fulfilled, for which we propose the following four different steps:

IFSC-1 Seeds Selection

Five different approaches are proposed for this step. Four methods are based on a greedy randomized construction in which, according to their corresponding greedy function, potential points (those that are not selected yet as a seed) are evaluated and the better choices are placed on a restricted candidate list (RCL) from which the seed is randomly selected. The fifth method is a semi random approach and for this reason is not a greedy randomized construction. It is called semi random because the selection of the seeds is not completely done at random. A pseudocode for each method is presented in Appendix II.

The first three approaches are based on distance, and the last two approaches are based on the dispersion and workload content of the points. The first method is referred to as the *"P-Dispersion Algorithm"* because points selected as seeds are the most dispersed relative to each other based on a modified version of the greedy algorithm of Erkut et al. (1991) that solves the P-dispersion problem. The greedy function computes the sum of distances between the seeds and a potential point. Those points with the larger sums are placed in the RCL from which a seed is randomly selected.

The second approach is referred to as the *"Neighborhood algorithm"* and it also consists of a greedy randomized construction in which the greedy function evaluates the points according to the number of points that are located within a "*neighborhood*" defined by a threshold distance. Those points that have more "*neighbors*" are placed in the RCL. Once a seed is selected, its neighbors are discarded as potential candidates. This procedure is repeated until all seeds have been selected. If no potential points remain and not all the seeds have been selected yet, then all the points that were discarded are now considered as potential points from which the remaining seeds are randomly selected. This method as it will be shown in the following section turned out to be most inefficient. However, it was included because it was observed during a preliminary experimentation that some of the solutions selected as the final districting for an instance corresponded to the set of seeds found by this method. Thus, it is not dominated by the other methods.

The third approach is similar but it is based on a semi random construction. Hence, it is named the *"Semi-random algorithm"*. It randomly selects a point as a seed, but those points located in its neighborhood are discarded as potential points as it is done in previous method. Again this procedure is repeated until the set of seeds has been defined and also if not all the seeds have been selected and all the points have been discarded, the required seeds are selected randomly among the discarded points. Given that this method selects the seeds randomly but discards the neighbors of the seed, it is not a complete random approach.

The fourth method is called the *Angle method* because its greedy function is based on the location of the points over the service region for which we compute the angle of the point with respect to the depot located in the origin. Once each angle is computed, the service region is partitioned into *m* radial sectors of equal size. The RCL is formed by the point whose location is closest to the corresponding division of each district and also by those points that are located either immediately before or after that division until all the candidates of the RCL have been selected from which the seed is randomly chosen.

The last method is referred to as the *Workload method* and its greedy function is similar to the previous method except that instead of dividing the region into sectors of equal size, the region is divided into sectors of approximately equal workload content. Workload content for this method is measured only by the total number of points that demand a service, which are equally divided into the *m* districts. The RCL is formed by those points whose location better approximates the corresponding division of each district and also by those points located either immediately before or after the division that corresponds to each district, until all the candidates of the RCL has been selected. For this method, as well as for the *Angle method*, an odd value of the RCL size is required in order to place in the RCL the point that better approximates the division of a district and also those located before and after that division, and so on until all the candidates have been selected.

IFSC-2 Compact allocation of points

Once the set of seeds has been selected, districts are formed around the seeds. Unassigned points are allocated to a district, for which we propose four procedures. Each of them attempts to create a feasible solution and the next step is applied only if no feasible solution has been constructed. This procedure not only attempts to create a feasible solution, but also enhances compactness by assigning points to their closest seed but only if adjacency requirements are fulfilled.

The first and second procedures also consider assigning a point to a district only if capacity limits of the districts are respected, reason for which we refer to them as feasible allocation of points. They differ in that each follows a different decision rule when assigning points, but both attempt to assign a point to its closest seed respecting the capacity limits of the districts. If no feasible solution was constructed during either of these two steps, a third procedure is applied in which infeasible allocations of points are allowed. However, due to the updating of adjacency during each assignment of points, the algorithm always attempts to make feasible assignments first and only if no feasible assignment is possible, then it is allowed to assign a point to a district even if capacity limits are not respected.

The last procedure is performed with the aim of finding feasibility for the case in which an infeasible solution was constructed, and it consists of reallocating points among adjacent districts. If no feasible solution can be constructed the procedure is stopped and the set of seeds is discarded. Details of each procedure are presented as it follows:

IFSC-2.1 Feasible allocation by seeds (FAS).

Each district is formed around its corresponding seed. Once seeds have been assigned to a district, we repeat the following procedure during a number of iterations or until a feasible solution is constructed. First step consists of selecting a seed at random. Then, a determined number of points are explored with the aim of assigning one point to the seed. The candidate points are those that are closest to the seed and are presented in a list sorted in ascending order in terms of the distance to the seed. A point is assigned to the district if capacity limits and adjacency requirements are respected. This procedure is repeated a determined number of iterations.

We suggest that the number of iterations may be around two to three times the number of points. During each iteration, either one point or none is assigned to the randomly selected seed. Also, as it is done during all the procedure, adjacency is updated each time that a point is assigned to a district. The procedure is stopped if a feasible solution is found or the stopping limit condition regarding a maximum number of iterations is reached. Given that during this step capacity limits are respected, a feasible solution is found when all the points have been assigned.

IFSC-2.2 Feasible allocation by points (FAP).

If a solution found by FAS is not feasible because some of the points were not able to be assigned to a district, during this step an attempt to assign these points to a district respecting capacity limits is performed. In a sequential order , a point is selected and districts are explored with the aim of finding a district adjacent to the point with available capacity. If the point was assigned to a district, adjacency and capacity of the districts are updated, otherwise the point is discarded and another point is selected. The process is repeated until all the points have been assigned or attempted to be assigned during a determined number of iterations. If a feasible solution is obtained, which means that all the points are currently assigned to a district, we stop the procedure.

IFSC-2.3 Infeasible allocation of unassigned points (IAUP).

If a solution found by FAP is still infeasible, then we assign the points even if capacity limits are not respected. However, given that each time a point is assigned to a district the adjacency is updated, the procedure attempts to assign the remaining points to the adjacent district with more capacity available with the expectation of performing a feasible allocation or at least violate the capacity limits of a district as little as possible (avoiding the creation of districts with an excessive number of points with respect to others). First, districts are sorted with respect to the remaining capacity in terms of pickups or deliveries. Then the unassigned points are selected in a sequential order, and the districts are analyzed starting with the district with more available capacity in terms of pickups or deliveries (according to the workload that the point represents), with the aim of finding an adjacent district with enough capacity to receive the point.

If no feasible allocation is possible, then the point is assigned to the adjacent district with more available capacity even if the assignment violates the capacity limits. The procedure is repeated until all the points have been assigned, resulting in an infeasible solution that will be processed by the next procedure in an attempt to achieve feasibility.

IFSC-2.4 Feasible-Reallocation of points (F-R)

This step is performed when an infeasible solution is constructed, meaning that all the points have been assigned but capacity limits are not respected for one or more districts. The procedure consists of moving points between adjacent districts with the aim of finding a feasible solution. It is possible that no feasible solution exists, due to the adjacency structure.

Figure 4.3 shows an example of an instance for which no feasible solution can be found. Suppose that each node represents a customer that requires either a pickup (*P*) or a delivery (*D*) and we want to define two districts for which capacity limits are: α= 1 unit and β= 2 units. It is possible to observe that no feasible solution exists since each district should contain one of the pickups. It is not possible to do so because one of the districts would be disjoint or not connected to the depot.

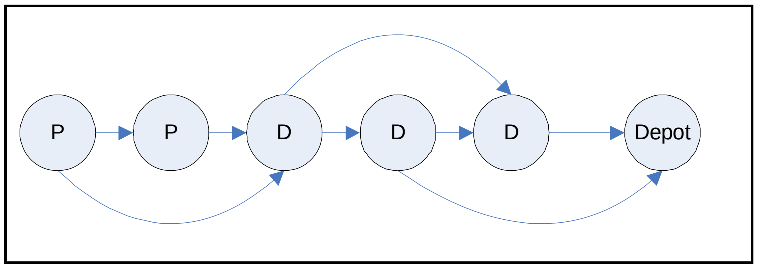


Fig. 4.3: Illustration of an instance with no feasible solution.

To move points between adjacent districts, different decision rules can be followed to determine which point should be moved and which district should receive the point. The best decision rule depends on the current districting configuration, and it motivates the implementation of a hyperheuristic as part of this step that will be referred to as Hyperheuristic-Feasibility (Hyp-F). Details of the hyperheuristic are presented after the description of the general procedure of the feasible reallocation step (F-R).

Given that each point requests a single service (whether a pickup or a delivery), feasibility in terms of each service can be addressed separately because capacity limits with respect to one of the services are not affected by a move of a point that requests the other service. However, we may notice that a move of a point that requires one of the services has an impact on the possible moves to be done with respect to the other service because of the adjacency relationship among points and districts. For this reason we propose an iterative procedure in which, during a number of iterations, it is attempted to attain feasibility with respect to one type of service before addressing the other type of service. Once feasibility is achieved for one of both operations, the rest of the iterations address the service that still remains infeasible until a feasible solution is achieved or stopping conditions are met.

For those solutions in which capacity is violated only in terms of one of the service activities, during each iteration only points that represent that service are analyzed for possible reallocations among adjacent districts.

During the procedure of reallocating points, as it will be explained later, some of the low level heuristics proposed for the H-F allow infeasible moves. This means that moves that result in worse values of the infeasibility metrics are allowed. Hence, the best solution found over all the iterations that are performed for a specific service is maintained with the aim that if at the end of the procedure no feasible solution was found, some iterations over the best solution are done in a final attempt to get a feasible solution, but in this case allowing only moves that improve the infeasibility metrics.

To evaluate the feasibility of a solution the following metrics are defined:

a) *Infeasibility*^{P}: Feasibility metric that accounts for the total number of pickups assigned to a district that exceeds the capacity limits,

b) *Infeasibility*^{D}: Feasibility metric that accounts for the total number of deliveries assigned to a district that exceeds the capacity limits.

A Feasible Solution (FS) is obtained when previous metrics have the value of zero units.

c) *W_dispersion*: This is a secondary metric that measures the dispersion among the workload content allocated to the districts by computing the difference between the maximum and minimum district workloads. This metric mainly helps to balance the workload of the districts but it also contributes to measure the feasibility improvement on a current solution, since it is expected to get feasibility as long as the workload content may be balanced among the districts. Since the only objective is feasibility, this metric is considered secondary and it allows moves that do not improve the infeasibility metrics, but decrease the dispersion on the workload content among the districts. A stopping limit is established for the number of consecutive iterations in which the moves performed only improve this metric.

In section 5.2 the stopping rules for the F-R procedure are described, considering that the number of iterations in case that capacity is violated for a single service should be greater than those for the case in which the procedure alternatively iterates over both types of services. A flow chart of the general procedure of the F-R is presented in figure 4.4.

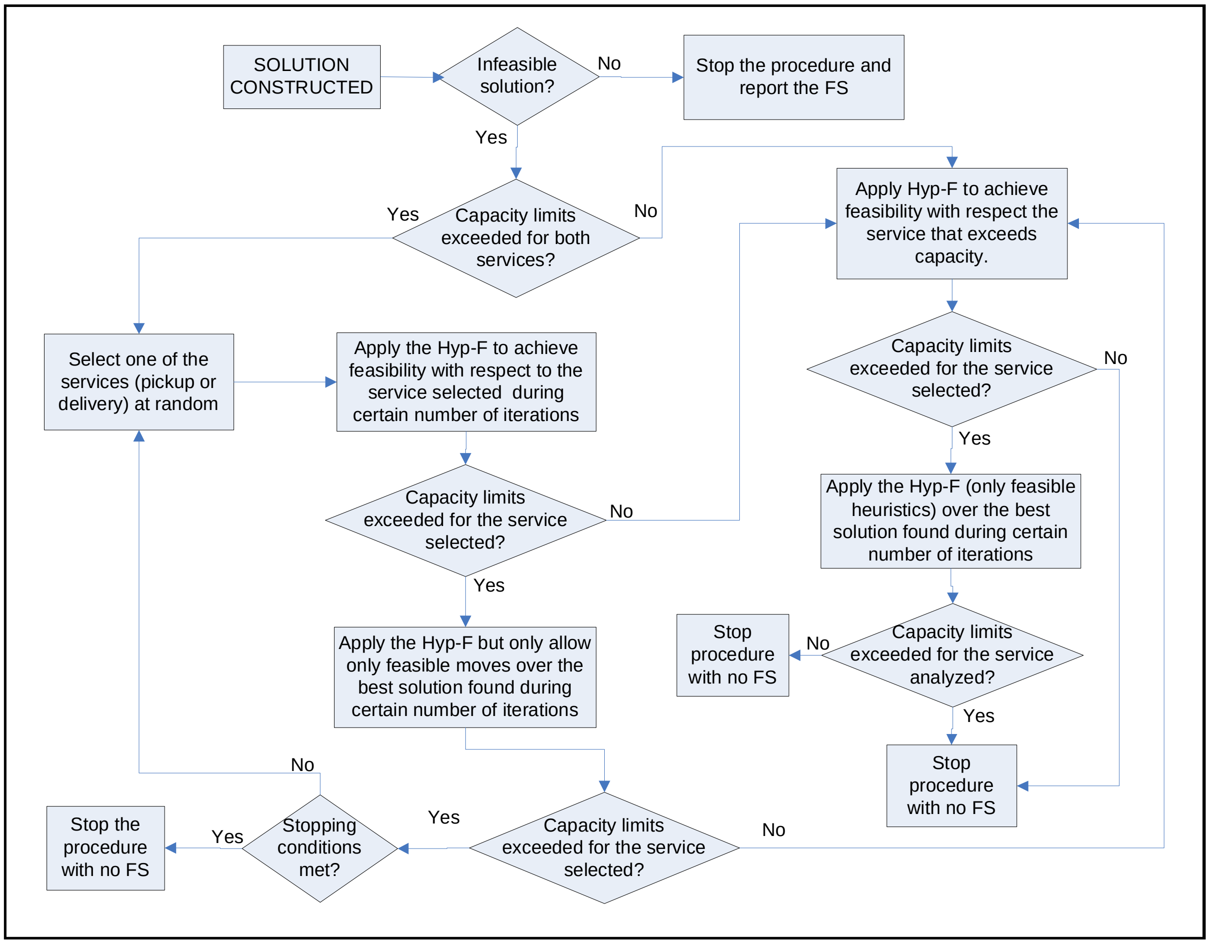

SOLUTION CONSTRUCTED
Infeasible solution?
No
Stop the procedure and report the FS
Yes
Capacity limits exceeded for both services?
Yes
No
Apply Hyp-F to achieve feasibility with respect the service that exceeds capacity.
Select one of the services (pickup or delivery) at random
Apply the Hyp-F to achieve feasibility with respect to the service selected during certain number of iterations
Capacity limits exceeded for the service selected?
No
Yes
Apply the Hyp-F (only feasible heuristics) over the best solution found during certain number of iterations
Capacity limits exceeded for the service selected?
No
Yes
Apply the Hyp-F but only allow only feasible moves over the best solution found during certain number of iterations
Stop procedure with no FS
No
Capacity limits exceeded for the service analyzed?
Yes
Stop procedure with no FS
No
Stop the procedure with no FS
Yes
Stopping conditions met?
Yes
Capacity limits exceeded for the service selected?
No

Fig. 4.4 Flow chart of the F-R procedure.

Hyperheuristic-Feasibility (Hyp-F) description

The H-F procedure is performed to reallocate points related to a single service at a time, according to the process described both in the previous paragraph and figure 4.4. Table 4.1 and Table 4.2 present the details of the design issues of the H-F, based on the guidelines used in (Soubeiga, 2003). The metrics proposed to evaluate each solution are then described along with the low level heuristics.

Table 4.1 Design details of the H-F procedure (a).

Level 0: Problem representation	
a) Complete or partial solutions?	The procedure initiates with a complete but infeasible solution obtained in previous step of the construction phase.
b) Acceptance criteria?	Three of the low level heuristics accept only solutions that improve the objective function (Only improvements criterion). The other three heuristics consider an AM (all moves) criterion since worse solutions are accepted.
c) Single or multiple objectives?	Single objective: feasibility. However a secondary metric is also defined and a determined number of moves in which only this secondary metric is improved are allowed.
d) Single solution or a population ?	A single solution is created at a time.
Level 1: Low level heuristics	
a) How many?	Six low level heuristics are proposed based on different decision rules that may be followed to determine the pair of districts from which a point is reallocated.
b) How are they applied?	They are randomly selected.
c) How long are they applied for?	They are either applied a single iteration or in a steepest descent fashion, which is randomly determined.
d) Are they metaheuristics?	All of them are based on a TS approach.

Table 4.2 Design details of the H-F procedure (b).

Level 2: High level Heuristic	
a) Learning mechanism, heuristic ranking?	A simple hyperheuristic is proposed with no learning mechanism based on a combination of a "simple random" with a "random descent" hyperheuristic. For each iteration, the hyperheuristic randomly selects one of them.
b) Definition of a decision point?	Decision point at which a low level heuristic is chosen is defined in terms of the number of iterations that the heuristic selected is applied, which it is randomly determined between two options: the low level heuristic is applied during a single iteration in which a single point may be reallocated, or in a steepest descent fashion. Also as stopping rule, when a feasible solution is found the procedure is stopped (in terms of the type of service activity that is currently analyzed).
c) Actual selection of heuristics?	Selection of the heuristics is randomly performed.

Low level heuristics description:

The heuristics attempt to find a pair of adjacent districts such that a point can be reassigned from one (the *sending* district) to the other (the *receiving* district). Each of the heuristics has a TS function that aids in escaping local optima. The first three heuristics allow only feasible moves. This means that a district can receive a point only if has enough capacity. For all the heuristics, the districts are first sorted in descending order by workload content according to the type of service that is currently selected. After this initial step the heuristics differ.

Heuristic 1 designates the district at the top of the list as the sending district. Then starting from the bottom of the list, it seeks a receiving district that is adjacent to the sending district until a feasible and non tabu move can be performed. If no feasible move is found, the sending district is discarded and the same procedure is repeated (new sending district from top of list) until a feasible move is found or all possibilities are exhausted.

Heuristic 2 is similar and designates the sending district in the same way but the list is searched from top to bottom to find a receiving district. Heuristic 3 designates the district at the bottom of the list as the receiving district and seeks a sending district by searching the list from top to bottom. Heuristic 4 is similar to heuristic 1 in that the sending district is designated in the same way and the list is searched in the same direction. However, a move is always made on the first search of the list. If a feasible move is found it is executed. If no feasible move is found, the entire list is examined and the non tabu move that least increases the infeasibility metric is performed. Heuristics 5 and 6 are similar to Heuristics 2 and 3, respectively, but also may allow infeasible moves.

The following notation is defined to identify each heuristic:

F= if a district allows only feasible moves,
I= for those districts that accept both feasible and infeasible moves (under the conditions previously described),
MM=for a heuristic that selects the first and second district among those with more workload content,
ML= for a heuristic that selects the first district among those with more workload content but the second district among those with less workload content,
LM= for a heuristic that selects the first district among those with less workload content but the second district among those with more workload content.

For example, FMM denotes for a heuristic that allows only feasible moves and selects the first district among those with more workload content and the second district also among those with more workload content (independently of whether the heuristic selects the *receiving* or *sending* district first).

The six heuristics are implemented in the black box of the hyperheuristic at the beginning of the procedure and none of them is modified during the iterations. As it was mentioned previously, each heuristic is applied either in single applications or in a steepest descent fashion. In addition, the following stopping rules are defined:

a) Three consecutive iterations in which both feasibilities metrics do not improve.

b) Five consecutive iterations in which only *W_dispersion* metric improves.

Once a point has been reallocated, the improvement of the infeasibility metrics with respect to previous iteration is computed as described in equation (75):

$$Impr(Metric) = \frac{Metric^{iter-1} - Metric^{iter}}{Metric^{iter-1}}, \qquad (75)$$

where *Metric* refers to any of the infeasibility metrics previously defined.

TS features of the low level heuristics:

The search space consists of all the possible moves between a pair of adjacent districts, which differs among the low level heuristics depending on the decision rule used to select the pair of adjacent districts and on the criterion used to include infeasible moves as part of the search space. Moves that lead to assign a point to a district from which has recently moved to another district are classified as "*tabu active"* during a number of iterations (*permanence*) in order to avoid cycling back to previously visited solutions. Aspiration criteria allows a *tabu active* move if the resulting solution is better than the currently known best solution. The following variables are defined:

Tabu(i,j)=iteration in which point *i* leaves district $j, i \in V, j \in J$,

Move(i,j)=solution in which point *i* is assigned to district $j, i \in V, j \in J$,

TA=set of *tabu active* moves,

Tabu tenure= Tabu permanence which is the number of iterations that a move is not allowed or *tabu active*,

iter= Current iteration counter,

Titer= number of iterations in which the tabu tenure is incremented twice its current value.

And the following condition establishes if a move is considered *tabu active:*

$$Move(i,j) \in TA \text{ if } Tabu(i,j) + Tabu\,tenure \geq iter \tag{76}$$

See Appendix III for a pseudocode of the algorithm of the hyperheuristic and the six low level heuristics.

4.2.2 Phase II: Local Search (LS)

Once an initial feasible solution has been constructed, it is now improved by a local search (LS) technique. The metaheuristic employed is a Tabu Search and we propose three neighborhood structures:

a) 1-Step LS (1-S)
b) 2-Steps LS (2-S)
c) K-Steps/Pairs LS (K-S/P)

These neighborhood structures can be searched in a combined fashion. For this we propose two methods, resulting in total five neighborhood structures:

d) Hyperheuristic LS (HypLS)
e) 2-Iterations LS (2-IterLS)

The procedure consists of an evaluation over a search space, determined by each of the neighborhood structures previously mentioned, that consists of the solutions found once a move or interchange of points among adjacent districts is performed with the aim of finding a better districting configuration. We programmed five procedures that differ from each other in the neighborhood structure over which the solutions are explored with the aim of finding a better solution.

An iteration of the LS phase consists of the evaluation of all the solutions over a determined search space by equation (1). Solutions evaluated are those that result by the move or interchange of points among adjacent districts according to the neighborhood structure under consideration. In case of ties, we evaluate the solutions by equation (77) for the 1-S, 2-S, HypLs and 2IterLS and by equation (78) for the K-S/P. Equation (77) measures the dispersion of the workload content of each district with respect to an average workload assigned to the districts and equation (78) measures the difference on the workload content between the pair of districts analyzed. The solution that presents the less dispersion or difference on the workload content is preferred.

$$WDispersion = \sum_{j \in J} \left| W_j - \overline{W} \right| \tag{77}$$

$$PWDispersion = abs(W_{d1} - W_{d2}), \tag{78}$$

where

$$W_j = Std \sum_{i \in V} wd_i X_{ij} + Stp \sum_{i \in V} wp_i X_{ij} + D_j / Sp, \tag{79}$$

$$\overline{W} = \frac{\sum_{j \in J} W_j}{|J|} \tag{80}$$

and *d1*, *d2* are the pair of adjacent district selected during the iteration of the K-S/P.

Also, it is important to mention before describing each of the neighborhood structures proposed, that during each iteration it is allowed to select solutions that resulted to be worse than the current solution found, with the aim of enhancing diversity over the search space. However, a list is maintained with the tree overall best solutions found for all the neighborhood structures except the K-S/P for which only the overall best solution is maintained. At end of the procedure, an attempt to improve those best solutions is made.

The procedure reports as the final districting configuration the overall best solution that was found for a given set of seeds from which an initial solution was constructed. Finally, the best overall solution is reported as the final solution. We report both the value of the objective function defined by (1) that is the weighted sum of the maximum workload content and compactness metrics. We also report the dispersion of the workload content among the districts with respect to an average workload as defined by equation (77).

4.2.2.1 Description of Neighborhood Structures

This section presents the details of the neighborhood structures proposed. Each of them implements a TS short term memory with an aspiration criterion that allows a *tabu active* move only if the resulting solution is better than the current best solution. The search space consists of the solutions that resulted after the move or interchange of points among adjacent districts. The best solution found is reported after a number of iterations. As it was previously mentioned, in the case of ties, the solution that with the less dispersion on the workload content of the districts is selected.

The first and second neighborhood structures consists of the solutions that result after the evaluation of *"moves"* of points between adjacent districts, which means that a single point is translated from one district to an adjacent district. The third neighborhood structure also evaluates solutions that result of *"exchanges"* of points, but restricts the search space over a pair of adjacent districts while the first and second search over all the adjacent districts. The fourth and fifth neighborhood structures are a combination of the first and third neighborhood structures. For an interchange of points, since adjacency is a requirement for any move it is necessary to have at least two pairs of connected points between the pair of districts under consideration.

1-Step LS (1-S)

First neighborhood structure is a greedy approach that consists of a quick evaluation of all the feasible moves between adjacent districts. The best solution is selected and the corresponding move of a point is performed during each of the iterations. Given that the best move may result in a worse solution than current solution, during the procedure a list of the three best solutions is maintained. At the end of the procedure a final attempt is made to improve these three best solutions in hopes of finding a better solution with a small amount of additional effort. The overall best solution found is reported as the final solution for the given initial feasible solution.

2-Steps LS (2-S)

This neighborhood structure consists of an evaluation of all the possible moves between adjacent districts including also infeasible moves. This part of the procedure is referred to as the *first step*. For each solution found during the *first step*, now the search space consists of all the feasible solutions that result from moving a point between adjacent districts. This is referred to as the *"second step*". Among all the solutions evaluated in the "*second step"*, the best solution is selected but only the move done during the *"first step"* is performed. If it turns out that it is an infeasible solution, then the move selected during the *"second step"* is also performed so that we always end up with a feasible solution. Also, in case of ties, the solution that results in the lower dispersion metric according to equation (79) is selected and as it is done in the 1-S procedure, a list of the three best solutions is maintained and at the end of the procedure a final attempt is made to improve these three solutions, but the search is done by evaluations of a single step. The overall best solution found is reported as the final solution for the given initial feasible solution.

The second step is performed with the aim to avoid a completely greedy selection of a solution as it is done in 1-S, by searching beyond the solutions that result after the move of a point and also exploring infeasible moves during the first step but recovering feasibility during the second step. However, as it is shown during the numerical experimentation stage of this research, it resulted in long computation times for large size instances, reason for which the algorithm based on this neighborhood structure was only tested with small instance sizes.

k-Steps/Pair LS (k-S/P)

This neighborhood structure is an extension of the algorithm proposed by Kerninghan and Lin (1970) for graph partitioning. It is an improvement routine that tries to find a better partition given a current, feasible partition among pairs of adjacent districts. This procedure evaluates both moves and exchanges of points among a pair of adjacent districts. This procedure is not completely greedy as the 1-S algorithm because it explores the possible moves and also exchanges of points in a determined number of consecutive steps. However, it restricts the search to a single pair of adjacent districts, while the 1-S and 2-S are more global by searching all pairs of adjacent districts.

In this procedure, during a *"first step",* all the solutions that resulted after a move or interchange of points between a selected pair of adjacent districts are evaluated and the best of them is selected. Also in case of ties, the solution that results in the lower dispersion metric according to equation (78) is selected. Based on the solution that resulted from the first step, a second step searches over all the solutions that result of moving a point or interchanging a pair of points among the selected districts, and again, the best solution is selected. This procedure is repeated during *k* steps. At the end, the best overall solution among the *k* steps is selected. The suggested value of *k* is $\lfloor n/2 \rfloor$, where *n* is the maximum number of points allocated to one of the two districts under consideration.

To select the pair of districts, the first district is selected randomly and among its adjacent districts, the second one is randomly selected. To enhance diversity, during an iteration the probability of the selected districts is decreased by a determined percentage, and the probability of the rest of the districts is equally increased so that the probabilities of the districts sum to one. During all the iterations, the overall best solution is maintained. This is because during an iteration is possible to select worse solutions than the current solution to enhance diversity. However, at the end of the procedure the overall best solution is reported as the final districting configuration for the given feasible initial solution.

Hyperheuristic-LS (HypLS)

This neighborhood structure is a combination of the 1-S and kS-P by a hyperheuristic. The hyperheuristic proposed is very similar to that for the F-R step. Also the details of the procedure are described based on the guidelines used in (Soubeiga, 2003) and summarized in Table 4.3. The hyperheuristic randomly selects one of the two neighborhood structures and explores the solutions according to the characteristics defined by the 1-S or kS-P search space. The best solution is selected and next iteration the hyperheuristic randomly selects the neighborhood structure over which solutions will be evaluated. This procedure is repeated a determined number of iterations, maintaining also a list with the three best solutions to perform a final attempt to improve those solutions at the end, based on 1-S. The best overall solution found is reported as the final solution for the given initial feasible solution.

Table 4.3: Design details of the H-LS (a).

Level 0: Problem representation	
a) Complete or partial solutions?	The procedure initiates with a complete and feasible solution obtained from phase I.
b) Acceptance criteria?	All solutions that resulted of a move or interchange of points are accepted as long as they lead to a feasible solution even though may be worse than current solution.
c) Single or multiple objectives?	Single objective defined in equation (1) by a weighted combination of the two objectives related to workload balance and compactness.
d) Single solution or a population of solutions?	A single solution is created as a result of an iteration.

Table 4.4: Design details of the H-LS (b).

Level 1: Low level heuristics	
a) How many?	Two: 1-S and k-S/P
b) How are they applied?	They are randomly selected.
c) How long are they applied for?	They are applied during a single iteration from which the best solution is reported.
d) Are they metaheuristics?	Yes all of them are based on a TS approach.
Level 2: High level Heuristic	
a) Learning mechanism, heuristic ranking?	A simple hyperheuristic is proposed with no learning mechanism based on a "simple random" approach.
b) Definition of a decision point?	Decision point at which a low level heuristic is chosen is defined in terms of the number of iterations that the heuristic selected is applied. Stopping rules for the hyperheuristic are based on a maximum amount of iterations.
c) Actual selection of heuristics?	Selection of the heuristics is randomly performed.

2-Iterations LS (2-IterLS)

The search space defined here is also a combination of the neighborhood structures defined in 1-S and kS-P, that results in the union of both search spaces. Given a current solution, the procedure consists of evaluating all the possible solutions according to each search space defined by 1-S and kS-P. The best solution found is selected. This procedure is repeated a certain number of iterations and also the three best solutions are maintained so that a final attempt to improve them is done based on 1-S. The best overall solution found is reported as the final solution of the initial feasible solution generated.

See Appendix IV for a pseudo code of the algorithms based on each neighborhood structure.

4.3 Complexity analysis of the heuristics

In this section an analysis of the computational complexity of the five heuristics based on the five neighborhood structures previously described is presented, for which we analyze the computations required to perform the two main phases of the heuristics: IFSC and LS. All the heuristics have in common the IFSC procedure and differ in the LS phase according to the neighborhood structure over which solutions are evaluated with the aim of finding a better solution. We first present the analysis of the IFSC procedure and then the analysis of each LS neighborhood structure. At the end we present the computations required for each of the five algorithms.

4.3.1 IFSC Complexity Analysis

As it will be shown during the analysis, the F-R turned out to be the most significant procedure for the number of operations required to construct the feasible initial solution. However, this procedure is not performed for all the solutions that are constructed and improved, only for the cases in which an infeasible solution was previously constructed and it is required to reallocate points among adjacent districts with the aim of obtaining a feasible solution. For this reason, we present an analysis of the complexity for both situations, when a feasible solution is obtained without requiring the F-R procedure and when an infeasible solution was constructed and the F-R is applied with the aim of obtaining feasibility.

IFSC-1 Seeds Selection

Given that five different methods to select the set of seeds we present the complexity of each method, reporting the worst case scenario for the total complexity of the construction of an initial feasible solution. For the procedures that required to sort either angles or distances, we used the "Quicksort routine" for which we consider the average case performance of $O(n \log n)$ comparisons.

The "P-Dispersion" requires $O(JV \log V)$ time to select the set of seeds, the "Semi-Random" requires $O(JV)$, the "Neighborhood" $O(JV^2)$, and the "Angle" and "Workload" procedures each of them require $O(V \log V + JV)$ time. We can observe that the "Neighborhood" procedure turned out to required the maximum number of computations among the five methods to select a set of seeds. We will consider this number of computations as the worst case for the selection of a set of seeds.

IFSC-2 Compact allocation of points

This procedure consists of four sub-procedures in which the first three procedures attempt to assign a point to a district, and the last procedure is applied in order to obtain feasibility by the reallocation of points (F-R). The computations required for the FAS are $O(V^4 \log V)$, the FAP requires $O(JV^3)$, the IAUP requires $O(V^3 J + JV^2 \log J)$ and the F-R $O(V^5 J^2)$. For the cases in which the F-R procedure is required, the total number of computations to construct an initial feasible solution are the sum of the computations required to select a set of seeds, for the FAS, FAP, IAUP and F-R procedures, resulting in $O(V^5 J^2)$. When the F-R procedure is not required, the time required to construct an initial feasible solution is $O(V^4 \log V + JV^3)$.

4.3.2 LS Complexity Analysis

In this section we present the analysis of complexity for each of the local search procedures of the five heuristics proposed, considering both cases previously described for the construction of an initial feasible solution.

1-Step LS algorithm

The time required to perform the local search of this algorithm is $O(V^3 J)$. When the F-R procedure is required in the construction of an initial feasible solution, the total time required for the 1-S algorithm is $O(V^5 J^2)$. When the F-R procedure is not required, the 1-S algorithm requires $O(V^4 \log V + JV^3)$.

2-Step LS algorithm

This algorithm requires for the local search $O(V^4 J^2)$ computations, resulting the LS algorithm that requires more time over the five procedures proposed. However, given that the time required to construct an initial solution is even bigger when the F-R procedure is applied, the total time required for the 2-S algorithm turns out to be the same as for the 2-S algorithm, which is $O(V^5 J^2)$. When the F-R procedure is not required, the 2-S algorithm requires $O(V^4 \log V + V^4 J^2)$.

K-Steps/Pairs LS algorithm

The time required to perform the local search phase for this algorithm is $O(V^4 + V^2 J)$. When the F-R procedure is required in the construction of an initial feasible solution, the total time required for the K-S/P algorithm is $O(V^5 J^2)$ When the F-R procedure is not required, the algorithm requires $O(V^4 \log V + JV^3)$. As we can observe, for both cases, the complexity of this algorithm is the same as for the 1-S algorithm.

HypLS and 2-Iter LS algorithm

The computations required for each of the combined LS algorithms are the same, requiring $O(V^4 + V^3 J)$ time to perform the local search and $O(V^5 J^2)$ total time for the entire procedure in the cases in which the F-R is required and $O(V^4 \log V + JV^3)$ otherwise. As we can observe, for the entire procedure the computations required for both cases in which it is required or not the F-R is the same as in the 1-S and K-S/P algorithms.

Chapter 5

Experimentation and results

To test the performance of the proposed solution procedure, a set of instances was generated. All problem instances were solved on a 2.00 GHz Pentium processor with 2 GB of RAM running under Windows XP. Section 5.1 describes the instance generation procedure. Section 5.2 presents the stopping rules conditions. Section 5.3 presents the numerical results. Section 5.4 describes the analysis to the heuristic's performance and Section 5.5 presents an analysis of the methods to select the set of seeds.

5.1 Instance generation.

Five types of instances were defined: *Symmetric*, *Semi-Symmetric*, *Asymmetric*, *Urban* and *Parcel-based* instances. The first four types are randomly generated and last type is made using data provided by the company under consideration, which was obtained by sending some GPS in the routes to store the location of stops made during some days. Further details will be provided in this chapter.

A symmetric instance is that in which the optimal solution consists of "symmetric" districts, that is, districts with the same workload content and the same shape. These instances are the most unrealistic type, but are included in the numerical experimentation in order to measure the performance of the algorithm for larger instances in which CPLEX is not able to find neither the optimal nor at least an integer solution. For this type of instances, even though the graph is not complete, Euclidean distances are computed even if the points are not connected to guarantee symmetry.

We also include a variant of this type of instances that we referred as *Semi-Symmetric*. These instances differ to the former type in that the distances are computed by finding a shortest path among the points that are not connected by an edge instead of computing Euclidean distances for all the points as it is done in the symmetric instances. Semi-symmetric instances are included in the numerical experimentation since they are a type of instance that CPLEX can solve more easily (for smaller instance sizes), allowing more optimal solutions to be found for comparison purposes.

Asymmetric instances are more general instances and consists of points uniformly distributed over a plane. Urban instances consist of points distributed over a region that resembles the structure of the metropolitan area of Monterrey, N.L. Mexico. As it was previously mentioned, this last type of instance is made by data provided by the parcel company. It consists of the locations of the stops made by the vehicles during a representative day, obtained by a GPS unit that was carried in the delivery vehicles. The edge set is formed by real distances between adjacent points, considering as a rule that there must be an edge connecting a pair of points only if there are no intermediate points between them. Details of the generation of each of the instance types will be described in the following subsections.

5.1.1 Symmetric, Semi-Symmetric and Asymmetric instances generation

Five different instance sizes were defined, which are classified by the number of points and districts:

- Small instances: 50 points/5 districts and 200 points/10 districts;
- Medium instances: 450 points/15districts;
- Large instances: 1000 points/20 districts and 1500 points/30 districts.

To generate all the types of instances, the location of points is first determined over a plane using polar coordinates (which are then converted to Cartesian coordinates) with the depot located at the origin. Then the workload of each point is randomly assigned (either a pickup or delivery) and the edge set is defined according to each type of instance. The last step consists of determining the shortest path among all pairs of points including the depot.

For the case of a symmetric instance, we first generate one of the districts and then we rotate this district to form the rest of the districts symmetrically, such that districts consists of subgraphs of equal size. There must exist an empty and small region of equal size separating the districts with respect to each other. Workload content is also equal for all the districts, such that the farthest point to the depot has the same distance for all, and each point of a district represents the same workload for the corresponding symmetric points in the rest of the districts.

The details of the procedure to generate the instance are as follows: for each district, the radius of the farthest point is estimated by defining a range from which the value is randomly selected. The range is defined as a function of the instance size as it will be described later in this section. Once the districts are defined as well as the spanning tree that connects the edges within a district, m edges are included in E to connect the m closest points to the depot. Also, in order to connect the districts between each other, the point with the maximum angle with respect to the depot of a district is selected and the edge that connects that point to the closest point of an adjacent district is included in E. The symmetric edge for the rest of the districts is also included. Finally a random edge that connects points within a district as well as the corresponding symmetric edges for the rest of the districts are also included in E. Euclidean distances are computed for all the points independently of the edge set definitions. For the case of the Semi-Symmetric instances, Euclidean distances are computed only for the points that are connected by an edge, and for the rest as it was previously mentioned, all shortest paths are found using the Floyd-Warshall algorithm, (Floyd, 1962).

For an asymmetric instance, all points are uniformly generated over the plane and a spanning tree is computed to include the corresponding edges in E, as well as the edges to connect the m closest points to the depot. Additional edges are randomly selected. The distance matrix is found following the same procedure than for the Semi-Symmetric instances.

Even though these instances do not resemble the urban structure of a city, the size of the region is defined according to the size of the metropolitan region of Monterrey city, Mexico. This region has 3159.4 km^2 and includes eight adjacent cities:

1. Apodaca : 183.5 km^2
2. Escobedo: 191 km^2
3. Guadalupe: 151.3 km^2
4. Juárez: 277.8 km^2
5. Monterrey: 451.30 km^2
6. San Nicolás de los Garza: 86.8 km^2
7. San Pedro Garza García : 69.40 km^2
8. Santa Catarina : 984.50 km^2

Figure 5.1 shows a map of the region. Garcia city is not included as part of the metropolitan region by the parcel company because it is assigned to a different service region. The figure and data were obtained from the Enciclopedia de los Municipios en Mexico, 2005.

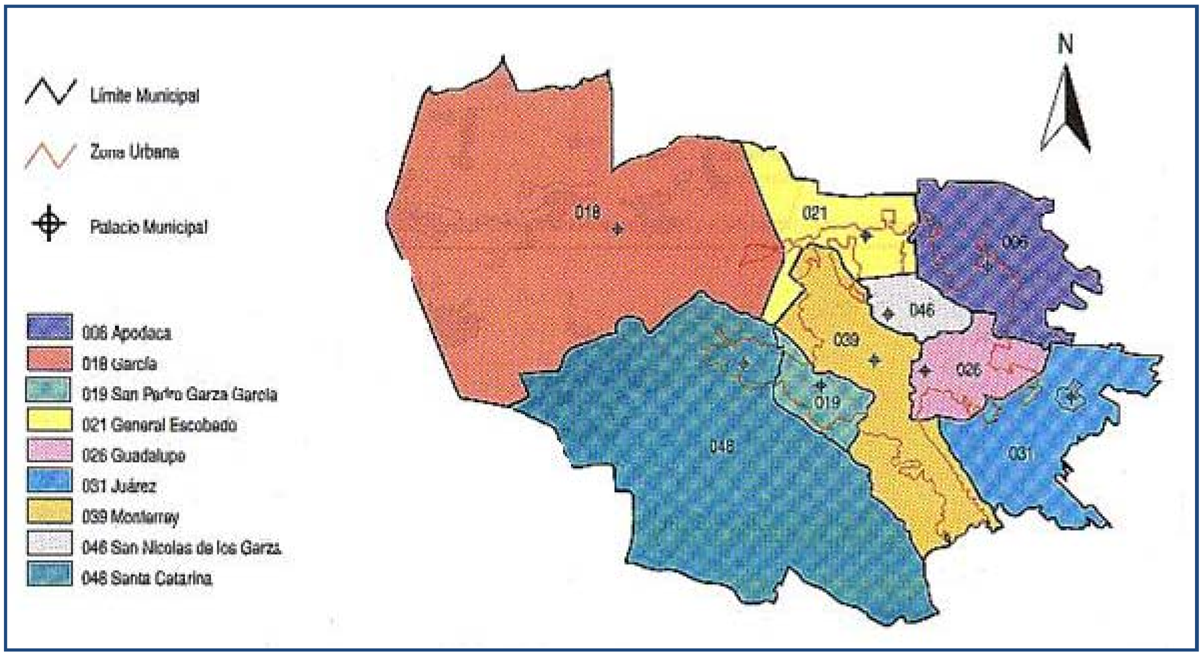


Fig. 5.1 Metropolitan region of Monterrey city.

We consider for the large instance size that the corresponding region of the instance should be approximately equal in size to the metropolitan region of Monterrey city, assuming a circular shape, and a scaled sub region for the rest of the instances according to their size. For this, the value of the farthest point to the depot is generated among a range that contains the desired radius value for each instance size, with a tolerance of +/-5%. Table 5.1 presents the corresponding area of the region, the radius, and the range for each instance size.

Table 5.1: Corresponding radius for each instance size.

Instance (points)	Area	Radius	Range		
1500	3159.4	31.71224	30.12662	-	33.29785
1000	2106.266667	25.89293	24.59829	-	27.18758
450	947.82	17.36951	16.50103	-	18.23798
200	421.2533333	11.57967	11.00069	-	12.15865
50	105.3133333	5.789836	5.50034	-	6.07933

Regarding the stopping times for the pickup and delivery activities, we fixed them at a realistic value for all the instances generated, considering that the service activities are done in an urban region and that a pickup usually requires more time than a delivery: 5 minutes (1/12 hour) for the average stopping time that a delivery requires, 10 minutes (1/6 hour) for a pickup. We define three levels of average speed, assuming that all the vehicles assigned to the districts travel on average at the same speed over the entire service region: 25 kilometers/hour, 30 kilometers/hour and 35 kilometers/hour. These values were validated by the parcel company operating in Monterrey, Mexico.

Each point was assigned a pickup or a delivery at random with equal probability. The limits on the number of pickups and deliveries were set using equations (82) and (83), respectively.

$$\alpha = \left\lceil \sum_{i \in V} wp_i / J \right\rceil + tolerance \qquad (82)$$

$$\beta = \left\lceil \sum_{i \in V} wd_i / J \right\rceil + tolerance \qquad (83)$$

Two levels of capacity are defined: tight (T) and less restricted (LR). The tolerance term is determined according to the instance size and the tightness of capacity. Table 5.2 shows the expressions used to compute the tolerances:

Table 5.2 Tolerance definition.

Tight (T)	Less restricted (LR)
Floor($V \cdot 0.15 / J$)	Max(2, Floor($V \cdot 0.22 / J$))

The edge set is formed such that the graph is connected and incomplete. For this, a spanning tree is first found. Then the edges that connect the depot to its *m* closest points are added to the graph. Finally, points are connected to their nearest neighbors at random until the average number of edges per point is in a range of two to four edges. This is the realistic range for a real urban road network. The relative weighting factor was varied over three values: λ=0.25, 0.5, and 0.75.

We generated three replicates for each of the five instance sizes and for each of the three types of instances. Each instance was solved varying the three values of the relative weighting factor, the two levels of capacity limits and the three values of speed resulting in a total of 3 x 5 x 3 x 3 x 2 x 3 = 810 instances.

5.1.2 Urban instances generation

As we have previously mentioned, these instances are generated such that they may resemble the structure of the metropolitan region of Monterrey, excluding also Garcia city since this city is not served by the depot located in Monterrey by the parcel company that motivated the study of this problem. Points were generated such that their location correspond to places over the region in which there actually exists a home, office or a building that may be visited by the service vehicles. This means that points cannot be located over the mountains that are found in the metropolitan region such as the "*Cerro de la Silla*", "*Cerro de Topochico*", "*Cerro de las Mitras*" and "*Cerro de la Huasteca*".

To define the location of the points for each of the instances of this type, we generated two sets of what we call *"base points"*, from which points are randomly selected to form each of the instances. Both sets of *base points* respect the conditions mentioned in previous paragraph. The first set of *base points* consists of 2211 points that are located over the entire metropolitan region under consideration and includes the data of the customers provided by the parcel company that correspond to 1176 points as well as additional points in order to cover the regions in which there were no demand points for the routes of the parcel company.

The depot is located in the same place as it is for the parcel company, which is approximately at the center of the geographical region. Among this set of *base points,* for the larger size instances of 1000 and 1500 points, the points are randomly selected to resemble a geographical region that covers the entire metropolitan area. The workload content of each point is randomly determined following the same procedure as for the asymmetric instances. Figure 5.2 and 5.3 show two different views of the base points over the metropolitan region of Monterrey by the use of Google Earth version 4.3.7284.3916 and Figure 5.4 shows them over a plane as rendered by GPS Trackmaker Version 13.3.

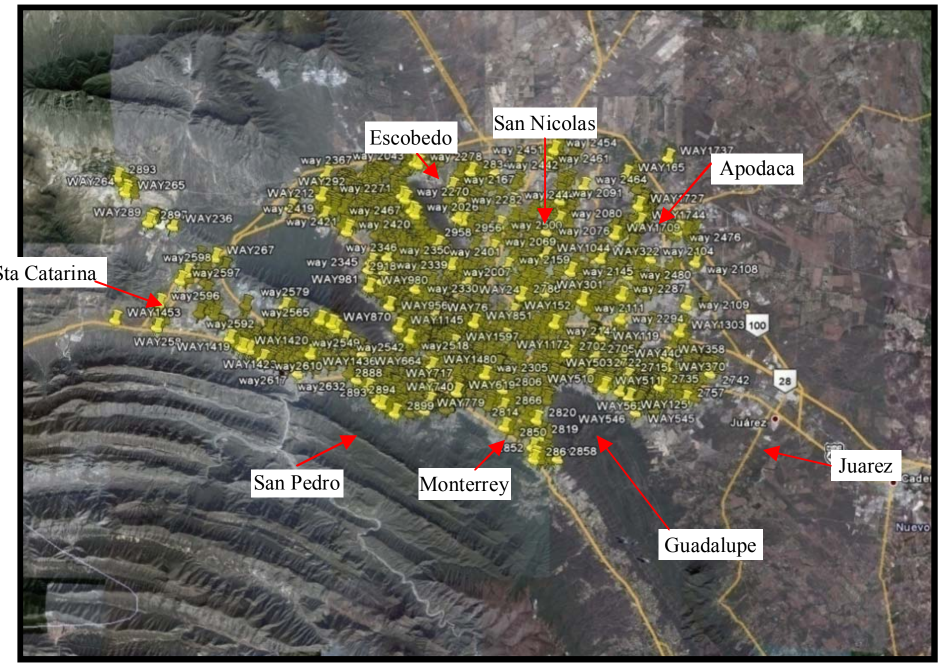


Fig. 5.2 View (a) of the location of the *base points* for the large size instances.

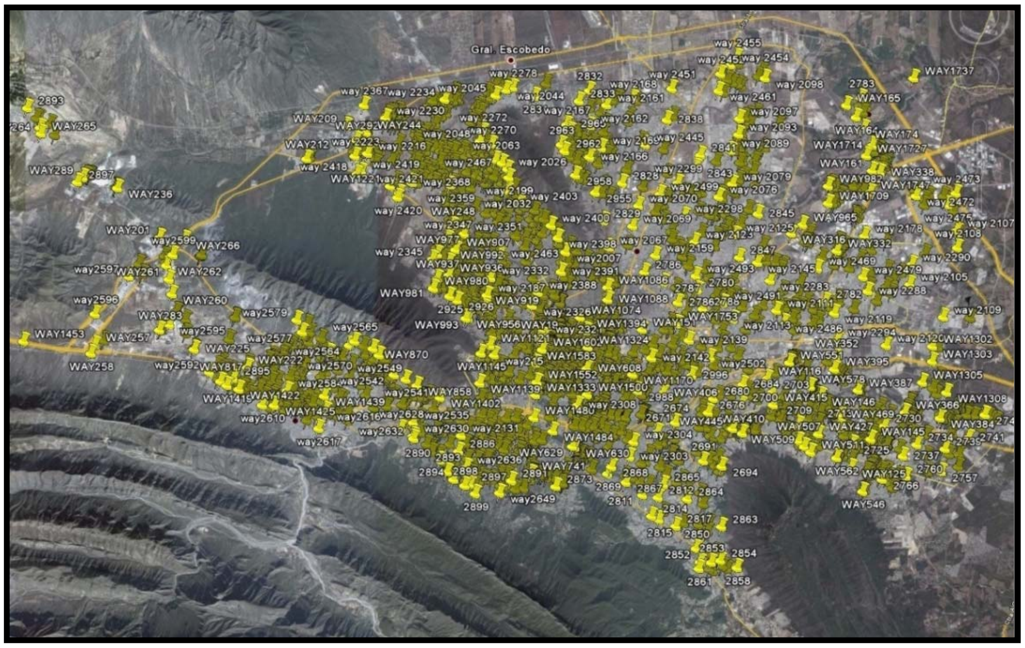


Fig. 5.3 View (b) of the location of the *base points* for the large size instances.

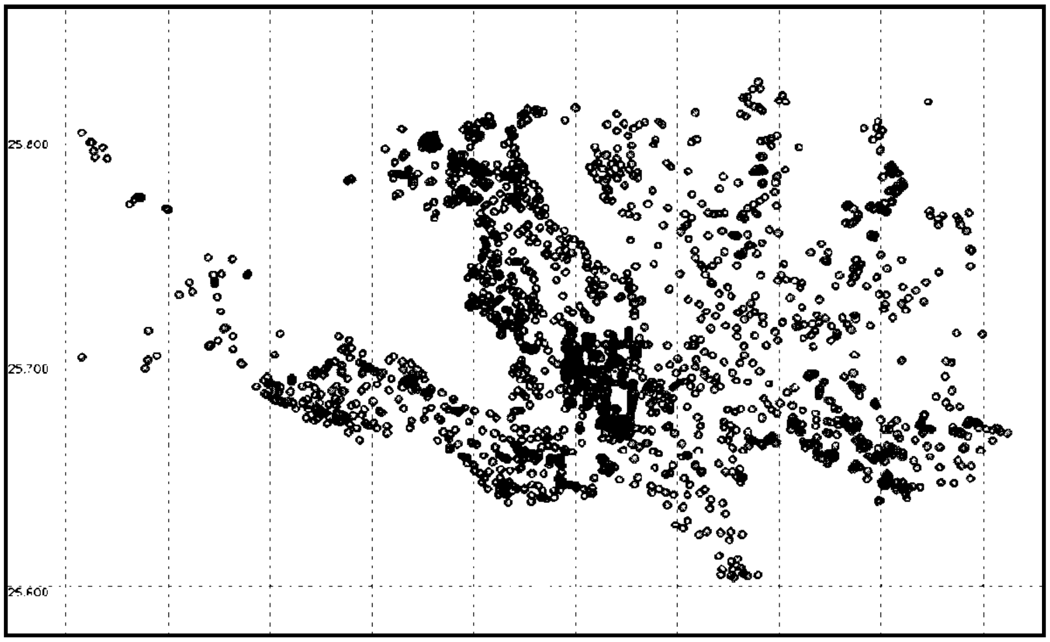


Fig. 5.4 Location of the *base points* for the large size instances.

The second set of *base points* covers a smaller region that includes only Monterrey, and part of the cities of San Pedro, San Nicolas and Guadalupe. This set consists of 1185 *base points* and it is created to generate the small and medium size instances such that the region may be of smaller size than that of the large size instances. For each of the five instance sizes, three replicates were generated, and each of them were tested varying over the two capacity limits, the three values of the relative weighting factor as well as the three values of speed, resulting in total 5x3x2x3x3=270 instances.

Figure 5.5 and 5.6 show two views of the set of points located on a map by Google Earth and figure 5.7 shows the view of the points over a plane by GPS Trackmaker.

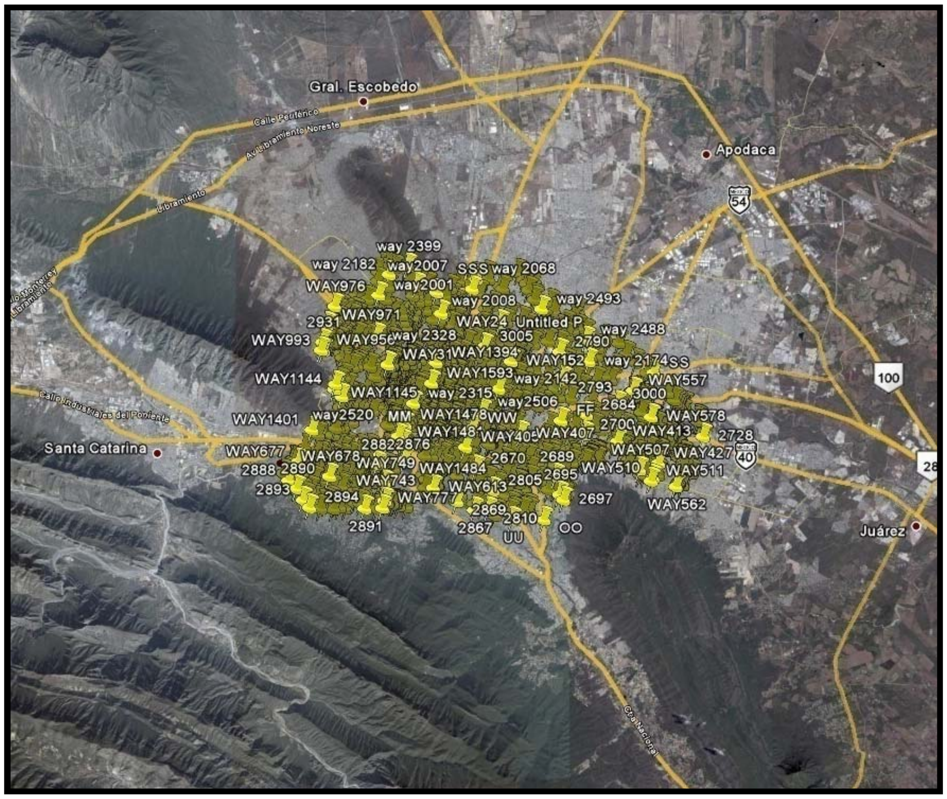


Fig. 5.5 View (a) of the location of the *base points* for the small and medium size instances.

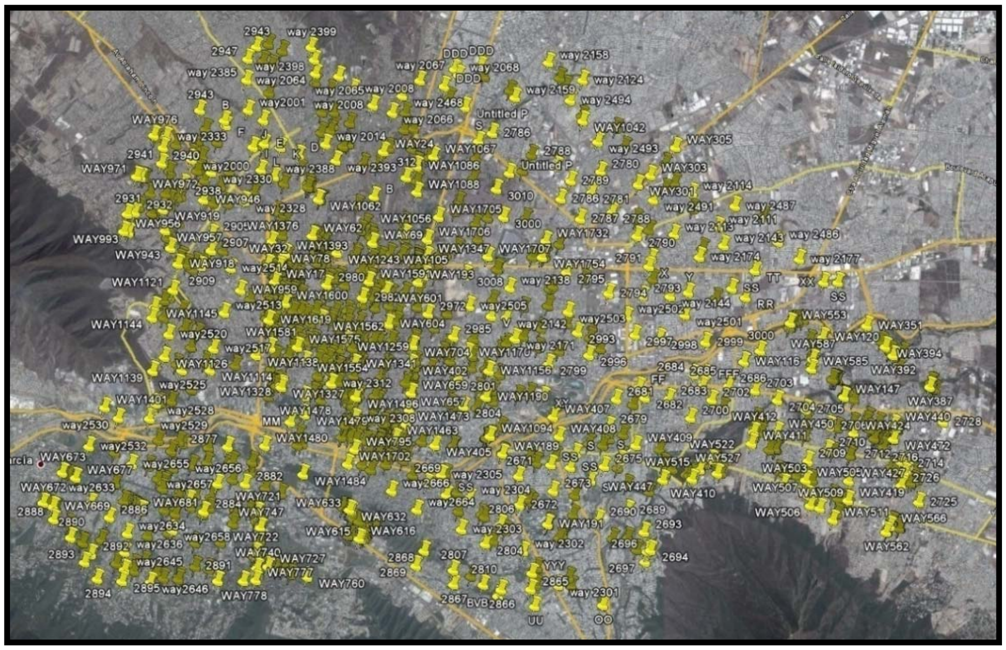


Fig. 5.6 View (b) of the location of the *base points* for the small and medium size instances.

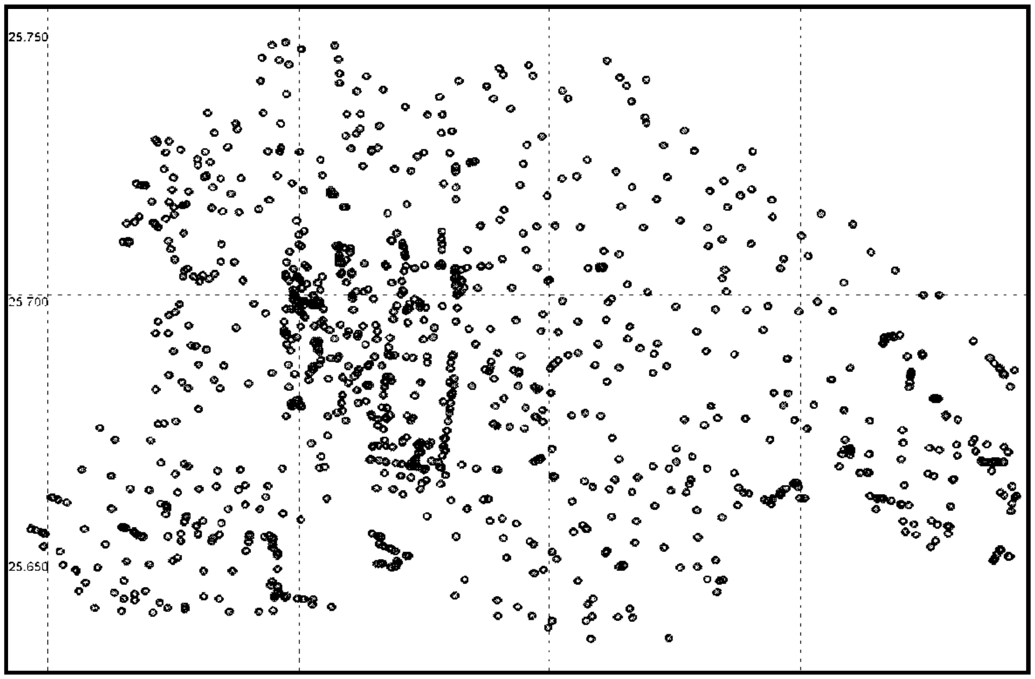


Fig. 5.7 Location of the *base points* for the smaller and medium size instances.

5.1.3 Parcel instances generation

This type of instance is generated by considering the data provided by a parcel company whose operations are performed over the metropolitan region of Monterrey, Nuevo Leon, Mexico, excluding the city of Garcia. Data was collected from some days in which a GPS was sent with each of the vehicles and the stops made over the route were registered. Given that information provided from the GPS is not totally accurate, since it includes any type of stops that may be due to traffic signs or any other stop made during the route, we had to form a representative instance from the data registered by the GPS that may resemble the structure of the districting configuration of the parcel company. Approximately among 30 to 50 points were taken from each route and with the required adjustments to exclude points that are very close to each other and may not represent different customers we constructed the instance with the real location of 1109 customers over the metropolitan region of Monterrey, as it is illustrated by figure 5.8 and 5.9 shows the location of the points over a map and on a plane respectively.

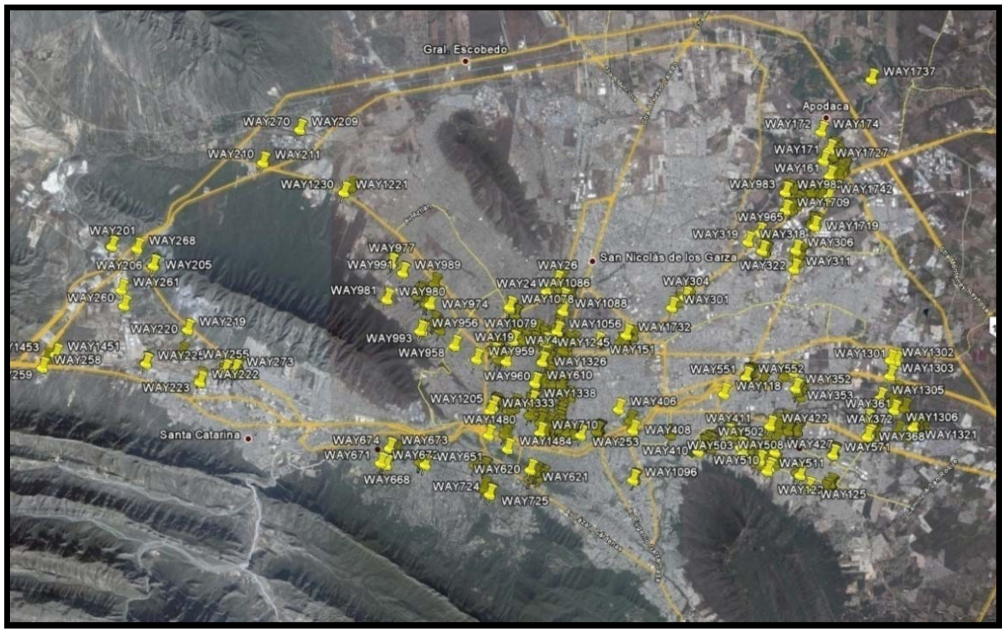


Fig. 5.8 Location of the points on a map for the Parcel instance.

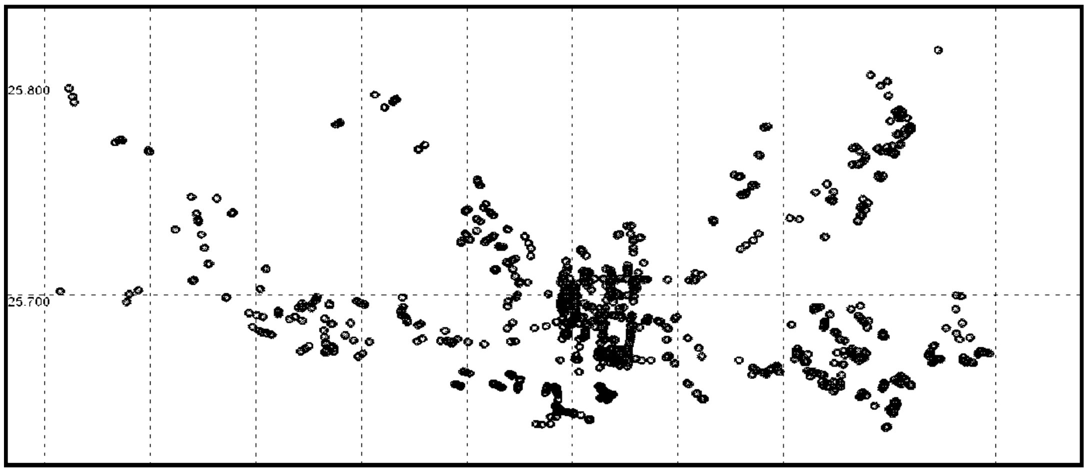


Fig. 5.9 View of the location of the points on a plane for the parcel instance.

The set of edges was defined according to the following rule: for each pair of points, an edge will be included in the graph if the points are adjacent which means that there are no intermediate points to cross between the pair of points. Each point may have in average around three to four edges, with a maximum of six and a minimum of two edges. The depot was defined so that it has fifteen edges connecting the depot with its closest point. Previous considerations allows the instance to be created such that it may resemble the structure of the region under consideration.

Each edge represents the real distance between the pair of points, and it is computed respecting the street structure, which implies that for this type of instance, points that may appear to be close may actually be far apart due to the existence of obstacles such as bridges, drainage canals, and private streets. Figure 5.10 illustrates the computation of the distance between a pair of points.

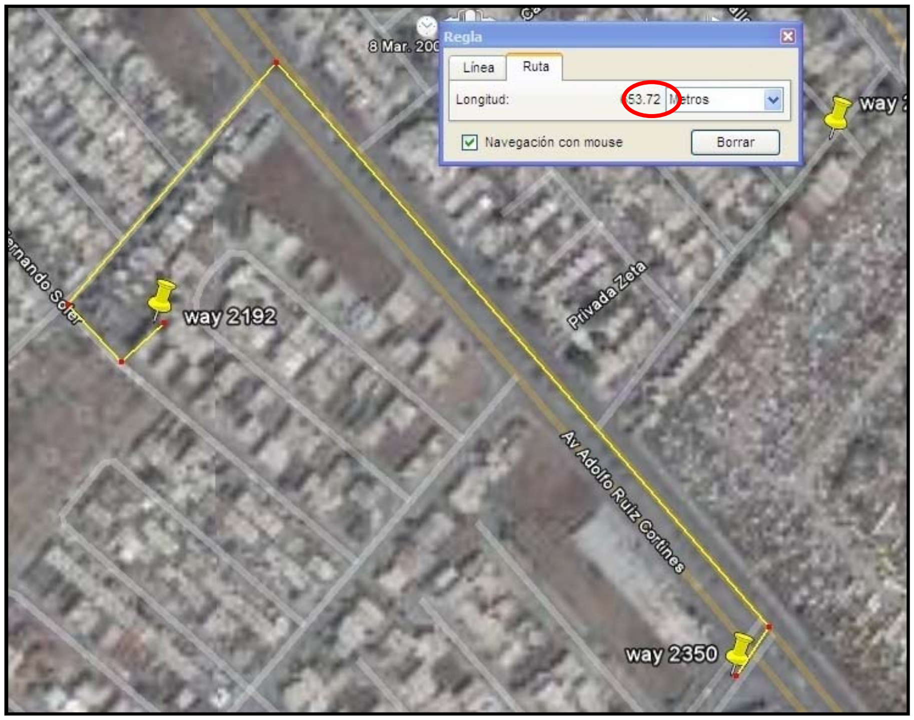


Fig. 5.10 Illustration of the computation of the real distances for the parcel instances.

Once the edge and distance matrix was defined, we randomly assigned the workload content of each point, generating a single replicate of the instance which was solved for the case of each of the two capacity limits that were defined for the rest of the instance types and also varying the three values of the relative weighting factor as well as the three values of the speed.

In total 2x3x3= 18 instances were tested. Results are compared to the current districting design of the parcel company. Figure 5.11 illustrates the location of the points over a map, without any districting configuration and figure 5.12 shows the current districting design of the parcel company with 28 districts:

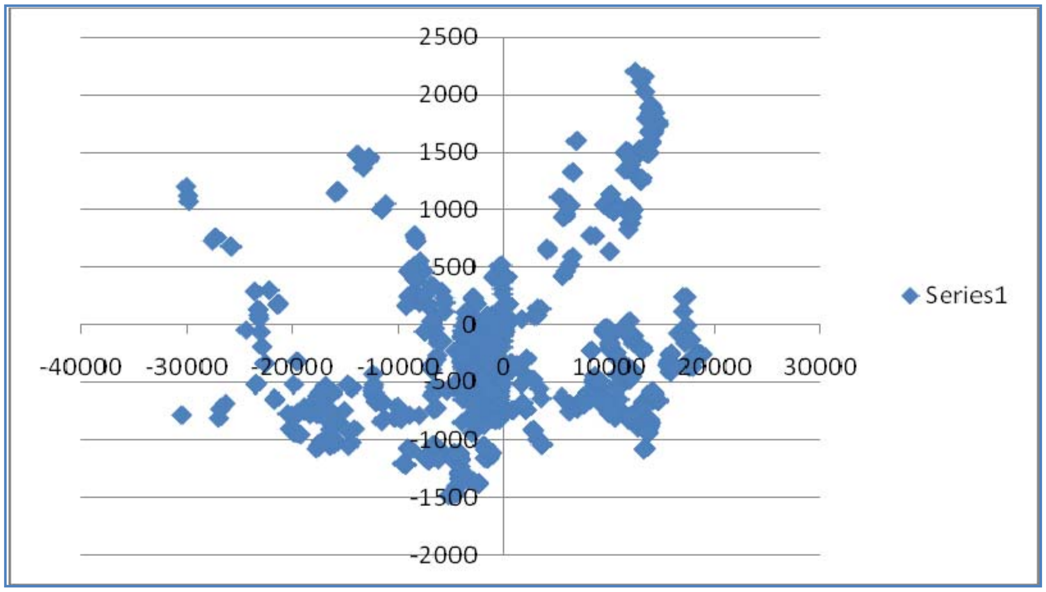


Fig. 5.11 General view of the points for the parcel instance.

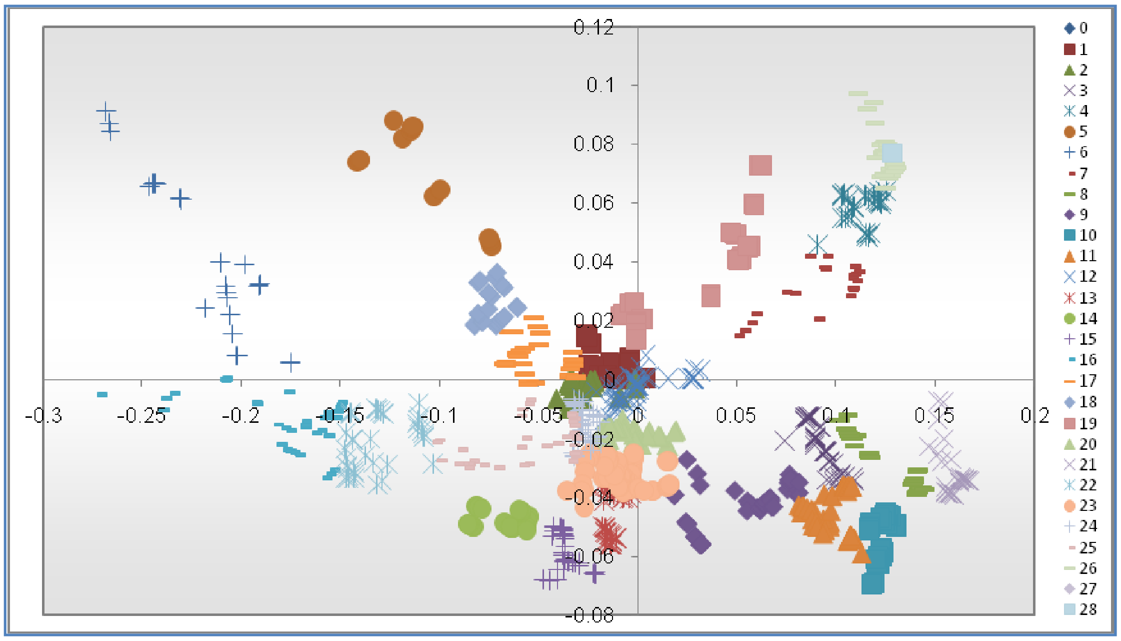


Fig. 5.12 Districting configuration of the parcel instance.

Section 6 contains the comparison of the solution value of both the objective function defined by equation (1) and the dispersion of the workload content among districts as defined in equation (81).

5.1.4 Total number of experiments performed

Considering all the types of instances, a total of 810+270+18= 1,098 instances were generated. From these instances, only the smallest size instances were able to be solved by CPLEX, which correspond only to the smallest size instances (50_5 and 200_10) of the four types of instances: symmetric, semi-symmetric, asymmetric and urban type. In total, considering all the variants on the parameters, 54 instances of each size and type were solved, resulting in total 2x4x54=432 instances that were solved by each of the two optimization models based on the farthest point line haul metric defined in Chapter 3.

The best integer solution found by CPLEX is reported, for both the weighted sum of the metrics for the compactness and balance of the workload metric defined by equation (1) as well as the value of the dispersion that corresponds to the objective function of the second optimization model defined by equation (10). For the heuristics, the value of the objective function defined by equation (1) as well as the value of the dispersion of the workload content among the districts defined by equation (10) is also reported.

The 1,098 instances were solved by each of the five proposed heuristics except for the 2-steps LS, since it turned out to be computationally inefficient for the larger size instances. For this heuristic we present results for all the instance types of 50_5, 200_10 and 450_15 size, as well as the Parcel instances. In total for this 666 instances were solved by the 2-S heuristic. The runs performed by the five heuristics accounts to 1098x4= 4392+66= 5058 runs. All the heuristics evaluate the solutions based on the farthest point line haul metric, defined by equation (1), and in case of ties, they select the solution with less dispersion as it is described in equations (77) for all the algorithms except for the k-Steps/Pairs that uses equation (78).

5.2 Stopping Rules.

A limit time of 3600 seconds was set for the small size instances solved by CPLEX. For the medium size instances of 450 points a limit time of 7200 was set but CPLEX could not find an integer solution, even with no limit time the memory ran out. Hence only the small size instances were solved by CPLEX. For the all the heuristics we set a maximum limit time of 3600 seconds as it was done for CPLEX, however as results will show, only the larger instances reached this limit time.

We also define some stopping conditions as a function of the instance size. Table 5.3 presents the number of iterations that were established for the F-R procedure as well as the LS procedures of each heuristic according to the instance size. We also define a maximum number of points to explore during the FAS procedure in which it is attempted to assign a point to a district according to the instance size.

Table 5.3: Stopping conditions according to the instance size.

Instance (points)	F-R procedure	LS procedure	FAS procedure
50	15 iterations	40 iterations	5 points
200	25 iterations	40 iterations	10 points
450	25 iterations	40 iterations	15 points
1000	30 iterations	50 iterations	25 points
1500	40 iterations	70 iterations	35 points
Parcel-1109	40 iterations	60 iterations	25 points

We also defined for all the heuristics some stopping conditions for the cases in which the F-R may be required to be applied, since there might be some cases in which it was possible to find a feasible solution during whether the FAS or FAP procedures.

The stopping conditions are divided in two categories: the cases in which capacity is violated in terms of a single service activity (pickup or delivery) and the cases in which capacity is violated for both activities and the procedure alternates the iterations performed to attempt obtaining feasibility with respect to each activity. Table 5.4 and 5.5 summarize these stopping conditions.

Table 5.4: Stopping conditions for the F-R procedure (a).

Capacity is violated in terms of a single service activity (pickup or delivery)	
Max_iter =Maximum number of iterations for the H-F procedure applied to achieve a feasible solution with respect to each service activity (*P*=pickups, *D*=deliveries).	$Max\_iter^{P} = \lceil MaxF \rceil + Infeasibility^{P}$, $Max\_iter^{D} = \lceil MaxF \rceil + Infeasibility^{D}$
Max_Best =Maximum number of iterations for the H-F applied over the best solution previously found.	$Max\_Best^{P} = \left\lceil \frac{MaxF}{2} \right\rceil + Infeasibility^{P}$, $Max\_Best^{D} = \left\lceil \frac{MaxF}{2} \right\rceil + Infeasibility^{D}$
Capacity is violated in terms of both service activities (pickup and delivery)	
Tot_iters= Total number of iterations in which the H-F is applied with respect to one of the types of service activities. The H-F procedure includes the iterations performed over the current solution and if required over the best solution found.	$Tot\_iters = 5 + \lceil 0.01 \cdot V \rceil$
Max_iter	$Max\_iter^{P} = \left\lceil \frac{MaxF}{4} \right\rceil + Infeasibility^{P}$, $Max\_iter^{D} = \left\lceil \frac{MaxF}{4} \right\rceil + Infeasibility^{D}$
Max_Best	$Max\_Best^{P} = \left\lceil \frac{MaxF}{8} \right\rceil + Infeasibility^{P}$, $Max\_Best^{D} = \left\lceil \frac{MaxF}{8} \right\rceil + Infeasibility^{D}$

Table 5.5: Stopping conditions for the F-R procedure (b).

Capacity is violated in terms of both service activities (pickup and delivery) cont.	
Max_{iter}^{*} ▬ Maximum number of iterations for the H-F applied over the current solution when only one of the services remains infeasible after a number of consecutive iterations in which it was applied alternating the services that is referred as *iters*.	$Max\_iter^{*} = \lceil (MaxF - iters)/4 \rceil + Infeasibility^{P}$ $Max\_iter^{*} = \lceil (MaxF - iters)/4 \rceil + Infeasibility^{D}$
Max_{Best}^{*} ▬ Maximum number of iterations for the H-F applied over the best solution previously found when only one of the services remains infeasible.	$Max\_Best^{*} = \lceil (MaxF - iters)/8 \rceil + Infeasibility^{P}$ $Max\_Best^{*} = \lceil (MaxF - iters)/8 \rceil + Infeasibility^{D}$

5.3 Numerical Results.

For each of the instances solved by CPLEX and the heuristics, we compute a gap between the best integer solution reported by CPLEX (that in some cases corresponds to the optimal) and each of the five heuristics proposed as described by equation (84). Positive gaps are obtained when CPLEX finds a better solution than the heuristics. Gap is computed for the objective function defined by equation (1) found by CPLEX and the heuristics. A positive Gap indicates that CPLEX found a better solution than the heuristic, and a negative gap indicates that the solution found by the heuristic is better than the best integer solution found by CPLEX under the limit time that was set.

$$Gap = \frac{Heur\_sol - CPLEX\_sol}{CPLEX\_sol} \tag{84}$$

Also, for the case of the symmetric instances, the value of the symmetric optimal solution that is known with advance is compared to the solution found by the heuristic and an analogous gap to that computed with respect to CPLEX solutions is reported.

It is important to point out that the inclusion of constraints (17) to (19) for the second optimization model was required given that during the experimentation we observed that constraint (7) allows D_j to take any value greater or equal to the time to travel from the depot to the farthest point, which for the first model did not represent any problem since we want to minimize the maximum workload content of a district and constraints (7) are enough to guarantee that we compute the value of the maximum workload content of a district. However, for the second optimization model constraints (7) did not guarantee that D_j takes the exact value of the time to travel from the depot to the farthest point of each district, and allow the model to give the required value to D_j so that all the districts seem to have the same workload content resulting in a zero value of the variance. For this reason, the experimentation was redone and constraints (17) to (19) were included in the second optimization model.

We also want to report that even though that we feed an initial solution for the second optimization model, there were some instances in which CPLEX did not report an integer solution during the limit time that was set. This is because the first model did not include the Yij variables, and hence we only feed to the second model the values of the Xij variables, reason for which it was not a complete initial solution for the model. We also observe that for the instances of 200_10, the integer solution reported by CPLEX resulted with some empty districts, which makes even more difficult for the second model to obtain a feasible solution. For this cases, we report the best integer solution reported by CPLEX and we compute the corresponding dispersion.

5.3.1 Results by type and instance size

In this section we present the results found summarized by type and instance size. Table 5.6, 5.7 and 5.8 present the results for the Asymmetric, Semi-Symmetric and Urban instances with respect to CPLEX solutions. CPLEX found at least an integer solution only for the size instances of 50_5, 200_10 and 450_15 for which the maximum, average and minimum gap is shown. Table 5.8 presents the results for the Symmetric instances with respect to the "Optimal symmetric" solution that is known with anticipation because of the structure of this type of instance, and the table present the gaps for CPLEX, and the five heuristics for each instance size (for CPLEX only the instance sizes that could be solved).

We can observe from Table 5.6 that CPLEX did not find the optimal solution for the instances of 200_10, and as we can observe from the table, all the heuristics found a better solution than the best integer solution reported by CPLEX for this instance size. For the smaller size instances of 50_5, CPLEX found the optimal solution for almost all the instances, and in all the cases reported a better or equal solution than the heuristics. We can also observe that the heuristics reported in average small gaps, and a maximum gap of less than 15%. Among the heuristics, the 1-S reported the smallest gaps.

Table 5.6 Gaps with respect to CPLEX for the Asymmetric instances by instance size.

SIZE	**GAP with respect to CPLEX**	**ASYMMETRIC**				
		2-S	**K-S/P**	**1-S**	**HypLS**	**2-IterLS**
50_5	**max**	0.07606	0.13267	0.08730	0.13925	0.12692
	average	0.00725	0.04004	0.01588	0.02093	0.01203
	min	0.00000	0.00000	0.00000	0.00000	0.00000
200_10	**max**	-0.21486	-0.19276	-0.19794	-0.20084	-0.20060
	average	-0.41733	-0.37214	-0.40205	-0.40342	-0.40772
	min	-0.57094	-0.53633	-0.55535	-0.55232	-0.54543

As we can observe in Table 5.7, for all the instances of 50_5, the heuristics found the optimal solution that CPLEX found, and for the case of the instances of 200_10, all the heuristics found a best solution than the integer solution reported by CPLEX or at least the same solution found by CPLEX, except for the K-S/P heuristic that in some instances CPLEX found a better solution, however as we can observe, gaps are quite small.

Table 5.7 Results for the Semi-Symmetric instances with respect to CPLEX by instance size.

SIZE	GAP with respect to CPLEX	SEMI-SYMMETRIC				
		2-S	K-S/P	1-S	HypLS	2-IterLS
50_5	max	0.00000	0.00000	0.00000	0.00000	0.00000
	average	-0.00107	0.00000	0.00000	0.00000	0.00000
	min	-0.01011	0.00000	0.00000	0.00000	0.00000
200_10	max	0.00000	0.05697	0.00000	0.00000	0.00000
	average	-0.45756	-0.42499	-0.45540	-0.46049	-0.46116
	min	-0.59868	-0.56347	-0.59868	-0.59868	-0.59868

Regarding the urban instances we can observe in Table 5.8 small gaps for the smaller instances of 50_5 for which CPLEX found the optimal solution and even for some of the instances the heuristics found also the optimal solution and on average present small gaps with respect to CPLEX. For the instances of 200_10 all the heuristics found a better solution than the best integer solution reported by CPLEX.

Table 5.8 Results for the Urban instances with respect to CPLEX by instance size.

SIZE	GAP with respect to CPLEX	URBAN				
		2-S	K-S/P	1-S	HypLS	2-IterLS
50_5	max	0.00000	0.12911	0.00018	0.04565	0.00000
	average	0.00000	0.01637	0.00002	0.00572	0.00000
	min	0.00000	0.00000	0.00000	0.00000	0.00000
200_10	max	-0.29542	-0.24098	-0.23136	-0.28321	-0.28790
	average	-0.51376	-0.45043	-0.49329	-0.50453	-0.51675
	min	-0.70590	-0.65553	-0.68156	-0.69891	-0.72161

For the Symmetric instances we can observe in Table 5.9 that CPLEX found the optimal solution for the 50_5 instances and as well as all the algorithms except the K-S/P. For the latter, small gaps are presented on average, and only for some instances that are the tight instances a gap of 20% was found. For the instances of 200_10 the heuristics did not find the optimal solution neither CPLEX, and on average present small gaps, even though there were some instances with quite bigger gaps of around 20 to 30% for all the instances sizes.

Table 5.9 Results for the Symmetric instances with respect to the optimal symmetric solution by instance size.

SIZE	GAP with respect to the Optimal Symmetric	SYMMETRIC					
		CPLEX	2-S	K-S/P	1-S	HypLS	2-IterLS
50_5	max	0.00000	0.00000	0.20224	0.00000	0.00000	0.00000
	average	0.00000	0.00000	0.03336	0.00000	0.00000	0.00000
	min	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
200_10	max	1.55344	0.22334	0.37775	0.22334	0.23084	0.22334
	average	0.16309	0.04859	0.17315	0.04859	0.08682	0.02933
	min	-1.00000	0.00000	0.08048	0.00000	0.00000	0.00000
450_15	max		0.21164	0.28140	0.22616	0.26886	0.26160
	average		0.06062	0.12696	0.07794	0.11443	0.09237
	min		0.00000	0.02804	0.00000	0.00000	0.00000
1000_20	max			0.38083	0.18808	0.23955	0.24019
	average			0.12535	0.07167	0.09078	0.07868
	min			0.04102	0.02865	0.01743	0.01743
1500_30	max			0.34573	0.33690	0.34573	0.33690
	average			0.10375	0.13772	0.12852	0.08862
	min			0.00000	0.00000	0.00000	0.00000

5.3.2 Results by type, instance size and capacity limits

In this section we present the results found summarized by type, instance size and the two capacity limits defined for the instances. Table 5.10, 5.11, 5.12 and 5.13 presents the results for the Asymmetric, Semi-Symmetric, Urban and Symmetric instances, in comparison to CPLEX solutions, respectively.

In table 5.10 we can observe that the tight instances resulted more difficult to solve in general, and smaller gaps are found for the instances of 50_5 with respect to CPLEX that in most of the cases found the optimal solution for these instances, and for the 200_10 instances the gaps with respect to the solution found by CPLEX and the algorithms are similar for both type of capacity limits, in which the heuristics found a better solution than the reported by CPLEX.

Table 5.10 Results for the Asymmetric instances vs. CPLEX by instance size and capacity limits

SIZE	CAPACITY	GAP with respect to CPLEX	ASYMMETRIC				
			2-S	K-S/P	1-S	HypLS	2-IterLS
50_5	Less restricted	max	0.01396	0.08264	0.03061	0.04564	0.01396
		average	0.00438	0.03110	0.00925	0.00903	0.00320
		min	0.00438	0.03110	0.00925	0.00903	0.00320
	Tight	max	0.07606	0.13267	0.08730	0.13925	0.12692
		average	0.01012	0.04899	0.02251	0.03283	0.02087
		min	0.01012	0.04899	0.02251	0.03283	0.02087
200_10	Less restricted	max	-0.21486	-0.19276	-0.19794	-0.20084	-0.20060
		average	-0.41084	-0.37448	-0.39350	-0.39931	-0.40256
		min	-0.57094	-0.53633	-0.54468	-0.54450	-0.54543
	Tight	max	-0.24750	-0.21661	-0.25302	-0.23635	-0.24443
		average	-0.42382	-0.36980	-0.41060	-0.40754	-0.41287
		min	-0.55652	-0.50011	-0.55535	-0.55232	-0.53779

In table 5.11 we can observe that for the Semi-Symmetric instances, given that the heuristics found the optimal solution as well as CPLEX did it for the smaller size instances, there is no difference with respect to the capacity limits. For the case of the instances of 200_10 we can observe that the tight instances resulted either easier to be solved than the less restricted, or at least the heuristics found the same solution than CPLEX. For the tight instances, all the heuristics found a better solution than the reported by CPLEX.

Table 5.11 Results for the Semi-Symmetric instances vs. CPLEX by instance size and capacity limits.

SIZE	CAPACITY	GAP with respect to CPLEX	SEMI-SYMMETRIC				
			2-S	K-S/P	1-S	HypLS	2-IterLS
50_5	Less restricted	max	0.00000	0.00000	0.00000	0.00000	0.00000
		average	0.00000	0.00000	0.00000	0.00000	0.00000
		min	0.00000	0.00000	0.00000	0.00000	0.00000
	Tight	max	0.00000	0.00000	0.00000	0.00000	0.00000
		average	0.00000	0.00000	0.00000	0.00000	0.00000
		min	0.00000	0.00000	0.00000	0.00000	0.00000
200_10	Less restricted	max	0.00000	0.05697	0.00000	0.00000	0.00000
		average	-0.44501	-0.40638	-0.44069	-0.44368	-0.44501
		min	-0.59868	-0.55672	-0.59868	-0.59868	-0.59868
	Tight	max	-0.30760	-0.28311	-0.30760	-0.30760	-0.30760
		average	-0.47012	-0.44359	-0.47012	-0.47730	-0.47730
		min	-0.59484	-0.56347	-0.59484	-0.59674	-0.59674

For the case of the urban instances, in Table 5.12 we observe similar gaps for both type of capacities. In the case of the 50_5 instances, the 2-IterLS found the optimal solution as well as CPLEX did it and for the 1-S algorithm it resulted that some instances of less restricted capacity could not find the optimal solution but presents gaps very small. For the instances of 200_10 size, all the heuristics found a better solution than CPLEX and results do not differ too much between both types of capacity.

Table 5.12 Results for the Urban instances vs. CPLEX by instance size and capacity limits.

SIZE	CAPACITY	GAP with respect to CPLEX	URBAN				
			2-S	K-S/P	1-S	HypLS	2-IterLS
50_5	Less restricted	max	0.00000	0.02875	0.00018	0.01168	0.00000
		average	0.00000	0.00783	0.00004	0.00271	0.00000
		min	0.00000	0.00000	0.00000	0.00000	0.00000
	Tight	max	0.00000	0.12911	0.00000	0.04565	0.00000
		average	0.00000	0.02490	0.00000	0.00872	0.00000
		min	0.00000	0.00000	0.00000	0.00000	0.00000
200_10	Less restricted	max	-0.31338	-0.24705	-0.30749	-0.30609	-0.31563
		average	-0.51458	-0.45661	-0.50240	-0.50373	-0.51341
		min	-0.70590	-0.65553	-0.68156	-0.69702	-0.70568
	Tight	max	-0.29542	-0.24098	-0.23136	-0.28321	-0.28790
		average	-0.51294	-0.44424	-0.48418	-0.50534	-0.52009
		min	-0.69921	-0.65100	-0.67780	-0.69891	-0.72161

Regarding the symmetric instances, we can observe in table 5.13 that results do not differ too much between both types of capacity and depending on the instance size there are cases in which some heuristics present bigger gaps with the less restricted capacity such as for the K-S/P algorithm and the 200_5 size, and there are cases in which the tight capacity resulted in bigger gaps such as for the same heuristic and the instance of 1000_20 size.

Table 5.13 Results for the Symmetric instances with respect to the optimal symmetric by instance size and capacity limits.

SIZE	CAPACITY	GAP with respect to the Optimal Symmetric	SYMMETRIC					
			CPLEX	2-S	K-S/P	1-S	HypLS	2-IterLS
50_5	Less restricted	max	0.00000	0.00000	0.20224	0.00000	0.00000	0.00000
		average	0.00000	0.00000	0.06672	0.00000	0.00000	0.00000
		min	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
	Tight	max	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
		average	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
		min	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
200_10	Less restricted	max	1.55344	0.22334	0.37775	0.22334	0.23084	0.03767
		average	0.29240	0.04859	0.18327	0.04859	0.09761	0.00646
		min	-1.00000	0.00000	0.09358	0.00000	0.02820	0.00000
	Tight	max	1.45398	0.22334	0.26715	0.22334	0.23084	0.22334
		average	0.03377	0.04859	0.16303	0.04859	0.07603	0.05220
		min	-1.00000	0.00000	0.08048	0.00000	0.00000	0.00000
450_15	Less restricted	max		0.14265	0.27613	0.20954	0.26160	0.26160
		average		0.02928	0.13545	0.05099	0.10783	0.09143
		min		0.00000	0.04152	0.00000	0.00000	0.00000
	Tight	max		0.21164	0.28140	0.22616	0.26886	0.22616
		average		0.09197	0.11847	0.10488	0.12103	0.09331
		min		0.02089	0.02804	0.02999	0.01349	0.00495
1000_20	Less restricted	max			0.31193	0.11724	0.23536	0.16267
		average			0.12026	0.06612	0.09425	0.06791
		min			0.04914	0.02865	0.04558	0.01743
	Tight	max			0.38083	0.18808	0.23955	0.24019
		average			0.13044	0.07723	0.08732	0.08944
		min			0.04102	0.02865	0.01743	0.03291
1500_30	Less restricted	max			0.34573	0.33690	0.34573	0.33690
		average			0.09848	0.14400	0.11974	0.09391
		min			0.00000	0.00000	0.00000	0.00000
	Tight	max			0.34573	0.33248	0.34132	0.26889
		average			0.10902	0.13144	0.13731	0.08333
		min			0.00000	0.00000	0.00000	0.00000

5.3.3 Objective function and dispersion of the workload content values by instance type and size

In this section we present the minimum, average and maximum value of the objective function found for each instance as defined by equation (1) and also for the dispersion of the workload content among the districts as defined by equation (77). Tables 5.14 to 5.21 present the results for each instance type.

In Table 5.14 we can observe that for the asymmetric instances, the objective function values are very similar to those found by CPLEX for the instances of 50_5, resulting the 2-S algorithm with better objective function values, but very close to those found by the 1-S. For the 200_10 instances all the algorithms found a better solution than CPLEX, and also the 2-S resulted with better solutions but very close to those reported by the 1-S. For the 450_15 instances it is possible to observe that even though for the smaller size instances the 2-S resulted in better solutions, for this instance size that was the biggest size instance solved by this algorithm, the 1-S performed better. This make us think that the 2-S algorithm that consumes such long time for large size instances, would not actually provide better solutions as those algorithms that are more efficient.

In table 5.15 the dispersion of the workload content is presented. We can observe that for the case of the hyperheuristic, there are instances of the biggest size of 1500_30 in which the heuristic found a solution with a dispersion of zero value, and on average it resulted in the smallest value of the workload dispersion for the medium and large instance sizes. For the smaller size instances of 50_5 the 2-S found solutions with less workload dispersion, and for the 200_10 the 1-S resulted in better solutions.

Table 5.14 Objective function values for the Asymmetric instances by instance size.

SIZE	METRIC	ASYMMETRIC					
		OBJECTIVE FUNCTION					
		CPLEX	2-S	K-S/P	1-S	HypLS	2-IterLS
50_5	max	0.94953	0.94953	1.01772	0.94953	0.97601	0.96096
	average	0.86612	0.87254	0.90121	0.88002	0.88443	0.87673
	min	0.73923	0.73923	0.74909	0.73923	0.73923	0.73923
200_10	max	2.27857	1.12006	1.25560	1.13878	1.15585	1.12936
	average	1.74385	0.98316	1.06500	1.00986	1.00833	1.00082
	min	1.21589	0.86576	0.95652	0.89946	0.90280	0.88679
450_15	max		1.38493	1.42181	1.21598	1.34886	1.22486
	average		1.15369	1.21059	1.08904	1.12892	1.08770
	min		1.04996	1.06012	1.00393	1.01917	0.99298
1000_20	max			1.67838	1.53757	1.53481	1.53268
	average			1.37688	1.27487	1.28518	1.27241
	min			1.17574	1.08738	1.12186	1.10148
1500_30	max			1.89246	1.65811	1.69469	1.85812
	average			1.44037	1.34197	1.22138	1.37479
	min			1.18246	1.13763	0.83142	1.14144

Table 5.15 Dispersion values for the Asymmetric instances by instance size.

SIZE	METRIC	ASYMMETRIC					
		WORKLOAD DISPERSION					
		CPLEX	2-S	K-S/P	1-S	HypLS	2-IterLS
50_5	max	1.96579	2.83872	6.54424	3.45351	2.60502	2.17911
	average	0.50514	0.68649	3.12133	0.97097	0.79962	0.70234
	min	0.07215	0.14797	1.30597	0.07630	0.05924	0.07630
200_10	max	0.29184	19.44014	32.63868	14.42094	21.96877	16.79725
	average	0.20634	10.73822	19.05335	8.86711	10.17375	9.37112
	min	0.08525	3.67042	6.60917	3.05867	3.71981	2.38643
450_15	max		142.19948	149.83945	157.77258	126.58720	149.47059
	average		72.96754	80.95720	78.53688	60.07404	70.14381
	min		8.05624	16.87058	41.50168	20.37104	37.51399
1000_20	max			457.17610	433.35831	309.17219	413.04939
	average			274.21254	253.70523	208.14440	245.56166
	min			129.59389	55.16285	131.88706	57.51319
1500_30	max			873.37492	874.74138	503.83913	745.16830
	average			553.44177	502.07141	106.37232	465.58103
	min			313.46919	235.95466	0.00000	223.70942

In Table 5.16 we present the objective function values for the Semi-Symmetric instances. We can observe that for the smallest size instance, the heuristic found the optimal solution as it was done by CPLEX and found better solutions than CPLEX for the 200_10 size. For the rest of the instance sizes, the heuristics found similar solutions, resulting the most difficult the biggest size instance of 1500_30.

Table 5.16 Objective function values for the Semi-Symmetric instances by instance size.

SIZE	METRIC	SEMI-SYMMETRIC					
		OBJECTIVE FUNCTION					
		CPLEX	2-S	K-S/P	1-S	HypLS	2-IterLS
50_5	max	0.95086	0.95086	0.95086	0.95086	0.95086	0.95086
	average	0.85708	0.85617	0.85708	0.85708	0.85708	0.85708
	min	0.69069	0.69069	0.69069	0.69069	0.69069	0.69069
200_10	max	2.64807	1.13396	1.21859	1.13396	1.13396	1.13396
	average	1.83609	0.95339	1.01112	0.95707	0.94787	0.94701
	min	0.94208	0.81348	0.86883	0.82625	0.78923	0.78923
450_15	max		1.36855	1.49135	1.40093	1.50382	1.48777
	average		1.04916	1.11393	1.05578	1.09213	1.06793
	min		0.87095	0.87923	0.87095	0.87095	0.87095
1000_20	max			1.42977	1.42478	1.42727	1.41765
	average			1.18175	1.15756	1.16359	1.14938
	min			1.03348	1.01123	1.03348	1.00278
1500_30	max			2.14385	2.15251	2.01899	2.13735
	average			1.55418	1.51379	1.34227	1.54330
	min			1.09899	1.03854	1.07672	1.06930

Regarding the dispersion of the workload content, we can observe in Table 5.17 that for this type of instances, the dispersion is quite small, and only the HypLS algorithm found solutions with an average value of 155 seconds, but for the rest of the heuristics, the dispersion resulted in less than 20 seconds for all the instance sizes.

Table 5.17 Dispersion values for the Semi-Symmetric instances by instance size.

SIZE	METRIC	SEMI-SYMMETRIC					
		WORKLOAD DISPERSION					
		CPLEX	2-S	K-S/P	1-S	HypLS	2-IterLS
50_5	max	0.00000	0.33333	1.66667	0.00000	0.00000	0.00000
	average	0.00000	0.09877	0.66667	0.00000	0.00000	0.00000
	min	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
200_10	max	17.06306	0.33333	3.50000	0.00000	0.00000	0.00000
	average	6.67522	0.19444	1.19444	0.00000	0.00000	0.00000
	min	0.00000	0.00000	0.33333	0.00000	0.00000	0.00000
450_15	max		0.25477	7.51649	1.00954	1.31706	1.06212
	average		0.14305	4.40926	0.18877	0.67163	0.20079
	min		0.00000	2.16667	0.00000	0.00000	0.00000
1000_20	max			11.50000	0.83333	3.00000	1.66667
	average			8.62963	0.38889	1.45062	0.44444
	min			7.00000	0.00000	0.33333	0.00000
1500_30	max			19.66667	4.33333	457.17610	2.66667
	average			13.95136	1.11111	154.24457	1.59368
	min			7.83333	0.00000	0.83333	0.00000

For the symmetric instances, we observe in Table 5.18 that the heuristics found the optimal solution for all the instances of 50_5 except the K-S/P. In comparison to the results obtained for the asymmetric instances shown in table 5.14, we can observe smaller values of the objective function, especially if we focus our attention to the largest size instances of 1500_30, from which we can conclude that the asymmetric instances resulted more difficult to be solved.

Regarding the values of the dispersion of the workload content of the symmetric instances, in table 5.19 is possible to observe that for this type of instance the dispersion resulted in very small values, even for the largest size instances of 1500_30, for which the heuristics found for some instances a solution in which the workload content is equally distributed among the districts.

Table 5.18 Objective function values for the Symmetric instances by instance size.

SIZE	METRIC	SYMMETRIC					
		OBJECTIVE FUNCTION					
		CPLEX	2-S	K-S/P	1-S	HypLS	2-IterLS
50_5	max	0.94445	0.94445	0.99654	0.94445	0.94445	0.94445
	average	0.87384	0.87384	0.90264	0.87384	0.87384	0.87384
	min	0.77580	0.77580	0.77580	0.77580	0.77580	0.77580
200_10	max	1.90615	1.11275	1.12306	1.11275	1.11957	1.11275
	average	0.99633	0.92052	1.02261	0.92052	0.95217	0.90250
	min		0.71559	0.90643	0.71559	0.71559	0.71559
450_15	max		0.97724	0.99841	0.98595	1.00110	0.97724
	average		0.88751	0.94079	0.90162	0.92962	0.91170
	min		0.70511	0.83611	0.70511	0.85636	0.82893
1000_20	max			1.03070	1.01369	1.03070	1.01369
	average			0.95314	0.91217	0.92592	0.91636
	min			0.83794	0.73362	0.81199	0.76445
1500_30	max			1.36603	1.50609	1.49927	1.31107
	average			1.10151	1.14247	1.12954	1.08796
	min			0.83099	0.82323	0.82841	0.78309

Table 5.19 Dispersion values for the Symmetric instances by instance size.

SIZE	METRIC	SYMMETRIC					
		WORKLOAD DISPERSION					
		CPLEX	2-S	K-S/P	1-S	HypLS	2-IterLS
50_5	max	0.00000	0.33333	0.66667	0.00000	0.00000	0.01596
	average	0.00000	0.05556	0.13889	0.00000	0.00000	0.00075
	min	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
200_10	max	16.17730	0.33333	8.16667	0.00000	1.00000	0.83333
	average	10.39527	0.20771	3.43541	0.00000	0.19136	0.12037
	min	0.00188	0.00000	0.83333	0.00000	0.00000	0.00000
450_15	max		2.00000	7.33333	4.00000	2.33333	2.16667
	average		0.55864	4.90883	1.08333	0.66696	0.52160
	min		0.00000	3.33333	0.00000	0.00000	0.00000
1000_20	max			15.16667	1.33333	5.00000	3.50000
	average			10.12144	0.76852	2.47222	1.40741
	min			4.00000	0.00000	1.16667	0.00000
1500_30	max			37.33333	23.00000	2.00000	5.66667
	average			12.55556	6.10185	1.18966	2.13889
	min			0.00000	0.00000	0.00000	0.00000

Table 5.20 presents the objective function values for the urban type instances. It is possible to observe in the table, smaller values than those found for the asymmetric instances, and also very close values to those found by CPLEX for the smaller size instances of 50_5 for which CPLEX found the optimal solution. In case of the 200_10 instances, we observe that all the heuristics found a better solution than CPLEX, and the 2-S found a better solution than the other procedures. However, we can observe that for the instances of 50_5 the 2-IterLS reported the same results than CPLEX and the 2-S. For the 200_10 instances, the 2-S reported better values for the minimum objective function but the 2-Iter reported better values for the maximum and average objective function. For the larger instance sizes, the 1-S reported better solutions on average but the 2-Iter reported smaller maximum and minimum objective function values.

Table 5.20 Objective function values for the Urban instances by instance size.

SIZE	METRIC	URBAN					
		OBJECTIVE FUNCTION					
		CPLEX	2-S	K-S/P	1-S	HypLS	2-IterLS
50_5	max	0.93661	0.93661	0.94306	0.93661	0.93661	0.93661
	average	0.73699	0.73699	0.74759	0.73700	0.74066	0.73699
	min	0.54227	0.54227	0.54861	0.54227	0.54861	0.54227
200_10	max	1.98814	0.91608	1.01847	1.03535	0.92841	0.91917
	average	1.59931	0.74589	0.85331	0.78066	0.76122	0.74101
	min	1.26096	0.54231	0.63532	0.58732	0.55847	0.53018
450_15	max		0.95961	1.06564	0.96283	0.94059	0.93126
	average		0.82420	0.86449	0.85214	0.82589	0.79397
	min		0.58710	0.62924	0.58726	0.62924	0.59365
1000_20	max			1.43967	1.41477	1.31399	1.36568
	average			1.16988	1.12227	1.12348	1.10705
	min			0.93130	0.90656	0.97140	0.92908
1500_30	max			1.66709	1.57777	1.52848	1.48707
	average			1.33567	1.24259	1.30543	1.28487
	min			0.97898	0.92996	0.97076	0.85244

Regarding the dispersion of the workload content among this type of instances that resemble the structure of a urban region, we can observe in Table 5.21 relatively large values of dispersion. However, if we observe the smaller size instance of 50_5 for which CPLEX found the optimal solution, the heuristics found similar values of dispersion and this value tends also to be big. Hence for this type of instance, given its characteristics, it resulted difficult to have a perfect balance of workload content among the districts and as long as the instance size of the instances increases it is expected that the value of the dispersion tends to increase.

Table 5.21 Dispersion values for the Symmetric instances by instance size.

SIZE	METRIC	URBAN					
		WORKLOAD DISPERSION					
		CPLEX	2-S	K-S/P	1-S	HypLS	2-IterLS
50_5	max	764.61300	784.74437	795.34310	794.62386	840.05954	949.01457
	average	584.92330	589.73779	627.28868	604.57349	605.04910	644.24126
	min	445.11200	439.55939	445.93141	445.11160	424.33735	445.11160
200_10	max	267.38600	1869.22931	2161.32844	1758.99071	1715.59088	1879.00128
	average	143.13093	1284.89590	1400.27025	1268.27201	1240.43266	1249.48070
	min	59.36390	914.91277	960.33445	941.34388	805.68118	757.42086
450_15	max		2739.69385	3265.45663	2977.65324	2901.46206	3161.96745
	average		2195.35235	2232.97055	2237.50232	2125.33201	2155.09940
	min		1773.21289	1672.01362	1704.35172	1643.94637	1474.71149
1000_20	max			8220.29673	8264.06014	7472.33124	6447.01348
	average			5938.91967	5630.09660	5452.43349	5055.76068
	min			4293.05354	4062.05036	4100.00381	3935.10474
1500_30	max			11118.55023	12153.24513	12336.32887	10941.97639
	average			8840.01673	8936.94271	9044.27607	8361.15860
	min			6774.82096	5941.24238	5941.24238	6688.72410

5.3.4 Results of the Parcel instance in comparison to the current solution.

In this section we present the results for the parcel instance in comparison to the current solution according to the data provided by the company. Tables 5.22 and 5.23 present the results.

In Table 5.22 we can observe the gaps with respect to the current solution. It is possible to observe that all the heuristics found a better solution because for all the cases we have negative gaps that indicates that current solution resulted in a worst value. From all the methods we can observe that the most negative values corresponds to the 2-Iter algorithm, meaning that better solutions were found by this algorithm. Worst solutions correspond to the 2-S algorithm. Regarding the capacity levels, we can observe that results are very similar, but on average, more negative gaps are found for the tight instances, meaning that better solutions than the current solution were found. Regarding the values of lambda, better solutions were found when lambda has the value of 0.25, which means that more weight is given to the compactness metric.

Table 5.22 Gap with respect to the current solution of the parcel instance.

VARIANTS	Gap with respect to current solution	PARCEL INSTANCE				
		2-S	**K-S/P**	**1-S**	**HypLS**	**2-IterLS**
ALL	**max**	-0.19247	-0.19919	-0.19387	-0.22020	-0.23475
	average	-0.29536	-0.31405	-0.34229	-0.34900	-0.37110
	min	-0.38604	-0.43666	-0.50448	-0.47008	-0.49870
TIGHT	**max**	-0.19247	-0.21803	-0.25260	-0.23544	-0.24928
	average	-0.29461	-0.32299	-0.38697	-0.36062	-0.38231
	min	-0.38390	-0.43666	-0.50448	-0.47008	-0.49870
LESS RESTRICTED	**max**	-0.19311	-0.19919	-0.19387	-0.22020	-0.23475
	average	-0.29611	-0.30510	-0.29761	-0.33738	-0.35990
	min	-0.38604	-0.39753	-0.38832	-0.43967	-0.46920
Lambda=0.25	**max**	-0.38373	-0.33188	-0.38826	-0.43949	-0.46904
	average	-0.38489	-0.39958	-0.44634	-0.45479	-0.48388
	min	-0.38604	-0.43666	-0.50448	-0.47008	-0.49870
Lambda=0.5	**max**	-0.30710	-0.31797	-0.31015	-0.35164	-0.37518
	average	-0.30807	-0.33365	-0.35693	-0.36402	-0.38711
	min	-0.30900	-0.34924	-0.40387	-0.37638	-0.39900
Lambda=0.75	**max**	-0.19247	-0.19919	-0.19387	-0.22020	-0.23475
	average	-0.19312	-0.20891	-0.22360	-0.22820	-0.24233
	min	-0.19371	-0.21846	-0.25334	-0.23619	-0.24982

In table 5.23 we can observe that in general, the heuristics found better objective function values than the current solution, however current solution presents less dispersion than the heuristics, but it does not differ too much. Regarding computational times, we can observe that the all the heuristics except the 2-S have an average of less than 1800 seconds (half of an hour) which is a good time given that we are solving a strategic problem. The 2-S algorithm on the other hand, presents longer computational times, reaching and exceeding the limit time. Given that this is an strategic problem it may be considered reasonable that the heuristic requires on average 2.38 hours. However, the heuristic could only construct and improve a single solution, while the other heuristics are able to generate more solutions.

Table 5.23 Objective function and dispersion values of the parcel instance.

METRIC	PARCEL INSTANCE					
	CURRENT SOLUTION	2-S	K-S/P	1-S	HypLS	2-IterLS
	OF					
max	2.35216	1.44958	1.57153	1.43884	1.31834	1.24886
average	1.96205	1.35761	1.32108	1.26150	1.24784	1.20252
min	1.57174	1.26729	1.22863	1.16549	1.20075	1.17908
METRIC	WORKLOAD DISPERSION					
max	6333.65870	7788.14042	8414.61597	7221.64446	8644.82118	7341.21931
average	5379.47538	6318.98775	7011.99013	5978.91691	7252.88124	6206.77851
min	4525.73240	5066.48535	5857.33099	4988.45573	6029.14154	5183.34200
METRIC	COMPUTATIONAL TIME					
max		9962.70400	4041.40300	1146.68700	1569.92200	1690.34300
average		8585.09383	1632.66022	1125.19478	1538.04172	1612.27472
min		7208.25000	1479.25000	1103.93700	1506.26700	1535.26600

5.3.5 Computational times per instance type and size

Tables 5.24 to 5.27 presents the computational time for each of the instance sizes required by CPLEX and the heuristics in CPU-seconds. In table 5.24 we present the time for the asymmetric instances. The time reported for CPLEX includes the time to solve the first and second optimization models, that for the case of the instances of 200_10 reached the limit time in most of the cases. For the heuristics, we can observe considerable small computational times of less than 2 seconds for the instances of 50_5, and less than 33 seconds for the instances of 200_10 size. For the rest of instances in which CPLEX could not find any integer solution, the algorithms present small computational gaps, and only for the largest size instances reached the limit time of 3600 seconds. However, for the instances of 1000_20 which most approximates to the instances size of a real instance, all the heuristics takes less than 1800 seconds (half of an hour), which is a reasonable amount of time given that we are dealing with a strategic problem. The HypLS algorithm resulted with the smaller times in most of the instance sizes, including the largest size instances of 1500_30.

Table 5.24 Computational time for the Asymmetric instances.

SIZE	AVERAGE TIMES	ASYMMETRIC					
		CPLEX	2-S	K-S/P	1-S	HypLS	2-IterLS
50_5	max	7203.24000	20.10900	0.98600	0.56300	0.81200	1.31200
	average	3200.69089	13.98128	0.79598	0.41485	0.65300	1.06320
	min	166.48500	8.63900	0.59400	0.29600	0.53000	0.89000
200_10	max	7209.93000	2083.14200	20.48500	16.43700	20.64000	32.90600
	average	4604.93819	1504.95961	17.86404	14.00572	17.88026	28.15261
	min	0.64100	1064.32800	14.81400	12.17200	14.92300	23.32900
450_15	max		4695.94800	150.40600	140.04800	167.88900	260.78000
	average		4147.51370	119.59681	116.51735	132.65069	209.38937
	min		3611.86000	86.43900	95.87700	93.29600	163.75100
1000_20	max			1793.29600	1173.82700	339.12500	1555.92300
	average			1441.03896	1032.09939	295.02496	1303.13004
	min			1230.31400	924.79900	268.25100	1159.09300
1500_30	max			3779.56500	3829.18700	3754.78100	3898.06100
	average			3610.31274	3733.51220	2546.77924	3730.87119
	min			3271.64300	3608.01500	1506.00000	3603.82800

Table 5.25 shows the computational time for the Semi-Symmetric instances, in which we can observe that given the special structure of this type of instances, they resulted easier to solve for the heuristics and also for CPLEX than the Asymmetric instances. In this type of instances, computational times in general are small, and only for the largest size instance of 1500_30 the HypLS heuristic reached the maximum computational time for some instances, presenting the biggest times, which is was the opposite as for the Asymmetric instances in which this heuristic presented the smaller times for this instance size. . For the case of the 1-S algorithm we can observe that it has less than 1800 seconds and in average less than 1300 seconds for the largest instance.

Table 5.25 Computational time for the Semi-Symmetric instances.

SIZE	**AVERAGE TIMES**	**SEMI-SYMMETRIC**					
		CPLEX	**2-S**	**K-S/P**	**1-S**	**HypLS**	**2-IterLS**
50_5	**max**	471.62500	3.96900	10.37500	7.18700	9.26500	13.90600
	average	70.33074	2.89839	9.03409	5.93739	8.30209	12.02143
	min	7.12500	2.04600	7.13900	4.90800	7.00100	9.56500
200_10	**max**	3605.58000	258.90700	84.81200	53.35900	78.15700	112.75000
	average	3603.08130	160.31030	61.69493	43.55681	58.16463	80.46946
	min	3602.12000	77.71900	32.71900	32.13900	33.62500	41.45400
450_15	**max**		2413.54400	84.81200	53.35900	78.15700	112.75000
	average		1779.66294	61.69493	43.55681	58.16463	80.46946
	min		1122.68700	32.71900	32.13900	33.62500	41.45400
1000_20	**max**			608.37500	431.92100	450.78000	737.46800
	average			494.72872	351.37283	404.31037	585.83657
	min			307.89000	285.73300	306.04700	375.86100
1500_30	**max**			2233.57900	1595.54800	4097.26500	2774.37400
	average			1815.40643	1286.22300	2638.16837	2251.16904
	min			1414.78100	993.57900	1218.51400	1564.35900

In table 5.26 we can observe the computational time required for the Urban instances. We can observe that in general the heuristics present small times, and only for the largest size instance reached the limit time as it was done for the asymmetric instances. However for the instances of 1000_20 and smaller, computational times are quite small for all the heuristics. For this type of instance, we observe that the 1-S resulted with the smaller computational times for all the instance sizes, and the K-S/P resulted with the biggest time for the largest size instances. Also as it happens in all the types of instances, the 2-S resulted in long computational times reason for which it was tested only in the small and medium size instances.

Table 5.26 Computational time for the Urban instances.

SIZE	AVERAGE TIMES	URBAN					
		CPLEX	**2-S**	**K-S/P**	**1-S**	**HypLS**	**2-IterLS**
50_5	**Max**	1174.65600	10.06300	0.92100	0.42300	0.64000	1.03100
	Average	467.11759	6.58956	0.70831	0.31500	0.51341	0.80439
	Min	300.70300	4.34400	0.51600	0.23400	0.37600	0.62300
200_10	**Max**	3606.01000	406.15600	13.35900	6.59300	11.73400	17.86000
	Average	3603.76296	237.32409	9.72185	5.47674	8.64683	12.01083
	Min	3601.76000	62.18700	5.71800	3.95300	5.45300	6.10900
450_15	**Max**		3151.53200	96.28000	59.04700	90.81400	120.89100
	Average		1868.59059	86.28944	48.86133	84.35357	94.97598
	Min		1155.56200	77.81100	44.03100	74.21900	81.93700
1000_20	**Max**			577.96800	413.09300	558.00000	558.00000
	Average			952.36226	603.70213	878.69080	878.69080
	min			1447.76600	806.31200	1242.11000	1242.11000
1500_30	**max**			5307.62200	4097.26500	4097.26500	4259.99900
	average			3912.29444	3497.02524	3840.50806	3867.53633
	min			3604.01600	2653.71900	3607.31200	3605.61000

Table 5.27 presents the results for the Symmetric instances. We can observe similar times than those found for the Semi-Symmetric instances, that as we may remember, both types of instances are symmetric but the Semi-Symmetric differ in that the optimal solution may not correspond to the symmetric solution because the distances between points are not Euclidean for all the points as it is for the Symmetric instances. Only the Hyp-LS algorithm reached the time limit for the largest size instance of 1500_30. In general we can observe small times for all the types of instances, and only for the largest size average times of 2000 to 3000 seconds. The heuristic that resulted with smaller times is the 1-S, requiring less than 1550 seconds for the instance of 1500_30 which is less than half of an hour, which is very efficient for a strategic activity.

Table 5.27 Computational time for the Symmetric instances.

SIZE	AVERAGE TIMES	SYMMETRIC					
		CPLEX	**2-S**	**K-S/P**	**1-S**	**HypLS**	**2-IterLS**
50_5	**max**	621.96900	3.96900	11.37600	7.12600	10.09400	15.32600
	average	84.94474	2.89839	9.53502	6.14987	8.66322	12.43248
	min	6.15700	2.04600	6.51500	4.98600	6.31300	8.26300
200_10	**max**	3605.53000	258.90700	72.32800	50.61000	65.17200	98.34400
	average	2201.91593	160.31030	60.00980	45.04163	56.56943	78.36443
	min	0.00000	77.71900	47.82700	37.32700	46.98500	62.71800
450_15	**max**		2413.54400	72.32800	50.61000	65.17200	98.34400
	average		1779.66294	60.00980	45.04163	56.56943	78.36443
	min		1122.68700	47.82700	37.32700	46.98500	62.71800
1000_20	**max**			660.76600	379.21800	533.70300	770.64000
	average			561.68274	317.43774	488.34600	620.73741
	min			447.03100	194.78100	418.37500	460.23600
1500_30	**max**			2855.84400	1523.92200	4097.26500	2917.28000
	average			2224.10326	982.32620	2873.36076	2430.00152
	min			1506.00000	546.99900	1778.57700	1582.31200

Appendix V presents an analysis of the computational time required for the different types and sizes of instances. Also we present the corresponding fitted curves to compare them with respect to the complexity analysis that was performed in Section 4.2. Results from this analysis indicate that a cubic model best fits to the average and worst cases computational times. This indicates that in practice the complexity of the heuristics is actually better than the estimated worst case complexity.

5.4 Analysis of the heuristics' performance.

Appendix VI presents a statistical analysis to compare the results found by the heuristics. For this we performed a Wilcoxon signed test on the differences between the solutions found by the methods. The results show that the 2-IterLS algorithm found the best solutions for all of the five instance types solved.

5.5 Analysis of the methods to select the set of seeds.

We analyzed with more detail the impact that the seed methods have on the quality of the final solution found by each algorithm and also the impact that the methods have in the ability to construct a feasible solution. In Appendix VII we present this analysis. In Section VII.A we determine the percentage of final solutions that correspond to each of the seed methods. The results show that the method with the largest percentage is the Workload method, with 33.42%. and the Semi-Random method resulted with the lowest percentage of 6.29%.

In Section VII.B we also explore the impact of the seed method on the quality of the final solution reported by each heuristic, and also the percentages in which a feasible solution was constructed by each of the seed methods. We performed a Two-Way ANOVA analysis for a set of the instances solved, considering two instances (one for each capacity level) of the medium size instances of 450_15 for each type of instance.

The results indicate that the seed method has a strong impact and is significant for the quality of the solution reported, and also in the ability to construct a feasible solution. The type of instance is significant, since results vary according to the instance type and also according to the capacity restrictiveness of the instance. We also observed that in general the 2-S or 2-IterLS resulted with the best solutions. Interaction between the Seed method and the heuristic is observed in most of the instances analyzed. We can conclude that including different seed methods as part of the procedure contributes to make a more robust method to different characteristics of instances that may be solved.

Chapter 6

Conclusions and Recommendations for Future Research

In this chapter we present some conclusions in Section 6.1 from the research that was performed and the numerical results found. We also present some research extensions of this work that are propose for future research in section 6.2.

6.1 Conclusions

In conclusion, we were able to obtain a mathematical formulation of the problem as well as a solution methodology to solve the problem. Difficulty of the problem motivated us to employ a heuristic procedure. Solution methodology was tested with data sets with different characteristics.

The mathematical formulation of the logistics districting problem accounts for the requirements of a parcel company that serves a determined region. The model proposed aims to balance the workload content among the districts and to form districts of compact shape, for which a hierarchical optimization model is proposed. We propose also some variants for the workload content metric used in the optimization model, that were not tested during the experimentation.

The solution methodology includes five heuristics. All of them consist are hybrid algorithms that combine elements of the metaheuristics GRASP and Tabu Search, and have in common the procedure to construct the initial feasible solution and differ from each other in the neighborhood structure.

We generated a reasonable number of instances of a wide variety of types and sizes, including an instance generated by data from a parcel company. We solved the instances with the heuristics and with CPLEX. However, CPLEX could only find integer solutions for the small size instances. For most of the instances of 50_5 size, CPLEX found the optimal solution. For the instances of 200_10, CPLEX only reported an integer solution because it reached the time limit of 2 hours. In section 5.3 we present the gaps for the heuristics with respect to CPLEX.

The results show small gaps for the heuristics with respect to the optimal solutions found by CPLEX, and all the heuristics found better solutions than those reported by CPLEX for the 200_10 instances. For the symmetric instances, we compared the heuristic solutions with respect to the symmetric optimal solution. For some instances, the heuristics found the optimal solution, including some cases in which the instance was solved with tight capacity limits. For the largest size instance of 1500_30, the heuristics present an average gap between 8.33% and 13.73%, which is reasonable even though the maximum gaps are larger.

Low computational times were observed for all the heuristics except for the 2-S heuristic, which could solve only the small and medium size instances in a reasonable time. The remainder of the heuristics reached the 3600 second time limit only when solving the largest size instances, and only for certain instance types. For the symmetric and semi-symmetric instances, only some of the instances solved by the HypLS heuristic reached the limit, and the rest of the heuristics had a computational time of less than 2800 seconds. The urban instances were the most time and all the heuristics reached the time limit when the 1500_30 instances were solved. The exception was the 1-S that had an average solution time of less than 3500 seconds, but a maximum computational time of almost 4100 seconds. This indicates that for some instances it reached the time limit. Overall, a maximum time of 5300 seconds, which is less than one hour and a half, was reported. This solution time was for an urban instance of 1500_30 size. These are good solution time for a strategic level problem.

For the instances generated from the data from the parcel company, we observed that the solutions found by the heuristics are better than the current design, but current solution presented less dispersion in the workload content among the districts, however the solution is worst with respect to the compactness. We believe that this is mainly because the company performs a sweep procedure to design the districts, without considering the urban structure of the service region. We can also observe that computational times for the parcel instance are very good, and all the heuristics except the 2-S, had computational times of less than 4100 seconds, and for the 1-S and the combined algorithms the maximum computational time was always less than 30 minutes. Compared to the time that the company requires for the redesign of their districting configuration (around 3 weeks), the heuristics are fast and produce solutions of good quality.

We also solved this instance by the 2-S algorithm, which consumed a lot of time requiring to construct and improve a single solution almost 10,000 seconds, which is almost 3 hours. As we have previously mentioned, given that the problem is a strategic type, it may be reasonable that the heuristic takes this time. However, since the other procedure proposed required much less time to find the solution and the quality of the solutions is similar and even in some cases better, the 2-S algorithm did not turn out to be a good method for this problem.

A statistical analysis to test which of the heuristics found better solutions was performed. From this analysis we conclude that the 2-IterLS is the best. The 1-S had the lowest times, but the solutions reported by the 2-IterLS are better.

Regarding the seeds methods proposed for the construction phase, we observed that the method from which the biggest proportion of best solutions reported was the Workload method, and the smallest proportion corresponds to the Semi-Random method. We performed a statistical analysis and the results indicate that the seed method impacts the quality of the solution and there is interaction with the local search heuristic employed. Also, we observe that the results differ according to the instance type and restrictiveness of the capacity limits. Hence, we can conclude that including different seed methods makes the algorithm more robust to different characteristics of the instances solved.

6.2 Recommendations for Future Research

For future research we propose to formulate a stochastic version of the problem and analyze different demand scenarios. The problem may also be solved as a bi-objective optimization problem to find the efficient frontier instead of a single solution. We also propose to analyze different metrics of the workload content of a district and its effect on the performance of the heuristics, such as the closest point and centroid line hauls metrics proposed here that were not explored during the numerical experimentation.

Another extension is to propose a decomposition approach in which the sub problems consist of defining each of the districts and the master problem selects a set of districts so that all the points are allocated to a district. We may also try to find a better mathematical formulation for the problem that may allow CPLEX to solve larger instances. A stronger lower bound could also be developed. A Reactive GRASP could be tested. The RCL size could be determined by value instead of cardinality.

References

[1] Altman, M., Mc Donald K., McDonald G. (2005) “From Crayons to Computers: The Evolution of Computers use in redistricting”. *Social Science Computer Review*, 23: 334-346.

[2] Altman M., (1997) "Is Automation the Answer: The Computational Complexity of Automated Redistricting," *Rutgers Computer and Law Technology Journal*. 23(1): 81-142.

[3] Altman, M. (1998) “Districting Principles and Democratic Representation”. *Doctoral Thesis*, California Institute of Technology.

[4] Baker, J.R., Clayton, E.R. and Moore L.J. (1989) “Redesign of primary response areas for county ambulance services”. *European Journal of Operational Research.* 41: 23-32.

[5] Beasley, J.E. (1984) “Fixed Routes”. *The Journal of the Operational Research Society*. 35(1): 49-55.

[6] Beardwood, J., Halton J.H., Hammersley J.M. (1959) “The shortest path through many points”. *Proceedings Cambridge Philosophy Society*, 55: 299-328.

[7] Beullens, P, Muyldermans, L. , Cattrysse D., Van Oudheusden D. (2003) “A guided local search heuristic for the capacitated arc routing problem”. *European Journal of Operational Research*, 148: 629-643.

[8] Blais, M., Lapierre, S.D. and Laporte G. (2003) “Solving a home-care districting problem in an urban setting”. *Journal of the Operational Research Society,* 54(11):1141–1147.

[9] Blumenfeld (1985) “Use of continuous space modeling to estimate freight distribution cost”. *Transportation Research A*, 19A: 173-187.

[10] Bodily S. (1978) “Police sector design incorporating preferences of interest groups for equality and efficiency” *Management Science*, 24: 1301-1313.

[11] Bong, C.W., Chai, W.Y, Wong, C.W. (2002) “State-of-the-Art Multiobjective Metaheuristic for Redistricting” *IEEE*, 2: 763-769.

[12] Caballero-Hernández S.I., Ríos-Mercado R.Z., López F. and Schaeffer S.E. (2007). “Empirical Evaluation of a Metaheuristic for a Commercial Territory Design with Joint

Assignment Constraints". *Proceedings of the 12th Annual International Conference on Industrial Engineering, Theory, Applications and Practice (IJIE'07*), 422-427, Cancún, México.

[13] Caro F., Shirabe, T., Guignard, M. and Weintraub A. (2004) "School redistricting: embedding GIS tools with integer programming". *Journal of the Operational Research Society*, 55(8):836–849.

[14] Clarke, G., Wright J.W. (1964) "Scheduling of Vehicles from a Central Depot to a number of Delivery Points" *Operations Research*, 12: 568-581.

[15] Christofides and Eilon (1969) "Expected distances in distribution problems" *Operations Research Quart.,* 20: 437-443.

[16] Church R.L. and Murray A.T. (1993) "Modeling school utilization and consolidation". *Journal of Urban Planning and Development*, 119 (1): 23–38.

[17] Daganzo, C.F. (1984a) "The length of tours in zones of different shape". *Transportation Research B*, 18: 135-146

[18] Daganzo, C.F. (1984b) "The distance traveled to visit N points with a maximum of C stops per vehicle: An analytic model and an application" *Transportation Science*, 18: 331-350.

[19] Daganzo, C.F. (1987a) "Modeling distribution problems with time windows: Part I." *Transportation Science*, 21: 171-179.

[20] Daganzo, C.F. (1987b) "Modeling distribution problems with time windows: Part II: two customer types." *Transportation Science*, 21: 180-187.

[21] Daganzo, C.F. (1991) "Logistics Systems Analysis". *Lecture Notes in Economics and Mathematical Systems #361*. Springer-Verlag. Heidelberg, Germany

[22] D'Amico, S.J., Wang S.-J., Batta, R. and Rump, C.M. (2002) "A simulated annealing approach to police district design" *Computers and Operations Research*, 29 (6): 667-684.

[23] Deckro R. (1970-1977) "Multiple Objective Districting: A general Heuristic Approach using multicreria". *Operational Research Quarterly*. 28(4 Part 2): 953-961.

[24] Diamond, J.T. and Wright, J.R. (1987) "Multiobjective analysis of public-school consolidation". *Journal of Urban Planning and Development*, 113(1):1–18.

[25] Drexl A. and Haase K. (1999) "Fast approximation methods for sales force deployment".

Management Science, 45(10): 1307-1323.

[26] Elizondo R., Boyd E.A., and Beauregard, M. "Evaluating school facility capacity and attendance boundaries using a large-scale assignment algorithm". *Economics of Education Review*, 16(2):155–161, 1997.

[27] Erkut, E., Neuman S. (1991) "Comparison of four models for dispersing facilities" *INFOR*, 29(2): 68-86.

[28] Fernández E., Kalcsics J., Nickel S., Ríos-Mercado R. (forthcoming) "A Novel Maximum Dispersion Territory Design Model arising in the implementation of the WEEE-Directive". *Journal of the Operational Research Society*.

[29] Feo T.A. and Resende M.G.C. "Greedy randomized adaptive search procedures". *J. Of Global Optimization*, 6: 109-133, 1995.

[30] Fisher M.L. and Jaikumar R. (1981) *"A generalized Assignment Heuristic for Vehicle Routing"* Networks, 11: 109-124.

[31] Fleischmann B. and Paraschis J.N. (1988) "Solving a large-scale districting problem- a case- report". *Computers and Operations Research*, 15(6): 521-533.

[32] Floyd, Robert W. (1962) *"Algorithm 97: Shortest Path"*. Communications of the ACM 5 (6): 345.

[33] Galvao L.C., Novaes A.G. de Cursi J.E.S, Souza J.C. (2006) "A multiplicatively-weighted Voronoi diagram approach to logistics districting". *Computers & Operational Research*, 33: 93-114.

[34] Garey, M.R. and Johnson D.S. (1979) "Computers and Intractability: A guide to the Theory of NP-Completeness". W.H. Freeman and Company, New York, EUA.

[35] Garfkinkel R., Nemhauser G. (1970) "Optimal Political Districting by implicit enumeration techniques". *Management Science* , 16(8): 495-408

[36] Gendreau M., Guertin F., Potvin, J.-Y and Taillard E. (1999) "Parallel tabu search for real-time vehicle routing and dispatching" *Transportation Science*, 33: 381-390.

[37] Gendreau M., Guertin F., Potvin, J.-Y. (2006). "Neighborhood search heuristics for a dynamic vehicle dispatching problem with pick-ups and deliveries". *Transportation Research Part C*. 14: 157-174.

[38] Glover, F. (1986) "Future Paths for Integer Programming and Links to Artificial Intelligence", *Computers and Operations Research*, 5, 533-549.

[39] Glover F. and Laguna M. “Tabu Search”. Kluwer Academic Publishers, 1997.

[40] Glover F., and Laguna M., “Tabu Search”, *Modern Heuristic Techniques for Combinatorial Problems*, C.R. Reeves, ed., Blackwell Scientific Publishing, 71-140, 1993.

[41] Golden, B.L, Wong R.T. (1981) “Capacitated Arc Routing Problems”. *Networks*, 11: 305-315.

[42] Grilli di Cortona,P., Manzi, C., Pennisi, A. , Ricca, F. and Simeone, B. (1999) “Evaluation and optimization of electoral systems”. *SIAM: Monographs on Discrete Mathematics and Applications*, Philadelphia.

[43] Han A.F.W., Daganzo, C.F. (1984) “Distributing nonstorable items without transshipment”. *Transportation Research Record*, 1061: 32-41.

[44] Hardy, W. (1980) “Vehicle Routing Efficiency: A comparison of Districting Analysis and the Clarke-Wright Method”. *American Journal of Agricultural Economics*, 62(3): 534-536.

[45] Haugland, D. Ho, S.C, Laporte G. (2007) “Designing delivery districts for the vehicle routing problem with stochastic demands”, *European Journal of Operational Research*, 180(3): 997-1010.

[46] Hall R.W. (1986) “Discrete models/continuous models”. *Omega International Journal of Management Science*. 14: 213-220.

[47] Hess S.W., Weaver J.B., Siegfeldt H.J., Whelan J.N and Zitlau P.A. (1965) “Nonpartisan political redistricting by computer”. *Operations Research*, 13:998-1008.

[48] Hojati M. (1996) “Optimal political districting”. *Computers and Operations Research*, 23: 1147-1161.

[49] Howick R.S. and Pidd M. (1990). “Sales force deployment models”. *European Journal of Operational Research*, 48(3):295-310.

[50] Hill, A.V., Benton W.C. (1992) “Modelling intra-city time-dependent travel speeds for vehicle scheduling problems”. *Journal of Operational Research Society*. 43(4): 343-351.

[51] Kalcsiscs, J., Nickel, S. and Shröder M. (2005) “Toward a unified territorial design approach: Applications, algorithms and GIS integration” *Top*, 13(1):1-74.

[52] Jalliet (1988) “A priori solution of a traveling salesman problem in which a random subset of the customers are visited” *Operations Research*, 36: 929-936.

[53] Keeney, R.L. (1972) "A method for districting among facilities" *Operations Research*. 20: 613-618.

[54] Kernighan, B. W., and Lin S. (1970) "An efficient heuristic procedure for partitioning graphs", *The Bell System Technical Journal*, 49, 291-307.

[55] Langevin A, Soumis F. (1989) "Design of multiple-vehicle delivery tours satisfying time constraints". *Transportation Research-B*, 23B(2): 123-138.

[56] Langevin A., Mbaraga P. (1996) "Continuous approximation models in freight distribution: an overview". *Transportation Research-B*. 30(3): 163-188.

[57] Mehrotra A., Johnson E., Nemhauser G. (1998) "An optimization based heuristic for political districting". *Management Science*, 44(8): 1100-1114.

[58] Moonen M**.** (2004) "Patrol deployment, districting, and dispatching within the urban police: state of the art." Leuven, Belgium: Centre for Industrial Management, Katholieke Universiteit Leuven, 68, HV 8080 .P2 M77.

[59] Morrill R.L. (1981) "Political redistricting and geographic theory". Association of American Geographers, Washington D.C.

[60] Muyldermans L, Cattrysse D, Oudheusden D, Lotan T. (2002). "Districting for salt spreading operations", *European Journal of the Operational Research*, 139(3): 521–532.

[61] Muyldermans, L., Cattrysse D. and Van Oudheusden, D. (2003) "District Design for arc-routing applications". *Journal of Operational Research Society*, 54: 1209-1221.

[62] Newell, G.F. Daganzo C.F. (1986a), "Design of multipl-vehicle tours-I A ring radial network", *Transportation Research B*, 20B(5): 345-363.

[63] Novaes, A.G.N., Graciolli, O.D.(1999) "Designing multi-vehicles delivery tours in a grid-cell format". *European Journal of Operational Research*, 119: 613-634.

[64] Novaes, A.G.N., Souza de Cursi, J.E., Graciolli, O.D. (2000) "A continuous approach to the design of physical distribution systems". *Computers & Operations Research*. 27: 877-893.

[64] Pierskalla, W.P. and Brailer, D.J. (1994) "Operations Research and the Public Sector", *Handbooks in Operations Research and Management Science, chapter Applications of operations research in health care delivery*, 6: 469–498. Elsevier Science B.V., North-Holland, Amsterdam.

[65] Ríos-Mercado R.Z., Fernández E. (2009) *"A Reactive GRASP for a Commercial*

Territory Design Problem with Multiple Balancing Requirements", Computers and Operations Research, 36(3): 755-766, 2009.

[66] Rosenfield D.B. Engelstein I. Feigenbaum D. (1992) "An application of sizing service territories". *European Journal of Operations Research*. 63 (2): 164-172.

[67] Tavares-Pereira, F. Fiegueira J.R., Mousseau, V., Roy, B. (2007b) "Comparing two territory partitions in districting problems: Indices and practical issues" *Socio-Economic Planning Sciences*. In Press, Corrected Proof.

[68] Young, H.P. (1988), "Measuring the compactness of Legislative Districts". *Legislative Studies Quarterly*, 13(1): 105-115.

[69] Van Oudheusden D, Cattrysse D, Lotan T. (1999) "On the importance of districting and its potential impact on routing". *Proceedings of 8th World Conference on Transport Research*. Amsterdam: Pergamon; 521-531.

[70] Wong K.F, Beasley J.E. (1984) "Vehicle routing using fixed delivery areas". *OMEGA, International. Journal of Management Science*. 12(6): 591-600.

[71] Enciclopedia de los Municipios en Mexico, 1st edition by INAFED, 2005. Web site: http://www.e-local.gob.mx/wb2/ELOCAL/ELOC_Enciclopedia.

[72] Minitab Release 14.12.0 Web site: www.minitab.com.

[73] Zoltners A.A. (1976). "Integer programming models for sales territory alignment to maximize profit". *Journal of Marketing Research*. 13(4): 426-430.

[74] Zoltners A.A. (1979). "Sales Management: New Developments from Behavioral and Decision Model Research, chapter A unified approach to sales territory alignments" *Marketing Science Institute*, 360-376.

Zoltners A.A. and Sinha (1983). "Sales territory alignment- a review and model" *Management Science*, 29(11): 1237-1256.

APPENDIX I. Literature Review classification

The following table summarizes the related work to logistics districting and the main characteristics and differences of each work with respect to the problem addressed in this dissertation and the solution methodology proposed:

Table I.1: Literature review summary

Literature	**Criteria**		**Division**		**Agglomeration**						**Main characteristics of the problem and solution method differences with respect to the proposed methodology**
								Allocation			
	Single	Multiple	Voronoi	Construction and Sweep Algorithm	Graph partitioning	Clustering	Improvement and local search	Multi-Kernel growth	Transportation problem	Integer programming	
Keeney (1972)	X		X								-A facility is assigned to each district instead of a central depot. -Objective consists of minimizing travelling distance. -No graph formulation of the problem. -Graphical solution method.
Deckro (1977)		X				X					-General problem with multiple objectives. -Clustering technique forming districts that fall between ranges of the criteria in a lexicographic order.
Hardy (1980)	X								X		-A facility is assigned to each district instead of a central depot. -Comparison of the Clark-Wright savings algorithm for VRP and a districting first-routing last approach by an approach based on the transportation problem.
Wong et al. (1984)	X						X				-They address the "Vehicle routing using fixed delivery areas" problem. -The objective is to minimize distance travelled and there is no attempt to balance the workload content among the districts. -Location of the customers is known and only demand varies.

Daganzo (1984b)	X			X		X					-The methodology consists on partitioning a region into zones of nearly rectangular shape elongated toward the source -The number of points is large compared to the capacity of the vehicles.
Newell and Daganzo (1986a)	X			X							-The underlying network of roads is a dense ring-radial network. -Approximation method for the design of multiple-vehicle delivery tours. -The objective is minimize total distance of the tours.
Han and Daganzo (1986)	X			X							-Problem related to perishable items not parcel items. -The objective is to minimize total costs.
Langevin and Soumis (1989)	X			X							-Continuous approximation model and a ring radial grid. -The methodology partitions the region into approximately rectangular delivery zones that are arranged into concentric rings around the depot. -Points are randomly located with a density as a function of the radius.
Rosenfield et al. (1992)	X										- Each district contains a service facility at its center. -Analyze the tradeoff between the variable cost of delivery and the fixed cost of the facilities
Novaes and Graciolli (1999)		X		X							- A rectangular grid structure for the representation of the variables is assumed and a ring-radial model. -Sweep approach.
Novaes et al. (2000)		X				X					-Continuos approach for the previous problem (Novaes and Graciolli 1999).
Muyldermans et al. (2002)		X			X		X	X			-Graph formulation but demand occurs at the edges. - A facility to serve each district is assumed.

Muyldermans et al. (2003)	X				X		X	X		X	-Graph formulation but demand occurs at the edges. - A facility to serve each district is assumed.
Haugland et al. (2005).	X						X				-Two-stage stochastic problem with recourse. -Objective is to minimize the expected travel time of each district. -Propose a tabu search and a multistart local search algorithms, but differs in that we present a hybrid algorithm that combines both metaheuristics.
Galvao et al. (2006)		X	X								-Extended Novaes et al. (2000) to introduce some improvements to the ring-radial model. -Multiplicatively-weighted (MV)-Voronoi diagram approach -Continuous approximation problem.
Tavares-Pereira et al. (2007a)		X					X				-Workload content is measured only by the amount of points assigned to the district, while the proposed methodology includes the line haul distance to the depot, and also distinguishes between the times required by each kind of service demanded.
Novaes et al. (2008)		X	X								- Continuous location-districting model. -Combine a Voronoi diagram with an optimization algorithm.

Proposed methodology		X			X		X				- A central depot is assumed for which the vehicles depart to service the region. -The problem is formulated as a graph. -Two types of workload are considered: pickups and deliveries. -Two objectives are optimized: balanced workload content among the districts and compactness -No ring-radial structure or approximation method is proposed. -A heuristic procedure is proposed based on a multi-start approach and local search.

APPENDIX II. Pseudo code of the seed selection algorithms

The following variables are defined including those previously defined:

RCL=Set of points that are placed in the restricted candidate list.
Neigh(i)= Set of points that are located within a determined distance defined by a *threshold* value from point *i* and then considered to be at the neighborhood of point *i*, $i \in \mathrm{V}$,
Del= Set of discarded points formed by the points at the neighborhood of the seed points,
$seed_j$= Point that is selected as a seed for district *j*, $j \in J$,
Sdist= Distance among the two first seeds selected,
$angle_i$=Angle of point *i* measured in degrees, $i \in \mathrm{V}$,
$work_i$= Number of pickups and deliveries assigned to the point *i*, $i \in \mathrm{V}$.

The pseudocode of the algorithm for each method are as follows:

P-dispersion algorithm

```
Set $RCL = \{\phi\}$,
for $k$=1,…,|$RCL$| do
        Set $\underset{i \in V-RCL}{\mathrm{argmax}}\{d_{i0}\} \in RCL$
Select $seed_1$ randomly from $RCL$,
Set $V'=V-\{seed_1\}$ and $RCL = \{\phi\}$,
for $k$=1,…, |$RCL$| do
        Set $\underset{i \in V'-RCL}{\mathrm{argmax}}\{d_{i,seed_1}\} \in RCL$
Select $seed_1$ randomly from $RCL$,
Set $V'=V'-\{seed_2\}$, $RCL = \{\phi\}$ and $Sdist= d_{seed1,seed2}$,
for $j$=3,…,$m$ do
        for $k$=1,…, |$RCL$| do
                Set $\underset{i \in V'-RCL}{\mathrm{argmax}}\{Sdist + \sum_{i \in V'}\sum_{k=1}^{j-1} d_{i,seed_k}\} \in RCL$,
        Select $seed_j$ randomly from $RCL$,
        Set $V'=V'-\{seed_j\}$, $RCL = \{\phi\}$ and $Sdist = Sdist + \sum_{k=1}^{j-1} d_{seed_j,seed_k}$
```

Semi-Random algorithm

```
Set $Del = \{\phi\}, V' = V$,
for $j$=1,…,$m$ do
    if $|V' - Del| \neq \{\phi\}$ then
        Select $seed_j$ randomly from $V'$-$Del$
        $Set\ \ Del = Del \bigcup \{i \in V' - Del \mid d_{i,seed_j} \leq threshold\}$
    else
        Select $seed_j$ randomly from $Del$
```

Neighborhood algorithm

```
Set $RCL = \{\phi\}$, $Del = \{\phi\}$ and $V' = V$,
for $j$=1,…,$m$ do
    if $|V'| \neq \{\phi\}$ then
        for $i$=1,…,$V'$ do
            Set $Neigh(i)$= $\{ k \in V', k \neq i \mid d_{ik} \leq threshold\}$
        for $k$=1,…, $|RCL|$ do
            Set $\underset{i \in V' - RCL}{\text{argmax}}\{|Neigh(i)|\} \in RCL$,
        Select $seed_j$ randomly from $RCL$,
        Set $Del = \{k \mid k \in Neigh(seed_j)\}$
        Set $V'$=$V'$-$\{seed_1\}$-$Del$ and $RCL = \{\phi\}$,
    Else
        Select $seed_j$ randomly from $Del$
        Set $Del = Del - \{seed_j\}$,
```

Angle algorithm

```
Define $V' = [v \in V \mid angle_i \leq angle_{i+1}, i \geq 1]$,
Set $\overline{A} = \dfrac{\underset{i \in V}{Max}\{angle_i\}}{m}$
Select $seed_1$ randomly from $V'$,
Set $RCL = \{\phi\}$, $seed_1 \notin V'$ and $A = \overline{A} + \text{angle}_{seed_1}$
for $j$=2,…,$m$ do
    if $A > \underset{i \in V}{\text{Max}}\{angle_i\}$ then
        Set $A = A - \underset{i \in V}{\text{Max}}\{\text{angle}_i\} + \underset{i \in V}{\text{Min}}\{\text{angle}_i\}$
    Set $a = \text{Min}\{i \in V' \mid \text{angle}_i \geq A\} \in RCL$,
    for $k$=2,…,( $|RCL|$-1)/2 do
        Set $\text{Min}\{k \in V' - RCL \mid \text{angle}_k \geq \text{angle}_a\} \in RCL$
        Set $\text{Max}\{k \in V'-RCL \mid \text{angle}_k \le \text{angle}_a\} \in RCL$
    Select $seed_j$ randomly from $RCL$,
    Set $seed_j \notin V'$, $RCL = \{\phi\}$ and $A = \overline{A} + \text{angle}_{seed_j}$,
```

Workload algorithm

```
Define $V' = [v \in V \mid work_i \le work_{i+1}, i \ge 1]$,
Compute $work_i = wp_i + wd_i \quad \forall i \in V$ and $\overline{W} = \frac{\sum_{i \in V} work_i}{m}$
Set $RCL = \{\phi\}$ and $p$=0,
for $j$=1,…,$m$ do
    Set $a = \text{Min}\{i \in V' \mid \sum_{k=p+1}^{i} work_k \ge \overline{W}\} \in RCL$,
    for $k$=2,…,( |$RCL$|-1)/2 do
        Set $\text{Min}\{k \in V'-RCL \mid work_k \ge work_a\} \in RCL$
        Set $\text{Max}\{k \in V'-RCL \mid work_k \ge work_a\} \in RCL$
    Select $seed_j$ randomly from $RCL$,
    Set $seed_j \notin V'$, $RCL = \{\phi\}$
    Set $p$ to the value of the position of the seed in set $V'$.
```

APPENDIX III. Pseudo code of the low level heuristics for the H-F.

The following parameter is defined:

bigM=a very big value,

And the following variables are defined in addition to previously defined:

Pcap$_j$= remaining capacity of district *j* in terms of pickups, $j \in J$,

Dcap$_j$= remaining capacity of district *j* in terms of deliveries, $j \in J$,

Best_sol= Value of the *Sum_neg* metric that corresponds to the current best solution known,

BDC=Best districting configuration that corresponds to *Best_sol*,

$$none = \begin{cases} 1 & \text{if a move to be performed has been for current iteration} \\ 0 & \text{otherwise} \end{cases},$$

Ibest= best value of the *Sum_neg* metric over a set of infeasible moves evaluated,

Ipoint= corresponding point of the move that leads to *Ibest,*

Isending= corresponding district that sends the *Ipoint* that leads to *Ibest,*

Ireceiving= corresponding district that receives the *Ipoint* that leads to *Ibest,*

$$adj(i,j) = \begin{cases} 1 & \text{if point } i \in V, X_{ij'} = 1 \text{ and } \exists \; e_{ik} \in E \text{ such that } X_{kj} = 1, j' \neq j; \quad j, j' \in J; \; k \in V \\ 0 & \text{otherwise} \end{cases},$$

$$border_i = \begin{cases} 1 & \text{if point } i \in V \text{ and } \exists \; j \in J \text{ such that } adj(i,j) = 1 \\ 0 & \text{otherwise} \end{cases},$$

The pseudocodes of the algorithms are as follows:

Hyperheuristic-Feasibility

```
Compute $Infeasibility^S$ and $W\_dispersion$ metrics of the current solution
for the type of service $S$, currently attempted to achieve feasibility; $S=P,D$.
Set $Best\_sol = Infeasibility^S$ and establish current districting configuration
as $BDC$,
Determine $Pcap_j, Dcap_j$ and $work_j$ $\forall j \in J$,
Initialize $Tabu(i,j) = -tperm$ and compute $adj(i,j)$ $\forall i \in V, j \in J$,
Do
    Select a low level heuristic randomly among:
        -FMM    -IMM
        -FML    -IML
        -FLM    -ILM
    Determine randomly the iterations (one or steepest descent),
    Apply the low level heuristic according to the iterations selected,
Until one of the stopping conditions is met,
return $Best\_sol$ and $BDC$.
```

FMM heuristic algorithm

```
Set $iter=0$ and $tperm'=tperm$,
Do
    Set $J'=J$, $iter=iter+1$ and $none=0$,
    if module($iter$,$Titer$)=0 then

        $tperm'=2 \cdot tperm'$,
    while $none=0$ do
        Set $sending = \underset{j \in J'}{\operatorname{argmax}}\{work_j\}$, $J' = J' - \{sending\}$,
        Set $Q = \{i \in sending \mid i \text{ is a } border\ point\}$,
        while $Q \neq \{\phi\}$ and $none=0$
            Select $i \in Q$ and set $Q=Q-\{i\}$ and
            $K = \{j \in J - \{sending\} \mid adj(i,j)=1,\ wp_i \le Pcap_j \text{ and } wd_i \le Dcap_j\}$,
            while $none=0$ and $K \neq \{\phi\}$
                Set $receiving = \underset{k \in K}{\operatorname{argmax}}\{work_k\}$,
                if $Tabu(i,receiving)+tperm' < iter$ then
                    MOVE($i$,$sending$,$receiving$),
                else
                    ASPIRATION($i$,$sending$,$receiving$),
                    Set $K=K-\{receiving\}$,
Until one of the stopping conditions is met.
```

FML heuristic algorithm

```
Set $iter$=0 and $tperm'$=$tperm$,
Do
    Set $J'=J$, $iter=iter+1$ and $none$=0,
    if module($iter$, $Titer$)=0 then
        $tperm'=2 \cdot tperm'$,
    while $none$=0 do
        Set $sending = \operatorname{argmax}_{j \in J'}\{work_j\}$, $J' = J' - \{sending\}$,
        Set $Q = \{i \in sending \mid i \text{ is a } border\ point\}$,
        while $Q \neq \{\phi\}$ and $none$=0
            Select $i \in Q$ and set $Q=Q-\{i\}$ and
            $K = \{j \in J - \{sending\} \mid adj(i,j) = 1,\ wp_i \leq Pcap_j \text{ and } wd_i \leq Dcap_j\}$,
            while $none$=0 and $K \neq \{\phi\}$
                Set $receiving = \operatorname{argmin}_{k \in K}\{work_k\}$,
                if $Tabu(i,receiving)+tperm' < iter$ then
                    MOVE($i$,$sending$,$receiving$),
                else
                    ASPIRATION($i$, $sending$,$receiving$),
                    Set $K=K-\{receiving\}$,
Until one of the stopping conditions is met.
```

FLM heuristic algorithm

```
Set $iter$=0 and $tperm'$=$tperm$
Do
    Set $J'=J$, $iter=iter+1$, $none$=0,
    if module($iter$, $Titer$)=0 then
        $tperm'=2 \cdot tperm'$,
    while $none$=0 do
        Set $receiving = \operatorname{argmin}_{j \in J'}\{work_j\}$,
        Set $J' = J' - \{receiving\}$ and $K = J - \{receiving\}$,
        while $none$=0 and $K \neq \{\phi\}$
            Set $sending = \operatorname{argmax}_{k \in K}\{work_k\}$ ,and $K = K - \{sending\}$,
            Set $Q = \{i \in sending \mid adj(i,receiving) = 1,\ wp_i \leq Pcap_{receiving}$
                    and $wd_i \leq Dcap_{receiving}\}$,
            while $none$=0 and $Q \neq \{\phi\}$
                Select $i \in Q$ and set $Q=Q-\{i\}$,
                if $Tabu(i,receiving)+tperm' < iter$ then
                    MOVE($i$,$sending$,$receiving$),
                else
                    ASPIRATION($i$,$sending$,$receiving$),
Until one of the stopping conditions is met.
```

IMM heuristic algorithm

```
Set $iter$=0 and $tperm'$=$tperm$,
Do
    Set $iter$=$iter$+1, $none$=0, $Ibest$= $bigM$, $Ipoint$=0, $Isending$=0 and $Ireceiving$=0,
    if module($iter$, $Titer$)=0 then
        $tperm'$=2·$tperm'$,
    Set $sending = \underset{j \in J}{\mathrm{argmax}}\{work_j\}$,
    Set $Q = \{i \in sending \mid i \text{ is a } border\ point\}$,
    while $Q \neq \{\phi\}$
        Select $i \in Q$ and set $Q$=$Q$-$\{i\}$,
        Set $K = \{j \in J - \{sending\} \mid adj(i,j) = 1\}$,
        while $K \neq \{\phi\}$
            Set $receiving = \underset{k \in K}{\mathrm{argmax}}\{work_k\}$,
            if $wp_i \leq Pcap_{receiving}$ and $wd_i \leq Dcap_{receiving}$ then
                if $Tabu(i,receiving)+tperm' < iter$ then
                    MOVE($i$,$sending$,$receiving$),
                    Set $K = \{\phi\}$ and $Q = \{\phi\}$,
                else
                    ASPIRATION($i$,$sending$,$receiving$),
                    Set $K$=$K$-$\{receiving\}$,
            else
                INFEASIBLE($i$,$sending$,$receiving$,$Ibest$),
                Set $K$=$K$-$\{receiving\}$,
    if $none$=0
        IMOVE($Ibest$, $Ipoint$,$Isending$, $Ireceiving$),
Until one of the stopping conditions is met
```

IML heuristic algorithm

```
Set $iter$=0 and $tperm'$=$tperm$,
Do
    Set $iter$=$iter$+1, $none$=0, $Ibest$= $bigM$, $Ipoint$=0, $Imoving$=0 and $Ireceiving$=0,
    if module($iter$, $Titer$)=0 then
        $tperm'$=2·$tperm'$,
    Set $moving = \underset{j \in J}{\mathrm{argmax}}\{work_j\}$,
        Set $Q = \{i \in sending \mid i \text{ is a } border\ point\}$,
        while $Q \neq \{\phi\}$
            Select $i \in Q$ and set $Q$=$Q$-$\{i\}$,
            Set $K = \{j \in J - \{sending\} \mid adj(i,j) = 1\}$,
            while $K \neq \{\phi\}$
                Set $receiving = \operatorname{argmin}_{k \in K}\{work_k\}$ and $K$-$\{receiving\}$,
                if $wp_i \leq Pcap_{receiving}$ and $wd_i \leq Dcap_{receiving}$ then
                    if $Tabu(i,receiving)+tperm' < iter$ then
                        MOVE($i$,$sending$,$receiving$),
                        Set $K = \{\phi\}$ and $Q = \{\phi\}$,
                    else
                        ASPIRATION($i$,$sending$,$receiving$),
                else
                    INFEASIBLE($i$,$sending$,$receiving$,$Ibest$),

        if $none$=0
            IMOVE($Ibest$, $Ipoint$,$Isending$, $Ireceiving$),
    Until one of the stopping conditions is met.
```

ILM heuristic algorithm

```
Set $iter$=0 and $tperm'$=$tperm$,
Do
    Set $iter$=$iter$+1, $none$=0, $Ibest$=$bigM$, $Ipoint$=0, $Imoving$=0 and $Ireceiving$=0,
    if module($iter$, $Titer$)=0 then
        $tperm'$=2·$tperm'$,
    Set $receiving = \operatorname{argmin}_{j \in J}\{work_j\}$ and $K = J - \{receiving\}$,
    while $none$=0 and $K \neq \{\phi\}$
        Set $sending = \operatorname{argmax}_{k \in K}\{work_k\}$ and $K = K - \{sending\}$,
        Set $Q = \{i \in sending \mid adj(i, receiving) = 1\}$,
        while $Q \neq \{\phi\}$
            Select $i \in Q$ and set $Q$=$Q$-$\{i\}$,
            if $wp_i \leq Pcap_{receiving}$ and $wd_i \leq Dcap_{receiving}$ then
                if $Tabu(i,receiving)+tperm' < iter$ then
                    MOVE($i$,$sending$,$receiving$),
                    Set $K = \{\phi\}$ and $Q = \{\phi\}$,
                else
                    ASPIRATION($i$,$sending$,$receiving$),
            else
                INFEASIBLE($i$,$sending$,$receiving$,$Ibest$),
    if $none$=0
```

```
          IMOVE(Ipoint,Isending, Ireceiving),
Until one of the stopping conditions is met.
```

```
MOVE(i,sending,receiving)
Set $i \in receiving$ and $i \notin sending$ and $none$=1,
Update:
    Workload capacities: $Pcapj$, $Dcapj$ and $work_j$, $j$=sending, receiving,
    Feasibility metrics and improvement: $Infeasibility^s$, $W\_dispersion$,
    $Impr(Infeasibility^s)$ and $Impr(W\_dispersion)$,
    $adj$ and $Tabu$ matrixes,
if $Infeasibility^s < Best\_sol$ then
    Set $Best\_sol = Infeasibility^s$ and $BDC$= current districting configuration.
```

```
ASPIRATION(i,sending,receiving)
Set $i \in receiving$ and $i \notin sending$ and compute $Infeasibility^{s}{'}$,
if $Infeasibility^{s}{'} < Best\_sol$ then
    Set $Best\_sol = Infeasibility^{s}{'}$ and BDC= current districting configuration,
    Set $none$=1, and $Infeasibility^s = Infeasibility^{s}{'}$,
    Update:
        Workload capacities: $Pcapj$, $Dcapj$ and $work_j$ for $j$=sending, receving,
        Remaining feasibility metric and improvement: $W\_dispersion$,
        $Impr(Infeasibility^s)$ and $Impr(W\_dispersion)$,
        $adj(i,j)$ and $Tabu(i,j)$ matrixes,
else
    Set $i \notin receiving$ and $i \in sending$.
```

```
IMOVE( Ipoint,Isending, Ireceiving)
Set $Ipoint \in Ireceiving$ and $Ipoint \notin Isending$ and $none$=1,
Update:
    Workload capacities: $Pcapj$, $Dcapj$ and $work_j$, $j$=Isending, Ireceiving,
    Feasibility metrics and improvement: $Infeasibility^s$, $W\_dispersion$,
    $Impr(Infeasibility^s)$ and $Impr(W\_dispersion)$,
    $adj(i,j)$ and $Tabu(i,j)$ matrixes,
if $Infeasibility^s < Best\_sol$ then
    Set $Best\_sol = Infeasibility^s$ saving the current districting configuration.
```

```
INFEASIBLE(i,sending,receiving,Ibest)
Set $Infeasibility^{s}{}' = Infeasibility^{s}$, $P'cap_j = Pcap_j$ and $D'cap_j = Dcap_j$ , $j$=sending, receiving,
Set $i \in receiving$ and $i \notin sending$,
Update:
    Temporal workload capacities: $P'capj, D'capj$, $j$=moving, receiving,
    Temporal feasibility metric: $Infeasibility^{s}{}'$,
if $Infeasibility^{s}{}' < Ibest$ then
    Set $Ibest = Infeasibility^{s}{}'$, $Ipoint=I$, $Isending=sending$ and $Ireceiving=receiving$,
Set $i \notin receiving$ and $i \in sending$.
```

APPENDIX IV. Pseudo code of the five hybrid heuristics.

The following variables are defined in addition to previously defined:

Distr$_i$ = District where point *i* is currently assigned,

OFbest1, *OFbest2* and *OFbest3*= Three best known objective function values, where OFbest1 is the overall best solution value.

XBest1, *XBest2*, *XBest3* = corresponding binary values of the three best solutions, where *XBest1* defines the overall best solution.

$adj^{copy}(i,j)$= copy of the current adjacency matrix that is updated according the temporal move performed in the first step so that the second step evaluates moves conditioned on the first step.

candidate=1 if the point under revision is potential to be moved during a second step in order to get a feasible solution given that first step lead to an infeasible solution and only points that may be moved from the district in which capacity is violated are considered; 0 otherwise.

Tot_iter= maximum number of iterations set for a LS algorithm.

The pseudocodes of the LS algorithms are as follows:

1-S_LS(OF, X_{ij})

```
Set:
        iter=0, tperm'=tperm
        OFbest1=μ1, OFbest2=μ2, OFbest3=μ3,
        $XBest1_{ij} = 0,\quad XBest2_{ij} = 0,\quad XBest3_{ij} = 0 \quad \forall i \in V, j \in J,$
        i*=0, imin=0, jmin=0, j2_prev=0,
Do
        FIRST STEP:
        for (i = 1; i ≤ |V|; ++i) {
                iter++,
                if ($border_i$ =1){
                        i*=i;
                        $X_{i*,Distr_{i*}} = 0$,
                        $PCap_{Distr_{i*}} += wp_{i*}$,  $DCap_{Distr_{i*}} += wd_{i*}$
                        for (j = 1; j ≤ |J|; ++j) {
                                if (adj(i,j)=1 & $PCap_j \geq wp_{i*}$ & $DCap_j \geq wd_{i*}$
                                 & Tabu(i*,j)+tperm< iter ){
                                        $X_{i*,j} = 1$,
                $PCap_j -= wp_{i*}$ and $DCap_j -= wd_{i*}$
                O$F = \lambda W + (1-\lambda)Z$
                if (*OF<minOF*){
                        *minOF=OF*,
                        *imin=i\*, jmin=j,*
                        $WDispersion* = \sum_{j \in J} \left| W_j - \overline{W} \right|$
                }
                else if (*OF=minOF*){
                $WDispersion = \sum_{j \in J} \left| W_j - \overline{W} \right|$
                if *(WDispersion<WDispersion\*){*
                        *minOF=OF*,
                        *imin=i\*, jmin=j,*
                        *WDispersion\*=WDispersion,*
        }}}
        ASPIRATION CRITERIA:
        else if *(adj(i,j)=1 &* $PCap_j \geq wp_{i*}$ $\& DCap_j \geq wd_{i*}$ ){
            $X_{i*,j} = 1$,
            $PCap_j -= wp_{i*}$ and $DCap_j -= wd_{i*}$
            O$F = \lambda W + (1-\lambda)Z$
            if (*OF<minOF*){
                    *minOF=OF*,
                    *imin=i\*, jmin=j,*
                    $WDispersion* = \sum_{j \in J} \left| W_j - \overline{W} \right|$
            }
            else if (*OF=minOF*){
                    $WDispersion = \sum_{j \in J} \left| W_j - \overline{W} \right|$
                    if *(WDispersion<WDispersion\*){*
                            *minOF=OF*,
                            *imin=i\*, jmin=j,*
                            *WDispersion\*=WDispersion,*
            }}
            OFBEST(*OF*, $X_{ij}$),
            $X_{i*,j} = 0$ ,
            $PCap_j += wp_{i*}$ and $DCap_j += wd_{i*}$,
        }
        $X_{i*,Distr_{i*}} = 1$,
        $PCap_{Distr_{i*}} -= wp_{i*}$, $DCap_{Distr_{i*}} -= wd_{i*}$
}
```

```
        Perform the best move found:
        if (imin>0){
                $X_{i\min,Distr_{i*}} = 0$, $X_{imin,jmin} = 1$
                $PCap_{Distr_{i*}} += wp_{i*}$, $DCap_{Distr_{i*}} += wd_{i*}$
                        $PCap_{jmin} -= wp_{i*}$, $DCap_{jmin} -= wd_{i*}$
        }
        Update:
                adj(i,j), Tabu(i,j) matrixes and capacity of the districts (Pcap and
                Dcap)

Until one of the stopping conditions is met.

LAST_SEARCH(OFbest1, OFbest2, OFbest3, $XBest1_{ij}$, $XBest2_{ij}$, $XBest3_{ij}$)

Return OFbest1 and $XBest1_{ij}$
```

2-S_LS(OF, X_{ij})

```
Set:
    iter=0, tperm'=tperm
    OFbest1=μ1, OFbest2=μ2, OFbest3=μ3,
    $XBest1_{ij} = 0,\quad XBest2_{ij} = 0,\quad XBest3_{ij} = 0 \quad \forall i \in V, j \in J,$
    i*=0, imin=0, jmin=0, j2_prev=0,
    $adj^{copy}(i,j)$= adj (i,j) $\forall i \in V, j \in J,$
Do
    FIRST STEP:
    for ($i = 1; i \le |V|; ++i$) {
        iter++,
        if ($border_i$ =1){
        i*=i;
        $X_{i*,Distr_{i*}} = 0$,
        $PCap_{Distr_{i*}} += wp_{i*}$, $DCap_{Distr_{i*}} += wd_{i*}$
        for ($j = 1; j \le |J|; ++j$) {
            if (adj(i,j)=1 & Tabu(i*,j)+tperm< iter ){
                $X_{i*,j} = 1$,
                $PCap_j -= wp_{i*}$ and $DCap_j -= wd_{i*}$
            }
            if ($PCap_j \ge wp_{i*}$ & $DCap_j \ge wd_{i*}$)
                $OF^{first} = \lambda W + (1-\lambda)Z$
            else
                $OF^{first} = \infty$
            Update $adj^{copy}(i,j)$ matrix
```

```
SECOND STEP:
if (O$F^{first} \neq \infty$) {
        OFBEST( O$F$ ),
        for ($p$ = 1; $p \leq |V|$; ++$p$) {
                if ($border_i$ =1){
                        $p$*=$p$;
                        $X_{p^*,Distr_{p^*}} = 0$,
                        $PCap_{Distr_{p^*}} += wp_{p^*}$,  $DCap_{Distr_{p^*}} += wd_{p^*}$
                        for ($m$ = 1; $m \leq |J|$; ++$m$) {
                                if $(adj^{copy}(i,j)=1$ & $Tabu(i^*,j)+tperm< iter$
                                & $PCap_j \geq wp_{i^*}$ & D$Cap_j \geq wd_{i^*}$ ){
                                        $X_{p^*,m} = 1$,
                                        $PCap_m -= wp_{p^*}$
                                        $DCap_m -= wd_{p^*}$
                                        O$F$ = $\lambda W + (1-\lambda)Z$
                                        if ($OF$<$minOF$){
                                                $minOF$= O$F$ ,
                                                $imin=i^*$, $jmin=j$,
                                                $i2min=p^*$,$j2min=m$,
                                                $minOF^{first} = OF^{first}$,
                                                $WDispersion^* = \sum_{j \in J} \left| W_j - \overline{W} \right|$

                                        }
                                        OFBEST($OF$, $X_{ij}$),
                                }
                                ASPIRATION CRITERIA:
                                else if $(adj^{copy}(i,j)=1$ & $PCap_j \geq wp_{i^*}$  &
                                $DCap_j \geq wd_{i^*}$ ){
                                        $X_{p^*,m} = 1$,
                                        $PCap_m -= wp_{p^*}$
                                        $DCap_m -= wd_{p^*}$
                                        O$F$ = $\lambda W + (1-\lambda)Z$
                                        if ($OF$<$minOF$){
                                                $minOF$= O$F$ ,
                                                $imin=i^*$, $jmin=j$,
                                                $i2min=p^*$,$j2min=m$,
                                                $minOF^{first} = OF^{first}$,
                                        $WDispersion^* = \sum_{j \in J} \left| W_j - \overline{W} \right|$
                                }
                                else if ($OF=minOF$){
                                        $WDispersion = \sum_{j \in J} \left| W_j - \overline{W} \right|$
                                        if($WDispersion < WDispersion^*$ ) {
                                                $imin=i^*, jmin=j,$
                                                $i2min=p^*, j2min=m,$
                                                $minOF^{first} = OF^{first},$
                                                $WDispersion^* = WDispersion,$
                                        }
                                }
                                OFBEST($OF$, $X_{ij}$),
 }}}}}
SECOND STEP (only moves that lead to a feasible solution)
 else {
        for ($p = 1; p \le |V|; ++p$) {
                $candidate=0,$
                if ($Xpj=1$ & $adj^{copy}(i,j)=1$ & $PCap_j<0$ & $wp_p = 1$ )
                        $candidate$=1,
                else if ($Xpj=1$ & $adj^{copy}(i,j)=1$ & $DCap_j<0$ & $wd_p = 1$ )
                        $candidate$=1,
                if($candidate$=1){
                        for ($m = 1; m \le |J|; ++m$) {
                                if ($adj^{copy}(i,j)=1$ & $Tabu(i^*,j)+tperm < iter$
                                & $PCap_j \ge wp_{i^*}$ & $DCap_j \ge wd_{i^*}$ ){
                                        $X_{p^*,m} = 1,$
                                        $PCap_m -= wp_{p^*}$
                                        $DCap_m -= wd_{p^*}$
                                        $OF = \lambda W + (1-\lambda) Z$
                                        if ($OF<minOF$){
                                                $minOF = OF$ ,
                                                $imin=i^*, jmin=j,$
                                                $i2min=p, j2min=m,$
                                                $minOF^{first} = OF^{first},$
                                                $WDispersion^* = \sum_{j \in J} \left| W_j - \overline{W} \right|$
                                        }
                                        OFBEST($OF$, $X_{ij}$),
                                }
                                ASPIRATION CRITERIA:
```

```
                    else if $(adj^{copy}(i,j)=1$ & $PCap_j \geq wp_{i*}$ & $DCap_j \geq wd_{i*}$ ){
                        $X_{p*,m} = 1$,
                        $PCap_m -= wp_{p*}$
                        $DCap_m -= wd_{p*}$
                        $OF = \lambda W + (1-\lambda)Z$
                        if ($OF<minOF$){
                            $minOF = OF$,
                            $imin=i*$, $jmin=j$,
                            $i2min=p*$,$j2min=m$,
                            $minOF^{first} = OF^{first}$,
                            $WDispersion^* = \sum_{j \in J} |W_j - \overline{W}|$
                        }
                        else if ($OF=minOF$){
                            $WDispersion = \sum_{j \in J} |W_j - \overline{W}|$
                        if ($WDispersion<WDispersion^*$){
                                $imin=i*$,    $jmin=j$,
                                $i2min=p*$,
                                $j2min=m$,
                                $minOF^{first} = OF^{first}$,
                                $WDispersion^*=$
                                $WDispersion$,
                        }}
                        OFBEST($OF$, $X_{ij}$),
}}}}}}
PERFORM THE BEST MOVE (only first step)
if ($imin$>0){
        $X_{i\min,Distr_{i*}} = 0$, $X_{imin,jmin} = 1$
        $PCap_{Distr_{i\min}} += wp_{i\min}$, $DCap_{Distr_{i\min}} += wd_{i\min}$
        $PCap_{jmin} -= wp_{i\min}$, $DCap_{jmin} -= wd_{i\min}$
        OF= $minOF^{first}$,
}
PERFORM ALSO SECOND STEP (since first was infeasible)
if($minOF^{first} = \infty$){
        $X_{i2\min,Distr_{i2\min}} = 0$, $X_{i2\min,j2\min} = 1$
        $PCap_{Distr_{i2\min*}} += wp_{i2nin}$, $DCap_{Distr_{i2\min}} += wd_{i2\min}$
        $PCap_{jmin} -= wp_{i2\min}$, $DCap_{jmin} -= wd_{i2\min}$
        OF= $minOF$,
}
Update:
```

```
            adj(i,j), Tabu(i,j) matrixes and capacity of the districts (Pcap and Dcap)

Until one of the stopping conditions is met.
LAST_SEARCH(OFbest1, OFbest2, OFbest3, $XBest1_{ij}$, $XBest2_{ij}$, $XBest3_{ij}$)
Return OFbest1 and $XBest1_{ij}$
```

```
k-S/P_LS(OF, $X_{ij}$)
Set:
    iter=0, tperm'=tperm
    OFbest1=μ1, OFbest2=μ2, OFbest3=μ3,
    $XBest1_{ij} = 0, \quad XBest2_{ij} = 0, \quad XBest3_{ij} = 0 \quad \forall i \in V, j \in J,$
    $Prob_j = \frac{1}{|J|},$
Do:
    SelectPair($Prob_j$)
    k-S(OF, $X_{ij}$, d1, d2),
    Set:
        OF=OFcop,
        $X_{ij}$=$Xcop_{ij}$   $\forall i \in V, j \in J,$
    Update:
        adj(i,j) matrix, Tabu(i,j) matrixes and capacities of the districts (Pcap
        and Dcap)
Until one of the stopping conditions is met.

Return OFbest1 and $XBest1_{ij}$
```

```
k-S(OF, $X_{ij}$, d1, d2)
Set:
    iter=0, tperm'=tperm
    $Pairs = \{\phi\},$
    OFcop=OF,
    $Xcop_{ij}$=$X_{ij}$ $\forall i \in V, j \in J,$
    $Wcop_j$=$W_j$ $\forall j \in J,$
    n=maximum number of points assigned either to d1 or d2,
    $K = \lfloor n/2 \rfloor,$

    $i, k, i', k' \in Pairs$ if $\exists\ j \neq j';\ j, j' \in J;\quad i \neq k \neq i' \neq k';\ i, k, i', k' \in V$ such that $adj(i, j) = 1$,
$adj(k, j) = 1, adj(i', j') = 1$ and $adj(k', j') = 1$
```

```
        $npairs$ =total number of pairs of adjacent points between $d1$ and d$2$
        $mink$=0,  $minOF = \infty$, $minkOF = \infty$,
for ($k$ = 1; $k \leq K$; ++$k$) {
        FROM $d1$ to $d2$:
                Initialize:
                $OFcop$=$OF$,
                $Xcop_{ij}$=$X_{ij}$ $\forall i \in V, j \in J$,
                Evaluate:
                Moves($OFcop$, $Xcop_{ij}$, $d1$, $d2$, $Wcop_{d1}$,$Wcop_{d2}$)
                Update best solution:
                $mind1$=$imin$,
                $mind2$=$0$,
                $BestminOF$=$minOF$,
                $PWDispersion^* = abs(Wcop_{d1} - Wcop_{d2})$
        FROM $d2$ to $d1$:
                Initialize:
                $OFcop$=$OF$,
                $Xcop_{ij}$=$X_{ij}$ $\forall i \in V, j \in J$,
                Evaluate:
                Moves($OFcop$, $Xcop_{ij}$, $d2$, $d1$, $Wcop_{d1}$, $Wcop_{d2}$)
                Update best solution:
                if ($minOF$< $BestminOF$) {
                        $mind2$=$imin$,
                        $mind1$=$0$,
                        $BestminOF$=$minOF$,
                }
                else if ($minOF$= $BestminOF$) {
                        $PWDispersion = abs(Wcop_{d1} - Wcop_{d2})$
                        if($PWDispersion$<$PWDispersion^*$){
                                $mind2$=$imin$,
                                $mind1$=$0$,
                                $BestminOF$=$minOF$,
                                $PWDispersion^*$=$PWDispersion$
                }}
        INTERCHANGHES:
        if ($Pairs \neq \{\phi\}$) {
                Initialize:
                $OFcop$=$OF$,
                $Xcop_{ij}$=$X_{ij}$ $\forall i \in V, j \in J$,
                Evaluate:
                Exchanges($OFcop$, $Xcop_{ij}$, $Pairs$, $npairs$, $Wcop_{d1}$,$Wcop_{d2}$)
                Update best solution:
                if ($minOF$< $BestminOF$) {
                    $mind1=imin1$,
                    $mind2=imin2$,
                    $BestminOF=minOF$,
            }
            else if ($minOF= BestminOF$) {
                $PWDispersion = abs(Wcop_{d1} - Wcop_{d2})$
                if($PWDispersion<PWDispersion*$){
                    $mind2=imin1$,
                    $mind1=imin2$,
                    $BestminOF=minOF$,
                    $PWDispersion*=PWDispersion$
        }}}
        PERFORM BEST MOVE FOR STEP $k$
        if ($mind1 \neq 0$) {
            $Xcop_{mind1,d1} = 0$,
            $Xcop_{mind1,d2} = 1$,
        }
        if ($mind2 \neq 0$) {
            $Xcop_{mind2,d2} = 0$,
            $Xcop_{mind2,d1} = 1$,
        }
        Update:
            $adj(i,j)$ matrix, $Tabu(i,j)$ matrixes and capacities of the districts ($Pcap$
            and $Dcap$)
        Set $Xmin_{ij}=Xcop_{ij}$,
        SELECT BEST SOLUTION OVER THE $k$ STEPS
        if($minOF<minkOF$){
            $mink=k$,
            $Xkmin_{ij} = Xmin_{ij}$,
            $minkOF=minOF$,
            $PWDispersion** = abs(W_{d1} - W_{d2})$
        }
        else if ($minOF=minkOF$){
            $PWDispersion = abs(W_{d1} - W_{d2})$
            if ($PWDispersion<PWDispersion**$)
            $mink=k$,
            $Xkmin_{ij} = Xmin_{ij}$,
            $minkOF=minOF$,
            $PWDispersion**= PDispersion$,
        }
}
UPDATE BEST SOLUTION OVER THE $K$ STEPS
if ($mink \neq 0$) {
        $Xcop_{ij}=Xkmin_{ij}$,
        $OFcop=minkOF$,
}
Update:
        $adj(i,j)$ matrix, $Tabu(i,j)$ matrixes and capacities of the districts ($Pcap$ and $Dcap$)
Return $OFbest1$, $XBest1_{ij}$, $OFcop$, $Xcop_{ij}$,
```

SelectPair($Prob_j$)

```
Set:
        $rmax_0=0$,
        $d1=0$, $d2=0$,
DEFINE RANGES
for ($j = 1; j \le |J|; ++j$) {
        $rmin_j = rmax_{j-1}$
        $rmax_j = rmin_j+Prob_j$
}
SELECT $d1$
$Random$= rand()
Set $d1 = j \mid rmin_j \le Random < rmax_j$
SELECT $d2$
While ($d2$=0){
        $Random$= rand()
        if ($\exists\, X_{id1} = 1$ and $X_{kj} = 1$ such that $e_{ik} \in E$, and $rmin_j \le Random < rmax_j$;
            $i \ne k$;   $j \ne d1$;   $i,k \in V$;   $j, d1 \in J$){
                Set $d2=j$.
}
UPDATE PROBABILITIES:
        $Prob_{d1}= Prob_{d1}- \alpha$,
        $Prob_{d2}= Prob_{d2}-\alpha$,
        for ($j = 1; j \le |J|; ++j$) {
                if ($j \ne d1$ & $j \ne d2$){
                        $Prob_j = Prob_j + \frac{2\alpha}{|J|-2}$,
        }
Return $d1$ and $d2$
```

Moves(OF, X_{ij}, $sending$, $receiving$, $Wcop_{d1}$,$Wcop_{d2}$)

```
Set:
        $imin=0$,
    $OFmin=\infty$,
EVALUATIONS:
for ($i = 1$; $i \le |V|$; $++i$) {
    if ($border_i = 1$ & $adj(i,receiving)=1$ & $X_{i,sending}=1$ & $PCap_{receiving} \ge wp_i$ &
    $DCap_{receiving} \ge wd_i$ & $Tabu(i,receiving)+tperm< iter$ ){
        $X_{i,sending} = 0$,
        $X_{i,receiving} = 1$,
        $PCap_{sending} += wp_i$, $DCap_{sending} += wd_i$,
        $PCap_{receiving} -= wp_i$, $DCap_{receiving} -= wd_i$,
        $OF = \lambda W + (1-\lambda)Z$,
        OFBEST($OF$, $X_{ij}$),
        $X_{i,sending} = 1$,
        $X_{i,receiving} = 0$,
        $PCap_{sending} -= wp_i$, $DCap_{sending} -= wd_i$,
        $PCap_{receiving} += wp_i$, $DCap_{receiving} += wd_i$,
        if ($OF<minOF$){
            $minOF=OF$,
            $imin=i$,
            $minsend=sending$,
            $minreceive=receiving$,
            $PWDispersion^* = abs(W_{receiving} - W_{sending})$,
        }
        else if($OF<minOF$){
            $PWDispersion = abs(W_{receiving} - W_{sending})$
            if ($PDispersion<PDispersion^*$){
                $minOF=OF$,
                $imin=i$,
                $minsend=sending$,
                $minreceive=receiving$,
                $PWDispersion^*=PWDispersion$,
            }}}
            ASPIRATION
            else if ($border_i = 1$ & $adj(i,receiving)=1$ & $X_{i,sending}=1$ &
            $PCap_{receiving} \ge wp_i$ & $DCap_{receiving} \ge wd_i$ ){
                $X_{i,sending} = 0$,
                $X_{i,receiving} = 1$,
                $PCap_{sending} += wp_i$, $DCap_{sending} += wd_i$,
                $PCap_{receiving} -= wp_i$, $DCap_{receiving} -= wd_i$,
                $OF = \lambda W + (1-\lambda)Z$,
                                OFBEST($OF$, $X_{ij}$),
                                $X_{i,sending} = 1$,
                                $X_{i,receiving} = 0$,
                                $PCap_{sending} -= wp_i$ ,  $DCap_{sending} -= wd_i$ ,
                                $PCap_{receiving} += wp_i$ ,  $DCap_{receiving} += wd_i$ ,
                               if ($OF<minOF$){
                                     $minOF=OF$,
                                     $imin=i$,
                                     $PDispersion^* = abs(W_{receiving} - W_{sending})$,
                               }
                               else if($OF<minOF$){
                                     $PWDispersion = abs(W_{receiving} - W_{sending})$
                                     if ($PWDispersion<PWDispersion^*$){
                                           $minOF=OF$,
                                           $imin=i$,
                                           $PWDispersion^*=PWDispersion$,
            } }}}
Return (minOF, imin)
```

Exchanges(*OF*, *X_{ij}*, *Pairs*, *npairs*,*d1*,*d2*,*W_{d1}*,*W_{d2}*)

```
Set:
      $imin$=0,
      $OFmin$=∞,
EVALUATIONS:
for ($a$ = 1; $a \le npairs$; $++a$) {
      for ($b$ =a+ 1; $b \le npairs$; $++b$) {
         FIRST PAIR OF ADJ POINTS
         $p1=Pairs(a,1)$, $q1=Pairs(a,2)$,
         SECOND PAIR OF ADJ POINTS
         $p2=Pairs(b,1)$, $q2=Pairs(b,2)$,
         EVALUATE: $p1$ to $d2$, $q2$ to $d1$
            if ( $PCap_{d2} \ge wp_{p1}$ & $DCap_{d2} \ge wd_{p1}$ & $PCap_{d1} \ge wp_{q2}$ & $DCap_{d1} \ge wd_{q2}$ &
            $Tabu(q2,d1)+tperm< iter$ & $Tabu(p1,d2)+tperm< iter$ ){
                  $X_{p1,d1} = 0$ , $X_{p1,d2} = 1$, $X_{q2,d2} = 0$ , $X_{q2,d1} = 1$
                  O$F$ = $\lambda W + (1-\lambda)Z$,
                  OFBEST($OF$, $X_{ij}$),
                  $X_{p1,d1} = 1$ , $X_{p1,d2} = 0$, $X_{q2,d2} = 1$ , $X_{q2,d1} = 0$
                  if ($OF<minOF$){
                        $minOF=OF$,
                        $imin1=p1$,
                imin2=q2,
                $PWDispersion^* = abs(W_{d1} - W_{d2})$,
            }
            else if($OF<minOF$){
                $PWDispersion = abs(W_{d1} - W_{d2})$
                if ($PWDispersion<PWDispersion^*$){
                    minOF=OF,
                    imin1=p1,
                    imin2=q2,
                    PWDispersion*=PWDispersion,
    }}}
    ASPIRATION
    else if ( $PCap_{d2} \geq wp_{p1}$ & $DCap_{d2} \geq wd_{p1}$ & $PCap_{d1} \geq wp_{q2}$ &
    $DCap_{d1} \geq wd_{q2}$ ){
            $X_{p1,d1} = 0, X_{p1,d2} = 1, X_{q2,d2} = 0, X_{q2,d1} = 1$
            $OF = \lambda W + (1-\lambda)Z$,
            OFBEST($OF$, $X_{ij}$),
            $X_{p1,d1} = 1, X_{p1,d2} = 0, X_{q2,d2} = 1, X_{q2,d1} = 0$
            if ($OF<minOF$){
                minOF=OF,
                imin1=p1,
                imin2=q2,
                $PDispersion^* = abs(W_{d1} - W_{d2})$,
            }
            else if($OF<minOF$){
                $PWDispersion = abs(W_{d1} - W_{d2})$
                if ($PWDispersion<PWDispersion^*$){
                    minOF=OF,
                    imin1=p1,
                    imin2=q2,
                    PWDispersion*=PWDispersion,
    }}}
  EVALUATE: q1 to d1, p2 to d2
    if ( $PCap_{d1} \geq wp_{q1}$ & $DCap_{d1} \geq wd_{q1}$ & $PCap_{d2} \geq wp_{p2}$ & $DCap_{d2} \geq wd_{p2}$ &
    Tabu(q1,d1)+tperm< iter & Tabu(p2,d2)+tperm< iter ){
            $X_{p2,d1} = 0, X_{p2,d2} = 1, X_{q1,d2} = 0, X_{q1,d1} = 1$
            $OF = \lambda W + (1-\lambda)Z$,
            OFBEST($OF$, $X_{ij}$),
            $X_{p2,d1} = 1, X_{p2,d2} = 0, X_{q1,d2} = 1, X_{q1,d1} = 0$
            if ($OF<minOF$){
                minOF=OF,
                imin1=p2,
                    imin2=q1,
                    $PWDispersion^* = abs(W_{d1} - W_{d2})$,
                }
                else if($OF<minOF$){
                    $PWDispersion = abs(W_{d1} - W_{d2})$
                    if (PWDispersion<PWDispersion*){
                        minOF=OF,
                        imin1=p2,
                        imin2=q1,
                        PWDispersion*=PWDispersion,
            }}}
            ASPIRATION
            else if ( $PCap_{d1} \geq wp_{q1}$ & $DCap_{d1} \geq wd_{q1}$ & $PCap_{d2} \geq wp_{p2}$ &
            $DCap_{d2} \geq wd_{p2}$ ){
                $X_{p2,d1} = 0, X_{p2,d2} = 1, X_{q1,d2} = 0, X_{q1,d1} = 1$
                O$F = \lambda W + (1-\lambda)Z$,
                OFBEST($OF$, $X_{ij}$),
                $X_{p2,d1} = 1, X_{p2,d2} = 0, X_{q1,d2} = 1, X_{q1,d1} = 0$
                if ($OF<minOF$){
                    minOF=OF,
                    imin1=p2,
                    imin2=q1,
                    $PWDispersion^* = abs(W_{d1} - W_{d2})$,
                }
                else if($OF<minOF$){
                    $PWDispersion = abs(W_{d1} - W_{d2})$
                    if (PDispersion<PDispersion*){
                        minOF=OF,
                        imin1=p2,
                        imin2=q1,
                        PWDispersion*=PWDispersion,
}} }}}
Return (minOF, imin1,imin2)
```

Hyperheuristic_LS(OF, X_{ij})

```
Set:
    iter=0, tperm'=tperm
    OFbest1=µ1, OFbest2=µ2, OFbest3=µ3,
    $XBest1_{ij} = 0, \quad XBest2_{ij} = 0, \quad XBest3_{ij} = 0 \quad \forall i \in V, j \in J$,
    $Prob_j = \frac{1}{|J|}$,
Do
        SELECT A LS ALGORITHM
            $Random$= rand()
            if($Random \le 0.5$) {
                    APPLY k-Steps/Pairs LS algorithm
                     SelectPair($Prob_j$)
                     k-S($OF, X_{ij}, d1, d2$)
                     Set:
                            $OF=OFcop$,
                            $X_{ij}=Xcop_{ij} \quad \forall i \in V, j \in J$,
            }
            else {
                     APPLY 1-Step LS algorithm
                     1-S_LS($OF, X_{ij}$)
            }
            Update:
                    $adj(i,j)$ matrix, $Tabu(i,j)$ matrixes and capacities of the
                    districts ($Pcap$ and $Dcap$)

Until one of the stopping conditions is met,
LAST_SEARCH($OFbest1$, $OFbest2$, $OFbest3$, $XBest1_{ij}$, $XBest2_{ij}$, $XBest3_{ij}$)
Return $OFbest1$ and $XBest1_{ij}$
```

```
2-Iters_LS(OF, X_ij )
Set:
       $iter$=0, $tperm'=tperm$
       $OFbest1=\mu1$, $OFbest2=\mu2$, $OFbest3=\mu3$,
       $XBest1_{ij} = 0, \quad XBest2_{ij} = 0, \quad XBest3_{ij} = 0 \quad \forall i \in V, j \in J$,
       $Xcop_{ij}=X_{ij} \quad \forall i \in V, j \in J$,
       $OFcop=OF$,
       $Prob_j = \frac{1}{|J|}$,
       Do
            Set:
                    $OFcop=OF$ and $Xcop_{ij}=X_{ij} \quad \forall i \in V, j \in J$,
            EVALUATE with k-Steps/Pairs LS algorithm
                    SelectPair($Prob_j$)
                    k-S($OFcop, Xcop_{ij}, d1, d2$)
            Set:
                  $OF\_first=OF$ and $X\_first_{ij}= X_{ij} \quad \forall i \in V, j \in J$,
                   $Dispersion^* = \sum_{j \in J} |W_j - \overline{W}|$
            EVALUATE with 1-Step LS algorithm
                1-S_LS($OFcop, Xcop_{ij}$)
            SELECT BEST SOLUTION
            if ($OF > OF\_first$){
                $OF=OFcop$
                $X_{ij}=Xcop_{ij} \quad \forall i \in V, j \in J,$
            }
            else if ($OF < OF\_first$){
                $OF=OF\_first,$
                $X_{ij}=X\_first_{ij} \quad \forall i \in V, j \in J,$
            }
            else if ($OF = OF\_first$){
                $Dispersion = \sum_{j \in J} \left| W_j - \overline{W} \right|$
                if ($Dispersion<Dispersion*$){
                    $Dispersion* = Dispersion,$
                    $OF=OFcop$
                    $X_{ij}=Xcop_{ij} \quad \forall i \in V, j \in J,$
                }
                else{
                    $OF=OF\_first,$
                    $X_{ij}=X\_first_{ij} \quad \forall i \in V, j \in J,$
                }
            }
            Update:
                $adj(i,j)$ matrix, $Tabu(i,j)$ matrixes and capacities of the districts
                ($Pcap$ and $Dcap$)
Until one of the stopping conditions is met,
LAST_SEARCH($OFbest1$, $OFbest2$, $OFbest3$, $XBest1_{ij}$, $XBest2_{ij}$, $XBest3_{ij}$)
Return $OFbest1$ and $XBest1_{ij}$
```

LAST_SEARCH(*OFbest1, OFbest2, OFbest3, XBest1*$_{ij}$*, XBest2*$_{ij}$*, XBest3*$_{ij}$)

```
Set:
    Set Tot_iter=5
    1-Step_LS($OFbest1$, $XBest1_{ij}$)
Set:
    Set Tot_iter=3
    1-Step_LS($OFbest2$, $XBest2_{ij}$)
Set:
    Set Tot_iter=2
    1-Step_LS($OFbest3$, $XBest3_{ij}$)
Return $OFbest1$ and $XBest1_{ij}$
```

```
**OFBEST(*OF*)**
    Set *OFbest1* and $XBest1_{ij}$ as the current solution
    For iter=1 to 5
        **1-Step LS**
    end
    Set *OFbest2* and $XBest2_{ij}$ as the current solution
    For iter=1 to 3
        **1-Step LS**
    end
    Set *OFbest3* and $XBest1_{ij}$ as the current solution
    For iter=1 to 2
        **1-Step LS**
    end
Return *OFbest1* and $XBest1_{ij}$
```

APPENDIX V: Analysis of the computational time of the heuristics and fitted curves.

In this section we present an analysis of the computational times and the fitted curves according to the instance size (measured by the number of points) based on the maximum (worst case) and average computational time. This analysis is done based on the results for the 1-Step LS algorithm, given that this algorithm resulted in the good solutions and can be considered as an average case with respect to the rest. Also, we selected this algorithm because there is not too much difference among the computational times of the heuristics except the 2-Steps that resulted very inefficient and for this reason only some of the smallest instance sizes were tested with this algorithm. We used LAB Fit [48] and Minitab 14.0 for this analysis, and we found similar results by both software. Table V.1 summarizes the computational times for the 1-S for the worst and average cases according to the instance size measured by the amount of points.

Table V.1 Computational times for the 1-S LS algorithm

1-S Computational Time		
Number of Points	**Worst Case**	**Average Case**
50	0.563	0.3026713
200	16.437	7.8924306
450	140.048	63.494282
1000	1173.827	576.15302
1500	4097.265	2374.7717

V.1 Worst case analysis

In this section we present the curves found for the maximum computational time according to the instance size. In section 4.2 the complexity analysis of the heuristics was presented, for which we distinguish between the cases in which the F-R procedure was required or not, which does not depend on the instance size. For the 1-S heuristic, it was defined that algorithm requires $O(V^4 \log V$) time when the F-R procedure is not required. Figure V.1 shows the fitted line plot obtained by LAB Fit, which is obtained by a Power function of order cubic. From the Minitab analysis, we can observe in figure V.2 that actually a Squared order function may fit, with a R-Squared value of 99.5% and in figure V.3 we observe that for the Cubic order we have a R-Squared value of 100% which is consistent to the result found by LAB Fit

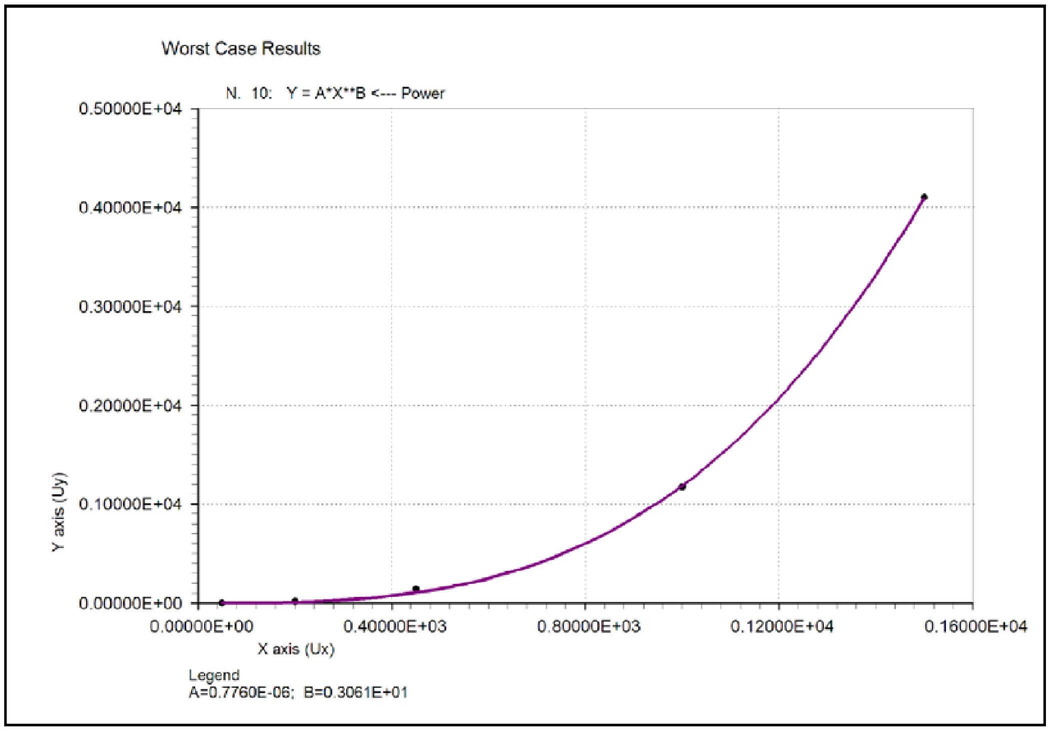


Fig V.1. Fitted Line Plot for the maximum computational times by LAB Fit.

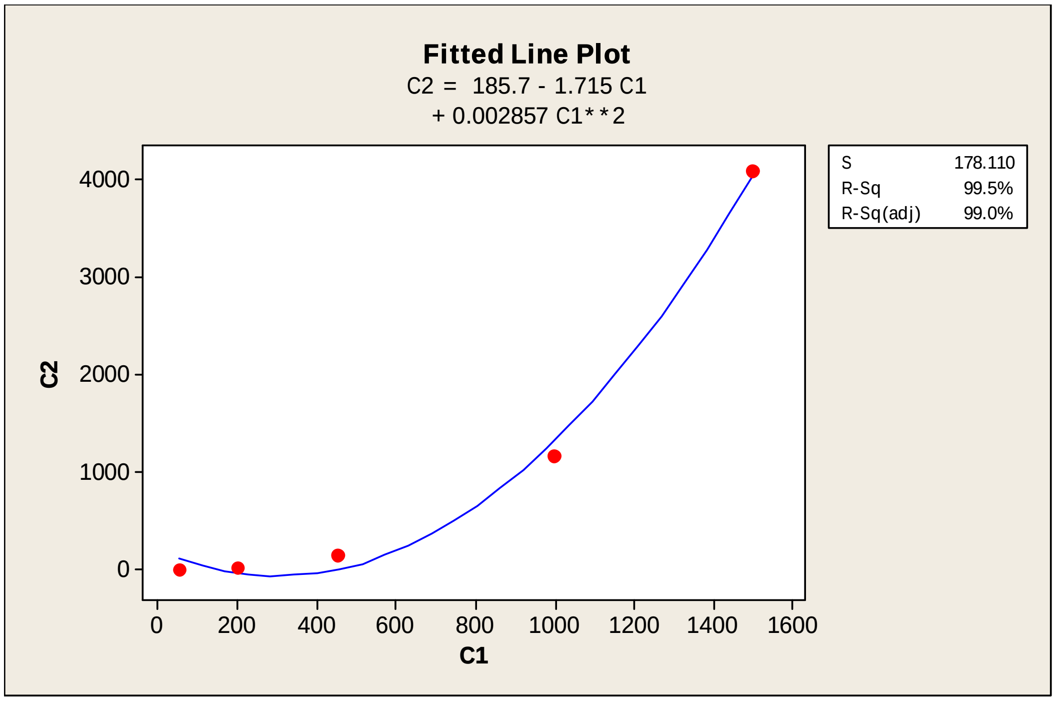


Fig V.2. Fitted Line Plot (Squared) for the maximum computational times by Minitab.

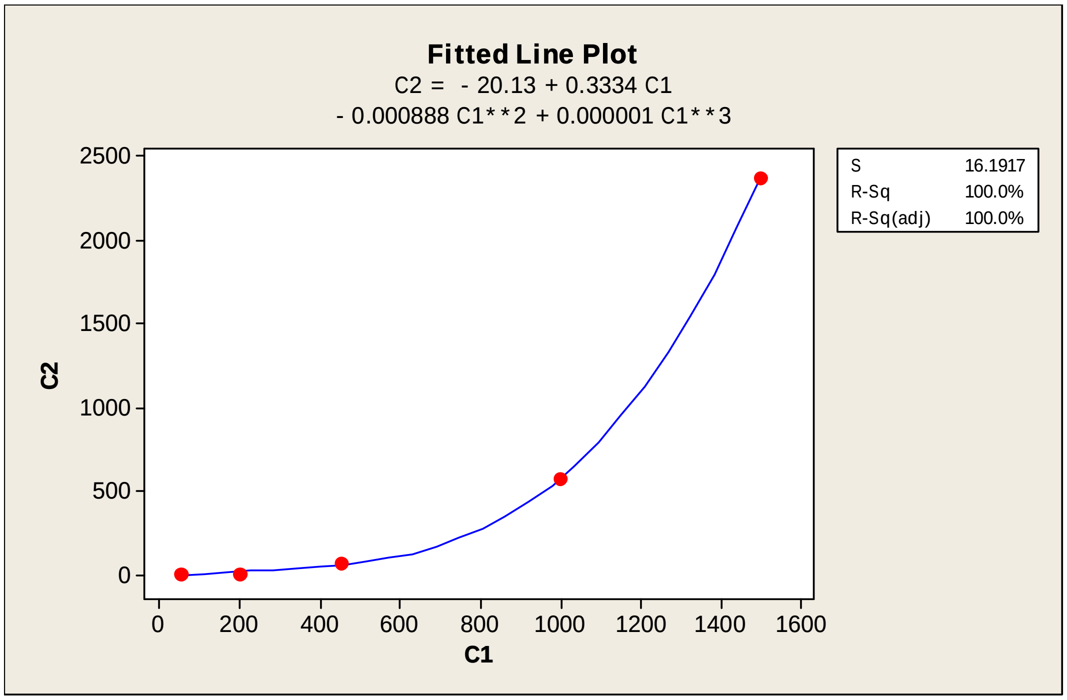


Fig V.3. Fitted Line Plot (Cubic) for the maximum computational times by Minitab.

The result found by the Fitting curves differs with respect to the complexity analysis computed in Section 4.2 in which the order of the function was of fourth or fifth depending on the requirement of the F-R procedure. This is due to the different stopping rules that were defined and also because of the amount of initial solutions that are attempted to construct, and that several procedures of the algorithms are guided by random numbers. For the complexity analysis we assumed the worst cases, and we can observe that in practice the performance of the algorithm is actually better than what was expected. We also present the regression analysis found by Minitab in figure V.4, from which we can observe that around 86% of the variability is described by the model, which is a reasonable value.

Analysis of Variance

Source	DF	SS	MS	F	P
Regression	1	10576469	10576469	18.49	0.023
Residual Error	3	1715924	571975		
Total	4	12292393			

S = 756.290 **R-Sq** = 86.0% **R-Sq(adj)** = 81.4%

Fig. V.4. Regression analysis for the Maximum computational times.

V.2 Average case analysis

In this section we present the curves found for the average computational time according to the instance size as it was done for the maximum computational time in section V.5. Figure V.1 shows the fitted line plot obtained by LAB Fit, which is obtained by a Power function of order cubic. From the Minitab analysis, we can observe in figure V.6 that actually a Squared order function may fit, with a R-Squared value of 99% and in figure V.7 we observe that for the Cubic order we have a R-Squared value of 100% which is consistent to the result found by LAB Fit and also with respect to the results of the worst case analysis.

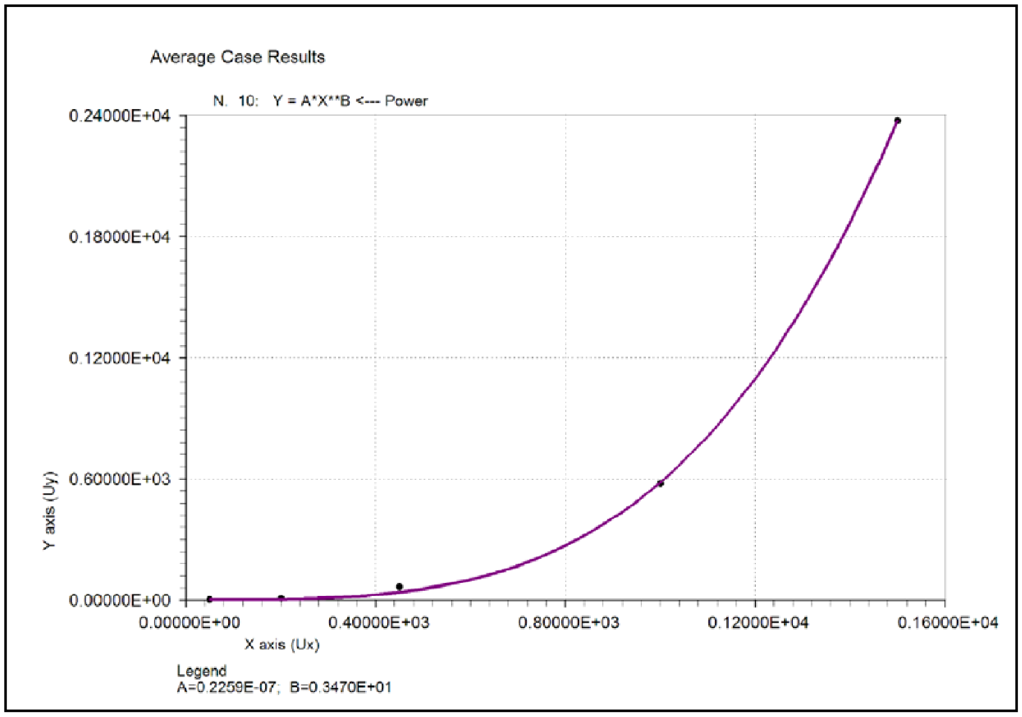


Fig V.5. Fitted Line Plot for the average computational times by LAB Fit.

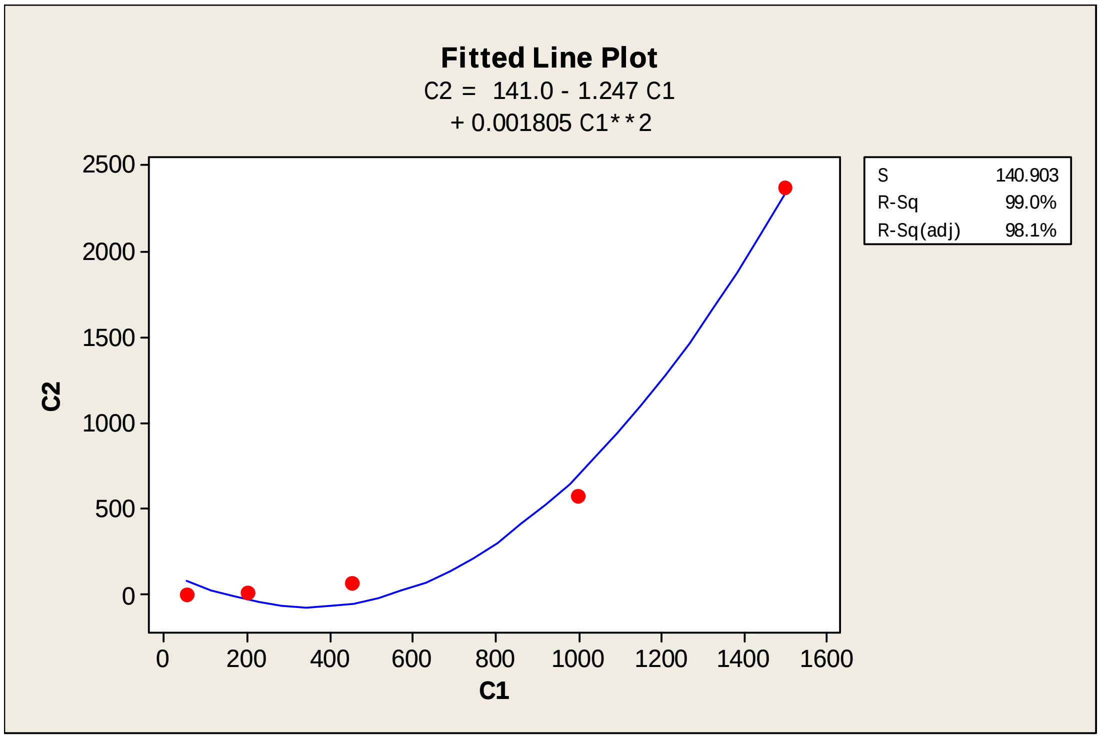


Fig V.6 Fitted Line Plot (Squared) for the average computational times by Minitab.

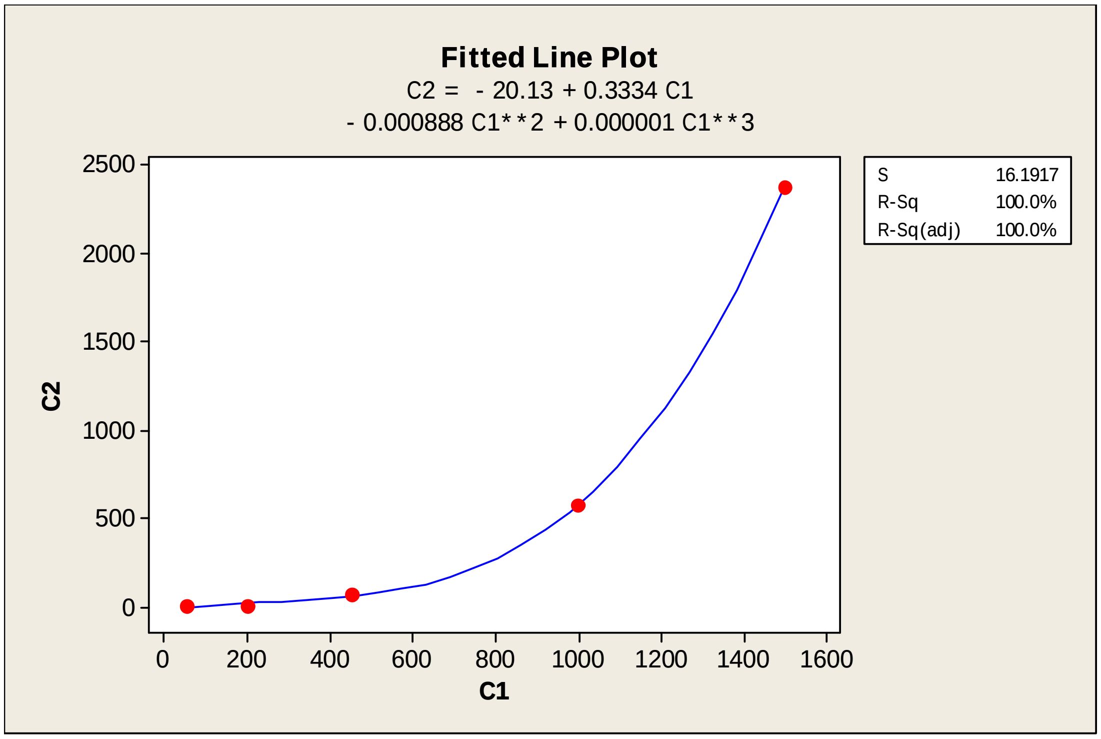


Fig V.7. Fitted Line Plot (Cubic) for the average computational times by Minitab.

We also present the regression analysis found by Minitab in figure V.8, from which we can observe that around 83% of the variability is described by the model, which is a reasonable value.

Analysis of Variance

Source	DF	SS	MS	F	P
Regression	1	3449048	3449048	14.80	0.031
Residual Error	3	699300	233100		
Total	4	4148347			

S = 482.804 **R-Sq** = 83.1% **R-Sq(adj)** = 77.5%

Fig. V.8. Regression analysis for the Maximum computational times.

APPENDIX VI: Test of Hypothesis to determine which heuristic resulted the best.

In this appendix we present a statistical analysis to determine which of the five heuristics resulted the best. For this, we made a Wilcoxon Signed Rank Test for the differences between pairs of methods using Minitab 14.0. Comparisons are based on the 1-S heuristic with respect to the rest of the methods. To determine which method has a better performance, we compare the differences of the solutions found for each heuristic under consideration.

The Null hypothesis establishes that there is no difference in the results found by both methods and for the alternative hypothesis we tested either that the difference is less than zero or greater. A negative median value of the differences indicates that the 1-S method performed better. Results indicates that the best seed method resulted the 2-Iter LS. Section VI.A presents an analysis of the results found for all the instance types except the Parcel instances, that were analyzed separately and results are presented in section VI.B.

VI. A: Analysis of the results obtained for the Asymmetric, Semi-Symmetric, Symmetric and Urban type instances.

In figure VI.1 we present the Wilcoxon signed rank test for the median difference between solutions found by the 1-S and 2-S methods. We reject that the heuristics provide equal solutions values, and conclude that 1S-2S>0, which indicates that 1-S resulted in greater values for the Objective Function and hence the 2-S method is better. We can observe an estimated mean of the difference of 0.003031.

```
Test of median = 0.000000 versus median > 0.000000

           N
          for  Wilcoxon        Estimated
       N  Test  Statistic     P    Median
1S-2S  648   329   40503.0  0.000   0.003031
```

Fig VI.1. Wilcoxon Signed Rank Test: (1-S) – (2-S)

Figure VI.2 presents the test for the difference of medians between the 1-S and K-S/P. We reject the null hypothesis that establishes that there is no difference between the solutions found by those algorithms and conclude that there is a negative value of the median difference, which indicates that the 1-S found better solutions than the K-S/P procedure.

```
Test of median = 0.000000 versus median < 0.000000

             N
           for   Wilcoxon        Estimated
         N  Test  Statistic     P    Median
1S-KS  1080   930    36702.5  0.000   -0.04045
```

Fig VI.2. Wilcoxon Signed Rank Test: (1-S) – (K-S/P)

In figure VI.3 we observe the test for the difference between 1-S and HypLS heuristics. We have enough evidence to reject the null hypothesis and conclude that there is difference between the solutions found by both algorithms and that 1-S found better solutions, as the median value of the differences is negative. This indicates that combining the 1-S and K-S/P by a hyperheuristic resulted with worse solutions than applying only the 1-S which is a simpler neighborhood structure.

```
Test of median = 0.000000 versus median < 0.000000

             N
           for   Wilcoxon        Estimated
         N  Test  Statistic     P    Median
1S-Hyp  1080   788   130732.5  0.000  -0.002473
```

Fig VI.3. Wilcoxon Signed Rank Test: (1-S) – (HypLS)

Figure VI.4 shows the Test of the difference between the 1-S and 2-IterLS algorithms. We have strong evidence to reject the Null Hypothesis that both algorithms have the same performance and conclude that the 2-Iter found better solutions given that the estimated median of the differences between the solutions is positive.

```
Test of median = 0.000000 versus median > 0.000000

             N
            for   Wilcoxon        Estimated
          N  Test  Statistic     P    Median
1S-2IT  1080   700   156521.5  0.000   0.003561
```

Fig VI.4. Wilcoxon Signed Rank Test: (1-S) – (2-IterLS)

Given that the 2-S and 2-IterLS algorithms resulted better than the 1-S, we made a test between these procedures to determine which is better. For the 2-S we must consider that only the small and medium size instances could be solved in reasonable time. Figure VI.5 presents the results of the Test, and we can observe a p-value of 0.048, which is very close to the value of α (0.05), hence we can conclude that there is no strong evidence to reject the null hypothesis that establishes that both methods have the same performance. Actually we observe that the estimated value of the median is very close to zero.

```
Test of median = 0.000000 versus median < 0.000000

            N
           for   Wilcoxon
         N  Test  Statistic      P  Estimated Median
2IT-2S  648   309    21323.0  0.048       0.000000000
```

Fig VI.5. Wilcoxon Signed Rank Test: (2-IterLS) – (2-S).

Hence, for the Asymmetric, Semi-Symmetric, Symmetric and Urban instances we can conclude that the 2-IterLS algorithm performed better, considering that even thought there is no strong evidence to conclude that 2-IterLS is better than 2-S, since these method did not solve in reasonable computational time the large size instances as it was done for the rest of the heuristics.

VI. B: Analysis of the results obtained for the Parcel Instances.

Figure VI.6 presents the results of the test between the 1-S and 2-S methods. We have strong evidence to reject the null hypothesis and conclude that the 1-S fond better solutions, since the difference between the solutions found by both algorithms resulted with a negative value.

```
Test of median = 0.000000 versus median < 0.000000

               N
             for   Wilcoxon          Estimated
          N  Test  Statistic     P     Median
1S-2S_1  18   18        0.0  0.000   -0.09523
```

Fig VI.6. Wilcoxon Signed Rank Test-Parcel: (1-S) – (2-S).

Figure VI.7 presents the results of the test between the 1-S and K-S/P methods. We may reject the null hypothesis that establishes that there is no difference between solutions found by both methods for α ≥0.041. Since this value is very close to 0.05 and we may also consider that there is not too much difference between the solutions found by both algorithms, which differs from the results of the rest of the instances presented in Section VI.A, where we observe a strong evidence that the 1-S performed better.

```
Test of median = 0.000000 versus median < 0.000000

               N
             for   Wilcoxon          Estimated
          N  Test  Statistic     P     Median
1S-KS_1  18   18       45.0  0.041   -0.04568
```

Fig VI.7. Wilcoxon Signed Rank Test-Parcel: (1-S) – (K-S/P).

Figure VI.8 presents the results of the test between the 1-S and HypLS methods. We performed the test under an alternative hypothesis that indicates that the median difference is greater than zero and also for the case that is less than zero. In both cases the p-value resulted greater than the level of significance value of α =0.05. For this reason we present the general test in which the alternative hypothesis indicates that results of both heuristics are different.

We observe a p-value of 0.338, hence we do not have evidence to reject the Null Hypothesis that establishes there is no difference between the solutions found by both methods. This result differs than the analysis performed for the rest of the instances, however it still indicates that combining 1-S and K-S/P by a hyperheuristic does not provide better results than applying the 1-S.

```
Test of median = 0.000000 versus median not = 0.000000

             N
            for   Wilcoxon          Estimated
          N  Test  Statistic      P     Median
1S-Hyp_1  18    18      108.0  0.338    0.01389
```

Fig VI.8. Wilcoxon Signed Rank Test-Parcel: (1-S) – (HypLS).

Figure VI.9 presents the results of the test between the 1-S and 2-IterLS methods, results indicates that for an α=0.05 we may reject the null hypothesis that indicates that both methods provided the same results and conclude that there is a positive difference between the solutions, hence the 2-Iter performed better than the 1-S. However, the p-value is very close to the significance level so the evidence is not as strong as that found for the rest of the instances in Section VI.A, but we still are able to conclude that the 2-Iter performed equal or better than the rest of the heuristics for all the instances types.

```
Test of median = 0.000000 versus median > 0.000000

           N
          for   Wilcoxon          Estimated
        N  Test  Statistic      P     Median
12-2IT  18    18      126.0  0.041    0.05920
```

Fig VI.9. Wilcoxon Signed Rank Test-Parcel: (1-S) – (2-IterLS).

APPENDIX VII: Analysis of the methods to select the set of seeds.

In this section we present an analysis of the five methods proposed to select a set of seeds, with the objective to determine the impact that the seed method has on the solution and analyze if there is a relation between the seed method and the quality of the solutions. Section VII.A analyze the percentage in which the final solution reported corresponds to each seed method. Section VII.B presents a statistical analysis of the impact that the seed method and the heuristics have on the quality of the solution, for which we used Minitab 14.0.

VII. A Corresponding proportion of the Final solution to each Seed method

For this analysis, we considered the results found by the 1-S heuristic for all the instance types and sizes. Tables VII.1 to VII.3 show the proportion in which the final solution reported for an instance corresponds to each seed method. Table VII.1 presents the results by instance size for each of the four types of instances: asymmetric, semi-symmetric, urban and symmetric. For each type of instance, 54 instances were solved, which includes the variants on the capacity restrictiveness, values of average speed, replicates and lambda variants. Table VII.2 presents the results that correspond to the parcel instances for which 18 instances were solved. Table VII.3 summarizes the results for all the types of instances per size as well as the global proportion that corresponds to each seed method for all the instances tested.

We can observe in table VII.1 that the proportions varies according to the instance type. For example, the Workload method resulted with the biggest proportions for the urban and asymmetric instances, but the P-Dispersion method resulted the best for the Semi-Symmetric and Symmetric instances. Results also vary according to the instance size.

Table VII.1 Proportion per instance size and type.

SEED METHOD	ASYMMETRIC					
	50_5	200_10	450_15	1000_20	1500_30	All sizes
P-Disp	0.37037	0.314815	0	0.018519	0	0.140741
Semi-Rand	0.092593	0.111111	0	0.166667	0	0.074074
Neigh.	0.12963	0	0.074074	0.037037	0	0.048148
Angle	0.296296	0.5	0.481481	0.444444	0.666667	0.477778
Workload	0.111111	0.074074	0.444444	0.333333	0.333333	0.259259
SEED METHOD	**SEMI-SYMMETRIC**					
	50_5	200_10	450_15	1000_20	1500_30	All sizes
P-Disp	1	0.5	0	0.481481	0.611111	0.518519
Semi-Rand	0	0	0	0	0	0
Neigh.	0	0.333333	0.666667	0.166667	0.388889	0.311111
Angle	0	0.166667	0	0.185185	0	0.07037
Workload	0	0	0.333333	0.166667	0	0.1
SEED METHOD	**URBAN**					
	50_5	200_10	450_15	1000_20	1500_30	All sizes
P-Disp	0.166667	0	0	0	0	0.033333
Semi-Rand	0	0.388889	0	0.166667	0.333333	0.111111
Neigh.	0.333333	0.055556	0	0	0	0.077778
Angle	0	0.166667	0	0.166667	0	0.066667
Workload	0.5	0.388889	1	0.666667	0.666667	0.511111
SEED METHOD	**SYMMETRIC**					
	50_5	200_10	450_15	1000_20	1500_30	All sizes
P-Disp	0.944444	0.166667	0.277778	0	0.333333	0.344444
Semi-Rand	0	0	0	0	0	0
Neigh.	0.055556	0.333333	0.333333	0.333333	0	0.211111
Angle	0	0.333333	0.222222	0.333333	0	0.177778
Workload	0	0.166667	0.166667	0.333333	0.666667	0.266667

In table VII.2 we present the proportions for the Parcel instances. We can observe that 100% of the solutions corresponded to the Workload method, in which the seeds are selected according to the dispersion of the workload over the region. Parcel instances are similar to the Urban instances and only differ in that for the Parcel instances, the location of the points corresponds to real points of demand. For the urban instances we observed that the biggest proportion corresponds also to the workload method.

Table VII.2 Proportion for the Parcel instances.

SEED METHOD	Parcel
	1109_28
P-Disp	0
Semi-Rand	0
Neigh.	0
Angle	0
Workload	1

In Table VII.3 we observe the global proportions that correspond to each seed method. The workload method resulted with the biggest fraction, while the smallest corresponds to the Semi-Random. There are some instance sizes in which none of the best solutions corresponded to the Semi-Random method. For the smallest size instances of 50_5 and 200_10 the P-Dispersion and Angle method respectively resulted with biggest proportions. For the rest of the instance sizes, the Workload method was the best.

Table VII.3 Proportion for all the instances types.

SEED METHOD	ALL INSTANCES TYPES						
	50_5	200_10	450_15	1000_20	1500_30	Parcel	ALL
Workload	0.152778	0.157407	0.486111	0.375	0.416667	1	0.334259
P-Disp	0.62037	0.24537	0.069444	0.125	0.236111	0	0.259259
Angle	0.074074	0.291667	0.175926	0.282407	0.166667	0	0.198148
Neigh.	0.12963	0.180556	0.268519	0.134259	0.097222	0	0.162037
Semi-Rand	0.023148	0.125	0	0.083333	0.083333	0	0.062963

From previous results we may notice that including different types of seed methods enhances the diversity over the search space and increase the likelihood to construct a feasible solution. Given that depending the type and size of instance the seed method that resulted in a biggest proportion of the best solutions reported may vary, including several types of seed methods makes the algorithm more robust to different types and sizes of instances.

VII. B Impact of the Seed Method on the final solution and the ability to construct a feasible solution.

In this section we present an analysis of the impact that the seed method has on the quality of the solution found by each heuristic. For this, we analyzed the results of a set of eight instances that corresponds to the medium size of 450_15. For this set, we selected two instances of each type: asymmetric, semi-symmetric, symmetric and urban, each solved by one of the capacity levels (less restricted and tight). These instances were solved with an average speed value of 30 kms./hr. and with a value of 0.5 for the relative weighting factor. It is important to recall that each of the five heuristics construct the same feasible initial solutions and only differ to each other in the local search.

Tables VII.4 to VII.11 and Figures VII.1 toVII.15 present the results for each instance type. We must recall that for each instance, up to five initial feasible solutions are attempted to be constructed by each seed method. Hence, for each instance we constructed in total up to 25 solutions. Each table shows the best solution that was found for each seed method and each of the heuristics.

The tables also present the seed method that corresponds to the best solution for each algorithm. This allows to analyze how much the quality of the solution may be affected if only one seed method would have been used in the construction phase. The tables also present in the last two columns, the proportion in which a feasible solution was constructed by each seed method and the proportion in which a feasible solution was constructed without requiring the F-R procedure.

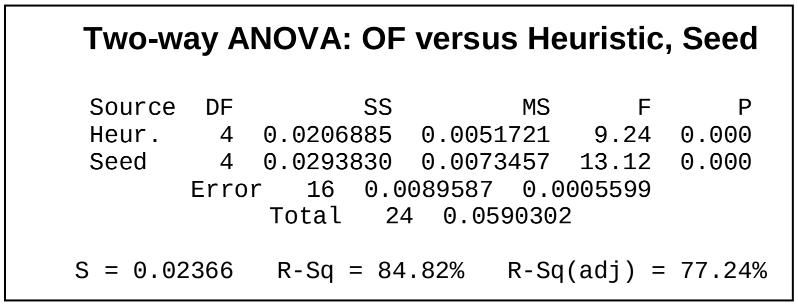
Asymmetric and Less Restricted instance (ASYM-LR)

Table VII.4 presents the values of the best solution found by each heuristic per seed method. We can observe that for this type of instance, the Angle method resulted the best. We also can notice that we were able to construct the 25 feasible initial solutions, but only 16% of those solutions did not require the F-R procedure to obtain feasibility, and most of this solutions were constructed with the Angle method. We also can notice that the best solution reported by each of the heuristics differ, for which the best is found by the 2-IterLS heuristic. We also observe the results found by a heuristic are not similar for the different seed methods. For instance, in the case of the 1-S algorithm, best solution found is 1.1556 but if we would have used only the neighborhood method, the resulting solution would have been 1.2758, around 10% bigger.

Table VII.4 Objective Function values per Seed Method and Heuristic, ASYM-LR.

SEED METHOD	**INSTANCE: ASYMMETRIC AND LESS RESTRICTED**						
	1-S	**K-S/P**	**2-S**	**HypLS**	**2-IterLS**	**%FS**	**%FS (no F-R)**
P-Dispersion	1.212466	1.287208	1.180128	1.206438	1.212466	1.00	0.00
Semi-Random	1.259279	1.289325	1.257479	1.25742	1.23285	1.00	0.00
Neighborhood	1.275855	1.350773	1.224901	1.230515	1.263233	1.00	0.00
Angle	1.155622	1.275975	1.155363	1.190741	1.150938	1.00	0.60
Workload	1.185871	1.202102	1.1841	1.194474	1.185871	1.00	0.20
Best OF value	1.155622	1.202102	1.155363	1.190741	1.150938	**All: 25 Solutions**	
Best Seed Meth.	Angle	Workload	Angle	Angle	Angle	1.00	0.16

In Fig. VII.1 we present the Two-way ANOVA from which we observe that both factors, the Heuristic and Seed method are significant and affect the quality of the solution found. In Fig. VII.2 we present the interaction plot for both factors, in which we can observe that there is interaction between the Heuristics and Seed methods. Is also possible to observe that the Angle method resulted better for all the algorithms except the K-S/P, for which the Workload resulted better.



Two-way ANOVA: OF versus Heuristic, Seed

```
Source  DF         SS         MS      F      P
Heur.    4  0.0206885  0.0051721   9.24  0.000
Seed     4  0.0293830  0.0073457  13.12  0.000
       Error   16  0.0089587  0.0005599
            Total   24  0.0590302

S = 0.02366   R-Sq = 84.82%   R-Sq(adj) = 77.24%
```

Fig VII.1 Two-way ANOVA, ASYM-LR.

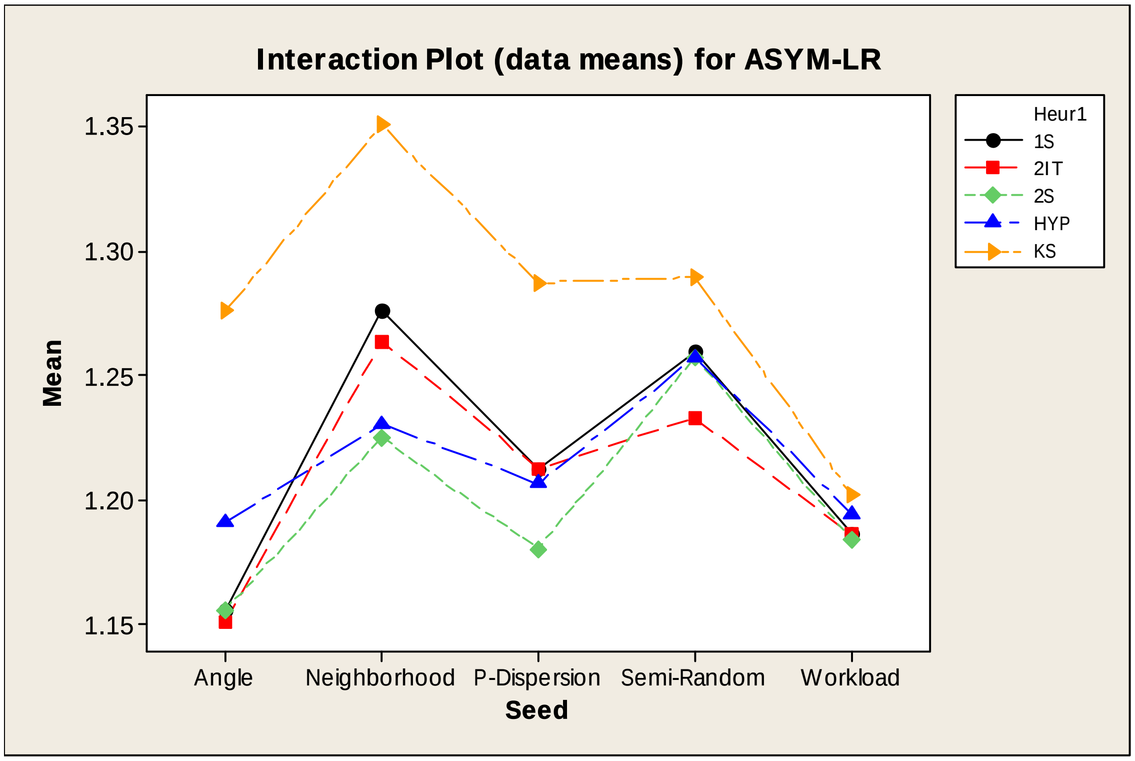


Fig VII.2 Interaction Plot, ASYM-LR.

Asymmetric and Tight instance (ASYM-T)

Table VII.5 presents the results of the same instance type but now solved with the tight level of capacity restrictiveness. We observe that now the best seed method corresponds to the Workload method. Angle and Workload methods are more similar than the rest of the methods because they select the seeds based on the dispersion of the points and workload within the region. We can observe that for both instances, either the less restricted or tight case, both seed methods are the best. We also observe in the table that not all the 25 solutions could be constructed as was the case for the less restricted instance, because it resulted more difficult. In this instance we can also observe that results found by the seed methods are very different, which indicates that the seed method impacts the quality of the solutions. For example, if K-S/P would have reported the solution found by the Neighborhood seed instead of the Angle, the value of the solution would have been around of 20% bigger.

Table VII.5 Objective Function values per Seed Method and Heuristic, ASYM-T.

SEED METHOD	**INSTANCE: ASYMMETRIC AND TIGHT**						
	1-S	**K-S/P**	**2-S**	**HypLS**	**2-IterLS**	**%FS**	**%FS (no F-R)**
P-Dispersion	1.250928	1.320985	1.219019	1.271257	1.250607	0.80	0.00
Semi-Random	1.262051	1.345179	1.246433	1.296183	1.237604	1.00	0.00
Neighborhood	1.378905	1.53425	1.336733	1.349565	1.358564	1.00	0.00
Angle	1.216347	1.269468	1.190982	1.238352	1.208699	1.00	0.20
Workload	1.212967	1.300957	1.191178	1.205245	1.18962	1.00	0.00
Best OF value	1.212967	1.269468	1.190982	1.205245	1.18962	**All: 25 Solutions**	
Best Seed Meth.	Workload	Angle	Angle	Workload	Workload	0.96	0.04

In figure VII.3 we present the ANOVA for this type of instance, in which we can observe that both factors, seed method and heuristics have a strong impact on the solution value. In figure VII.4 we present the interaction plot of this factors in which we can observe some interaction between the seed method and heuristic. For all the heuristics, the angle and workload methods resulted in the better solutions. The K-S/P heuristic found the worst solutions for all the seed methods.

Two-way ANOVA: OF versus Heuristic, Seed

```
Source   DF        SS          MS      F      P
Heur.     4  0.042601  0.0106503  18.14  0.000
Seed      4  0.096529  0.0241322  41.10  0.000
Error    16  0.009394  0.0005871
Total    24  0.148524

S = 0.02423   R-Sq = 93.68%   R-Sq(adj) = 90.51%
```

Fig VII.3 Two-way ANOVA, ASYM-T.

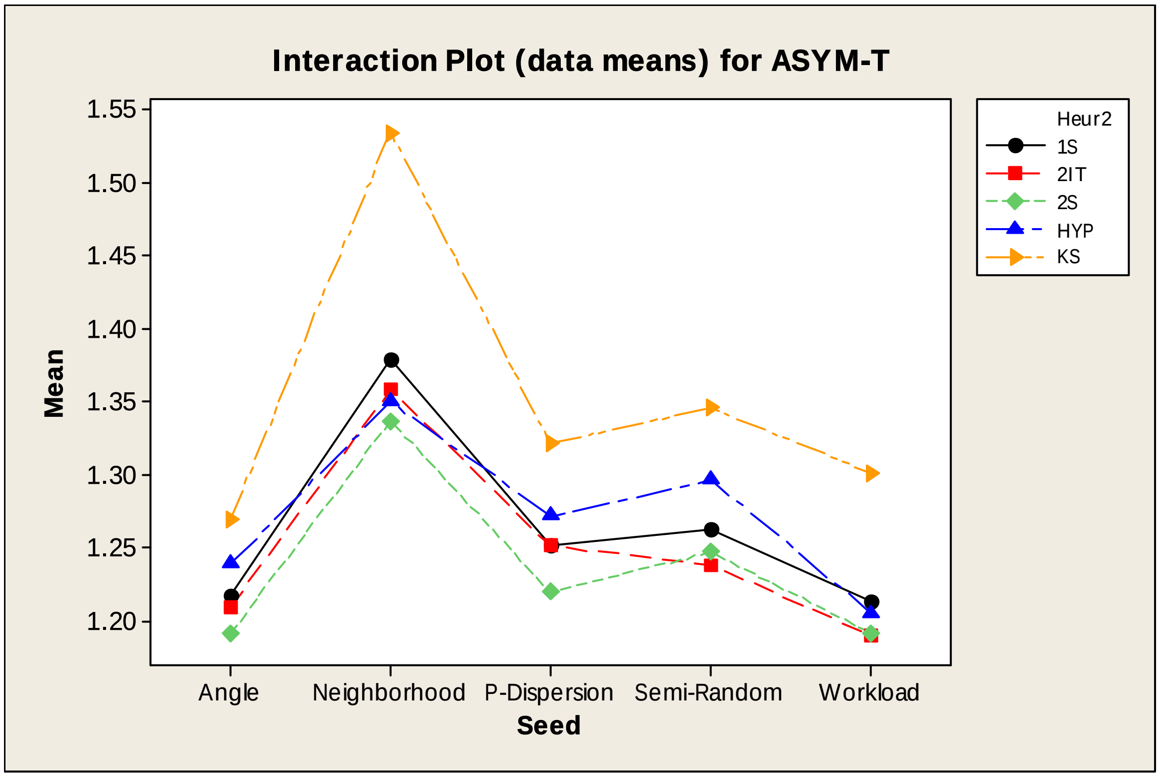


Fig VII.4 Interaction Plot, ASYM-T.

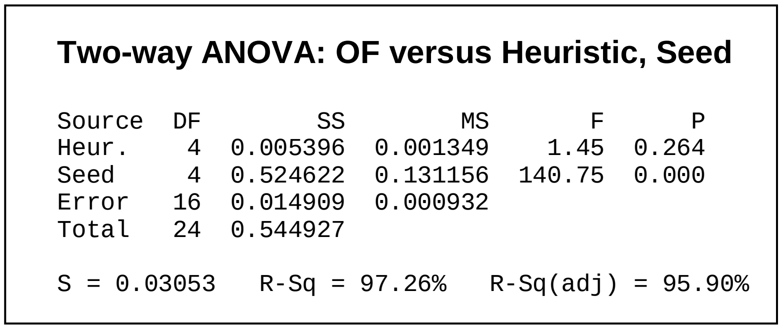
Semi-Symmetric and Less restricted instance (S-SYM-LR)

Table VII.6 presents the results for the Semi-Symmetric instance with the less restricted capacity limits. It is possible to observe that the Semi-Random method only could construct one feasible solution of the five attempts. For this type of instance, there are ties in the best solutions found by some of the seed methods. However, for all the algorithms the Angle resulted the best. Ties may indicate that for this type of instance there may be not too much difference on the quality of the solution if we wouldn't include several seed methods. However, if we compare the results found by the best and worst seed method of each algorithm there is a gap of around 44%. Hence, we can still consider that it is a good approach to include several types of seed methods. We can also observe in the last column that for the P-Dispersion and Neighborhood methods, none of the solutions required the F-R procedure, and for the rest of the methods except the Semi-Random, there were several cases in which this procedure was not required. This indicates that for this type of instance constructing a feasible solution is easier than for the asymmetric instances.

Table VII.6 Objective Function values per Seed Method and Heuristic, S-SYM-LR.

SEED METHOD	**INSTANCE: SEMI-SYMMETRIC AND LESS RESTRICTED**						
	1-S	**K-S/P**	**2-S**	**HypLS**	**2-IterLS**	**%FS**	**%FS (no F-R)**
P-Dispersion	1.056218	1.056218	1.056218	1.056218	1.047592	1.00	1.00
Semi-Random	1.316394	1.281889	1.316394	1.290515	1.316394	0.20	0.00
Neighborhood	0.922592	1.064845	0.922592	0.922592	0.913966	1.00	1.00
Angle	0.913966	0.932102	0.913966	0.922592	0.913966	1.00	0.60
Workload	0.913966	0.967146	0.913966	0.932102	0.914849	1.00	0.40
Best OF value	0.913966	0.932102	0.913966	0.922592	0.913966	**All: 25 Solutions**	
Best Seed Meth.	Angle, Workload	Angle	Angle, Workload	Neighb., Angle	Neighb., Angle	0.88	0.60

Figure VII.5 presents the ANOVA for this instance, in which we observe that the Seed Method is significant and impact the quality of the solution reported. However, for this type of instance there is no evidence that the heuristic method has an impact on the solution. Actually if we observe in Table VII. 6, the solutions reported by each of the methods do not differ too much.



Two-way ANOVA: OF versus Heuristic, Seed

```
Source  DF        SS        MS       F      P
Heur.    4  0.005396  0.001349    1.45  0.264
Seed     4  0.524622  0.131156  140.75  0.000
Error   16  0.014909  0.000932
Total   24  0.544927

S = 0.03053   R-Sq = 97.26%   R-Sq(adj) = 95.90%
```

Fig VII.5 Two-way ANOVA, S-SYM-LR.

In Figure VII.6 we present the interaction plot of both factors. If we observe there is not too much interaction between the factors and results of the heuristics for each seed method are very similar, with a slight variation for the K-S/P.

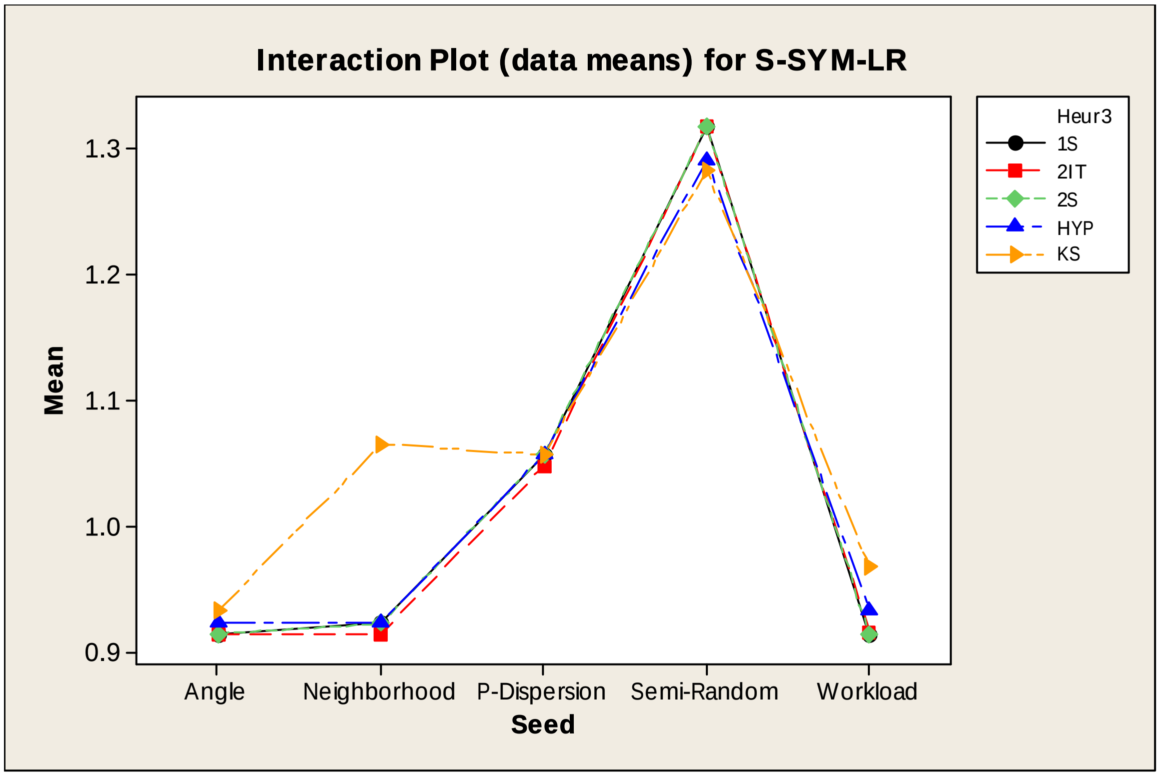


Fig VII.6 Interaction Plot, S-ASYM-LR.

Semi-Symmetric and Tight instance (S-SYM-T)

Table VII.7 presents the results for the Semi-Symmetric instance but now considering the tight level of capacity restrictiveness. In this case we can observe that the Semi-Random method could not find any feasible solution, because this instance resulted more difficult. For the rest of the heuristics we observe that the proportion of feasible solutions constructed without requiring the F-R procedure is the same as for the less restricted instance. We also observe ties in the best solutions found by each seed method, and also the Angle resulted the best for all the heuristics. The table also shows that there were two heuristics for which the P-Dispersion found the best solution as well as the Angle, which did not happen when the instance was solved with the less restricted capacity level.

Table VII.7 Objective Function values per Seed Method and Heuristic, S-SYM-T.

SEED METHOD	**INSTANCE: SEMI-SYMMETRIC AND TIGHT**						
	1-S	**K-S/P**	**2-S**	**HypLS**	**2-IterLS**	**%FS**	**%FS (no F-R)**
P-Dispersion	0.913966	1.064845	0.922592	0.922592	1.047592	1.00	0.80
Semi-Random						0.00	0.00
Neighborhood	1.047592	1.056218	0.922592	1.047592	0.913966	1.00	1.00
Angle	0.913966	0.932102	0.913966	0.922592	0.913966	1.00	0.20
Workload	0.913966	0.975773	0.931219	0.931219	0.913966	1.00	0.60
Best OF value	0.913966	0.932102	0.913966	0.922592	0.913966	**All: 25 Solutions**	
Best Seed Meth.	P-Disp., Angle, Workload	Angle	Angle	P-Disp., Angle	Neighb., Angle, Workload	0.80	0.52

In Figure VII.7 we presented the ANOVA results. We can observe that the R-Sq has a very small value as well as the R-Sq adjusted. Under this condition, the ANOVA indicates that none of the factors impact the solution reported. We assume that the main reason are the ties in the solutions found by the algorithms and also among the seed methods. Also if we observe in Figure VII.8 the interaction plot does not show a clear pattern.

Two-way ANOVA: OF versus Heuristic, Seed

```
Source   DF         SS         MS     F      P
Heur.     4  0.0155594  0.0038899  1.51  0.260
Seed      3  0.0196472  0.0065491  2.55  0.105
Error    12  0.0308739  0.0025728
Total    19  0.0660806

S = 0.05072   R-Sq = 53.28%   R-Sq(adj) = 26.02%
```

Fig VII.7 Two-way ANOVA, S-SYM-T.

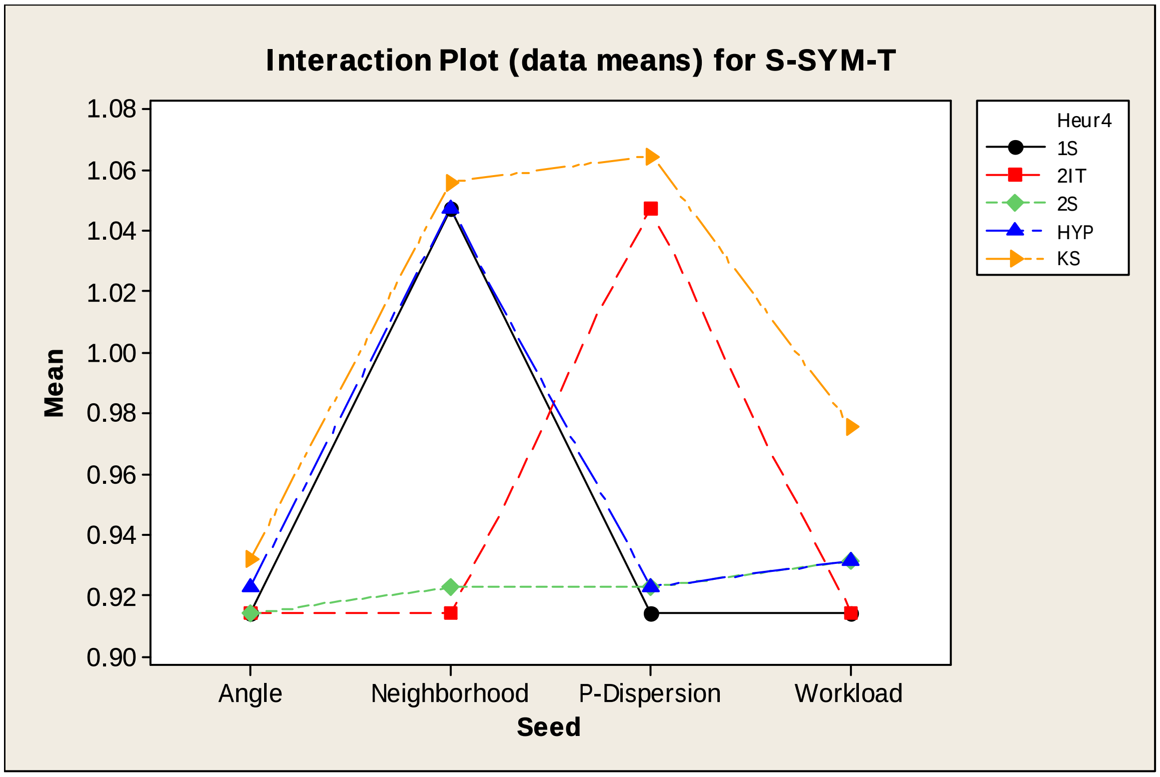


Fig VII.8 Interaction Plot, S-ASYM-T.

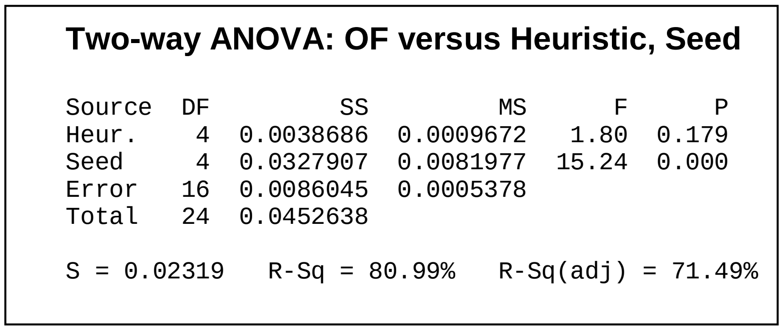
Symmetric and Less restricted instance (SYM-LR)

In this section we analyze the Symmetric instances, that differ from the Semi-Symmetric in that the optimal solution corresponds to symmetric districts always. Table VII.8 presents the results for the less restricted capacitated instance. It is possible to observe that the Workload method resulted the best for all the heuristics, and there is a tie only for the 2-S with the Angle method. We may also observe that all the Seed Methods found at least one feasible solution and only the Semi-Random and Neighborhood did not find the five feasible solutions. It is possible to observe also that for each seed method, at least there was an instance in which the F-R procedure was not required. This indicates that this type of instance because of the symmetry the F-R procedure is not required as much as for the Asymmetric instances for example.

Table VII.8 Objective Function values per Seed Method and Heuristic, SYM-LR.

SEED METHOD	**INSTANCE: SYMMETRIC AND LESS RESTRICTED**						
	1-S	**K-S/P**	**2-S**	**HypLS**	**2-IterLS**	**%FS**	**%FS (no F-R)**
P-Dispersion	1.062542	1.105157	1.044044	1.040222	1.025546	1.00	1.00
Semi-Random	0.972514	1.044044	0.972514	0.972514	1.017849	0.80	0.20
Neighborhood	0.972514	0.972514	0.963265	1.025546	0.972514	0.60	1.00
Angle	0.977116	0.973499	0.954016	0.972514	0.96425	1.00	0.40
Workload	0.954016	0.965728	0.954016	0.954253	0.916747	1.00	0.40
Best OF value	0.954016	0.965728	0.954016	0.954253	0.916747	**All: 25 Solutions**	
Best Seed Meth.	Workload	Workload	Angle, Workload	Workload	Workload	0.88	0.60

Figure VII.9 presents the ANOVA for this instance, we can observe that the Seed method resulted significative and impact the quality of the solution value reported. However, the Heuristic did not result significative, since the p-value is greater than the significance level of $\alpha=0.05$. Also if we observe in Table V.II. 8, results reported by the heuristics do not differ too much as for other types of instances. So this indicates that for this type of instance, all the heuristics could find solutions of relatively equal quality, but the seed method affects the solution that can be found. And also in Table V.II.8 we may observe that the gap between the best and worst solution found by a heuristic can be around of 10%.



Two-way ANOVA: OF versus Heuristic, Seed

```
Source  DF         SS         MS      F      P
Heur.    4  0.0038686  0.0009672   1.80  0.179
Seed     4  0.0327907  0.0081977  15.24  0.000
Error   16  0.0086045  0.0005378
Total   24  0.0452638

S = 0.02319   R-Sq = 80.99%   R-Sq(adj) = 71.49%
```

Fig VII.9 Two-way ANOVA, SYM-LR.

Figure VII.10 shows the interaction plot for both factors. We can observe that there is interaction between the seed method and the heuristic. And we also observe that the 2-IterLS algorithm found the best solution with the workload seed method.

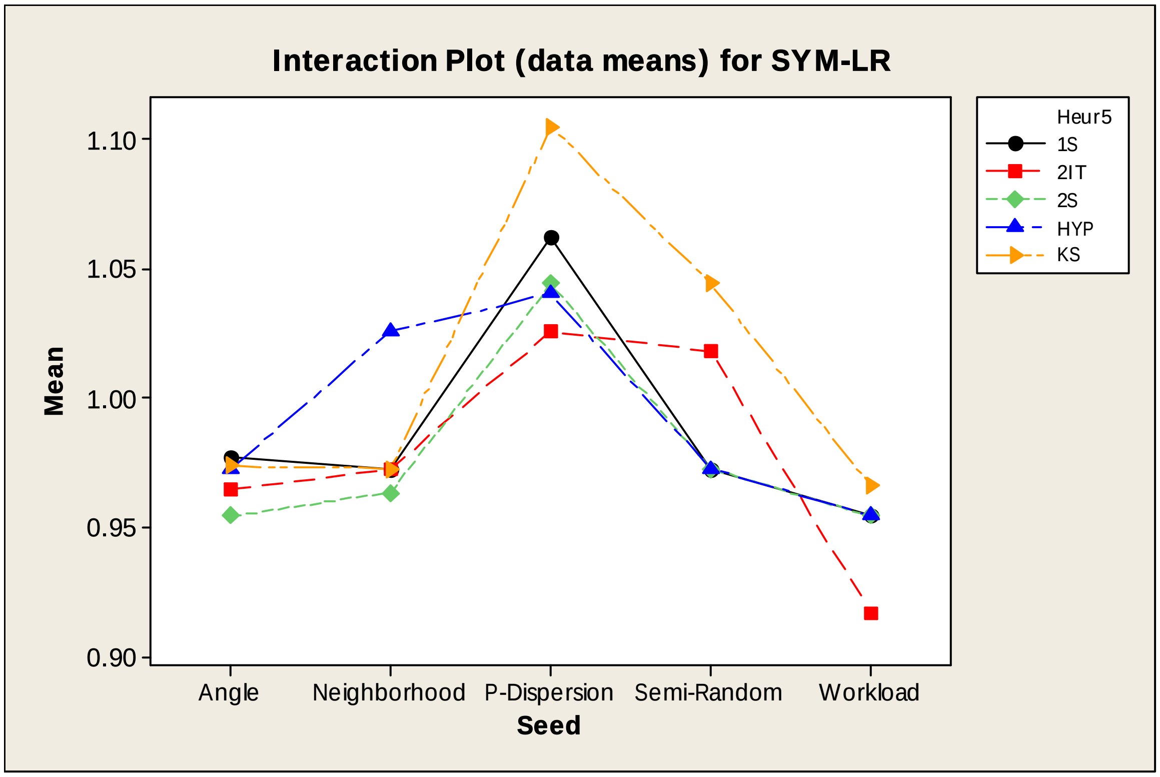


Fig VII.10 Interaction Plot, SYM-LR.

Symmetric and Tight instance (SYM-T)

Now we present the results for the symmetric instances with tight capacity limits in Table VII.9. We can observe that results differ too much with respect to the less restricted instance of this type, in which the best seed method resulted the Workload. However, for the tight capacitated instance, the P-Dispersion method resulted the best for all the algorithms and there is no tie among the results of the seed methods. All the seed methods found at least one initial feasible solution and even though the P-Dispersion method resulted the best, it failed to construct one of the five feasible solutions, and all of them required the F-R procedure. We can also observe that the 2-S and 1-S both found the best solution.

Table VII.9 Objective Function values per Seed Method and Heuristic, SYM-T.

SEED METHOD	**INSTANCE: SYMMETRIC AND TIGHT**						
	1-S	**K-S/P**	**2-S**	**HypLS**	**2-IterLS**	**%FS**	**%FS (no F-R)**
P-Dispersion	0.884749	0.930824	0.884749	0.921575	0.893998	0.80	0.00
Semi-Random	0.972514	1.000827	0.972514	0.981763	0.981763	0.20	0.00
Neighborhood	0.963265	0.965057	0.954253	0.963265	0.963265	1.00	0.40
Angle	0.954016	0.973499	0.954016	0.972514	0.954742	0.80	0.20
Workload	0.972514	1.000857	0.96425	0.935755	0.954253	1.00	0.40
Best OF value	0.884749	0.930824	0.884749	0.921575	0.893998	**All: 25 Solutions**	
Best Seed Meth.	P-Disp.	P-Disp.	P-Disp.	P-Disp.	P-Disp.	0.76	0. 20

Figure VII.11 presents the Two-way Anova. We can observe that the Seed method is significant and impacts the quality of the solution reported, but the heuristics are not significant because the p-value is equal to 0.179. This is similar as in the less restricted case of this type of instance, and we think that it is mainly because there is ties in the solutions reported by the heuristics and results do not differ too much. Figure VII.12 present the Interaction Plot of both factors, and we can observe that there is small interaction, but not as significant as was observed for other instances.

Two-way ANOVA: OF versus Heuristic, Seed

```
Source  DF         SS         MS      F      P
Heur.    4  0.0038686  0.0009672   1.80  0.179
Seed     4  0.0327907  0.0081977  15.24  0.000
Error   16  0.0086045  0.0005378
Total   24  0.0452638

S = 0.02319   R-Sq = 80.99%   R-Sq(adj) = 71.49%
```

Fig VII.11 Two-way ANOVA, SYM-T.

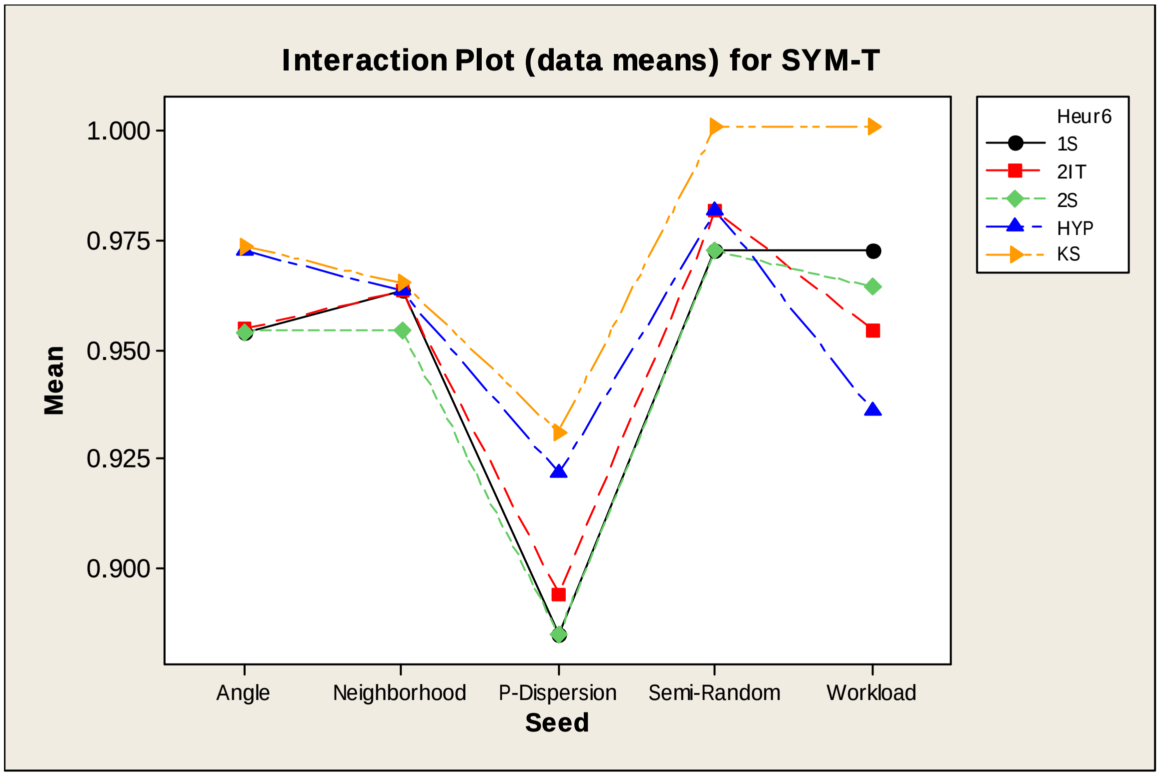


Fig VII.12 Interaction Plot, SYM-T.

Urban and Less Restricted instance (URB-LR)

In this section we present the analysis for the urban type instances with the less restricted capacity limit. This type of instances resulted the more difficult to construct a feasible solution. In Table VII.10 we can observe that only the Neighborhood and Workload methods were able to construct at least one initial feasible solution. And all the solutions constructed required the F-R procedure. This indicates that this type of instance was the most difficult, because the specific characteristics in which we attempt to create instances that resembles the structure of the Metropolitan region of Monterrey. We also can observe that the Workload resulted with the best solutions for all the heuristics, and that the 2-S found the best solution for this type of instance followed by the 2-IterLS.

Table VII.10 Objective Function values per Seed Method and Heuristic, URB-LR.

SEED METHOD	**INSTANCE: URBAN AND LESS RESTRICTED**						
	1-S	**K-S/P**	**2-S**	**HypLS**	**2-IterLS**	**%FS**	**%FS (no F-R)**
P-Dispersion						0.00	0.00
Semi-Random						0.00	0.00
Neighborhood	0.993024	1.008811	0.993024	0.992948	0.986883	0.20	0.00
Angle						0.00	0.00
Workload	0.848623	0.87572	0.799966	0.831518	0.825846	0.40	0.00
Best OF value	0.848623	0.87572	0.799966	0.831518	0.825846	**All: 25 Solutions**	
Best Seed Meth.	Workload	Workload	Workload	Workload	Workload	0.12	0

Figure VII.13 presents the ANOVA results, for which we only include the Neighborhood and Workload seed methods. Results indicates that the seed are significant and have an impact on the quality of the solution but the heuristics does not. This is mainly because the solutions found by the heuristics do not differ too much. We also present in Figure VII.14 the Interaction plot for both factors, from which we can observe that there is no interaction between the heuristics and seed methods.

```
Two-way ANOVA: OF versus Heuristic, Seed

Source   DF         SS         MS       F      P
Heur      4  0.0023999  0.0006000    2.34  0.216
Seed      1  0.0628876  0.0628876  244.98  0.000
Error     4  0.0010268  0.0002567
Total     9  0.0663143

S = 0.01602   R-Sq = 98.45%   R-Sq(adj) = 96.52%
```

Fig VII.13 Two-way ANOVA, URB-LR.

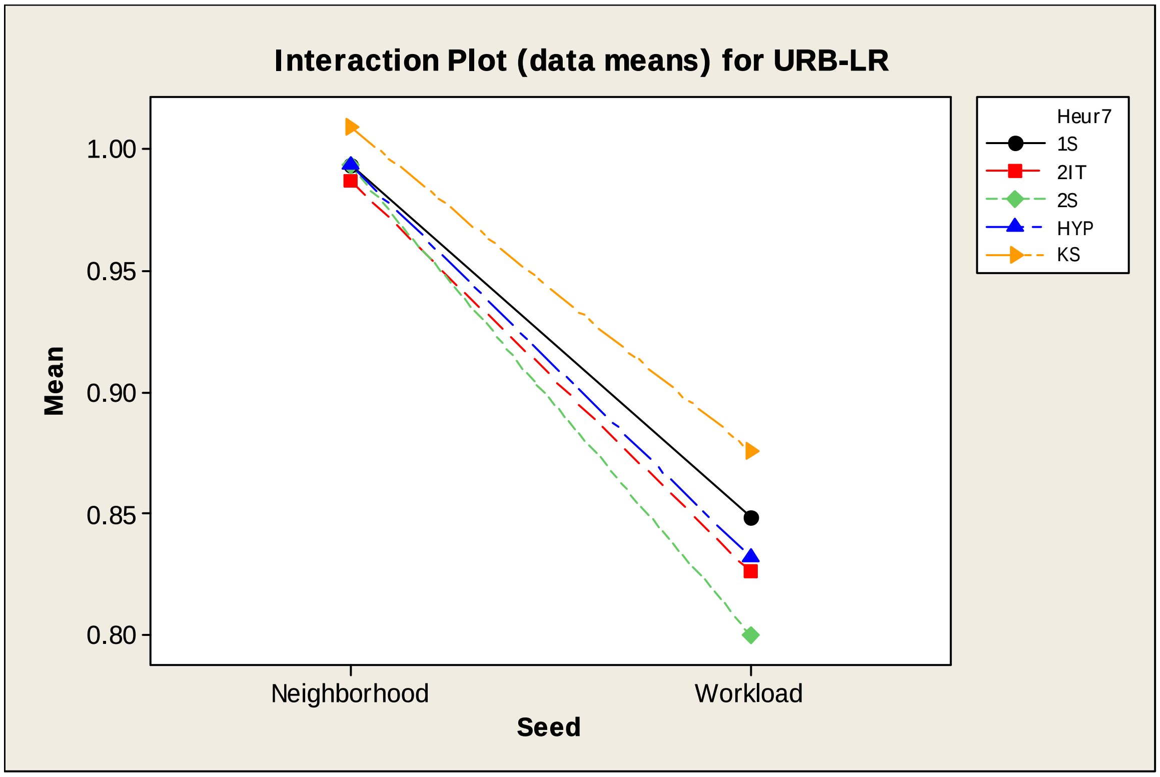


Fig VII.14 Interaction Plot, URB-LR.

Urban and Tight instance (URB-T)

The last instance analyzed is the Urban with tight capacity limits. Table VII.11 presents the results, which are similar to the less restricted case, because only the Neighborhood and Workload methods were able to find a feasible solution. Also all the solutions constructed required the F-R procedure, and the best solution corresponds to the Workload method. It is possible to observe that the 2-S reported the best solution, and for this heuristic solutions found by each seed method do not differ too much.

Table VII.11 Objective Function values per Seed Method and Heuristic, URB-T.

SEED METHOD	**INSTANCE: URBAN AND TIGHT**						
	1-S	**K-S/P**	**2-S**	**HypLS**	**2-IterLS**	**%FS**	**%FS (no F-R)**
P-Dispersion						0.00	0.00
Semi-Random						0.00	0.00
Neighborhood	1.036851	1.036834	0.880663	1.036775	0.903754	0.20	0.00
Angle						0.00	0.00
Workload	0.893199	0.91099	0.83338	0.843343	0.861466	0.40	0.00
Best OF value	0.893199	0.91099	0.83338	0.843343	0.861466	**All: 25 Solutions**	
Best Seed Meth.	Workload	Workload	Workload	Workload	Workload	0.12	0.00

Figure VII. 15 presents the ANOVA results, in which none of the factors resulted significant for this type of instance. However, we must recall that only 2 of the seed methods found a feasible solution. Therefore, the seed methods actually have a big impact on the solution reported and even more in the ability to construct a feasible solution. This indicates that it is the most difficult type of instance. We can also observe in Figure VII.16 the interaction plot of the factors, considering only the seed methods that could construct a feasible solution. In this case we observe more interaction than for the less restricted capacitated instance of the urban type.

Two-way ANOVA: OF versus Heuristic, Seed

```
Source  DF         SS         MS      F      P
Heur     4  0.0212620  0.0053155   2.52  0.196
Seed     1  0.0305255  0.0305255  14.48  0.019
Error    4  0.0084307  0.0021077
Total    9  0.0602183

S = 0.04591   R-Sq = 86.00%   R-Sq(adj) = 68.50%
```

Fig VII.15 Two-way ANOVA, URB-T.

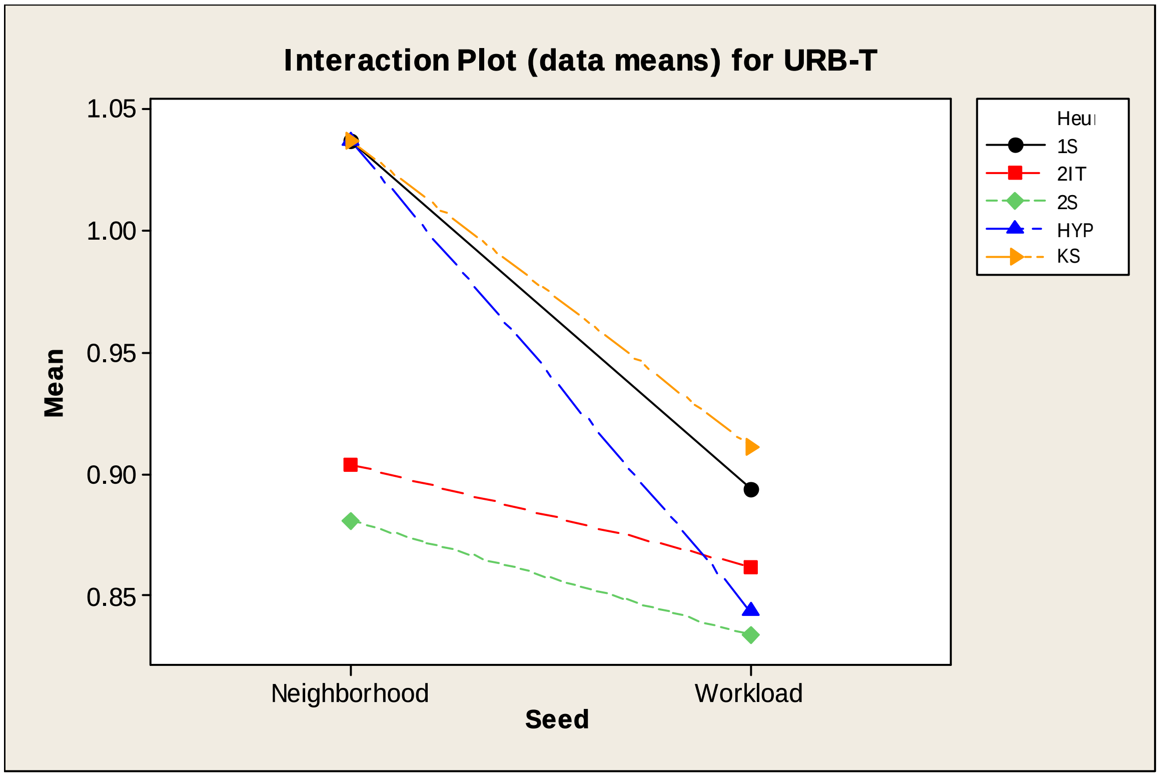


Fig VII.16 Interaction Plot, URB-T.

Finally, based on the statistical analysis that we performed, we can conclude that the seed method has a big impact on the quality of the solutions reported by the heuristics, and also in the ability to construct a feasible solution. The best seed method varies according to the instance type and capacity restrictiveness. Also, the heuristics impact on the quality of the solution, and either the 2-S or 2-IterLS resulted with the best solutions in general.