Doctoral Thesis

Hybrid Methodologies for Symmetric and Asymmetric Vehicle Routing Problems

Author
Rosa Herrero Antón

Advisor
Dr. Juan José Ramos González

PhD Program in
Informàtica Industrial: Tècniques Avançades de Producció
Departament de Telecomunicació i d'Enginyeria de Sistemes
Universitat Autonòma de Barcelona

Bellaterra, November 2015

Dr. Juan José Ramos González, Associate Professor at the Universitat Autònoma de Barcelona,

CERTIFIES:

That the thesis entitled **Hybrid Methodologies for Symmetric and Asymmetric Vehicle Routing Problems** and submitted by **Rosa Herrero Antón** in partial fulfillment of the requirements for the degree of Doctor, has been developed and written under his supervision.

Dr. Juan José Ramos González	Rosa Herrero Antón
Thesis Director	PhD Student

PhD Program in
Informàtica Industrial: Tècniques Avançades de Producció
Departament de Telecomunicació i d'Enginyeria de Sistemes
Universitat Autonòma de Barcelona

Sabadell, November 2015

"Time is life itself, and life resides in the human heart."
Michael Ende, Momo

"Someone is sitting in the shade today because
someone planted a tree a long time ago."
Warren Buffett

Acknowledgement

First, I am greatful to my advisor, Dr. Juanjo Ramos for his support along this thesis work, and also for giving me the opportunity to participate in very interesting experience during these last years.

There are two professors that I would especially like to express my gratitude. The first one is for me my second advisor Dr. Angel A. Juan, who guided and encouraged me during my PhD, even also giving me the chance to stay at Universitat Oberta de Catalunya. The second one is Dr. Marco Gavanelli, who led and supported me during my stay at Università degli Studi di Ferrara, I marveled at his technical research. Probably, this PhD thesis would have not been possible without their help.

I also want to thank to my colleagues Dr. Daniel Guimarans, Dr. Massimiliano Cattafi, Dr. José Cáceres for their willingness to help me, for their support in our articles and for the time enjoyed together.

Thanks to my mates who have shared with me all these years, some years at UAB and others despite the distance: Catya Zúñiga, José David Rojas, Adriana Ramírez, Héctor Delgado, Helena Martín, Lars Peters, Luis Blanco, Monica Gutierrez, David Bote... Thank you guys for the coffee time and so many interesting discussions, mostly without any relevance to this thesis, but otherwise essential for happiness!

I cannot forget to express my gratitude again to Catya Zúñiga, who I am lucky to count among my closest friends. Thank you for your friendship and love, without you finishing this thesis would have been almost impossible.

Por último, pero el más importante de todos, quiero agradecer a mi marido Javi su paciencia y amor, y también a toda mi familia, especialmente a mis padres Cruz y Remi, y a mi tío José María.

Gracias a todos!

Brief Contents

Contents

List of Tables and Figures

List of Algorithms

Introduction

Over the last decades, globalization has driven the adaptation of the transport and logistics sector to new social demands. At the same time, transport has been the backbone of globalization. This social need creates ambitious consumers who need their products quickly and an affordable price often unaware of their origin, transport mode or environmental aspects, among other factors. Nevertheless, to satisfy customer demands, it is needed to find the cheapest transport mode, which in turn means the improvement of transport logistics of the products. Therefore, these demands require an increasingly flexible service to meet customer requirements, and in addition companies want an efficient and productive transport.

In addition, it is important to know what kind of product demand is considered. Mainly, there are two types of demands: monotonous demand that requires a stable transport throughout months; and a more volatile demand which will depend on some perishable factors, such as seasons, trending or brief consumer need. This second demand has to be followed by a flexible transport network which has to be adaptable to new circumstances. It must also be considered the dependence on the product characteristics, such as dimensions and weight, or if any special condition is needed for fragility or conservation.

Furthermore, according to the previously mentioned factors, it must be selected the proper transportation mode. Transport could be carried out by sea, air or land. Air and sea routes are commonly used for large quantities or long distances, their itineraries are established and are not flexible. In contrast, land transport is really flexible and it allows to quickly adapts to the customer demand.

Therefore, road transport is a vital link in the increasingly complex sector of transport and logistics, which takes freight to ports and terminals and distribute urban goods between warehouses and retail outlets, in order to arrive to the customers.

The sector of transport and logistics in the Spanish economy represents the 3% of Gross Domestic Product (GDP). The total cost of this sector was estimated about 25,000 M€ in 2011. Additionally, freight transport represents 60% of the total cost which was about 15,000 M€ (OTLE, 2015).

The Spanish freight transport is characterized as predominantly national, with a share of 71% compared to 29% international. Road traffic transports 94.6% of the total tones nationwide in 2013 (OTLE, 2015).

Transportation costs cannot be neglected, improving the competitiveness of road

transport is essential for companies in this sector, not only at Spanish scale but also at worldwide. In Europe, transportation often accounts for between one-third and two-thirds of total logistics costs –i.e., between 9% and 10% of the Gross National Product (GNP) for the Europe economy and also between 10% and 20% of a products price, so transportations importance and key role is undeniable (Khooban, 2011).

The need for optimizing the road transportation affects to both public and private sectors, given that it is an ambivalent issue: on the one hand, it brings great benefits for individuals and economies; on the other hand, it may bring some negative side effects on quality of life and health, such as noise or air pollution, increasing greenhouse gases, waste products, even accidents death injuries.

Traveling Salesman Problems (TSP) and *Vehicle Routing Problems* (VRP) deal with the physical distribution of goods from a central depot to customers. The TSP was first formulated by Karl Menger (1930), the origin of the name "traveling salesman problem" is a bit of a mystery without any recognized creator. Nevertheless, the VRP first appeared defined by Dantzig and Ramser (1959). A wide number of related problems have been developed during the last decades, each of them considering different sets of characteristics and constraints. Usually, the main goal of this set of problems is to minimize distance-based costs associated with the distribution of products among customers while satisfying customers' demands.

Both problems belong to the field of combinatorial optimization, and they are one of the most challenging problems because of they belong to the NP-Hard problems class, (Lenstra & Kan, 1981), meaning that they are not solvable in polynomial time, i.e., there is no efficient algorithm to solve these problems that has been proved to solve correctly all scenarios, and whose worst-case running time is bounded by a polynomial time function which depends on the scenarios' size.

According to K. L. Hoffman et al. (2013), the TSP has commanded much attention of mathematicians and computer scientists specifically because it is easy to describe and really difficult to solve, in addition it has a lot of applications in the real life. Therefore, this problem has been a great engine of discovery for general purpose techniques in applied mathematics (Applegate et al., 2006). Some areas to which TSP research has made fundamental contributions are: Mixed-Integer Programming, Branch-and-Bound method, and some metaheuristics such as Local-Search algorithms, Simulated Annealing, Neural Network, and Genetic Algorithms.

The TSP algorithm developed by Held and Karp (1970) carried the best-known guarantee on the running time of a general solution method for the problem for over 40 years. Recently, a TSP solver named Concorde developed by Applegate et al. (2015) is the best-known exact method, it has been used to obtain the optimal solutions to 106 of the 110 *TSPLIB* instances; the largest having 85,900 cities. Currently, the heuristic of Lin and Kernighan (1973) effectively modified by Helsgaun (2000) holds the record for the best instances of problems with unknown optimal (*DIMACS TSP Challenge 2002*) with sizes ranging from 1,000 to 10,000,000 nodes.

The VRP is also one of the most researched problems, due to this field has been exploited dramatically partly driven by the industrial applicability. In the early years, spe-

cialized heuristics were typically developed for solving the VRP. Then, more generic solution schemes as metaheuristics were designed, among them it can be found research about Ant Colony Optimization, Genetic Algorithms, Greedy Randomized Adaptive Search Procedure, Simulated Annealing, Tabu Search, and Variable Neighborhood Search. The interest about hybrid optimization methods has grown very fast for the last decade.

From the industrial point of view, these problems characterize a family of different distribution problems which, one way or another, are presented in real problems. However, most of the real applications are not represented by the classical variants, for instance, most VRP related academic articles assume the existence of a homogeneous fleet of vehicles and/or a symmetric cost matrix. These assumptions are not always reasonable in real-life scenarios. In real problems, it is needed to consider real-life constraints, obtaining problems which are commonly known as *Rich VRP*.

Theoretical researches typically assume the symmetry of the distance-based costs associated with traveling from one place to another. In fact, classical benchmark instances are based on Euclidean distances between each pair of locations, which result in symmetric costs. However, this metric is just a lower bound of the real distance between two nodes connected by a transport network or highway. The real distance will depend upon the specific location of the nodes in the territory and also on the structure of the road network that connects them, which commonly are oriented networks, Rodríguez and Ruiz (2012b) suggest that real distances might not have to be symmetric. Furthermore, Rodríguez and Ruiz (2012a) demonstrate and measure, symmetric solutions (those obtained with symmetric and Euclidean distance matrices) have little in common with regard to sequence and total distance with real solutions (those obtained with asymmetric and real distances).

Another frequent assumption is the existence of a homogeneous fleet of vehicles with limited capacity. However, most road-transportation companies own a heterogeneous fleet of vehicles. This diversity in the vehicles' capacity might be due to the fact that different customers and locations might require different types of vehicles, e.g., narrow roads in a city, available parking spaces, vehicle weight restrictions on certain roads, etc. Another reason for owning vehicles with distinct capacities is the natural diversity that arises when vehicle acquisitions are made over time.

In this scenario, it becomes evident the need of developing new methods, models and systems to give support to the decision-making process so that optimal strategies can be chosen in road transportation.

The main goal of this thesis is to introduce hybrid methodologies that integrate several techniques to efficiently solve rich Vehicle Routing Problems with realistic constraints. This thesis is outlined in Figure 1, it starts with theoretical problems and evolves into more realistic scenarios tackling six combinatorial problems related to road transport. It is explored the potential of the *Lagrangian Relaxation* (LR) for solving rich and realistic problems. Due to the evolution of the scenarios chosen –from Traveling Salesman Problem to Asymmetric and Heterogeneous Vehicle Routing Problem– LR is combined into a hybrid method adding new techniques when required.

This thesis starts addressing the TSP, which is a well-known theoretical problem.

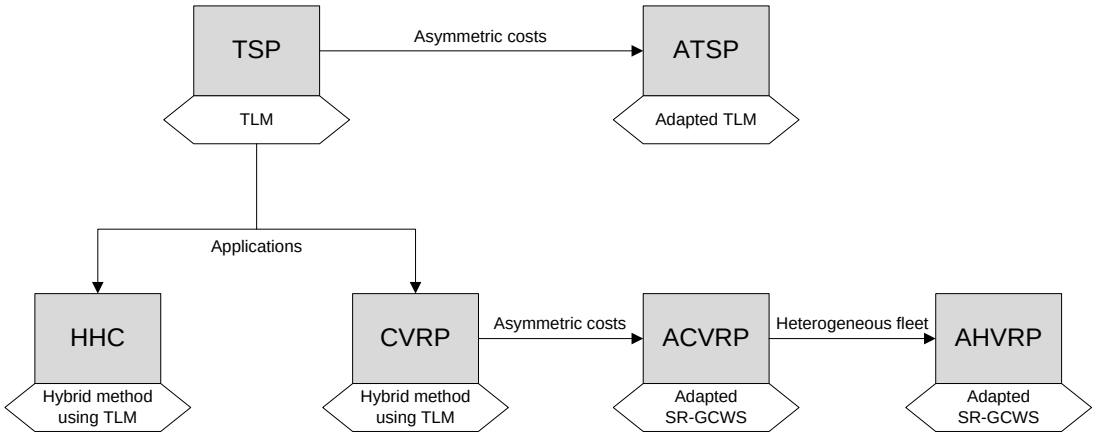


Figure 1: General outline of this thesis.

The problem deals with finding the shortest path of a salesman who is required to visit once and only once each different customers starting from a depot, and returning to the same depot. LR is used to exploit the structure of the problem reducing considerably its complexity by moving hard-to-satisfy constraints into the objective function, associating a penalty in case they are not satisfied. For that purpose, a metaheuristic –named Tailored Lagrangian Metaheuristic (TLM)– based on the Lagrangian Relaxation is developed.

The Asymmetric version of the TSP (ATSP) has been chosen in order to illustrate the effects of the frequent assumption of the symmetry of the distance-based costs associated with traveling from one place to another. Therefore, the proposed TLM is adapted for the asymmetric scenarios.

The Capacitated Vehicle Routing Problem (CVRP) consists of determining the optimal set of routes for a fleet of vehicles to deliver goods to a given set of customers, it is a generalization of the TSP. In the model proposed in this thesis, the CVRP has been divided into two subproblems, concerning customers' allocation and routing optimization separately. The first one aims to assign customers to vehicles fulfilling capacity limitations. Then, it is used to solve each independent route giving the best solution for a particular allocation. Thus, routing optimization process can be viewed as solving a set of independent symmetric TSP. A hybrid approach proposes a *Multi-Start Variable Neighborhood Descent* structure whose local search process is supported by *Constraint Programming* (CP) for solving the customers' allocation and our TLM metaheuristic for solving independently each TSP.

It is presented a practical application concerning the *Home Health Care* (HHC) service in the municipality of Ferrara, Italy. This real problem consists on assigning patients' services to nurses which travel to each patient's home. Therefore, it is defined the nurse itineraries which considers the following optimization aspects: the nurse workloads are balanced, patients are preferentially served by a single nurse or just a few ones, and

the overall travel time is minimized. A hybrid methodology based on CP and our TLM is considered for addressing this real problem.

Given the fact that urban networks are asymmetric and to contribute to closing the gap between theory and practice, the Asymmetric version of the CVRP (ACVRP) has been considered. A hybrid methodology based on the *randomized Clarke and Wright Savings* algorithm (SR-GCWS), developed by Juan et al. (2010), is adapted for the asymmetric scenarios. The Clarke and Wright Savings heuristic (CWS), presented by Clarke and Wright (1964), is one of the most commonly cited methods in the VRP literature. Our proposed algorithm combines a randomized savings heuristic with two local search processes specifically designed for the asymmetric nature of costs in real-life scenarios.

The ACVRP with *heterogeneous fleet of vehicles* (AHVRP) has been chosen in order to illustrate the effects of the frequent assumption of the existence of a homogeneous fleet of vehicles. Our proposed Asymmetric SR-GCWS is combined with the modified Clarke and Wright Savings heuristic presented by Prins (2002) for the HVRP. It is analyzed how routing costs vary when slight deviations from the homogeneous fleet are considered, i.e., how marginal costs/savings change when a few 'standard' vehicles in the homogeneous scenario are substituted by other vehicles with different loading capacity.

Objectives

The objectives of this thesis can been summarized as follows:

- The development of a metaheuristic aimed to tackle the Traveling Salesman Problem based on the Lagrangian Relaxation.
- This metaheuristic should be efficient regarding the solution values and the computational time. In addition, it should be flexible to be adaptable to the asymmetric scenarios.
- The integration of the developed metaheuristic into hybrid methodologies to tackle more complex problems, like the Capacitated Vehicle Routing Problem.
- The application of the developed metaheuristic in a realistic scenario within a hybrid methodology, like the application in the Home Health Care in Ferrara, Italy.
- The study of different variants focusing on the impact that causes the asymmetry of the costs and the heterogeneity of the fleet.
- The development of a hybrid methodology adapted to deal with the asymmetric nature of costs, more concretely, to solve the Asymmetric Capacitated Vehicle Routing Problem.
- The adaptation of the developed methodology considering heterogeneous fleets for solving the Asymmetric and Heterogeneous Vehicle Routing Problem.

Synopsis

The present thesis is structured in two parts, motivated by the fact that not all approaches work efficiently in both Symmetric and Asymmetric problems (Rodríguez & Ruiz, 2012b). Part I tackles the followings Symmetric Problems: the Traveling Salesman Problem, the Capacitated Vehicle Routing Problem, and the Home Health Care problem. Part II discusses Asymmetric Problems: the Asymmetric Traveling Salesman Problem, the Asymmetric Capacitated Vehicle Routing Problem, and the Asymmetric and Heterogeneous Vehicle Routing Problem.

The Symmetric Problems are structured as follows. Chapter 1 describes the Traveling Salesman Problem. It includes a description of its variants and applications, in addition to a review of the state of the art on TSP. It also includes an experiment of the behavior of the Subgradient convergence in a TSP problem. Then, our proposed Tailored Lagrangian Metaheuristic to solve the Traveling Salesman Problem is presented. It is described together with some unpublished results which are compare with optimal or best known solutions. The next two chapters (2-3) are applications of our proposed metaheuristic which has been included in hybrid methodologies using Constraint Programming in order to solve these complex problems. In chapter 2, the Capacitated Vehicle Routing Problem is solved including our metaheuristic in a Multi-Start Variable Neighborhood Descent approach. This chapter introduces a literature review on CVRP, and explains the adopted approach in detail. Finally, it is presented some numerical experiments and the main benefits of the presented approach are discussed. Chapter 3 describes an application concerning Home Health Care in the city of Ferrara, Italy, and how it can be addressed with a hybrid methodology using our metaheuristic and Constraint Programming. This problem is focused on societal needs and the balance with human and economic aspects.

The Asymmetric Problems are structured as follow. Chapter 4 adapts the proposed Tailored Lagrangian Metaheuristic to the Asymmetric Traveling Salesman Problem. It discusses the difference respect to the symmetric problem and presents unpublished results. Chapter 5 proposes a hybrid methodology for solving the Asymmetric Capacitated Vehicle Routing Problem. It combines a randomized version of a well-known Savings heuristic with local searches specifically adapted to deal with the asymmetric nature of costs. A computational experiment allows us to discuss the efficiency of this approach. Chapter 6 applies this hybrid methodology to the Asymmetric and Heterogeneous Vehicle Routing Problem and analyzes how routing costs vary when slight departures from the homogeneous fleet assumption are considered.

Finally, conclusions and contributions of this thesis together with related publications and future lines of research are presented in last chapter on page 133.

Methodologies

Lagrangian Relaxation

Lagrangian Relaxation (LR) is a well-known method to solve large-scale combinatorial optimization problems, which is commonly used to generate lower bounds. It was named for the French mathematician Joseph Louis Lagrange, presumably due to the occurrence of what we now call Lagrange multipliers in his calculus of variations, (Boyer, 1985). LR works by moving hard-to-satisfy constraints into the objective function, associating a penalty in case they are not satisfied.

LR is strongly related to the earlier decomposition method of Dantzig and Wolfe (1960). However, the origin of the Lagrangian approach, as it exists today, was developed by Held and Karp (1970, 1971). An excellent introduction to LR can be found in (Fisher, 2004) along with a discussion of some early examples of Lagrangian heuristics. For a theoretical review of LR and a review of some methods for the dual problem like the Subgradient algorithm, dual ascent methods, cutting plane method and column generation, see (Guignard, 2003).

LR is useful for exploiting the structure of the problem to reduce considerably the problem complexity. Thus, the Lagrangian Problem needs less computational effort to find solutions. However, one of its main drawbacks consist on finding optimal dual variables normally called Lagrangian multipliers. Furthermore, for a given problem, there may exist different Lagrangian relaxations depending on the relaxed constraint.

When the Lagrangian Problem can be formulated as an Integer Linear Problem or a Mixed Integer Problem, then it can be solved in a optimal way by embedding LR into the Branch-and-Bound (Geoffrion, 1974) or the Branch-and-cut (Edmonds, 1967) approaches. These methods provide both upper and lower bounds to the problem, and ensure that the optimal solution is found. However, if the problem is NP-hard, large problems are not expected to be solved in a reasonable time.

In other cases, especially when the dual function is non-differentiable, some techniques can be useful to maximize the dual function of the Lagrangian relaxation, such as the subgradient, bundle and surrogate. This techniques are simple and easy to implement and avoid using linear programming approaches. The Lagrangian multipliers start with initial values and direct them toward the optimal value, this direction is called subgradient. However, these techniques are often terminated before an optimal value is attained, offering only a lower bound of the best objective value.

Within these techniques, the most widely used is the Subgradient Optimization algorithm (Shor, 1985). If the relaxed problem can be minimized obtaining a subgradient, then it guarantees convergence where the Lagrangian multipliers are updated along the subgradient direction. One of the main drawbacks, remarked by Bertsekas (1999), is the need of solving optimally all the subproblems, which may be slow to be of real practical interest.

On the other side, the Surrogate Gradient technique needs only an approximate solution of the subproblem, so it may be desirable to obtain a proper direction with less effort. Nevertheless, it does not guarantee that the best bound obtained is equal to the optimal value (Zhao et al., 1999).

Finally, the Bundle technique can provide better directions than the Subgradient Algorithm. However, according to Hiriart-Urruty and Lemarechal (1993), to obtain each direction require solving the subproblem many times. Subgradients from past iterations are accumulated in a bundle, and a trial direction is obtained by quadratic programming based on the bundle information, for more information see (Kiwiel, 1996). This technique has not been of our interest given that it can increase the computational effort.

Literature review and Applications

Lagrangian Relaxation has been widely used in diverse applications, figure 2 reprinted a table from Fisher (2004, p. 1864). This table shows the first applications of LR in the 70's. It can be noticed how for one problem there may exist different Lagrangian relaxations depending on the relaxed constraint. Within this section a breve explanation of the most prominent applications will be given, we focus on the use of LR in both exact methodologies and heuristic approaches.

Regarding to *Manufacturing Scheduling Problems*, Luh and Hoitomt (1993) achieve good decomposition of three scheduling problems relaxing one or more sets of constraints: the first problem considers scheduling single-operation jobs in identical machines; the second problem deals with scheduling multiple-operation jobs with simple fork / join precedence constraints on identical machines; lastly, the *Job Shop Problem* is considered, where multiple-operation jobs with general precedence constraints are scheduled on multiple machine types. Nishi et al. (2010) address a LR-based method with cut generation for solving the hybrid flowshop scheduling problem to minimize the total weighted tardiness. Mao et al. (2014) study a real-world hybrid flowshop problem arising from the steel-making continuous casting process, which is the bottleneck of the iron and steel production process. Their paper is based on a time-index formulation and machine capacity relaxation, three LR subproblems are presented for addressing this scheduling problem: job-level problems, batch-level problems, and machine-level problems.

To enable efficient transportation in manufacturing systems, it is necessary to generate route planning of multiple Automated Guided Vehicles (AGVs) efficiently without collision among them. A mixed integer programming model for AGVs planning and control to minimize the total material handling cost is developed in (M. Chen, 1996) using a

Table 1 Applications of Lagrangian Relaxation

Problem	Researchers	Lagrangian problem
Traveling Salesman		
Symmetric	Held and Karp (1970, 1971)	Spanning tree
	Helbig Hansen and Krarup (1974)	Spanning tree
Asymmetric	Bazarra and Goode (1977)	Spanning tree
Symmetric	Balas and Christofides (1976)	Perfect 2-matching
Asymmetric	Balas and Christofides (1976)	Assignment
Scheduling		
n/*m* Weighted tardiness	Fisher (1973)	Pseudo-polynomial dynamic programming
1 Machine weighted tardiness	Fisher (1976)	Pseudo-polynomial DP
Power generation systems	Muckstadt and Koening (1977)	Pseudo-polynomial DP
General IP		
Unbounded variables	Fisher and Shapiro (1974)	Group problem
Unbounded variables	Burdet and Johnson (1977)	Group problem
0-1 variables	Etcheberry et al.(1978)	0-1 GUB
Location		
Uncapacitated	Cornuejols et al.(1977)	0-1 VUB
	Erlenkotter (1978)	0-1 VUB
Capacitated	Geoffrion and McBride (1978)	0-1 VUB
Databases in computer networks	Fisher and Hochbaum (1980)	0-1 VUB
Generalized assignment		
	Ross and Soland (1975)	Knapsack
	Chalmet and Gelders (1976)	Knapsack, 0-1 GUB
	Fisher et al.(1980)	Knapsack
Set covering-partitioning		
Covering	Etcheberry (1977)	0-1 GUB
Partitioning	Nemhauser and Weber (1978)	Matching

Figure 2: An original Table 1 from Fisher (2004, p. 1864). Copyright ©2004 INFORMS.

decomposition approach following the LR method. Rajagopalan et al. (2004) tackle the problem of simultaneously locating the pick-up/drop-off points along the periphery of a cell and determining the flowpath for an AGV based material handling system. Nishi et al. (2005) propose a distributed route-planning method for multiple mobile robots using an augmented Lagrangian decomposition and coordination technique. Their approach was also proposed for a distributed decision making system for integrated optimization of production scheduling and distribution planning, (Nishi et al., 2007).

The successful commercialization in the electric power industry depends on *Power Generation Systems* which is capable of capturing and producing power efficiently. In (Nowak & Römisch, 2000), a stochastic LR was applied for the weekly cost-optimal generation of electric power in a hydro-thermal generation system under uncertain load. Papavasiliou et al. (2011) describe wind power generation in terms of a representative set of appropriately weighted scenarios; they present a LR dual decomposition algorithm for solving the resulting stochastic problem. Z. Li and Shahidehpour (2003) describe a scheduling method for representing the thermal stress of turbine shafts as ramp rate constraints in the thermal commitment and dispatch of generating units using LR for optimal generation scheduling.

The *Facility Location Problem* is one of the strategic logistical drivers within the

supply chain which consist on locating a set of facilities and how to satisfy customers' demands from these open facilities so that the total cost which includes the facility set up cost as well as the transportation cost is minimized. Nezhad et al. (2013) use a LR heuristic for the uncapacitated single-source multi-product facility location problem. Gendron et al. (2013) present a Lagrangian-based Branch-and-Bound algorithm for the two-level uncapacitated facility location problem with single-assignment constraints. Escobar et al. (2014) propose a granular variable Tabu Neighborhood Search for the capacitated location-routing problem which uses LR method to group the customers into clusters in the location phase. Jena et al. (2014) propose a LR heuristic for large-scale dynamic facility location with generalized modular capacities, that is, a multi-period facility location problem in which the costs of capacity changes are defined for all pairs of capacity levels.

The usage of the Internet has grown substantially in recent times, this has resulted in high volumes of data traffic. Replication of content and placing them on multiple servers is a method that is used to reduce latency. The *Data Location Problem* in information networks deals with placing content so as to achieve better cost performance. Gavish and Pirkul (1986) propose a branch-and-bound algorithm where the bounds are computed by LR followed by subgradient optimization for the data allocation problem and server location problem. Nguyen et al. (2005) propose the *Overlay Distribution Network* as a cost-effective means to deliver the entertainment services which are expected to be increasingly interactive, on demand and personalized. They solved this problem with an efficient LR heuristic where content clustering is employed to improve the heuristic run time. Applegate et al. (2010) find a near-optimal solution with orders of magnitude speedup relative to solving even the LP relaxation via standard software for the optimal content placement for a large-scale video-on-demand system.

Given certain jobs and certain agents, the *Generalized Assignment Problem* deals with assigning each job to exactly one agent satisfying a resource constraint for each agent in order to determine a minimum assignment cost. Posta et al. (2012) propose an exact algorithm for solving the generalized assignment problem by a simple depth-first Lagrangian branch-and-bound method, improved by variable-fixing rules to prune the search tree. Wu et al. (2014) propose a Lagrangian-based branch-and-cut algorithm for the generalized assignment problem with min-max regret criterion under interval costs. Hanada and Hirayama (2011) use a distributed LR protocol for the over-constrained generalized mutual assignment problem where, with no centralized control, multiple agents search for an optimal assignment of goods that satisfies their individual knapsack constraints.

In a cross-dock, goods are unloaded from incoming trucks, consolidated according to their destinations, and then, loaded into outgoing trucks; all these with little or no storage in between, the objective is minimizing the total material handling cost. Nassief et al. (2015) present a new mixed integer programming formulation which is embedded into a Lagrangian relaxation that exploits the special structure of the problem to obtain bounds on the optimal solution value.

The *Set Covering Problem* (SCP) has been used as model in relevant applications, in

particular crew scheduling, where a given set of trips has to be covered by a minimum-cost set of pairings, i.e., a pairing is a sequence of trips that can be performed by a single crew. The most successful heuristic algorithms for large scale SCP's are based on LR, see (Caprara et al., 1996, 2000). Prins et al. (2006) deals with the bi-objective set covering problem, the proposed approach is a two-phase heuristic method which has the particularity to be a constructive method using the primal-dual Lagrangian relaxation.

With respect to the *Traveling Salesman Problems* and *Vehicle Routing Problems* (VRP), there has been some authors who use LR to find lower bounds, (Zamani & Lau, 2010; Smith et al., 1990). Nevertheless, we are interested in those who have exploited LR as exact methodologies or in heuristic approaches. It can be found a literature review on the use of LR for the TSP in chapter 1.

Balas and Christofides (1981) describe an algorithm for the *Asymmetric Traveling Salesman Problem* obtaining a Lagrangian relaxation based on the assignment problem. Their approach can be adapted to the symmetric TSP by using the 2-matching problem as a Lagrangian dual problem.

The *Generalized Traveling Salesman Problem* (GTSP) assumes that the nodes have been grouped into mutually exclusive and exhaustive node sets. It has to be found a minimum cost cycle which includes exactly one node from each node set. Fischetti et al. (1997a) apply Subgradient algorithm relaxing the fan inequalities plus the degree constraints. Moreover, for each tour among clusters found during the Lagrangian relaxation, they obtain a new feasible solution through procedure RP2. For the asymmetric GTSP, Laporte et al. (1987) relax the subtour prevention constraints and a set of constraints which ensure that each set is visited. The relaxed problem is solved as an assignment problem in a branch-and-bound algorithm. Using the same relaxation, Noon and Bean (1991) employs a Lagrangian relaxation to compute a lower bound and a heuristically determined upper bound are used to identify and remove arcs and nodes which are guaranteed not to be in an optimal solution.

Desrosiers et al. (1988) consider the problem of finding the minimum number of vehicles required to visit once a set of nodes subject to time window constraints, for a homogeneous fleet of vehicles located at a common depot. Their present an optimal solution approach using the augmented Lagrangian method. Two Lagrangian relaxations are proposed: in the first one, the time constraints are relaxed producing network subproblems which are easy to solve, but the bound obtained is weak; in the second relaxation, constraints requiring that each node be visited are relaxed producing shortest path subproblems with time window constraints and integrity conditions.

The *Plate-Cutting Traveling Salesman Problem* arises when parts are cut from large plates of metal or glass, it requires the determination of a minimum-length tour such that exactly one point must be visited on each given polygon. (Hoeft & Palekar, 1997) present a Lagrangian decomposition of the problem and develop lower bounds and heuristics based on this decomposition which are embedded in a variable r-opt heuristic for the overall problem.

The *Train Timetabling Problem* aims at determining a periodic timetable for a set of trains that does not violate track capacities and satisfies some operational constraints.

Caprara et al. (2002) propose a graph theoretic formulation for the problem using a directed multigraph. A novel feature of their formulation is that the variables in the Lagrangian relaxed constraints are associated only with nodes allowing a considerable speed-up in the solution of the relaxed problem. The relaxation is embedded within a heuristic algorithm which makes extensive use of the dual information associated with the Lagrangian multipliers.

For solving disruptions in the vehicle routing problem which are caused by vehicles breakdown or traffic accidents in the logistic distribution system, an urgency vehicle scheduling scheme is established based on the theory of disruption management. According to the characteristics of the problem, X. Wang et al. (2010) apply a Lagrangian relaxation approach to simplify and divide the *Urgent Multi-Depot Vehicle Routing Problem* into two parts. The column generation and saving approach method are used respectively to obtain the solution, and then the subgradient optimization method is used to iterate to get the Lagrangian multiplier. Moreover, in order to solve the infeasibility problem caused by the lack of convergence of the LR, an insertion algorithm is adopted to obtain a feasible solution of the original problem.

The *Vehicle Routing Problem with Time Windows* is a generalization of the VRP. A solution to the this problem must ensure that the service of any customer starts within a given time interval, a so-called time window. Fisher et al. (1997) present a Lagrangian decomposition in which variable splitting is used to divide the problem into two subproblems: a semi-assignment problem and a series of shortest path problems with time windows and capacity constraints. Kohl and Madsen (1997) propose a method based on a Lagrangian relaxation of the constraint set requiring that each customer must be served. They solved the Lagrangian dual problem using a combination of the Subgradient algorithm and the Bundle algorithm. Kallehauge et al. (2006) consider the Lagrangian relaxation of the constraint set requiring that each customer must be served by exactly one vehicle yielding a constrained shortest path subproblem. They present a stabilized cutting-plane algorithm within the framework of linear programming for solving the associated Lagrangian dual problem.

Lagrangian Dual Problem

Lagrangian relaxation is a method well suited for problems where the constraints can be divided into two sets: constraints under which the problem is solvable very easily; and constraints that make it very hard to solve. The main idea is to relax the problem by removing these second constraints and putting them into the objective function. Consider the following Integer Linear Problem (ILP):

$$\begin{aligned} \min_x \quad & cx \\ s.t. \quad & Ax \geq b \qquad (1) \\ & Dx \geq e \qquad (2) \\ & x \in \mathbb{Z}_+^n \end{aligned}$$

where x is $n \times 1$, b is $m \times 1$, e is $k \times 1$, and all other matrices have conformable dimensions. Let z_P be the optimal value of the ILP problem.

It is assumed that the constraints of this problem 1 and 2 are independent. Moreover, supposing that the set of constraints 2 can be optimized very easily whereas the set of constraints 1 makes the problem intractable.

The Lagrangian Dual problem obtained from this ILP problem by taking into the objective function the inequality $Ax \geq b$ (1) is:

$$L^* = \max_{u \in \mathbb{R}^m} L(u)$$

with Lagrangian function:

$$\begin{aligned} L(u) = \quad & \min_x \quad cx + u(Ax - b) \\ & s.t. \quad Dx \geq e \\ & \qquad x \in \mathbb{Z}_+^n \end{aligned}$$

This Lagrangian Dual problem relaxes a set of constraints and introduce a Lagrangian multiplier u for every constraint. Given that the relaxed set of constraints 1 is "greater than or equal to", the Lagrangian multiplier vector u is of appropriate dimensions m and non-negative components. Note also that the original ILP is a minimization problem, so its Lagrangian dual problem maximizes.

The original ILP problem can be relaxed into different problems depending on the relaxed constraints. If the chosen relaxation exploits the structure of the problem, the resulting dual problem can efficiently compute the optimal value for a fixed vector u and then it is easier to solve than the original problem.

Nemhauser and Wolsey (1988) proved that being $L(u)$ a relaxation of the original problem for all $u \in \mathbb{R}^m$, then:

- The feasible region is at least as large.
- The objective value is at least as small in $L(u)$ as for all feasible solutions in original problem, i.e., $L(u) \leq z_P$.

Figure 3 is a copy of Figure 1 from Hooker (2008, p. 1697) and depicts the previous proposition:

- The feasible region of the relaxed problem, $S' = \{x : Dx \geq e\}$, is at least as large as the feasible region of the original problem, $S = \{x : Ax \geq b, Dx \geq e\}$.
- For a given vector u, the penalized objective function $f'(x) = cx + u(Ax - b)$ is at least as small as the original objective function $f(x) = cx$ in the region S.

Finally, Nemhauser and Wolsey (1988) proved the next theorem: If $x \in \mathbb{Z}^n$ is an optimal solution of $L(u)$, it is feasible respect the original problem, and x and u are

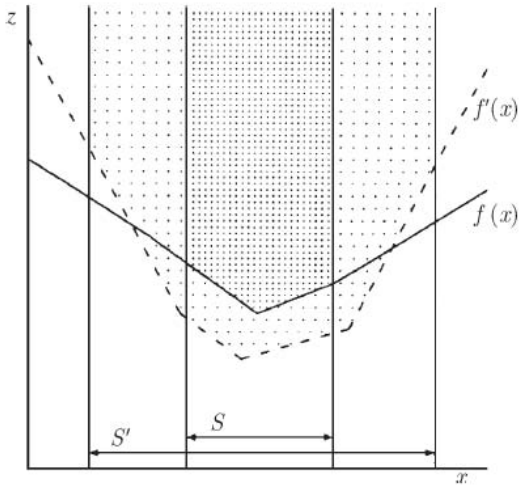


Figure 3: Epigraph of an optimization problem $min\{f(x) : x \in S\}$ (darker shaded area) and of a relaxation $min\{f'(x) : x \in S'\}$ (darker and lighter shaded areas). Reprinted from Hooker (2008, p. 1697, Figure 1). Copyright ©2009 SPRINGER.

complementary, then x is optimal solution of the original problem. In other words, if the Lagrangian multiplier u is the optimal solution of L^* and x is feasible respect 1, then x is the optimal solution of the original problem, and $L^* = Z_P$.

Nemhauser and Wolsey (1988) also proved that the dual objective function $L(u)$ is a piecewise function, but non-differentiable. If the original problem minimizes, then the Lagrangian dual problem is a maximization problem, $L^* = \max_{u \in \mathbb{R}^m} L(u)$, and the function $L(u)$ is concave. Whereas, if the original problem maximizes, then the Lagrangian dual problem is a minimization problem and the function is convex, see figure 4 from Fisher (2004, p. 1865).

Subgradient Algorithm

The most widely used approach to solve the Lagrangian dual problem is the Subgradient Algorithm, also known as Subgradient Optimization. This method provides from the method for finding lower bounds of Held and Karp (1971). That is designed to solve the problem of maximizing a piecewise linear concave function:

$$\max_{u \in \mathbb{R}^m} L(u), \qquad L(u) = \min_{x \in \mathcal{S}} cx + u(Ax - b)$$

where $\mathcal{S} = \{x \in \mathbb{Z}_+^n \mid Dx \geq e\}$.

A vector $\gamma \in \mathbb{R}^m$ is called a subgradient at u of a concave function $L : \mathbb{R}^m \to \mathbb{R}$ if it satisfies $L(u) \leq L(v) + \gamma(v - u)$ for all $v \in \mathbb{R}^m$. A subgradient is a straightforward generalization of a gradient. Nemhauser and Wolsey (1988) proved that:

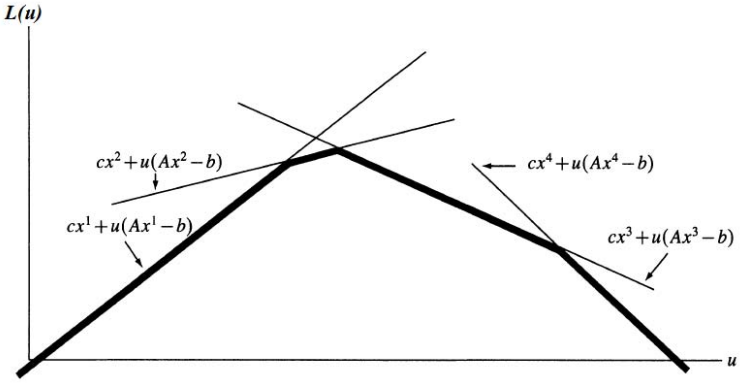


Figure 4: Form of the Lagrangian dual function $L(u)$: a concave piecewise non-differentiable function. Reprinted from Fisher (2004, p. 1865, Figure 1). Copyright ©2004 INFORMS.

- The vector $(Ax - b)$ is a subgradient at any $u \in \mathbb{R}^m$ for which x is an optimal solution of $L(u)$. Any other subgradient is a convex combination of theses primitive subgradient.
- x is optimal in the original problem if and only if $\gamma = 0$.

The Algorithm 1 shows the Subgradient Algorithm. Given an initial value u^0 a sequence $\{u^k\}$ is generated by the rule $u^{k+1} = u^k + \lambda_k \gamma^k$ where x^k is an optimal solution of the relaxed problem and λ_k is a positive scalar step size.

Algorithm 1 The Subgradient Algorithm

Initialize the multiplier u^0
while the subgradient $\gamma^k \neq 0$ **do**
  Solve the Lagrangian dual problem $L(u^k)$ with optimal solution x^k
  Check the subgradient $\gamma^k = Ax^k - b$
  Update the step size λ_k
  Update the multiplier $u^{k+1} = u^k + \lambda_k \gamma^k$
  $k \leftarrow k + 1$
end while

As explained before, one of the main drawbacks is the need of solving optimally the Lagrangian dual problem, which may be slow to be of real practical interest, (Bertsekas, 1999).

Convergence Criteria

The Subgradient algorithm is easy to program, the main difficulty of this algorithm lays on choosing a correct step size λ_k in order to ensure algorithm's convergence, (Reinelt,

1994). Convergence is guaranteed in the following rules:

a) If $\sum_k \lambda_k \to \infty$, and $\lambda_k \to 0$ as $k \to \infty$, then $L(u^k) \to L^*$ the optimal value of the Lagrangian Dual problem.

b) If $\lambda_k = \lambda_0 \rho^k$ for some parameter $\rho < 1$, then $L(u^k) \to L^*$ if λ_0 and ρ are sufficiently large.

c) If $LB \leq L^*$ and $\lambda_k = \delta_k \dfrac{LB - L(u^k)}{\| \gamma \|^2}$ with $0 < \delta_k < 2$, then $L(u^k) \to LB$, or the algorithm finds u^k with $LB \leq L(u^k) \leq L^*$ for some finite k.

The rule (a) guarantees convergence, but it is too slow to be of real practical interest. The rule (b) leads to faster convergence, but if the initial values of λ_0 and ρ are not sufficiently large, the geometric series $\lambda_0 \rho^k$ will tend to zero too rapidly, and the sequence $\{u^k\}$ will converge before reaching an optimal point.

In practice, rather than using rule (b) at each iteration, other two approaches can be considered: a geometric decrease can be achieved by reducing the value of λ_k every ν iterations, where ν is some natural problem parameter, for example, the number of variables; the second approach is reducing every ν iterations without improving the dual lower bound, aiming not to change the step size if it has a good convergence, and make a change if it has slow convergence.

Given that a dual lower bound LB is typically unknown, and in practice, it is more likely to know a primal upper bound $UB \geq L^*$. The step size rule (c) is used most commonly with an upper bound UB instead of LB. However, if $UB \gg L^*$, the term $UB - L(u^k)$ in the numerator will not tend to zero, and so the sequences $\{u^k\}, \{L(u^k)\}$ will not converge. The convergence holds if the parameter UB is a tight upper bound.

Stopping Criteria

The Subgradient algorithm reaches the optimal value when $\gamma = 0$. Other stopping criteria can be chosen: the criteria based on CPU time or number of iterations; and the criterion given by $\lambda_k < \varepsilon$, that is, the step size is so small that does not reach new values.

Metaheuristics

The problems addressed in this thesis are NP-hard and time-consuming combinatorial problems. Two major approaches are traditionally used to tackle these problems: exact methods and metaheuristics. Exact methods find exact solutions but are often impractical as they are extremely time-consuming for real problems. On the other hand, metaheuristics provide suboptimal (sometimes optimal) solutions in a reasonable time satisfying the deadline imposed in the industrial field to be met.

The naming metaheuristic was introduced by Glover and Laguna (1997), a metaheuristic "refers to a master strategy that guides and modifies other heuristics to produce solutions beyond those that are normally generated in a quest for local optimality".

An excellent review of this topic can be found in (Voß, 2001), the author concludes that a metaheuristic is "an iterative master process that guides and modifies the operations of subordinate heuristics to efficiently produce high-quality solutions".

Alba et al. (2009) analyze real-world problems and modern optimization techniques to solve them. They expose that the metaheuristic fields of application range from combinatorial optimization, bioinformatics, telecommunications to economics, software engineering, etc., which need fast solutions with high quality. The authors describe some fundamental characteristics of metaheuristics as follows:

- The goal is efficient exploration of the search space to find (nearly) optimal solution.
- Metaheuristic algorithms are usually nondeterministic.
- They may incorporate mechanisms to avoid getting trapped in confined areas of the search space.
- The basic concepts of metaheuristics permit an abstract-level description.
- Metaheuristics are not problem specific.

The family of metaheuristics includes Greedy Randomized Adaptive Search Procedure, Variable Neighborhood Search, Tabu Search, Ant Systems, Evolutionary methods, Genetic Algorithms, Scatter Search, Neural Networks, Simulated Annealing, among others.

Greedy Randomized Adaptive Search Procedure (GRASP) is a multi-start or iterative process in which each iteration consists of two phases: a construction phase –in which a feasible solution is produced– and a local search phase –in which a local optimum in the neighborhood of the constructed solution is sought. The best overall solution is kept as the result. In the construction phase, a feasible solution is iteratively constructed, one element at a time. At each construction iteration, the choice of the next element to be added is determined by ordering all candidate elements in a candidate list according to a greedy function. This function measures the (myopic) benefit of selecting each element. The heuristic is adaptive because the benefits associated with every element are updated at each iteration of the construction phase to reflect the changes brought on by the selection of the previous element. The probabilistic component of a GRASP

is characterized by the random choice of one of the best candidates in the list, but not necessarily the top candidate. This choice technique allows for different solutions to be obtained at each GRASP iteration. For a review see (Feo & Resende, 1995; Resende, 2008; Festa & Resende, 2009).

Variable Neighborhood Search (VNS), introduced for the first time by Mladenović and Hansen (1997), explores increasingly distant neighborhoods of the current incumbent solution, it exploits systematically the idea of neighborhood change, both in the descent to local minima and in the escape from the valleys which contain them. A review of this method can be found in (P. Hansen & Mladenović, 2003), together with a description of its most important variants. VNS is used for the TSP and its extensions, some relevant works are: a basic VNS for the euclidean TSP in (P. Hansen & Mladenović, 2006), a guided VNS methods for the asymmetric TSP in (Burke et al., 2001), and a VNS for the Pickup and Delivery in (Carrabs et al., 2007). Interesting results have been obtained even applying the simplest VNS algorithms, some examples in VRP researcher are Hasle and Kloster (2007) and Bräysy (2003).

One of the most important variants of the VNS is the Variable Neighborhood Descent (VND) method, its local search process performs an exhaustive exploration for each neighborhood changing to the next neighborhood in a deterministic order. All improving movements are recorded and sorted, so the best neighbor is constructed applying all independent changes in descending order. This way, solution values are improved faster than applying single movements. Two relevant works are a VND to the VRP with backhauls in (Crispim & Brandao, 2001) and a VND to take advantage of different neighborhood structures for the VRP in (Rousseau et al., 2002), between others.

Large neighborhood Search (LNS), which was proposed by (Shaw, 1998), is becoming more and more popular to solve routing problems. The idea is a local search that adopts a large neighborhood which makes less likely to fall in a local minimum. In LNS the neighborhood is implicitly defined by methods (often heuristics) which are used to destroy and repair an incumbent solution. For example, Rousseau et al. (2002) propose a LNS in which CP explores a neighborhood with three operators. These operators are combined in VND and a two phase process. Bent and Van Hentenryck (2004) describe an LNS heuristic for the VRPTW. Furthermore, Pisinger and Røpke (2007) propose a unified heuristic that works for several variants of routing problems and that uses an Adaptive LNS.

Tabu Search, originally proposed by Glover (1986), pursue local search whenever it encounters a local optimum by allowing non-improving moves; cycling back to previously visited solutions is prevented by the use of memories, called tabu lists, that record the recent history of the search, a key idea that can be linked to artificial intelligence concepts. A review on this field can be found in (Glover & Laguna, 1997, 2013). Some important researches related to VRPs are (Gendreau et al., 1999; Crispim & Brandao, 2001; Cordeau & Laporte, 2004; Brandao, 2011).

Ant Systems approach or Ant Colony Optimization, initially proposed by Dorigo and Gambardella (1997), is a natural metaphor on which it is based on how real ants are capable of finding the shortest path from a food source to their nest without using

visual clues by exploiting pheromone information. Some relevant works are (Bell & McMullen, 2004; Delisle et al., 2005, 2009; S. M. Chen & Chien, 2011; Uchida et al., 2012; Mavrovouniotis & Yang, 2013; Dorigo & Gambardella, 2014).

Genetic Algorithms are population based search techniques which mimics the principles of natural selection and natural genetics laid by Charles Darwin, that was proposed by Holland (1975). R. Cheng and Gen (1994) described a greedy selection crossover operator, which is designed for path representation and performed at gene level. It can utilize local precedence and global precedence relationship between genes to perform intensive search among solution space to reproduce an improved offspring. Diverse authors have contribute to improve the Genetic algorithms since then as (Nagata & Kobayashi, 2013), (Gen & Cheng, 2000), (Ray et al., 2007), (Deep & Mebrahtu, 2011), between others authors.

Most current metaheuristics are primal-only methods, except for the work of Boschetti and Maniezzo (2009), they propose to investigate the possibility of reinterpreting decompositions, with special emphasis on the related Benders and Lagrangian relaxation techniques, from a metaheuristic perspective. Other prominent authors are Marinakis et al. (2005, 2009), they present a hybrid method for solving the TSP proposing a combination of genetic algorithms, GRASP and LR. Constraints requiring that each node has two incident edges are relaxed obtaining a minimum spanning tree as the Lagrangian dual problem.

Most basic metaheuristics are sequential. Furthermore, some authors propose *Parallel Metaheuristics* to reduce the search time and to improve the quality of the solutions provided. For a discussion on how parallelism can be mixed with metaheuristics, the reader has several sources of information in the literature (Alba et al., 2013; Alba, 2005). Some works related to the problems addressed in this thesis are: a master-slave model for parallel Ant Colony Optimization has been implemented in multicore processors by Delisle et al. (2005) for the TSP; a multicore multi-population method for Ant Colony Optimization have also been proposed by Delisle et al. (2009) for the TSP; an algorithm defined by Dorronsoro et al. (2007) for the CVRP uses a master-slave model to distribute the most consuming operations (fitness evaluation and application of the operators) among the different processors of the parallel platform that is a grid system; and our own work presented in (Guimarans et al., 2013) which proposes a Multi-Start VND structure whose local search process is supported by Constraint Programming and our Tailored Lagrangian Metaheuristic.

Part I:
Symmetric Problems

Chapter 1

Traveling Salesman Problem

The Traveling Salesman Problem (TSP) is probably the best known and extensively studied problem in the field of Combinatorial Optimization. Reinelt (1994) expressed that this problem is undoubtedly the most prominent member of the rich set of combinatorial optimization problems. It is one of the few mathematical problems that frequently appears in the popular scientific press (Cipra, 1993) or even in newspapers (Kolata, 1991). It has a long history, dating back to the 19th century (A. Hoffman & Wolfe, 1985).

According to Bellman (1962), the TSP is the following: "A salesman is required to visit once and only once each of n different customers starting from a depot, and returning to the same depot. What path minimizes the total distance traveled by the salesman?" .

Given a set of cities along with the cost of travel between each pair of them, the TSP consists on finding the cheapest way of visiting all the cities and returning to the starting point. The "way of visiting all the cities" is simply the order in which the cities are visited; the ordering is called a tour or circuit through the cities, (Applegate et al., 2006).

According to K. L. Hoffman et al. (2013), the TSP has commanded much attention of mathematicians and computer scientists specifically because it is so easy to describe and so difficult to solve. Furthermore, the study of this problem has attracted many researchers from different fields, from both, theoretical approach and practical applications. Due to its characteristics, there is a vast amount of literature on it, as it is introduced on section 1.3.

Some of the first applications were: Whizzkids '96 Vehicle Routing, which problem consists of finding the best collection of routes for 4 newsboys to deliver papers to their 120 customers; the tour through MLB Ballparks where a baseball fan found the optimal route to visit all 30 Major League Baseball parks; the touring airports to find shortest routes through selections of airports in the world; USA trip which involved a chartered aircraft to visit cities in the 48 continental states, to mention some.

Nowadays, the TSP has several applications in addition to its original formulation, such as vehicle routing, planning, logistics, manufacturing, flow shop scheduling, genome sequencing, DNA universal strings, starlight interferometer program, scan chain op-

timization, coin collection scheduling, designing sonet rings, power cables, computer wiring, and frequency assignment in communication networks, among the most important ones.

Due to the wide range of applications and the complexity of the TSP problem, innovative and efficient algorithms are needed. Therefore, this chapter describes our proposed approach called Tailored Lagrangian Metaheuristic (TLM) which was introduced in (Herrero et al., 2010b), the results presented in this chapter are unpublished and improve the ones of our publication. The proposed metaheuristic is based on the Lagrangian Relaxation applied to the TSP. The presented approach combines the Subgradient Optimization algorithm with a heuristic to obtain a feasible primal solution from a dual solution. Moreover, it has been introduced a parameter to improve algorithm convergence. The main advantage is based on the iterative evolution of both upper and lower bounds to the optimal cost, providing a feasible solution in a reasonable number of iterations with a tight gap between the primal and the optimal cost. This metaheuristic provides a near-optimal solution in a reasonable time within a tight gap.

1.1 Problem Definition

The problem was first formulated by Karl Menger (1930) and a large number of heuristics and exact methods are known for its solution, some instances with thousands of cities can be solved.

The TSP belongs to the class of NP-Hard optimization problems (Lenstra & Kan, 1981). This means that no polynomial time algorithm is known for its solution. Like many combinatorial NP problems, there is no efficient algorithm to solve the TSP that has been proved to solve correctly all case scenarios, and whose worst-case running time is bounded by a polynomial time function which depends on the scenarios' size.

The symmetric TSP can be considered as a routing network, represented by a complete undirected graph $G = (I, E)$, connecting the customers set $I = \{1, 2, ..., n\}$ through a set of undirected edges $E = \{(i, j)|i, j \in I\}$. The edge $e = (i, j)$ in E has associated a travel distance c_e, it is assumed that distances satisfy the triangular inequality, that means c_e is supposed to be the lowest cost route connecting node i to node j. Note that it is not a capacitated problem, i.e., customers have not a demand to satisfy and the vehicle has not a limited capacity.

Solving the TSP consists on determining a route whose total travel distance is minimized, each customer is visited exactly once and the route starts and ends at the depot $(i = 1)$.

The classical formulation requires defining the binary variable x_e to denote that the edge $e = (i, j) \in E$ is used in the route. That is, $x_e = 1$ if customer j is visited immediately after i; otherwise $x_e = 0$. Thus, TSP can be mathematically formulated as follows:

$$\min \sum_{e \in E} c_e x_e \tag{1.1}$$

subject to

$$\sum_{e \in \delta(i)} x_e = 2 \ , \qquad \forall i \in I \tag{1.2}$$

$$\sum_{e \in E(S)} x_e \leq \mid S \mid -1 \ , \qquad \forall S \subset I \ , \ \mid S \mid \leq \frac{1}{2} \mid I \mid \tag{1.3}$$

$$x_e \in \{0, 1\} \ , \qquad \forall e \in E \tag{1.4}$$

where

- $\delta(i) = \{e \in E : \exists j \in I, \ e = (i, j) \text{ or } (j, i)\}$ represents the set of arcs whose starting or ending node is i.
- $E(S) = \{e = (i, j) \in E : i, j \in S\}$ represents the set of arcs whose nodes is in the subset S of vertices.
- $n = |I|$
- c_e is the associated cost to the undirected edge $e_{ij}(e_{ji})$.

Constraint (1.2) states that every node $i \in I$ must be visited once, that is, every customer must have two incident edges. Subtour elimination constraint (1.3) states that the route must be a Hamiltonian path, so it can not have any subcycle, (C. Miller et al., 1960).

A Hamiltonian path is a path that traverses every vertex of a connected graph once and only once, i.e., each customer is visited exactly once. A Hamiltonian cycle is a cycle that visits each vertex exactly once (except for the vertex that is both the start and end, which is visited twice). Hamiltonian paths and cycles are named after William Rowan Hamilton who invented the "Icosian Game", now also known as Hamilton's puzzle, which involves finding a Hamiltonian cycle in the edge graph of the dodecahedron. See figures 1.1 and 1.2.

Karp (1972) proved that Hamiltonian circuit is NP-complete. Given that the TSP can be formulated as finding a Hamiltonian circuit T whose weight $\sum_{e \in E(T)} c_e$ is minimum, it can be proved that the TSP is strongly NP-hard. The proof can be found in (Korte & Vygen, 2012).

In other words, it is evident that if the TSP contains a Hamiltonian circuit, then it is a solution. There are three kind of graph which ensure a solution:

- A symmetric and complete graph G.
- A symmetric, connected graph $G\{I, E\}$ which obeys $\mid I \mid = n > 2$ and $degree(i) \geq \frac{n}{2}$ $\forall i \in I$.
- A symmetric graph $G\{I, E\}$, which $\mid I \mid \geq 3$ and $degree(i) + degree(j) \geq \mid I \mid$ $\forall$ non-adjacent $i, j \in I$.

The proof can be found in (Basart, 2003). Nevertheless, the literature has been focus on symmetric and complete graphs.

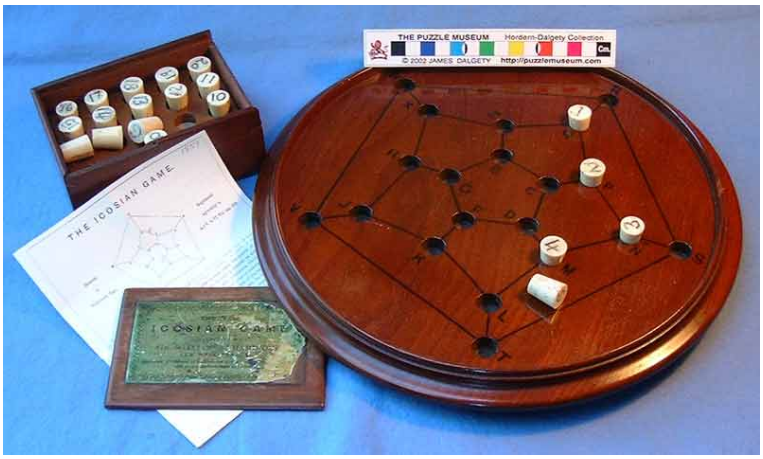


Figure 1.1: An original copy of Sir William Rowan Hamilton's famous "Icosian Game" in The Puzzle Museum. Copyright @ 2015 The Puzzle Museum James Dalgety.

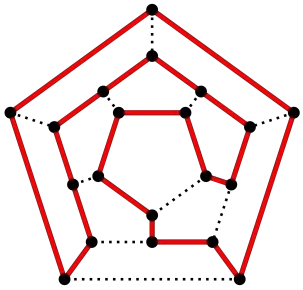


Figure 1.2: "Hamiltonian path". Copyright under CC BY-SA 3.0 via Wikimedia Commons by Christoph Sommer.

1.2 Its variants

There are several variants of the Traveling Salesman Problem, the most basic instance of the TSP is assuming that the distance cost is symmetric, i.e., the distance between two customers is the same in each opposite direction forming an undirected graph. On the other side, the *Asymmetric TSP* is a generalization of the TSP which assumes that the distance cost is asymmetric obtaining a directed graph. Usually, in real world, TSP appears with many side constraints. Most important restrictions are:

- Every customer has to be supplied within a certain time window. This problem is called the *TSP with Time Windows* and also can include other time data such as travel times between every pair of nodes, service times and a maximum tour duration. Normally, this problem considers the minimization of either the total time or cost of the tour.

- Customers want to delivery some goods to other customer. This problem is called the *TSP with Pick-Up and Delivery* and it is defined on a graph containing pickup and delivery vertices between which there exists a one-to-one relationship, that means that customers can be divided into two groups according to the type of service required (delivery or pick-up). The problem consists of determining a minimum cost tour such that each pick-up vertex is visited before its corresponding delivery customer. It can be a sequential ordering problem or a capacitated problem.

- Customers may have several predecessors. This problem is called the *Precedence-Constrained TSP* and it is a generalization of the *TSP with Pick-Up and Delivery* in which each customer may have several predecessors, i.e., another customers which have to been visited before the customer. Then, there exists a many-to-one relationship.

- The salesman must first delivery goods and then pickup goods. This problem is called the *TSP with Backhauls* where an uncapacitated vehicle must visit all the delivery customers before visiting a pickup customer. In other words, a set of locations must be routed before the rest of locations.

- There are more than one salesman. This problem is called the *Multiple TSP* and it is a generalization of the TSP in which more than one salesman is allowed. Given a set of customer, the objective of the problem is to determine a tour for each salesman such that the total tour cost is minimized and that each city is visited exactly once by only one salesman.

- The salesman must purchase a set of goods. This problem is called the *Traveling Purchaser Problem* and it deals with a purchaser who is charged with purchasing of all required products. He can purchase these products in several cities, but at different prices and not all cities offer the same products. The objective is to find a route between a subset of the cities, which minimizes total cost (travel cost + purchasing cost).

- Transportation of perishable goods. This problem is called the *Bottleneck TSP* and it is a variation with a different objective function. It considers a weighted graph and its objective is minimizing the weight of the most weighty edge of the cycle. Another similar problem is the *Maximum Scatter TSP* which is used in sequencing the riveting operations when fastening sheets of metal together. The goal is to maximize the length of a shortest edge in the tour.

- Visiting only a subset of the customers. This problem is called the *Prize Collecting TSP* where each customer has an associated prize, and a salesman calls for a minimum travel cost covering a customer subset whose total prize is not less than a given value.

- Some values are random. The *Stochastic TSP* can consider that some values are random like travel costs, customers' serve time or travel time, the purchased prices, the number of salesmen. Also the *Dynamic TSP* considers that customers can be deleted or inserted over time.

These problems are the most significant in real world, for the interested reader seeking more information on TSP variants or generalizations, we suggest (Gutin & Punnen, 2002).

1.3 Literature Review

As it can be noticed in the following literature review, the TSP problem has been solved mainly by exact methods and metaheuristic algorithms. The most important exact algorithms for this problem have been developed with linear programming, Branch-and-Bound and Cutting-Plane implementation. They work reasonably fast only for small problem sizes. The approximation methods are classified into metaheuristic and heuristic techniques. Metaheuristics, instead of exact algorithms, are extensively used to solve NP problems.

Dantzig et al. (1954) expressed the problem as an integer linear program and developed the cutting plane method for its solution. They showed the effectiveness of their method by solving a 49-city instance.

Significant progress has been made using Cutting-Planes and Branch-and-Bound techniques as in the work of (Grötschel & Holland, 1991), (Padberg & Rinaldi, 1987), (Padberg & Rinaldi, 1991), (Radharamanan & Choi, 1986) among others. It has been managed to exactly solve instances with up to 2392 cities.

One of the best-known exact method is named Concorde, a TSP Solver developed by Applegate et al. (2015), that employs an architecture that ensures the validity of the lower-bound computations using linear programming and a cutting-plane implementation. Applegate et al. (2006) explains how they solved optimally a instance of 85,900 locations taking over 136 CPU-years, and how they solved the World TSP of 1,904,711 cities yielding a gap of only 0.0474% respect the best tour length found by Helsgaun (2009) using a variant of his LKH heuristic algorithm.

The classical heuristic methods are Nearest Neighbor, 2-Opt, and 3-Opt methods, which are frequently used for constructing and improving initial solutions in metaheuristics. A survey of them can be found in (Nilsson, 2003).

Lin and Kernighan (1973) proposed an effective heuristic for the TSP, which is based on a k-Opt procedure with k variable in each step. Later, Helsgaun (2000) described an implementation even more effective of this heuristic, named LKH, for both symmetric and asymmetric problems. LKH currently holds the record for the best instances of problems with unknown optimal (*DIMACS TSP Challenge 2002*) with sizes ranging from 1,000 to 10,000,000 nodes.

Some of the latest heuristic approaches that are being well adapted to solve the NP-hard TSP are Genetic algorithm, Memetic Algorithm and Ant colony.

Genetic algorithms are population based search techniques which mimics the principles of natural selection and natural genetics laid by Charles Darwin, that was proposed by Holland (1975). R. Cheng and Gen (1994) described a greedy selection crossover operator, which is designed for path representation and performed at gene level. It can utilize local precedence and global precedence relationship between genes to perform intensive search among solution space to reproduce an improved offspring. Diverse authors have contribute to improve the Genetic algorithms since then as (Nagata & Kobayashi, 2013), (Gen & Cheng, 2000), (Ray et al., 2007), (Deep & Mebrahtu, 2011), between others authors.

The Memetic algorithms combine the recognized strength of population in Genetic algorithm with the intensification capability of a local search, i.e., all agents evolve solutions until they become local minima of a certain neighborhood, for an introduction see (Neri et al., 2012). The method proposed by Freisleben and Merz (1996) introduces a new recombination operator, called Distance Preserving Crossover, their algorithm uses the Lin-Kernighan heuristic as a local search engine for the Euclidean instances. Some of the latest publications in the field are (Bontoux et al., 2010), (Y. Wang et al., 2011), (Osaba & Díaz, 2012), and (Gutin & Karapetyan, 2010).

Regarding the Ant Colony approach, the natural metaphor on which it is based on how real ants are capable of finding the shortest path from a food source to their nest without using visual clues by exploiting pheromone information, initially proposed by Dorigo and Gambardella (1997). Some of the most prominent authors are (Dorigo & Gambardella, 2014), (S. M. Chen & Chien, 2011), (Karaboga & Gorkemli, 2011), and (Uchida et al., 2012).

In order to know about the closeness of the upper bound to the optimum value, one must also know a lower bound on the optimum value. Several relaxations have been considered for the TSP, such as the n-path relaxation, the assignment relaxation, the 2-matching relaxation, the 1-tree relaxation, and the linear programming relaxation.

For the interested reader seeking more information on this important problem, we suggest the seminal paper on the problem by Dantzig et al. (1954) and the books by Lawler et al. (1985) and by Reinelt (1994). For a survey of its variations, see (Gutin & Punnen, 2002). For a deep understanding of how algorithms for TSP work, see the book by Applegate et al. (2006), which besides providing a wide overview on TSP history and on its applications. For a global survey that summarize most of the research and provide extensive references, see (K. L. Hoffman et al., 2013), (Lawler et al., 1985) and (Laporte, 1992).

Lagrangian Relaxation

Finally, our interest is focused on the Lagrangian Relaxation for solving the TSP. There are two well known relaxations for the TSP: the 2-matching problem and the minimum spanning tree.

The first one was introduced by Balas and Christofides (1981) who relax the Subtour Elimination constraint obtaining a 2-matching problem as Lagrangian relaxed problem.

The 2-Matching problem can be solved in polynomial time (Edmonds & Johnson, 1973), an efficient implementation can be found in Pekny and Miller (1994).

The second one was proposed by Held and Karp (1970, 1971), they relax the set of constraints requiring that all vertices must have two incident edges except for the depot, and the obtained relaxed problem is a minimum Weight 1-Tree which is a variant of the minimum spanning tree.

The algorithm proposed by Held and Karp (1970, 1971) was the basis for the Subgradient algorithm. As explained previously on page 14, the main difficulty of this algorithm lays on choosing a correct step size. Some variations concerning the choice of initial step sizes and update of step sizes are (K. H. Hansen & Krarup, 1974), (Smith & Thompson, 1977), (Volgenant & Jonker, 1982), and (Balas & Toth, 1985). Make a special attention to Zamani and Lau (2010), who presents an effective procedure that finds lower bounds for the TSP based on the 1-Tree using a learning-based Lagrangian relaxation technique. The procedure can dynamically alter its step size depending upon its previous iterations, thus achieves better convergence.

An analysis of the Lagrangian relaxation introduced by Held and Karp (1970) can be found in Shmoys and Williamson (1990). Smith et al. (1990) analyze the Lagrangian relaxation introduced by Balas and Christofides (1981) and conclude that the lower bounds are a bit worse.

Extensive research has been devoted to Lagrangian heuristic, which combines a heuristic with the Lagrangian Relaxation. Fisher (2004) discusses of some early examples of Lagrangian heuristics for the *Generalized Assignment problem*. Barahona and Anbil (2000) present the Volume algorithm, that is an extension to the Subgradient algorithm to produce primal as well as dual solutions. They present successful experience for the *Set Partitioning*, the *Set Covering*, the *Max-Cut* and the *Plant Location*.

Respect to the TSP, the most relevant authors extending LR are: Christofides (1976) whose algorithm first finds a minimum cost spanning tree, then finds the minimum cost Eulerian augmentation of that tree, and finally shortcuts the corresponding Eulerian walk into a tour; Timsjo (1999) presents a solution strategy based on the Lagrangian relaxation of the 1-Tree for the TSP and using a Subgradient algorithm. The author applies a heuristic to modify the relaxed solution into a feasible primal one; In the work of Volgenant and Jonker (1982), it is proposed a Branch-and-Bound algorithm for the TSP based on the ascent method to calculate the 1-Tree bounds, and a heuristic is used to provide upper bounds throughout the computations; In (Belloni & Lucena, 2000), the authors present a Lagrangian Relax-and-Cut algorithm for the TSP, where violated constraints are attempted to be identified and are dualized for every Lagrangian relaxation problem eventually solved. For more information on Lagrangian Relax-and-Cut algorithm, see (Lucena, 2006); Other prominent authors are Marinakis et al. (2005, 2009), they present a hybrid method for solving the TSP proposing a combination of genetic algorithms, GRASP and LR. Constraints requiring that each node has two incident edges are relaxed obtaining a minimum spanning tree as the Lagrangian dual problem.

1.4 Lagrangian Dual Problem

The proposed LR uses the relaxation proposed by Held and Karp (1971) which relaxes the constraint set (1.2) requiring that all customers must be served expect for the depot ($i = 1$). Therefore, the obtained relaxed problem is a *Minimum Weight 1-Tree*.

A minimum weight 1-Tree consists of a *Spanning Tree* of minimal total length on the node set $I \setminus \{1\}$ combined with the shortest two edges incident to node 1 (depot). A spanning tree of a graph is just a subgraph that contains all the vertices and it forms a tree that is an undirected graph in which any two vertices are connected by exactly one path.

The Lagrangian Dual problem obtained relaxing the constraint set (1.2) is as follows:

$$\max_{u \in \mathbb{R}^n} L(u) \tag{1.5}$$

where:

$$L(u) = \min_{x \ 1-Tree} \sum_{e \in E} c_e x_e + \sum_{i \in I} u_i (2 - \sum_{e \in \delta(i)} x_e) \tag{1.6}$$

it weight the relaxed constraints with a multiplier vector u of appropriate dimension and unrestricted sign, defining the subgradient $\gamma_i^k = 2 - \sum_{e \in \delta(i)} x_e$.

Indeed, all subcycles can be avoided constructing the solution x as a 1-Tree, and a feasible solution of the TSP is a 1-Tree having two incident edges at each node (Held & Karp, 1971). The main advantage of this approach is that relaxing the set constraint 1.2, at the same time, it avoids the set of subtour elimination constraints (1.3) which contains an exponential number of constraints.

Furthermore, finding a minimum 1-Tree is relatively easy (Cormen et al., 2001). Prim's Algorithm and Kruskal's Algorithm are two algorithms commonly used for finding a minimum spanning tree.

Prim's Algorithm

An algorithm for solving the minimum weighted 1-Tree is Prim's algorithm, which was developed by Jarnik in 1930 and rediscovered by Prim in 1957 and Dijkstra in 1959.

Prim's Algorithm continuously increase the size of a tree starting with a single arbitrary vertex until it spans all the vertices. In each iteration it chooses the minimal weighted edge where one of its vertex has already been visited but the other has not been, given that, each pair of vertices must be connected by exactly one path. Its time complexity is $O(n^2)$, where n is the number of vertices.

Given a complete undirected graph $G = (I, E)$ representing the TSP, where I represent the set of nodes (customers and depot) and $E = \{(i, j) : i, j \in I\}$ the set of undirected edges. Let be the set of customers $I' = \{2, ..., n\}$ and their corresponding edges $E' \subset E$.

The Algorithm 2 shows the Prim's Algorithm for the 1-Tree. Remember that the so called 1-Tree is a tree and therefore, it has to be added two edges incidents to the depot ($i = 1$).

Algorithm 2 Prim's Algorithm for the 1-Tree

Initialization:
Find $e = (i, j) \in E'$ the minimal weighted edge.
Initialize $T = \{e\}$ the tree.
Initialize $V = \{i, j\}$ the set of visited nodes.
while $V \neq I'$ **do**
Find $e = (i, j) \in E'$ the minimal weighted edge with a visited node, $i \in V$, and an unvisited node, $j \in I' \setminus V$.
Add this edge to the tree: $T = T \cup \{e\}$.
Add the unvisited node: $V = V \cup \{j\}$.
end while

Find $e = (1, i) \in E$ the minimal weighted edge connected to the depot.
Add this edge to the tree: $T = T \cup \{e\}$.
Find $e = (1, j) \in E$ the second minimal weighted edge, i.e., $j \neq i$.
Add this edge to the tree: $T = T \cup \{e\}$.
return T.

1.5 An Experiment in Convergence

The purpose of this section is to show the importance of the step size on the Subgradient algorithm. An easy problem has been chosen to demonstrate its important, the Spanish Airports problem which requires visiting once each of the 37 airports of Spain, see Annex Appendix A. The travel distance has been obtained applying the Euclidean distance from the GPS coordinates, rounding it by integer values. Solving the problem 1.1 using the software IBM® ILOG® CPLEX® Optimization Studio (2015), its optimal value is $L^* = 7223$, however, using the Nearest Neighbor Heuristic, it is found an upper bound of value $UB = 8188$.

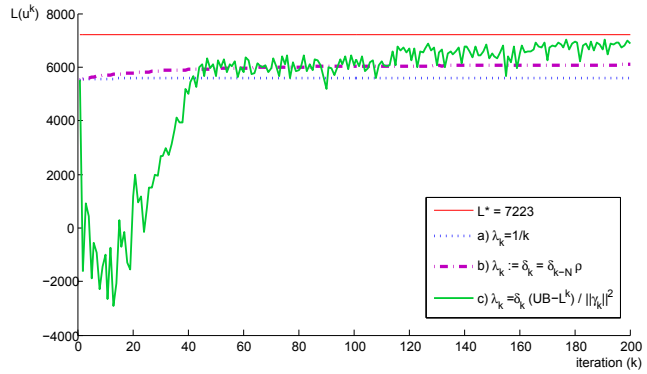
In order to compare the performance of the step size, the Subgradient algorithm is applied to the model of the Lagrangian relaxation of the TSP, see section 1.4. Figure 1.3 shows the convergence of the different step sizes for the Spanish Airports problem.

Fig. 1.3a shows convergence using the next three steps sizes. It should be noticed that despite initial perturbations the step size c) leads much faster convergence than the other steps.

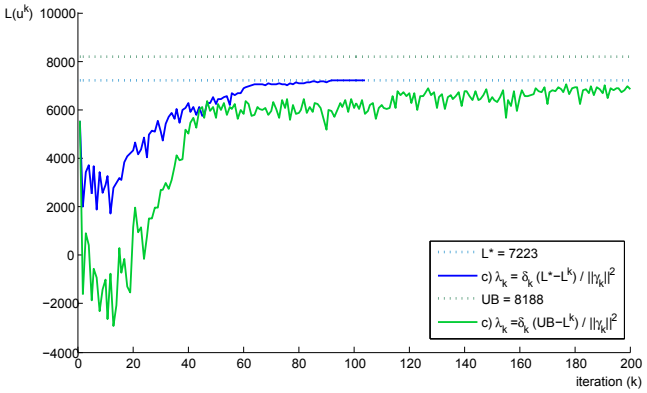
a) $\lambda_k = \frac{1}{k}$. This holds because $\sum_k \lambda_k \to \infty$, and $\lambda_k \to 0$ as $k \to \infty$.

b) $\lambda_k = \delta_{\{k-N\}} \cdot \rho$ for parameters $\delta_0 = 2$ and $\rho = 0.95$. This holds because the parameter δ_k is reduced every N iterations.

c) $\lambda_k = \delta_k \frac{UB - L(u^k)}{\|\gamma_k\|^2}$ and the parameter δ_k is reducing by $\rho = 0.95$ every N consecutive iterations without improving the lower bound with $\delta_0 = 2$. This holds if the parameter UB is tight upper bound.



(a) Rules a, b, and c.



(b) Rule c using an upper bound or the optimal value

Figure 1.3: Convergence of the Subgradient Algorithm with different step sizes for Spanish Airports problem.

Fig. 1.3b shows the convergence of the step size c) using different values. Using $UB = 8188$, the gap between the best Lagrangian value and the optimal value is of 5% at the iteration $k = 200$. However, using the optimal value $L^* = 7223$, it leads much faster convergence and finds an optimal solution at the iteration $k = 104$.

1.6 Tailored Lagrangian Metaheuristic

Due to the wide range of applications and the complexity of the TSP problem, innovative and efficient algorithms are needed. Therefore, we propose a metaheuristic methodology based on the Lagrangian Relaxation to solve the TSP. A metaheuristic "refers to a master strategy that guides and modifies other heuristics to produce solutions beyond those that are normally generated in a quest for local optimality" (Glover & Laguna, 1997).

Our Lagrangian metaheuristic uses the Subgradient optimization algorithm com-

bined with a heuristic which obtains a primal feasible solution from the dual variable. Compared with classical heuristics, our Lagrangian Metaheuristic provides both an upper and a lower bounds (UB and LB), and a posteriori quality check of the solution is obtained.

The main contribution of our method is that it has been introduced a parameter to improve the algorithm convergence on the optimal solution. That parameter is improved within the same process using the upper bound (UB) of the primal solutions obtained by the heuristic.

Our Tailored Lagrangian Metaheuristic metaheuristic in detail

Our proposed TLM metaheuristic is shown in Algorithm 3. A heuristic obtains a feasible solution from the dual variable, so it tries to improve the UB and hence, a better convergence using rule (c) is obtained. Eventually, the best of these feasible primal solutions may be provided if the method is stopped.

An optimal solution will be found if the method reaches $\gamma_i^k = 0$, i.e., the 1-Tree is a Hamiltonian cycle. Since we are interested in developing a methodology applicable to real problems where the computational time to solve instance is limited, it is added two stopping criteria based on the maximum number of iterations ($k < max_{iterations}$) and on a floating-point exception (step-size $\lambda_k < 10^{-15}$), see line 5.

The proposed metaheuristic uses the Nearest Neighbor Heuristic to obtain an initial value of the Upper Bound in line 3. This heuristic is one of the most commonly used to find UB, since that is easy to implement and it is computationally efficient, (Lin & Kernighan, 1973).

In line 6, Prim's algorithm is used to solve the Lagrangian Dual Problem of the TSP, as explained in section 1.4. Then, the subgradient is calculated in line 7, if its values are 0, the optimal solution is found and the iterative procedure is stopped in line 10.

The Proposed Heuristic

The proposed heuristic to improve the UB is applied when the 1-Tree is nearly a Hamiltonian cycle, that is when the subgradient satisfies $\| \gamma^k \|^2 < \xi$, see lines 11-12.

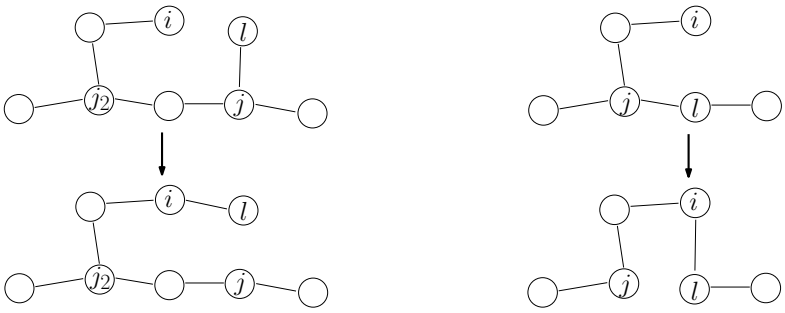
A parameter ξ is used to determine when to apply the heuristic, in line 11. A good estimation of the parameter ξ would avoid increasing the computation time excessively. First, its value may be large, for instance the value of the first iteration, $\xi = \| \gamma^1 \|^2$, but it should be updated whenever a better feasible solution is found according to $\xi = \| \gamma^k \|^2$. Eventually, the heuristic could find the optimal solution without detecting it, so the method would continue iterating until $LB = UB$.

As any solution is a 1-Tree, this criterion means that the solution has few vertices without two incident edges. This heuristic replaces an edge $e = (i, j)$ where j has some extra edges (i.e., $\gamma_j^k < 0$) for an edge $e = (i, l)$ where vertex l has one single edge (i.e., $\gamma_l^k = 1$) minimizing the cost of the exchange.

In the presented approach, two different moves have been defined (Figure 1.4): (a) unlimited move respect the vertices, and (b) limited move only replaces edges which

Algorithm 3 Tailored LR-based method for the TSP

Initialization:
  Initialize parameters $\delta_0 = 2; \rho = 0.95; \alpha_L = 1/3; \overline{L} = 0$
  Obtain an UB applying Nearest Neighbour Heuristic
  Initialize multiplier $u^0 = 0$
while $k < max_{iterations}$ **or** $\lambda_k < 10^{-15}$ **do**
  Solve the Lagrangian dual problem $L(u^k)$ with optimal solution x^k
  Update the subgradient $\gamma_i^k = 2 - \sum_{e \in \delta(i)} x_e^k$
  Check the subgradient:
    if $\| \gamma^k \|^2 = 0$ **then**
      An optimal solution is found $\rightarrow$ **Exit**
    else if $\| \gamma^k \|^2 < \xi$ **then**
      Apply a heuristic to improve the UB
    end if
  Check the parameter $\overline{L}$:
    if $L(u^k) > \overline{L}$ **then**
      It is updated, $\overline{L} = L(u^k) + \alpha_L(UB - L(u^k))$
    end if
    if $\overline{L} > UB$ **then**
      $\overline{L} = UB$
    end if
  Update the parameter δ_k:
    if $L(u^k) < LB$ **then**
      $\delta_{k+1} = \delta_k \rho$
    else
      $LB = L(u^k)$
      $\delta_{k+1} = \delta_k \frac{3-\rho}{2}$
    end if
  Update the step size $\lambda_k = \delta_k \dfrac{\overline{L} - L(u^k)}{\| \gamma^k \|^2}$
  Update the Lagrangian multiplier $u^{k+1} = u^k + \lambda_k \gamma^k$
  $k \leftarrow k + 1$
end while



(a) The unlimited move (b) The limited move

Figure 1.4: Movements of the proposed heuristic: swapping e_{lj} for e_{il}

both vertices i and l are connected to same vertex j. Our proposed heuristic is shown in Algorithm 4.

First, it iteratively applies the unlimited movement but being aware that it could divide the 1-Tree into disconnected trees. For each i with $\gamma_i^k = 1$, it swaps edge e_{lj} for e_{il} minimize the cost of the exchange:

$$\{i, j, l\} = argmin\Big\{c_{il} - c_{lj} \mid \gamma_i^k = 1,\ \gamma_j^k < 0,\ \gamma_l^k \geq 0,\ (l, j) \in T^k \text{ and } (i, l) \in E \setminus T^k\Big\} \tag{1.7}$$

Then, it iteratively applies the limited movement until it is found a feasible solution. For each pair $\{i, j\}$ where $\gamma_i^k = 1$ and j is the first connected node with $\gamma_j^k < 0$, it selects the neighbor l minimizing the cost of the exchange (1.7).

Convergence: new parameter $\overline{L}$

As mentioned, algorithm's convergence is critically influenced by the step-size λ_k. Using rule (c) explained on page 15, this value relies on either the LB or the UB, which are normally unknown or bad estimated. Therefore, convergence may not be assured for all cases. In order to overcome this limitations, we propose the use of a parameter $\overline{L}$, such that $LB \leq \overline{L} \leq UB$. By definition, this new parameter corresponds to a better estimation of the optimal value L^* than those obtained for LB and UB. The calculation of the step-size turns into (see line 28):

$$\lambda_k = \delta_k \frac{\overline{L} - L(u^k)}{\| \gamma_k \|^2}$$

Convergence is guaranteed if the term $\overline{L} - L(u^k)$ tends to zero. In turn, convergence efficiency can be improved as long as the new $\overline{L}$ parameter gets closer to the (unknown) optimal solution. The main idea is very simple: as the algorithm converges to the solution, new better lower bounds are known and new better upper bounds estimations can be obtained by using the heuristic designed to get feasible solutions. Therefore, in lines 14-20, the parameter $\overline{L}$ is updated according to the following conditions:

Algorithm 4 Heuristic to obtain feasible solution for a 1-Tree

Data: T^k the obtained 1-Tree
Initialize:
$V^+ = \{i \in I \mid \gamma_i^k = 1\}$ the subset of vertices which only has one edge
$V^- = \{j \in I \mid \gamma_j^k < 0\}$ the subset of vertices which has more than two edge

---

Unlimited Movement:
do $\parallel V^+ \parallel$ **times**
$\{i, j, l\} = argmin\Big\{c_{il} - c_{lj} \mid i \in V^+,\ j \in V^-,\ \gamma_l^k \geq 0,\ (l,j) \in T^k \text{ and } (i,l) \in E \setminus T^k\Big\}$

if nodes l and j are connected in $T^k \setminus (l,j) \cup (i,l)$ **then**
Swap these edges within the tree: $T^k = T^k \setminus (l,j) \cup (i,l)$
Update node i: $\gamma_i^k = 0$ and $V^+ = V^+ \setminus \{i\}$
Calculate γ_j^k
if $\gamma_j^k = 0$ **then**
Update node j: $V^- = V^- \setminus \{j\}$
end if
end if
end do

Limited Movement:
do $\parallel V^+ \parallel$ **times**
Find $\{i, j, l\} = argmin\Big\{c_{il} - c_{lj} \mid \gamma_l^k \geq 0,\ (l,j) \in T^k \text{ and } (i,l) \in E \setminus T^k\Big\}$ where $i \in V^+$, the node j is the nearest node of V^- connected to i, and l is a neighbor of j but not in the path i, j
Swap these edges within the tree: $T^k = T^k \setminus (l,j) \cup (i,l)$
Update node i: $\gamma_i^k = 0$ and $V^+ = V^+ \setminus \{i\}$
Calculate γ_j^k
if $\gamma_j^k = 0$ **then**
Update node j: $V^- = V^- \setminus \{j\}$
end if
end do
return T^k.

- It is initialized $\overline{L} = L(u^0) + \alpha_L(UB - L(u^0))$ with $0 < \alpha_L < 1$
- If $L(u^k) > \overline{L}$, it is updated $\overline{L} = L(u^k) + \alpha_L(UB - L(u^k))$
- If $\overline{L} > UB$, then $\overline{L} = UB$

Finally, in lines 21-27, the parameter δ_k is initialized to the value 2 and is updated as Zamani and Lau (2010) suggested: If the lower bound is not improved, δ_k is decreased, using the formula $\delta_{k+1} = \delta_k \rho$ with $0 < \rho < 1$. On the other hand, if the lower bound is improved, then its value is increased according to the formula $\delta_{k+1} = \delta_k \frac{3-\rho}{2}$ providing that $0 \leq \delta_k \leq 2$.

1.7 Computational Results

The methodology described in this chapter has been implemented in Java language. All tests have been performed on a non-dedicated server with an Intel Xeon Quad-Core i5 processor at 2.66GHz and 16GB RAM. In general, different processes were launched to solve different problems, while external applications were active at the same time. For this reason, CPU time is provided just for giving a rough idea of the algorithm computational performance.

A total of 64 symmetric TSP instances have been used to test the efficiency of the proposed approach. They have been obtained from the library TSPLIB (http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/ last updated August 6, 2008), a reference site with a large number of instances for the TSP, and related problems, from various sources and of various types.

The experiments have been conducted using the distance according to the specification included in the library. The number of iterations (300) used by (Reinelt, 1994) and the values $\delta_0 = 2$ and $\rho = 0.95$ suggested by Zamani and Lau (2010) are used without the setting of parameter $\alpha_L = 1/3$.

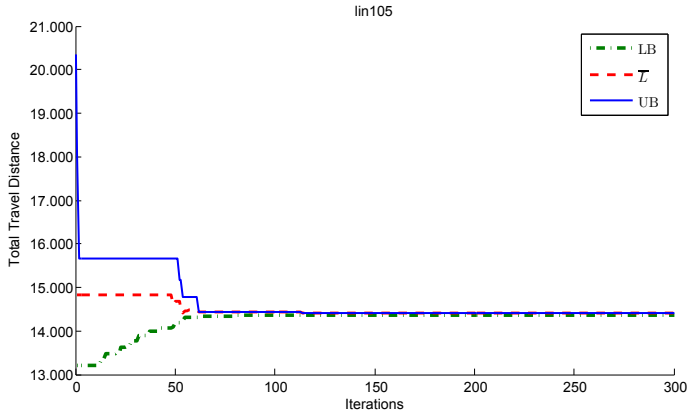
Size	Problems	$\%\Delta UB$	$\%\Delta LB$	$\%\Delta\overline{L}$
$n \leq 200$	27	3.15 %	1.96 %	-0.25 %
$200 < n \leq 500$	13	6.67 %	2.48 %	0.57 %
$500 < n \leq 1889$	24	12.53 %	1.74 %	3.02 %
	64	7.38 %	1.98 %	1.14 %

Table 1.5: Summary of results obtained applying the Tailored Lagrangian Metaheuristic for the Traveling Salesman Problem.

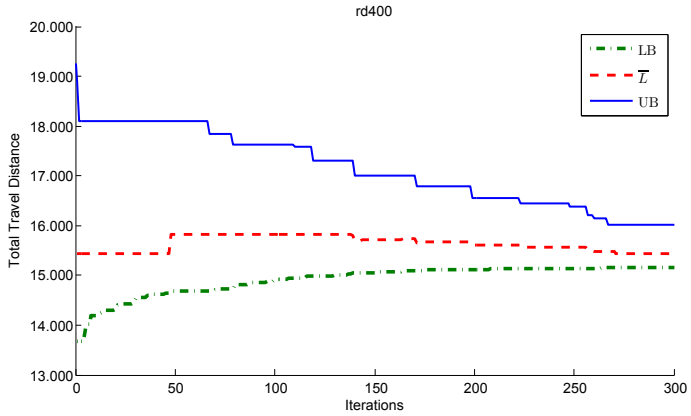
Table 1.5 presents the number of problems solved ordering by size, as well as the average gap of the obtained values UB, LB, and $\overline{L}$ from the best known value (BKS). The gap $\%\Delta\overline{L}$ was calculated as follows $100\left(\frac{\overline{L}-BKS}{BKS}\right)$. Therefore, if the gap is negative, the parameter $\overline{L}$ is smaller than the best known solution. Thus, the parameter $\overline{L}$ is

Problem	BKS	UB	$\%\Delta UB$	LB	$\%\Delta LB$	CPU_{time} (s)
eil51	426	440	3.29 %	422.42	0.84 %	0.226
st70	675	687	1.78 %	670.89	0.61 %	0.255
eil76	538	544	1.12 %	536.94	0.20 %	0.297
pr76	108159	108584	0.39 %	105112.85	2.82 %	0.250
rat99	1211	1237	2.15 %	1205.91	0.42 %	0.257
kroA100	21282	21319	0.17 %	20930.10	1.65 %	0.432
kroB100	22141	22931	3.57 %	21830.02	1.40 %	0.267
kroC100	20749	21270	2.51 %	20467.78	1.36 %	0.530
kroD100	21294	21792	2.34 %	21138.27	0.73 %	0.298
kroE100	22068	22905	3.79 %	21798.00	1.22 %	0.734
rd100	7910	8006	1.21 %	7898.65	0.14 %	0.499
eil101	629	629	**0.00%**	627.35	0.26 %	0.360
lin105	14379	14402	0.16 %	14370.14	0.06 %	0.237
pr107	44303	44420	0.26 %	39313.79	11.26 %	0.379
pr124	59030	60112	1.83 %	58048.18	1.66 %	0.463
bier127	118282	124502	5.26 %	117396.83	0.75 %	1.404
ch130	6110	6203	1.52 %	6074.46	0.58 %	0.351
pr136	96772	102534	5.95 %	95373.05	1.45 %	0.905
ch150	6528	6640	1.72 %	6488.12	0.61 %	1.279
kroA150	26524	27598	4.05 %	26280.90	0.92 %	1.608
kroB150	26130	27273	4.37 %	25725.39	1.55 %	1.577
pr152	73682	78848	7.01 %	66583.97	9.63 %	0.219
u159	42080	42636	1.32 %	41921.90	0.38 %	1.186
rat195	2323	2489	7.15 %	2290.58	1.40 %	2.443
d198	15780	17500	10.90 %	14372.36	8.92 %	3.093
kroA200	29368	30983	5.50 %	29056.19	1.06 %	3.231
kroB200	29437	31126	5.74 %	29158.27	0.95 %	2.960
ts225	126643	144299	13.94 %	115472.09	8.82 %	1.145
tsp225	3919	4261	8.73 %	3875.89	1.10 %	3.230
pr226	80369	84203	4.77 %	78451.35	2.39 %	3.155
gil262	2378	2473	3.99 %	2354.00	1.01 %	5.117
pr264	49135	50868	3.53 %	46236.48	5.90 %	5.180
a280	2579	2704	4.85 %	2565.40	0.53 %	7.202
pr299	48191	51053	5.94 %	47360.81	1.72 %	7.860
lin318	42029	45017	7.11 %	41768.56	0.62 %	5.695
rd400	15281	16061	5.10 %	15147.55	0.87 %	17.276
fl417	11861	13137	10.76 %	11144.00	6.05 %	12.179
pr439	107217	112768	5.18 %	105306.76	1.78 %	11.249
pcb442	50778	53840	6.03 %	50456.63	0.63 %	16.069
d493	35002	37397	6.84 %	34721.01	0.80 %	35.550
u574	36905	38985	5.64 %	36690.41	0.58 %	35.381
rat575	6773	7317	8.03 %	6718.70	0.80 %	44.725
p654	34643	43638	25.96 %	32487.42	6.22 %	3.858
d657	48912	52985	8.33 %	48420.58	1.00 %	115.474
u724	41910	46883	11.87 %	41618.86	0.69 %	160.662
rat783	8806	9674	9.86 %	8766.59	0.45 %	132.460
pr1002	259045	283008	9.25 %	256272.34	1.07 %	252.525
u1060	224094	250775	11.91 %	221469.74	1.17 %	411.675
vm1084	239297	265714	11.04 %	235787.50	1.47 %	294.671
pcb1173	56892	63499	11.61 %	56276.66	1.08 %	617.543
3d1291	50801	56274	10.77 %	49872.37	1.83 %	415.448
rl1304	252948	269195	6.42 %	248809.85	1.64 %	492.303
rl1323	270199	297319	10.04 %	265325.26	1.80 %	245.389
nrw1379	56638	64474	13.84 %	56284.49	0.62 %	1185.057
fl1400	20127	23358	16.05 %	19089.32	5.16 %	2968.874
u1432	152970	176933	15.67 %	152031.82	0.61 %	619.392
fl1577	22249	25631	15.20 %	20661.34	7.14 %	710.379
d1655	62128	69956	12.60 %	61177.18	1.53 %	1941.611
vm1748	336556	381719	13.42 %	331209.14	1.59 %	1456.419
u1817	57201	64062	11.99 %	56583.97	1.08 %	2465.488
rl1889	316536	351197	10.95 %	310987.88	1.75 %	1820.504
u2152	64253	73443	14.30 %	63753.38	0.78 %	1135.083
u2319	234256	283523	21.03 %	233605.99	0.28 %	18698.743
pr2392	378032	434652	14.98 %	372679.05	1.42 %	5759.861

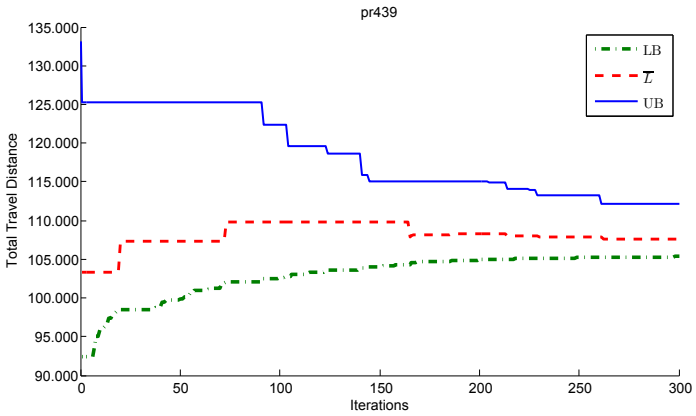
Table 1.6: Results obtained applying the Tailored Lagrangian Metaheuristic for the Traveling Salesman Problem



(a) Problem lin105



(b) Problem rd400



(c) Problem pr439

Figure 1.7: Convergence of LB, $\overline{L}$, and UB in some problems.

a good estimation with regard to an unknown lower bound or an initial upper bound obtained with a Nearest Neighbor Heuristic.

Table 1.6 presents results of a total of 64 instances comparing the results obtained to best known solution (BKS). The columns shows: UB best feasible value, $\%\Delta UB$ percentage distance from BKS of the UB, LB best dual value, $\%\Delta LB$ percentage distance from BKS of the LB, and t_{Final} final CPU time in seconds.

Figure 1.7 shows the evolution of upper and lower bounds on three runs. As it can be seen, the parameter $\overline{L}$ is updated according to the previous conditions explained, and it should be noticed that UB is improved every time the proposed heuristic finds a better value. The figure shows how LB, $\overline{L}$, and UB converge on BKS.

For the lin105 problem in Figure 1.7a, it has a quickly convergence in the first 100 iterations, however, it does not reach the optimal. The initial UB obtained with the Nearest Neighbor heuristic has a gap of 41.57%, our Lagrangian methodology dramatically improves this value up to 8.84% only in the second iteration. It shows the convergence of UB, LB and our parameter $\overline{L}$ with theirs final respective gaps of 0.16%, 0.06% and 0.16%.

Figure 1.7b shows rd400 problem, the initial UB obtained with the Nearest Neighbor heuristic has a gap of 26.06%, it shows the convergence of UB, LB and our parameter $\overline{L}$ with theirs final respective gaps of 5.10%, 0.87% and 1.03%.

For the pr439 problem in Figure 1.7c, the initial UB obtained with the Nearest Neighbor heuristic has a gap of 24.20%, it shows the convergence of UB, LB and our parameter $\overline{L}$ with theirs final respective gaps of 5.18%, 1.78% and 0.36%.

1.8 Conclusions

This chapter has presented a metaheuristic methodology based on the Lagrangian Relaxation technique. This scheme has been used to tackle the Traveling Salesman Problem, showing very good results both in terms of the quality of the solution and in terms of computational efficiency.

Our proposed Tailored Lagrangian Metaheuristic method uses the Subgradient algorithm combined with a heuristic. The motivation of our method is the common sense belief that dual solutions must obviously be relevant to primal solutions. The proposed heuristic is introduced in order to obtain a feasible solution from the dual variable providing both UB and LB, thus an a posterior quality check of the solution is obtained.

A new parameter $\overline{L}$ which is updated using the values of both UB and LB has improved significantly algorithm's convergence on the optimal solution.

In spite of optimality can not be always reached, the proposed method is able to provide a feasible solution with a tight gap between the primal and the optimal cost in a reasonable number of iterations.

Chapter 2

Capacitated Vehicle Routing Problem

This chapter presents an application of the Tailored Lagrangian Metaheuristic (TLM) proposed in Section 1.6. It presents an original hybrid approach to solve the Capacitated Vehicle Routing Problem (CVRP). The approach proposes a Multi-Start Variable Neighborhood Descent (VND) (P. Hansen & Mladenović, 2003) structure whose local search process is supported by Constraint Programming (CP) (Rossi et al., 2006) and our TLM.

On the one side, our metaheuristic is used to efficiently find the optimal routing solution for each transportation resource. On the other side, using the CP paradigm provides the required flexibility to model those operational constraints, beyond vehicle capacity, that are usually present in most real application cases. Due to this approach, adding these constraints is just a constraint modeling issue, i.e., no change on the solving strategy is required to deal with more realistic problems.

A probabilistic (Randomized) Clarke and Wright Savings (RCWS) (Juan et al., 2011) constructive method is used to generate initial solutions. This algorithm provides different good quality solutions that are used as seeds to launch the exploration of different regions of the search space. Therefore, the RCWS probabilistic behavior introduces a natural diversification mechanism and turns the scheme into an approach likely to be parallelized.

The CVRP has been selected mainly because there are huge amounts of models, techniques, benchmarks, and research on this topic. Hence, the proposed methodology can be easily compared —in terms of computational efficiency and solution quality— with previously existing approaches. Nevertheless, from the perspective of its industrial applicability, the basic CVRP model can be extended to tackle different realistic cases by means of the proposed optimization scheme. These cases include operational constraints beyond the basic vehicle capacity. Affordable examples are limitations on the total driving time of each route, incompatible customer-driver associations or constraints on the customer visiting periods (e.g., customer forbidden visit day when building daily routes for a distribution problem). The proposed optimization approach, as presented

in this chapter, is specifically designed to deal with those operational constraints that involve the allocation part of the VRP.

The approach presented in this chapter was first presented in (Herrero et al., 2010a) and (Guimarans et al., 2010) where we introduced a first methodology which combined CP and TLM within a VNS framework. The current approach was presented in (Guimarans et al., 2011b) and (Guimarans et al., 2013), it also integrates the RCWS-based Multi-Start approach which is significantly more efficient –both in terms of computational times and solutions' quality.

2.1 Problem Definition

The Vehicle Routing Problem (VRP) provides a theoretical framework for approaching the class of logistic problems dealing with physical distribution. This is among the most popular research areas in combinatorial optimization. It was first defined by Dantzig and Ramser (1959), and several variants of the basic problem have been proposed and studied later. These variants represent different types of operational constraints such as, for instance, time windows, pick up and delivery, heterogeneous fleets or multi-depot problems.

The Capacitated version of the VRP (CVRP) consists of determining the optimal set of routes for a fleet of vehicles to deliver goods to a given set of customers, therefore it is a generalization of the TSP. In the model proposed in this chapter, the CVRP has been divided into two subproblems, concerning customers' allocation and routing optimization separately. The first is aimed to assign customers to vehicles fulfilling capacity limitations. The latter is used to solve each independent route to optimality, giving the best solution for a particular allocation. Thus, routing optimization process can be viewed as solving a set of independent symmetric TSP.

The CVRP has been chosen in order to illustrate the benefits of the proposed methodology. The CVRP is the most basic VRP variant, which assumes a fleet of vehicles of homogeneous capacity housed in a single depot. The CVRP is a generalization of the Traveling Salesman Problem (TSP) and is therefore NP-hard (Savelsbergh, 1985).

The CVRP is defined over a complete graph $G = \{I, E\}$, where $I = \{1, 2, \ldots, n\}$ is the node set representing clients to be served plus the depot (node 1), and $E = \{e_{ij} = (i, j) \mid i, j \in I\}$ is the edge set representing connecting roads, streets, etc. Edges e_{ij} in E have an associated cost $c_{ij} > 0$, which is the traveling cost from node i to node j. It is usual to consider symmetric costs (i.e., $c_{ij} = c_{ji}, \forall i, j \in I$). Moreover, each node i in I has a demand $q_i \geq 0$. A fixed fleet of m identical vehicles, each one with capacity $Q \gg max\{q_i\}$, is available at the depot to accomplish the required delivery task.

Solving the CVRP consists of determining a set of $k \leq m$ routes with minimum total traveling cost and such that:

a) each customer is visited exactly once by a single vehicle,

b) each route starts and ends at the depot, and

c) the total demand of the customers assigned to a route does not exceed the vehicle capacity Q.

Therefore, a solution of a given CVRP instance is a set of k routes sharing a common starting and finishing node (the depot). In some cases, the fleet size is not fixed and minimizing the total number of used vehicles becomes an additional objective.

Given that our approach combines LR to optimize each individual route, and CP to quickly verify the feasibility, it is based on the classical decomposition into two subproblems: a resource allocation problem (to fit operational constraints), and a routing problem (to minimize the associated traveling costs).

Capacity problem

The proposed customers' allocation subproblem uses two lists of variables. A list R of size n, with integer domains $R_i \in [1..m] \mid i = 1, ..., n$, indicates which vehicle is serving the i^{th} customer. Q_v is a list of m variables with real domain $Q_v \in [0..Q]$ used to trace the cumulative capacity at each one of the m routes. Therefore, capacity constraints are enforced through domains definition since Q_v cannot get higher values than the maximum capacity Q satisfying constraint (c).

A set of dimension $m \times n$ of binary variables B has been introduced to relate R and Q_v values. For each vehicle $v \in V$, a list of n binary variables $B_{vi} \mid i \in I$ is defined, taking value 1 whenever customer i is assigned to vehicle v and 0 otherwise. Since each customer i is visited by a single vehicle, for all values of v the binary variable B_{vi} can take value 1 only once. This constraint is expressed in terms of the global constraint *cardinality_atmost* (Beldiceanu, Carlsson, & Rampon, 2005) aiming to ensure a faster propagation (Bessière, 2006).

The binary set B and allocation variables R are related through the following statement:

$$R_i = r_i \rightarrow B_{r_i i} = 1 \ \forall i \in I \tag{2.1}$$

Expression (2.1) states that the i^{th} element of the r_i list of B will have value 1 whenever the i^{th} component of R takes value r_i. The global constraint *cardinality_atmost* ensures propagation so all values of B_{vi} $v \in V \setminus r_i$ are set to 0 automatically. Therefore, cumulative capacities can be traced simply by using the following equation:

$$Q_v = \sum_{i \in I} B_{vi} q_i \ \forall v \in V \tag{2.2}$$

Routing Problem

The routing problem, tackled for each vehicle separately, can be viewed as a TSP instance. For each vehicle v, the related TSP can be considered as a complete undirected graph $G = (I_v, E_v)$, connecting assigned customers $I_v = \{i \in I \mid R_i = v\}$ through a set of undirected edges $E_v = \{(i, j) \in E \mid i, j \in I_v\}$. The solution is a path connected by

edges belonging to E_v that starts and ends at the depot ($i = 1$) and visits all assigned customers. Following the mathematical formulation presented in Section 1.1, a feasible solution of the TSP should, by definition, also satisfy constraints (a) and (b) of the CVRP, minimizing the total travel cost of the route.

2.2 Literature Review

The Clarke and Wright's Savings (CWS) constructive algorithm (Clarke & Wright, 1964) is probably the most cited heuristic to solve the CVRP. The CWS is an iterative method that starts out by considering an initial dummy solution in which each customer is served by a dedicated vehicle. Next, the algorithm initiates an iterative process for merging some of the routes in the initial solution. Merging routes can improve the expensive initial solution so that a unique vehicle serves the nodes of the merged route. The merging criterion is based upon the concept of savings. Roughly speaking, given a pair of nodes to be served, a savings value can be assigned to the edge connecting these two nodes. This savings value is given by the reduction in the total cost function due to serving both nodes with the same vehicle instead of using a dedicated vehicle to serve each node -as proposed in the initial dummy solution. This way, the algorithm constructs a list of savings, one for each possible edge connecting two demanding nodes. At each iteration of the merging process, the edge with the largest possible savings is selected from the list as far as the following conditions are satisfied: the nodes defining the edge are adjacent to the depot, and the two corresponding routes can be feasibly merged -i.e., the vehicle capacity is not exceeded after the merging. The CWS algorithm usually provides relatively good solutions, especially for small and medium-size problems, but it also presents difficulties in some cases (Gaskell, 1967). Many variants and improvements of the CWS have been proposed in the literature. For a comprehensive discussion on the various CWS variants, the reader is referred to Toth and Vigo (2002b) and Laporte (2007).

Monte Carlo Simulation (MCS) can be defined as a set of techniques that make use of random numbers and statistical distributions to solve certain stochastic and deterministic problems (Law, 2007). MCS has proved to be extremely useful for obtaining numerical solutions to complex problems that cannot be efficiently solved by using analytical approaches. Buxey (1979) was probably the first author to combine MCS with the CWS algorithm to develop a procedure for the CVRP. This method was revisited by Faulin and Juan (2008), who introduced an entropy function to guide the random selection of nodes. MCS has also been used by Fernández et al. (2000), Juan et al. (2008), Faulin et al. (2008) and Juan et al. (2009) to solve the CVRP. In this last paper, the authors make use of MCS to develop an efficient randomized version of the CWS heuristic, which we use in our approach to efficiently generate initial solutions.

Another way to address the VRP has been the use of complete methods, which ensure not only to find the solution but also, to prove its optimality. The main drawback of these techniques is that they may only deal with small instances, up to 100 customers (Cordeau et al., 2007). Numerous heuristics (like the ones mentioned above) and metaheuristics

have also been studied for different VRP variants. In most cases, these methods may solve larger instances but loosing optimality guarantees.

Using constructive heuristics as a basis, metaheuristics became popular for the VRP during the nineties. Some early examples are the Tabu Route method by Gendreau et al. (1994) or the Boneroute method of Tarantilis and Kiranoudis (2002). Tabu search algorithms, like those proposed by Taillard (1993) or Toth and Vigo (2003) are among the most cited metaheuristics. Genetic algorithms have also played a major role in the development of effective approaches for the VRP. Some examples are the studies of Berger and Barkaoui (2003), Prins (2004), Mester and Bräysy (2007) or Nagata (2007). Another important approach to the VRP is given by the Greedy Randomized Adaptive Search Procedure or GRASP (Feo & Resende, 1995; Resende, 2008; Festa & Resende, 2009). A GRASP algorithm is a multi-start or iterative process in which each GRASP iteration consists of two phases: a construction phase —in which a feasible solution is produced— and a local search phase —in which a local optimum in the neighborhood of the constructed solution is sought. The best overall solution is kept as the result. In the construction phase, a feasible solution is iteratively constructed, one element at a time. At each construction iteration, the choice of the next element to be added is determined by ordering all candidate elements in a candidate list according to a greedy function. This function measures the (myopic) benefit of selecting each element. The heuristic is adaptive because the benefits associated with every element are updated at each iteration of the construction phase to reflect the changes brought on by the selection of the previous element. The probabilistic component of a GRASP is characterized by the random choice of one of the best candidates in the list, but not necessarily the top candidate. This choice technique allows for different solutions to be obtained at each GRASP iteration.

Among metaheuristics, Variable Neighborhood Search (VNS), introduced for the first time by Mladenović and Hansen (1997), is a quite recent method with far less application examples in VRP research. However, interesting results have been obtained even applying the simplest VNS algorithms, e.g., (Hasle & Kloster, 2007). Embedding CP and LR approaches into a general VNS framework has also demonstrated to be an effective yet slow method to solve medium and large instances (Guimarans et al., 2011a). Combining these techniques provided a methodology able to reach good quality results and even to overcome some best-known solutions. However, the computational efficiency of this methodology is far from state-of-the-art algorithms and becomes an important issue to be addressed.

2.3 Technologies used

In this chapter we present a hybrid approach combining a randomized version of the CWS savings heuristic, the VNS metaheuristic, CP, and LR. Our approach aims at being an efficient procedure for obtaining quasi-optimal solutions in small- and medium-size CVRP instances and, at the same time, offers some additional advantages over other existing metaheuristics, namely: (a) it is a robust and flexible methodology that can be

easily adapted to consider additional constraints and costs; (b) it is able to generate a set of alternative good solutions in a reasonable time period; and (c) it can be easily executed in parallel.

This work has been published in Guimarans et al. (2011b), it is related to the PhD Thesis of Guimarans (2012) and a previous version can be found in Herrero et al. (2010a).

Our presented approach is much more competitive with state-of-the-art metaheuristics. Its efficiency has been significantly enhanced by including a multi-start procedure which makes use of a randomized CWS heuristic in order to quickly provide a set of different "good" initial solutions, over which a flexible local-search process is applied. Thus, the VNS diversification procedure is substituted by a multi-start approach, where different regions are explored thanks to the diversity of solutions provided by the randomized CWS algorithm. The local search process has also been enhanced with respect to the previous work by incorporating new data structures, which permit reducing the computational complexity. Finally, the methodology described in the present work has been parallelized to improve its efficiency.

Remember that our Tailored Lagrangian Metaheuristic is explained in Section 1.6, let us introduced the other used technologies in this approach.

Probabilistic Clarke and Wright Savings Algorithm

As discussed in Section 2.2, in the classic CWS algorithm, the edge with the largest possible savings is selected from the list at each iteration of the merging process, as far as the following conditions are satisfied: (a) the nodes defining the edge are adjacent to the depot, and (b) the two corresponding routes can be feasibly merged -i.e., the vehicle capacity is not exceeded. The approach presented in Juan et al. (2010), instead, assigns a selection probability to each edge in the savings list. This probability should be coherent with the savings value associated with each edge, i.e., edges with larger savings will be more likely to be selected from the list than those with smaller savings. In addition, this approach adds this biased random behavior without introducing too many parameters in the algorithm. Basically, different geometric statistical distributions during the randomized CWS solution-construction process are employed: every time a new edge is selected from the list of available edges, a value α is randomly selected from a uniform distribution in (a, b), where $0 < a \leq b < 1$. The α parameter defines the specific geometric distribution that will be used to assign exponentially diminishing probabilities to each eligible edge according to its position inside the sorted savings list. This way, edges with higher savings values are always more likely to be selected from the list, but the exact probabilities assigned are variable and they depend on the concrete distribution selected at each step.

Constraint Programming

CP is a powerful paradigm for representing and solving a wide range of combinatorial problems. Problems are expressed in terms of three entities: variables, their corresponding domains and constraints relating them. The problems can then be solved using

complete techniques such as depth-first search for satisfaction and branch and bound for optimization, or even tailored search methods for specific problems. Rossi et al. (2006) present a complete overview of CP modeling techniques, algorithms, tools, and applications.

Variable Neighborhood Search

A general VNS, as explained in (P. Hansen & Mladenović, 2003), is a recent metaheuristic which exploits systematically the idea of neighborhood change. The Variable Neighborhood Descent (VND) method starts from an initial solution x' and it is improved by a local search process.

The local search process for each neighborhood $N(x')$ of x' performs an exhaustive exploration. All improving movements are recorded and sorted, so the best neighbor $x'' \in N(x')$ is constructed applying all independent changes in descending order. This way, solution values are improved faster than applying single movements.

If this neighbor is better than the incumbent, the current solution is updated and neighborhoods' exploration is restarted. Otherwise, the algorithm keeps x' as the best solution found so far and continues exploring the next neighborhood. When the VND process reaches a local optimum, no solution improvement may be found according to defined neighborhoods.

Multi-Start Strategy

The VND-based local search process requires some type of diversification in order to overcome local optimality. Many techniques have been suggested to avoid getting trapped into a local optimum and aspire to find a global one. Among others, one possible way to achieve diversification is using a shaking mechanism within the VNS procedure. However, as more constraints are introduced in the problem, it usually becomes more efficient -in terms of computational time employed- to generate new feasible solutions from scratch than to apply complex shaking processes that might end in non-feasible solutions. This is especially certain if we consider that the Randomized version of the CWS used in this paper is a really fast method for generating different feasible and good solutions that can serve as initial solutions in our multi-start approach.

Thus, the Multi-Start strategy provides an appropriate framework which achieves diversification by re-starting the search from a new solution once a region has been extensively explored. Notice that each iteration includes two phases: a first one in which a new feasible solution is constructed, and a second one in which the initial solution is improved through a local search process.

2.4 The methodology in detail

The CVRP problem has been tackled using a RCWS-based Multi-Start approach. As it has been previously discussed, this strategy allows to ensure an efficient diversification

of the search space in order to: (a) avoid local minima, and (b) reach near-optimal solutions in reasonable times.

CP and TLM are used in the local search process within a VND structure. CP is used to check solutions feasibility. This formalism provides a fast and flexible method, able to model and include complex constraints while keeping a reasonable computational efficiency. In turn, out TLM method is applied to calculate routes every time a partial solution is generated. Using TLM allows reducing the computation time and algorithm's definition and complexity when compared to other routing post-optimization methods (Rousseau et al., 2002).

Pseudo-code for the Multi-Start strategy

A simplified scheme of the Multi-Start strategy is presented in Algorithm 5. The RCWS algorithm is used to find a good initial solution. Then, the VND method helps to reach a local minimum in the neighborhood of the solution.

The Multi-Start strategy generates $Total_{Threads}$ tasks within a thread pool. If a thread is not available for the task, the task waits in a queue for an active task to end. The algorithm stops when all tasks have been completed, or the maximum execution time is reached, whichever happens first. Each task executes two phases: find an initial solution and improve it in the search process. Starting from a different initial solution ensures certain diversification, overcoming local optimality.

Algorithm 5 Multi Start Approach

```
Let x be the best solution
Create a thread pool with Total_Threads threads.
repeat until Total_Threads threads end or until Max_Seconds time is consumed
  execute Max_Threads simultaneous threads:
    Generate an initial feasible solution x' using RCWS
    Improve x' to obtain x'' by using VND+CP+TLM
    if x'' is better than x then
      let x ← x''
    end if
end repeat
```

Pseudo-code for the Variable Neighborhood Descent procedure

A general VND has been implemented embedding CP and TLM methods. In the implemented algorithm, outlined in Algorithm 6, all four described moves (see section 2.4) have been selected to be used in local search neighborhoods.

In the exploration neighborhood (N_k), starting from the solution x', the k^{th} move is applied and the new solution's feasibility is checked using CP. Whenever it is proved feasible, TLM is used to recalculate only modified routes. This technique permits to consider only two routes per solution, reducing the computation time. Improvements

Algorithm 6 Variable Neighborhood Descent Algorithm

Initialize the set $LastModified \leftarrow V$; let x' be the initial solution
repeat until the stopping condition
  Set $k \leftarrow 1$;
  repeat until $k = k_{max}$
    Exploration of Neighborhood
      Find all neighbors $x'' \in N_k(x', LastModified)$.
      Check feasibility of capacity constraints using CP.
      Calculate the cost of modified routes using TLM
      if the solution x'' is better than x' **then**
        Include it in a list of improving changes
      end if
    Choose the best compatible neighbors
      Set $LastModified \leftarrow \emptyset$
      Sort the list of improving changes
      Apply the first improving changes
      Add in descending order the next compatible improvements
      Add the modified routes to $LastModified$
    if the list is empty **then**
      set $k \leftarrow k + 1$
    else
      set $k \leftarrow 1$
    end if
  end repeat
end repeat

are stored in a sorted list until no more feasible solutions are left in the k^{th} neighborhood. Then, all those which are independent, i.e., affect different route pairs, are applied in descending order on x' to get a better solution x''. This way, solution improvement is faster than applying a single change at each iteration.

After the first exhaustive exploration of each neighborhood, only those changes affecting routes modified by previous movements are explored in order to reduce the computation time. The modified routes are stored in the set $LastModified$. A similar approach may be found in Zachariadis and Kiranoudis (2010).

Inter-route Moves

The VNS metaheuristic is based on the exploration of different neighborhoods around a given feasible solution. In order to establish these neighborhoods, several moves are defined. In our approach, four different inter-route classical moves (Savelsbergh, 1988) have been identified to be used within the local search process:

(a) *Relocate* moves a customer from one route to another,

(b) *Swapping* exchanges two customers belonging to different routes,

(c) *Chain* is a specialization of 3-opt that swaps sections of two contiguous customers from different routes, and

(d) *Ejection chain* swaps the end portions of two different routes.

The use of our TLM ensures the partial optimality of most solutions from the routing perspective. The reason is that, since we are considering a relatively small number of customers per route, the proposed approach can quickly find the optimal solution to most TSP instances. In effect, the respective lower bounds (LB) and upper bounds (UB) converge rapidly, keeping their gap between 0 and 10^{-10}, which guarantees the solution optimality. In addition, it solves all routes in negligible times. Thus, it is an efficient alternative for intra-route optimization processes and avoids defining intra-route moves.

2.5 Computational Results

The methodology described in this paper has been implemented in Java and linked to the open-source CP software system ECLiPSe 6.0 (Apt & Wallace, 2007). All tests have been performed on a dedicated server with an Intel i5 processor at 2.66GHz and 16GB RAM. A total of 91 classical CVRP benchmark instances available at `www.branchandcut.org` have been used to test the efficiency of the proposed approach when dealing with this simple (in terms of constraints) but extensively tested scenario. In order to ensure fulfillment of the triangular inequality property, only those instances using Euclidean metrics have been selected. The selected problems also include 7 instances from (Christofides et al., 1979) (denoted in tables as C1-C5, C11, and C12) for further comparison with some recent metaheuristics.

As the algorithm has been designed to be run in a parallel computing environment, a test has been done over the set A of benchmark problems to determine the most suitable number of simultaneous threads. This parameter is to be fixed mainly according to computer's characteristics. In the particular server used in this work, up to 4 threads may be executed in parallel in order to keep a reasonable computational efficiency. In the performed test, adopting a parallelized approach permits reducing the total computation time significantly. In particular, for problems from the set A, the total computation time is 41% lower, on average, than the total time spent using a sequential approach. For this reason, all results presented in this paper correspond to a Multi-Start VND implementation with 4 parallel processes, since this approach has demonstrated to keep a reasonable balance between the time spent on calculating one single solution and the total execution time.

Table 2.1 shows results obtained for some representative problems from the selected benchmark sets. Due to algorithm's probabilistic behavior, the final solutions' quality depends on the total number of threads. For this reason, 100 total tasks have been considered for each problem, i.e., 100 pseudo-optimal solutions have been generated for

each benchmark instance. Table 2.1 summarizes information regarding the best solution found (OBS) for each problem, as well as the time required to reach this solution. These results are compared to the best known solutions (BKS) so far. Most sources give these values as integer numbers, obtained by rounding costs, except for the problems from (Christofides et al., 1979) where real values are usually given. From the detailed integer solutions, real costs have been calculated and reported. It should be remarked that the real cost of an integer optimal solution might not correspond to the optimal solution considering real costs. For this reason, negative gaps appear on this table. Thus, it can be deduced that the Multi-Start VND is able to match, and in many cases improve, the real value associated to the best known integer solutions. Concretely, the presented approach has been able to improve 23 best known solutions, considering real costs, out of the 91 tested instances. In addition, the gap is kept reasonably low for all considered instances, being the average gap 0.65 %. It remains lower, 0.17 %, for the problems selected in Table 2.1, which include most of the largest instances.

Furthermore, it should be remarked that these results have been obtained in competitive times even for some large instances. As shown in Table 2.1, most solutions for small problems are obtained in less than a second, while larger instances require higher yet reasonable computational times. In most cases, higher times are closely related to higher quality solutions, i.e., solutions with a negative gap.

Table 2.2 shows how these results are similar to other state-of-the-art metaheuristics. It provides a comparison between the proposed approach and some recent publications. The first three metaheuristics are: a hybrid algorithm of Simulated Annealing and Tabu Search (SA-TS) introduced in (Lin et al., 2009); a hybrid Electromagnetism-like heuristic (HEMA) proposed by Yurtkuran and Emel (2010); and a Particle Swarm algorithm (SR-2) described in (Jin Ai & Kachitvichyanukul, 2009). The next three metaheuristics correspond to our related work: a hybrid VNS (HVNS) presented in (Guimarans et al., 2011a); the randomized Clarke and Wright Savings (SR-GCWS) algorithm by Juan et al. (2010); and the Multi-Start VND presented in this chapter and in (Guimarans et al., 2011b).

Most publications only report results corresponding to the 14 instances from the workbench presented by Christofides et al. (1979). For this reason, few results corresponding to other problem sets are reported for the first three metaheuristics in Table 2.2. Moreover, some of the instances from (Christofides et al., 1979) include an additional constraint on the maximum route length that is not handled in the proposed version of the algorithm. Therefore, results for these instances have been omitted in this table.

It may be observed that the proposed approach is comparable in terms of quality and computational efficiency to these recent metaheuristics. Times needed by our approach to reach a pseudo-optimal solution are in most cases lower than those required by means of the other algorithms. It should be remarked that the proposed approach clearly improves the efficiency of the previous algorithms HVNS and SR-GCWS. Furthermore, the Multi-Start VND provides the lowest gap among all selected metaheuristics, only beated by the SR-GCWS approach. However, most of the higher gaps obtained with the proposed approach correspond to some of the largest instances, whose results are

Problem	# nodes	BKS	OBS	Gap (%) BKS-OBS	# routes	Time (s)
A-n32-k5	31	787.81	787.08	-0.09	5	0.633
A-n33-k5	32	662.76	662.11	-0.10	5	0.842
A-n33-k6	32	742.83	742.69	-0.02	6	0.480
A-n37-k5	36	672.59	673.59	0.15	5	1.948
A-n37-k6	36	952.22	950.85	-0.14	6	1.631
A-n38-k5	37	734.18	733.95	-0.03	5	2.546
A-n45-k6	44	944.88	944.88	0.00	6	1.622
A-n46-k7	45	918.46	918.13	-0.04	7	2.062
A-n54-k7	53	1171.78	1171.78	0.00	7	4.007
A-n55-k9	54	1074.46	1076.85	0.22	9	5.544
A-n63-k9	62	1622.14	1622.14	0.00	9	8.073
B-n31-k5	30	676.76	676.09	-0.10	5	0.657
B-n34-k5	33	791.24	789.84	-0.18	5	0.497
B-n35-k5	34	956.29	958.94	0.28	5	1.174
B-n38-k6	37	809.45	809.45	0.00	6	1.211
B-n39-k5	38	553.27	553.16	-0.02	5	1.577
B-n43-k6	42	747.54	746.98	-0.07	6	1.520
B-n45-k5	44	755.43	753.96	-0.19	5	1.011
B-n50-k7	49	744.78	744.23	-0.07	7	1.721
B-n50-k8	49	1316.20	1319.53	0.25	8	7.069
B-n51-k7	50	1035.71	1037.54	0.18	7	597.915
B-n57-k9	56	1603.63	1604.88	0.08	9	7.653
B-n64-k9	63	869.32	868.31	-0.12	9	287.953
E-n22-k4	21	375.28	375.28	0.00	4	0.337
E-n23-k3	22	568.56	568.56	0.00	3	0.422
E-n33-k4	32	838.72	837.67	-0.13	4	0.819
E-n51-k5 (C1)	50	524.61	527.98	0.64	5	17.164
E-n76-k10 (C2)	75	835.26	843.49	0.99	10	28.941
E-n101-k8 (C3)	100	826.14	841.16	1.82	8	195.271
F-n45-k4	44	724.57	727.75	0.44	4	4.459
F-n135-k7	134	1170.65	1179.09	0.72	7	630.427
G-n262-k25	261	5685.00	5722.00	0.65	25	1651.360
M-n101-k10 (C12)	100	819.81	821.40	0.19	10	51.395
M-n121-k7 (C11)	120	1042.11	1045.14	0.29	7	137.553
M-n151-k12 (C4)	150	1028.42	1052.52	2.34	12	834.642
M-n200-k17 (C5)	199	1291.45	1324.91	2.59	17	243.789
P-n16-k8	15	451.95	451.95	0.00	8	0.019
P-n19-k2	18	212.66	212.66	0.00	2	0.243
P-n20-k2	19	217.42	217.42	0.00	2	0.148
P-n21-k2	20	212.71	212.71	0.00	2	0.275
P-n22-k2	21	217.85	217.85	0.00	2	0.277
P-n23-k8	22	531.17	531.17	0.00	8	1.447
P-n40-k5	39	461.73	461.73	0.00	5	6.189
P-n45-k5	44	512.79	512.79	0.00	5	10.016
P-n50-k7	49	559.86	560.15	0.05	7	5.155
P-n51-k10	50	742.48	742.36	-0.02	10	5.156
P-n55-k10	54	697.81	698.00	0.03	10	5.331
P-n55-k8	54	592.17	581.17	-1.86	7	14.703
P-n76-k5	75	635.04	633.32	-0.27	5	92.627
P-n101-k4	100	692.28	693.54	0.18	4	839.622
Average				0.17		

Table 2.1: Results for 50 classical benchmark instances.

not reported for the SR-GCWS algorithm.

		SA-TS		HEMA		SR-2		HVNS		SR-GCWS		Multi-Start VND	
Problem	BKS	BS	t (s)	BS	t (s)	BS	t (s)	BS	t (s)	BS	t (s)	BS	t (s)
A-n32-k5	787.81	-	-	-	-	-	-	787.81	96.6	787.08	6.0	787.08	0.6
A-n33-k5	662.76	-	-	-	-	662.76	13.0	662.76	42.7	662.11	2.0	662.11	0.8
A-n33-k6	742.83	-	-	-	-	-	-	742.83	34.9	742.69	3.0	742.69	0.5
A-n37-k6	952.22	-	-	-	-	-	-	952.22	59.7	-	-	950.85	1.6
A-n38-k5	734.18	-	-	-	-	-	-	735.22	69.0	733.95	7.0	733.95	2.6
A-n45-k6	944.88	-	-	-	-	-	-	951.77	141.7	944.88	31.0	944.88	1.6
A-n46-k7	918.46	-	-	-	-	918.46	23.0	919.20	265.9	-	-	918.13	2.1
A-n63-k9	1622.14	-	-	-	-	-	-	1637.58	1254.3	1622.14	145.0	1622.14	8.1
B-n31-k5	676.76	-	-	-	-	-	-	-	-	676.09	1.0	676.09	0.7
B-n39-k5	553.27	-	-	-	-	-	-	-	-	553.16	17.0	553.16	1.6
B-n45-k5	755.43	-	-	-	-	755.43	20.0	-	-	754.22	20.0	753.96	1.0
B-n50-k7	744.78	-	-	-	-	-	-	-	-	744.23	2.0	744.23	1.7
E-n33-k4	838.72	-	-	-	-	-	-	-	-	837.67	7.0	837.67	0.8
E-n51-k5 (C1)	524.61	524.61	38.1	524.61	13.0	524.61	24.0	-	-	524.61	32.0	527.98	17.2
E-n76-k10 (C2)	835.26	835.26	118.3	849.77	19.0	844.42	57.0	-	-	835.26	21.7	843.49	28.9
E-n101-k8 (C3)	826.14	826.14	293.3	844.72	41.0	829.40	101.0	830.53	40888.9	-	-	841.16	195.3
M-n101-k10 (C12)	819.81	819.56	316	847.56	190.0	819.56	88.0	841.37	12897.9	819.56	338.0	821.40	51.4
M-n121-k7 (C11)	1042.11	1045.50	332.8	1042.11	319.0	1052.34	93.0	1092.13	39383.9	1043.88	74.5	1045.14	137.6
M-n151-k12 (C4)	1028.42	1038.71	701.4	1059.03	132.0	1048.89	223.0	1035.80	76933.9	-	-	1052.52	834.6
M-n200-k17 (C5)	1291.45	1311.70	1844.3	1302.33	201.0	1323.89	413.0	1325.07	195727.3	-	-	1324.91	243.8
P-n20-k2	217.42	-	-	-	-	-	-	217.42	0.9	217.42	41.0	217.42	0.2
P-n22-k2	217.85	-	-	-	-	-	-	217.85	4.7	217.85	9.0	217.85	0.3
P-n51-k10	742.48	-	-	-	-	-	-	742.48	427.3	741.50	19.0	742.36	5.2
P-n55-k8	592.17	-	-	-	-	-	-	580.96	247.7	-	-	581.17	14.7
P-n76-k5	635.04	-	-	-	-	-	-	633.32	10784.6	633.32	73.0	633.32	92.6
Average gap (%)		0.41		1.60		0.69		0.61		-0.05		0.23	

Table 2.2: Comparison between the proposed algorithm and other approaches for some classical benchmark instances.

As a final remark, it can be observed that the lowest gap (-1.86 %) corresponds to the problem P-n55-k8, where a solution considering only 7 vehicles (# routes) has been obtained. Although the best known solution for this problem uses 8 vehicles, feasible solutions with 7 vehicles and lower costs may be reached, as the one obtained with this approach. However, if only 7 vehicles are considered, the Multi-Start VND has finished slightly over the value 580.96 (576 considering integer costs), published for this problem in (Guimarans et al., 2011a), (Alba & Dorronsoro, 2008), and (Altinel & Oncan, 2005).

2.6 Conclusions

The described hybrid algorithm embeds CP and TLM within the VND metaheuristics framework by decomposing the CVRP into two subproblems concerning customers' allocation and routing optimization separately. This approach allows reducing the computation time during local search processes, since problems to be solved are far less complex than the original CVRP.

A fast and efficient algorithm such as the randomized CWS algorithm is used to feed the multi-start scheme by generating good initial solutions. Thus, the proposed optimization approach implements a flexible, efficient and robust optimization algorithm able to deal with some realistic problems, which means both the ability to tackle large instances and to represent real operational constraints. The characteristics of the resulting algorithm can be explained in the following way: flexibility involves the quality of the algorithm to be adapted to real problems; efficiency is related to the easiness of the algorithm to obtain optimal or quasi-optimal solutions in reasonable computation times; and robustness is related to the fact that the algorithm performs well even when no extensive fine-tuning processes are carried out on its parameters.

Regarding flexibility, this approach benefits from the CP capabilities to model different operational constraints. This constraints are present in most of the real application cases and, in general, affect to the allocation decisions. CP, which is not restricted by modeling limitations such as constraint linearity, facilitates the representation of allocation constraints without requiring any specific action on the solving method. Hence, the hybrid scheme can be easily adapted to different CVRP variants by simply adding the allocation constraints which properly model the feasible solutions of the problem. Since the VND optimization scheme is able to reach feasible solutions starting from non-feasible initial solutions, e.g., not fulfilling the maximum number of vehicles (Guimarans et al., 2010), the RCWS algorithm does not need to be modified in order to include operational constraints other than capacities. However, other capacity-like constraints, such as total driving time of each route, can be translated to a capacity constraint in order to obtain feasible solutions by means of the RCWS algorithm. Additional operational constraints may be added to the CP model, which will ensure solutions' feasibility along the local search process. Thus, this hybrid approach will be able to tackle complex instances related to real application cases by adding little modifications into the problem modeling, but neither into the optimization scheme nor algorithms.

The efficiency of the proposed algorithm is supported by the results presented in the

previous section. As discussed, the presented approach is able to match the best known solutions for benchmark problems of different sizes in reasonable computation times. The provided comparison proves that its efficiency is similar to other state-of-the-art metaheuristics, both in terms of time and solutions' quality.

The robustness of the algorithm is a consequence of the light requirements for fine tuning. Our TLM metaheuristic does not require any specific adjustment since all the convergence parameters are self-tuned. The CP-based subproblem depends just on the quality of the defined constraint model to properly describe the feasible solutions. The RCWS does not require any adaptation either. Only the VND movements could require different priorities depending on the problem being solved in order to get a better solution quality.

Our proposed Tailored Lagrangian Metaheuristic has been used to calculate routing cost for each vehicle separately. It solves all routes in negligible times and is an efficient alternative for intra-route optimization processes and avoids defining intra-route moves. Moreover, it ensures the partial optimality of most solutions from the routing perspective. The reason is that, since we are considering a relatively small number of customers per route, the proposed approach can quickly find the optimal solution to most TSP instances.

Chapter 3

Workload-Balanced and Loyalty-Enhanced Home Health Care

This chapter describes a practical application concerning the Home Health Care (HHC) service in the municipality of Ferrara, a mid-sized city in the North of Italy. The problem has been solved using a hybrid methodology based on Large Neighborhood Search (LNS) combining Constraint Programming (CP) and our Tailored Lagrangian Metaheuristic (TLM) proposed in Section 1.6.

Although sometimes it is necessary, no one likes to stay in a hospital, and patients who need to stay in bed but do not require constant medical surveillance prefer their own bed at home. At the same time, a patient in a hospital has a high cost for the community, that is not acceptable if the patient needs service only a few minutes a day. For these reasons, the Health Services providers in Europe and North America have tried to avoid the hospitalization of patients whenever possible. A patient that has to stay in a hospital has a high cost for the community, and saving money (and resources) means that those resources can be spent for providing a better service to other patients. Hence the current trend is to send nurses to visit patients at their home where patients perceive a higher quality of life, with their dear ones, and feel their illness more similar to a "normal" life situation. This reduces depression risk for the patient, and improves rehabilitation rate. Indeed, high quality Home Health Care following hospital dismissal has proved essential in reducing hospital readmissions, if able to cope with preventable complications.

The challenge is to deliver the service in a cost effective manner without a detriment of the service quality. These social and health management issues have interesting implications from the mathematical viewpoint, introducing a challenging combinatorial optimization problem. The problem consists on assigning patients' services to traveling nurses and defining the nurse itineraries so that the following optimization aspects are considered: the nurse workloads (including service as well as travel time) are balanced, patients are preferentially served by a single nurse or just a few ones, and the overall

travel time is minimized. These objectives are somehow conflicting and a reasonable trade off must be found. The complexity of the problem calls for suitable optimization-based algorithmic supports to decisions, in particular in the perspective of an increasing diffusion of the service. This HHC problem is currently solved by hand, starting from a partitioning of patients based on predefined zones. We describe a CP model that solves the HHC problem, and show significant improvements with respect to the current manual solution.

Arguments in favor of Home Health Care and how to reduce health related risks when clinical service is provided at home can be found also in (Giuliani et al., 2005) and (Mileo et al., 2011). When a patient requires service only for a limited time per day, or even per week, then it is a viable and convenient solution to send nurses to provide the service at the patient's home. Managing such a service gives rise to a family of challenging optimization problems referred to as Home Health Care routing and scheduling (see (Bertels & Fahle, 2006) and (Steeg & Schröder, 2008)).

The work presented in this chapter has a previous version presented in (Cattafi et al., 2012), it was in ECL^{i}PS^e (Schimpf & Shen, 2012), which is a logic programming system, so we defined a declarative semantics for the traveltime constraint. The current version is presented in (Cattafi et al., 2015), and it extends in several directions:

- a new implementation in the language Comet, which has more global constraints, that have powerful constraint propagation, and allowed us to improve the efficiency of the application;
- new search algorithms, based on Large Neighborhood Search, that let us get to better solutions in shorter time;
- a wider experimentation;
- the use of a new objective function.

3.1 Problem Definition: the Home Health Care service in Ferrara

At present, the HHC service in the city of Ferrara, Italy, is directly administrated by the local agency of the National Health Service (NHS), namely AUSL 109. All patients who are not self sufficient and in need of medical treatment are eligible for HHC, i.e., any person who is infirm, chronically ill, or disabled. A demand for assistance must be issued by the family doctor to AUSL 109, specifying the list of required medical treatments. If the demand is accepted, the patient is enrolled in the service, and the required treatments are entered into the request data base. Each request is characterized by a patient identifier (name and address), a medical treatment, and the specific day of the week when the treatment must be delivered (each patient can have more than one request per week).

Service is provided by a set of qualified nurses, all registered nurses with additional training, employed by the local NHS agency. Every day, each nurse who is on duty

starts the shift at the city hospital, visits the patients in the list delivering the required treatments and traveling by car from one patient's home to the next; finally the nurse returns to the hospital.

A total of 15 nurses is involved and they are organized as follows. On working days, i.e., from Monday to Friday, 12 nurses are on duty. 9 of them operate in the morning while 3 in the afternoon. During weekends, 4 nurses work on Saturday and 1 on Sunday or during holidays.

Working rules. A duty should last up to 7 hours and 12 minutes on a working day, and about 6 hours on Saturdays and holidays. This limit is not strictly enforced, provided that the threshold of 36 hours per week is not exceeded over a planning horizon of 4 weeks.

Each nurse works 5 days a week (on working days only) each alternate week and works 6 days the other week; the sixth day is usually a Saturday, but it can be a Sunday once every 14 weeks.

Lunch breaks are not present because shifts are either morning or afternoon. All nurses have same starting and ending times, and must start and end shifts at the hospital. Nurse unavailabilities are considered unexpected events, and one additional nurse is always available for such emergency.

Service	Time
First visit	1 hour
Enema	30 min
General check	15 min
Tracheotomy check and tube change	15 min
Central venous catheter	30 min
Electrocardiogram	30 min
Emogasanalysis	15 min
Mouth cleaning	5 min

Table 3.1: Excerpt of the services with average service time

Services consist of treatments and medications, and since their delivery time has almost no impact on the patient daily routine (patients stay in bed almost all day), there are no time windows. A treatment lasts from 5 to 60 minutes, depending on its specific characteristics (see Table 3.1 for an excerpt), but a patient in a day may need several treatments, that are carried out as a whole by a single nurse and handled as a single request. Therefore, requests duration is quite heterogeneous over the whole set of requests, as can be seen on Figure 3.2.

Ferrara has about 150,000 inhabitants, and the area administered by AUSL 109 is rather large and its population ageing. Most of the elderly patients live in the countryside, so there is a significant geographical dispersion of the requests. Therefore, the

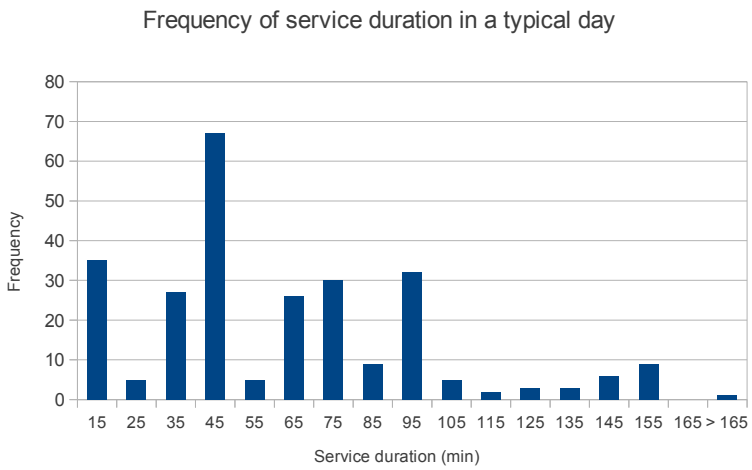


Figure 3.2: Histogram of the frequency of service durations in a typical day

service is characterized by a high variance of duration and a significant geographical dispersion of the requests.

As a sample of the set of requests, we consider the month of February 2010, with 3323 requests subdivided among 458 patients. Several patients are located either in town or in its suburbs, though quite a few live in the neighboring towns of Masi Torello and Voghiera, that are approximately $20km$ away.

At present, the digitization of data is limited to the patient data base, containing the address and the prescribed services for each patient. The duty definition and the management of nurse agendas is handled manually and supervised by the chief nurse, with frequent adjustments done by cell phone. As a record of their duty, nurses fill forms on paper which they return to the chief nurse at the end of their daily shift.

The problem data

Optimizing the HHC scheduling is essential to make the service cost-effective and to avoid the so called burn out phenomenon, i.e., nurses who get tired and act in an unfriendly manner to patients or even leave the job. At the same time, also the patient point of view should be accounted for. Specifically, a patient would like to receive cares always by the same nurse or by a few ones, if possible, since receiving clinical care involves physical and emotional contacts that one is not like to easily share with anyone. This introduces the concept of *loyalty* (i.e., the number of different nurses per patient) as a measure of the service quality as it is perceived from the patient viewpoint.

We start by examining the current solution approach, as it is manually carried out by the people in AUSL 109, and analysing its critical aspects. Nurses organize their duties themselves, solving the problem by hand according to the following procedure.

To simplify the assignment of patients to nurses, the territory pertaining AUSL 109 is statically partitioned into 9 zones (see Figure 3.3), considering factors like the

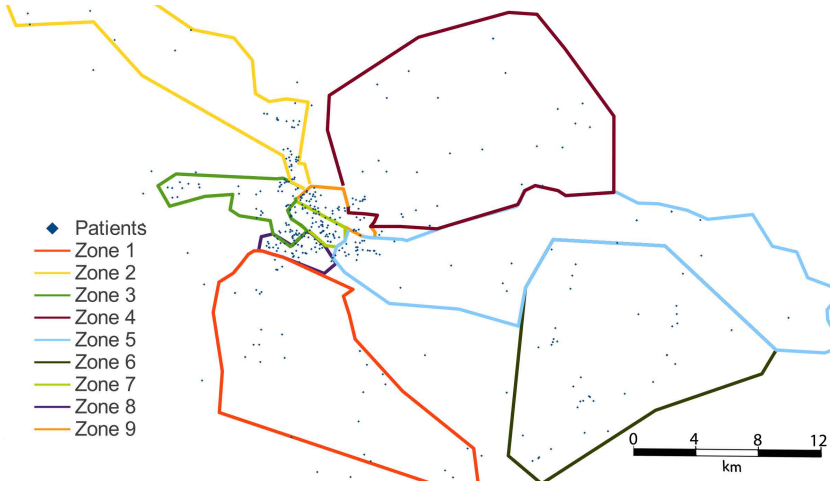


Figure 3.3: The 9 zones in which the city is divided, with shown the location of the patients

distribution of the population, its age and historical data. As we can see from Figure 3.3, some zones are in the city center, they are very small, but they have a large number of patients that require service. Other zones are in peripheral areas, and they are less densely populated. A nurse assigned to a central zone will probably have many patients to serve, while a nurse assigned to a peripheral zone will likely spend a significant amount of time traveling.

Each nurse receives in charge the services of all the patients belonging to one area. Then the nurse tries to fit the requests for the whole week into the working shifts, while complying with the maximum weekly workload allowed, since no overduty is permitted. The assignment of requests to daily shifts follows a sort of greedy criterion: nurses, by hand, insert as many requests as possible into the shifts. There are no given rules on which patients to select first: each nurse could use a different order, in principle. Such decisions are not driven by any optimization criteria, and the routing is not necessarily optimal.

If a nurse can not fulfill all the requests, some requests are forwarded to the chief nurse, who is in charge of the service coordination, and reassigns them to other colleagues. In such a case, a patient may be treated by different nurses at different days of the week, which goes to the detriment of the service quality. The three nurses working in the afternoon are in charge of the remaining requests, that are still assigned to the nurses on the basis of a similar geographical partition of the territory, each nurse dealing with the patients of three zones. Therefore, patients are supposed to be visited along the morning, but maybe they are visited in the afternoon. Currently, patients do not know the schedule.

Due to this greedy procedure, the nurse weekly schedules have very different workloads, and balancing this over the months leads to a detriment in loyalty. Nurses complain about such disparities, and have difficulties adapting their schedule to new patients,

new treatments, or to any other change. Moreover, if the workload balance could be improved by optimizing the routing component, nurses could be available at the hospital for other tasks. In addition, an improved routing plan would impact on the direct expenses related to fuel and car usage, which contribute to the overall cost.

The decisions to be taken are:

- partitioning the requests to a set of duties,
- for each duty determining the sequence of services to be carried out optimizing the traveling times by solving a Traveling Salesman Problem (TSP), and
- assigning each duty to a nurse to build the nurse weekly schedule.

Scheduling such a service poses several challenges, related to the criteria which characterize high quality solutions. A good solution should achieve at the same time:

- from the point of view of the NHS, the minimization of the travel time over the service time, since during travel a nurse is on duty but is not delivering any service;
- from the point of view of the nurses, a fair distribution of the workload, which can not be guaranteed by simply equally subdividing patients, due to heterogeneous requests;
- from the point of view of the patients, a high degree of *loyalty*, i.e., the number of different nurses who are in charge of a single patient should be minimal.

The problem we are facing contains several subproblems, namely assigning each request to one available nurse, scheduling optimally the services in each nurse's duty, and finding the optimal route for each nurse in order to avoid long useless trips. We will describe in the following the main components of the CP-based hybrid solution approach that has been developed to tackle such a difficult problem.

3.2 Literature Review

The efficient delivery of Home Health Care service is increasingly attracting the attention of the Artificial Intelligence and the Operations Research communities. Most of the contributions focus on the application aspects and often, beside the optimization approach, they describe the complete information system managing the service. Therefore the specific application aspects strongly characterize the optimization problems presented in each paper and the proposed solution methods. This is the reason why, in practice, each paper addresses a different problem and a thorough comparison can be hardly done.

The great majority of the contributions consider a service where patients/users specify a (more or less narrow) time window when they are available to receive the service. This feature is typical of the systems where the service is managed in conjunction with a Home Care. In Home Care services the presence of time windows is crucial since users are supposed to be independent and to move from their home. This is not the case of

Home Health Care where we are in the presence of patients that, instead of being at the hospital, are in their bed at home and are not supposed to move.

One of the first papers dealing with this class of problems is (E. Cheng & Rich, 1998). The problem accounts for time windows and a mix of full time and part time workers. In the scheduling of full time workers a lunch break must be included and the objective is to minimize the cost of over time of full time workers and the time of part time workers. The authors propose a mathematical programming model and a simple heuristic.

Eveborn et al. (2009) is a system adopted in Sweden for Home Care, although it is also able to consider some of the issues in HHC. Laps Care is based on a set partitioning formulation of the problem and uses an iterative method. The starting solution proposes a single route for each service; then routes are joined until no further improvement is possible. To escape from the local optimum, one of the routes is split into one route for each patient, and the joining phase restarts. Another integrated system is presented in (Begur et al., 1997) where the focus is mainly on the GIS component of the Decision Support System. The problem is reduced to a vehicle routing and solved with classical heuristic methods.

A Home Health Care system similar to ours is analyzed in (Borsani et al., 2006). One of the objectives is to balance the workload while some soft preference constraints for the nurse-patient assignment are accounted for as well as for the days of the week of the visits. Another objective is to minimize the number of uncovered visits that must be outsourced. The authors propose a simple assignment model ignoring the routing component, indeed the time of the visits is deterministically fixed to a constant time and this time includes the travel time. This assumption may be oversimplifying if we are in the presence of a wide territory to cover, in which the service and the travel times may vary from few minutes to one hour.

Rasmussen et al. (2012) considers a specific core subproblem that is the daily scheduling of nurses. In this problem the objective function accounts for many factors related mainly with the quality of service, such as the uncovered visits or a priority function, rather than the operational cost which is not easily applicable on a single day. The problem is an extension of the classical Vehicle Routing Problem with Time Windows, widely studied in the literature. The authors propose a set partitioning formulation and an exact solution method based on Branch and Price that exploits a special branching rule taking advantage of the specific problem structure.

In (Bertels & Fahle, 2006), (Elbenani et al., 2008) and (Steeg & Schröder, 2008) authors address HHC problems with rather tight time windows. Bertels and Fahle (2006) adopt an interesting hybrid approach, combining CP, local search and Linear Programming. The approach takes advantage of the presence of narrow time windows: due to time window constraints, in the HHC only few permutations correspond to feasible orderings. The instance in Ferrara does not have such constraints, while retaining the loyalty and the workload balancing aspects, so the only criterion for assigning the order of patients in a nurse's schedule is minimizing the route. However, this does not mean that the problem is under constrained or easier to solve. On the contrary, the search space is wider, as it is not pruned by simple constraints like time windows or nurse skills.

The number of feasible solutions, on the other hand, does not necessarily increase: in a real-world instance new nurses are hired only when necessary, so the actual workload of the nurses is very close to their maximum working hours. This means that if the routes are not close to the optimum, the daily-workload constraint becomes infeasible: one has to find the optimum of the TSP just to find a feasible solution to the HHC problem or to assess infeasibility.

Another work that uses a hybrid CP-Mathematical programming approach is that in (Nickel et al., 2012). Here the authors consider the whole problem of assigning patients to nurses on a weekly base and make the daily schedule. The objective is twofold: on the one hand they minimize the overtime and the route length, on the other hand they account for a loyalty factor privileging the assignment of the same nurse to the same patient if possible. Also in this case time windows are present. The decomposition of the problem in a master weekly scheduling and a daily scheduling is also done in order to be able to adjust the service to last minute changes, which is very common in Home Health Care. A completely different approach to the last minute variability arising in request issues is that in Koeleman et al. (2012) where a Markov process captures the stochastic aspects of the problem. Another stochastic model used in medium term planning is proposed in Lanzarone et al. (2010).

The above stated approaches consider the use of one transport mode during the tour of a nurse. Different modes of transport are discussed in (Rendl et al., 2012; Rest & Hirsch, 2012). Rendl et al. (2012) present a hybrid solution approach combining constraint programming and a metaheuristic to solve instances with up to 509 nurses and 717 visits. Additionally, Rest and Hirsch (2012) consider time-dependent traveling times for public transport. They propose Tabu Search-based solution approaches and solve instances with up to 89 nurses, 236 clients, and 388 visits.

Looking at the problem from a more abstract viewpoint, one may see some similarities with the classical Capacitated Vehicle Routing Problem (CVRP). In the CVRP a set of disjoint routes for a fleet of vehicles has to be found so that all customers (nodes) are visited, the required quantity of goods is delivered to each customer, the capacity of the vehicles is not exceeded and the objective function is minimized. The usual objective function is the overall traveled distance or the number of vehicles. The CVRP has been tackled by many solution methods (Baldacci et al., 2010; Pisinger & Røpke, 2007; Garrido et al., 2009). including branch and cut (Baldacci et al., 2010), metaheuristics (Pisinger & Røpke, 2007), and hyperheuristics (Garrido et al., 2009).

In our case we can see nurses as vehicles and patients as customers. There are some important differences with CVRP that make all the efficient method developed for the classical problem not applicable in our case. One difference concerns the capacity. As in CVRP we may consider nurse daily duty time as a capacity constraint, however, unlike the CVRP, in our case the sequence in which patients are visited matters on how the capacity is consumed. This actually turns our problem into a time constrained VRP which is not as easy as the CVRP and for which the classical CVRP methods are not so efficiently adapted. The other difference concerns the objective function. On the one hand, as in VRP, we would have to minimize the total traveled distance, in order

to make the service as efficient as possible, on the other hand we have to balance the workload among nurses. Thus this component of the objective function is a kind of bottleneck (min-max), that is difficult to address with OR methods. Finally, the loyalty is the component of the objective function that makes our problem very peculiar. The two criteria are often in contrast: any solution assigning to a single nurse all the services of the same patient would minimize the loyalty, but this would go to the detriment of workload balancing.

3.3 The Methodology in detail

A hybrid methodology based on Large Neighborhood Search is presented in this section, it combines Constraint Programming (CP) and the Tailored Lagrangian Metaheuristic (TLM) proposed in Section 1.6. This section is structured as follows: the input data is described, the CP model is presented, then the objective function is discussed, it is explained how the routing is addressed and finally the search strategy is presented.

LNS is becoming more and more popular to solve routing problems, an introduction can be found on page 17 and in (Shaw, 1998; Pesant et al., 1997). The idea is a local search that adopts a large neighborhood which makes less likely to fall in a local minimum. Exploring the neighborhood becomes a search problem itself, and it is done with CP. In this way, it is possible to take into account easily many side constraints.

The popularity of hybridization of CP and Local Search (Monfroy et al., 2004; Wallace, 2006) is witnessed by the existence of solvers tailored for this approach (Van Hentenryck & Michel, 2009), and the application to routing problems (Rousseau et al., 2002; Kilby et al., 2000). Various works deal with solving the TSP or its variant with CP or hybrid algorithms (Caseau & Laburthe, 1997; Pesant et al., 1998; Focacci et al., 2002). Rousseau et al. (2002) propose a large neighborhood search in which CP explores a neighborhood with three operators. These operators are combined in variable neighborhood descent and a two phase process. Kilby et al. (2000) consider vehicle routing problems with side constraints, and compare classical OR approaches with CP models. OR approaches tend to use cost-oriented heuristics, that make them suitable for problems with few side constraints, but fail to find an initial solution when the number of side constraints grows. On the other hand, CP approaches are better when the side constraints are more tight but usually provide worse solutions when the side constraints are not prevalent.

We believe that using CP to tackle this problem is a wise choice, because of the importance of the feasibility issues represented by the time constraints, of the many aspects which can benefit of CP's flexibility (such as the need to easily add and relax constraints to follow change in the requirements) and because CP provides an effective framework for the integration of components based on different approaches. In this section we will explain the CP model and we will deal with some of these issues.

Input Description

The input data consists of:

- a set $\mathcal{S}_{serv}$ of services, of size N_s; for each service s we know the patient pat_s, the day day_s and the duration dur_s;
- a matrix of distances D where the element $d_{i,j}$ is the travel time from patient i to patient j;
- the number of nurses N_n;
- the set of nurses $\mathcal{S}_{nurse}$ (i.e., $N_n = |\mathcal{S}_{nurse}|$ or $\mathcal{S}_{nurse} = \{1, \ldots, N_n\}$);
- the number of days N_d considered in the scheduling;
- the maximum time that a nurse can spend working each day; *MaxDayWL* is the amount of minutes available per day for each nurse (including service time and travel time); it currently corresponds to 7 hours and 12 minutes (432 minutes);
- the maximum time that a nurse can spend working in a week; *MaxWeekWL* is the number of minutes available per week for each nurse. It currently corresponds to 36 hours.

The CP Model

A constraint model is defined by a set of decision variables (unknowns), each ranging on a given domain, and a set of constraints, that each assignment must satisfy. Often, there is also an objective function, that should be minimized or maximized. A solution is an assignment of values taken from the domains to the corresponding variables such that all constraints are satisfied, and that minimizes (resp. maximizes) the objective function.

Usually an efficient model makes use of *global constraints*, i.e., constraints that involve many (possibly, all) variables in the model. Often, global constraints propagate more efficiently than small constraints that involve only two or three variables. We show a model that includes a number of global constraints.

Variables:

The main decision is assigning nurses to services. To each service s we associate a decision variable $Nurse_s$ with domain $\mathcal{S}_{nurse}$.

Other variables have functional dependency to the $Nurse_s$ variables. For each nurse n and day d, we have a decision variable $DayWL_{n,d}$ with domain $\{0..MaxDayWL\}$. This is the number of minutes worked by nurse n on day d.

Other variables represent service time and traveling time: $T^{svc}_{n,d}$ and $T^{trv}_{n,d}$ are, respectively, the service time and the travel time of nurse n in day d; they also range from 0 to the maximum day workload *MaxDayWL*.

We also have variables containing the weekly totals: each nurse n has a weekly travel time WT_n^{trv}, a weekly service time WT_n^{svc}, and a weekly workload WL_n, that is the sum of the travel and service time; their domain is $\{0..MaxWeekWL\}$.

Another decision to be taken is the order in which a nurse visits the assigned patients. This problem will be dealt with in "Addressing the Routing" on page 75.

Constraints:

The workload of a nurse n in a day d is the sum of the total service time and the travel time

$$DayWL_{n,d} = T_{n,d}^{svc} + T_{n,d}^{trv}, \tag{3.1}$$

while the (whole week) workload WL_n is simply the sum of the day workloads:

$$WL_n = \sum_{d=1}^{N_d} DayWL_{n,d} \quad (\forall n \in \mathcal{S}_{nurse}).$$

The service time is the sum of the duration of the services provided by nurse n in day d. In order to compute the service time of each nurse in each day (and to enforce at the same time that the total service time does not exceed the maximum day workload $MaxDayWL$), there can be different constraint models. An intuitive model could be to define a set of Boolean decision variables $X_{n,s}$ (variables taking values *true* or *false*) saying if nurse n takes service s, and add corresponding sum constraints. A more promising model can be obtained noticing similarities with classic problems such as bin packing and (multi)-knapsack problems (Korf, 2003; Fukunaga & Korf, 2007).

In a multi-knapsack problem, a set of N_{items} items with size $size_i$ have to be placed in a set N_k of containers without exceeding their capacities.

The global constraint *multiknapsack* (also called *binpacking*) (Shaw, 2004) is suited to solve such problems. In CP a multiknapsack problem can be simply defined with the constraint

$$multiknapsack(a, size, load)$$

where

- $a \equiv \{a_i\}$ is a set of decision variables a_i with domain

  $$a_i :: 1..N_k \qquad (\forall i \in 1..N_{items}),$$

  with the intended meaning that $a_i = j$ if only if item i is assigned to knapsack j;
- $size \equiv \{size_i\}$ is the array containing the size of each item
- $load \equiv \{load_j\}$ is a set of constrained variables that represent the actual load of each knapsack, i.e., the sum of the sizes of the items assigned to that knapsack. In order to solve a multi-knapsack problem, the $load_j$ domains are constrained to take values up to the capacity of the knapsacks:

  $$load_j \leq capacity_j \qquad (\forall j \in 1..N_{knapsacks}).$$

By using the techniques currently embedded in the *multiknapsack* constraint, Shaw (2004) obtains a speedup of several orders of magnitude with respect to the intuitive model, so we decided to take this second approach.

In the HHC problem, we can consider

- each service s as an item of size dur_s which has to be placed in one of the bins/knapsacks;
- to each pair $(nurse, day)$ we associate a knapsack of capacity $MaxDayWL$;
- the actual load of knapsack $(nurse, day)$ is the service time $T^{svc}_{nurse,day}$ of nurse $nurse$ in day day.

We can thus bind the *Nurse* decision variables with the whole T^{svc} vector. We impose the constraint

$$multiknapsack\,(Nurse, dur, T^{svc})\,,$$

which enforces that $(\forall n \in \mathcal{S}_{nurse}, \forall d \in \{1..N_d\})$:

$$T^{svc}_{n,d} = \sum_{\substack{s \in \mathcal{S}_{serv} \\ d = day_s}} [Nurse_s = n] \times dur_s$$

where the square brackets are the Iverson brackets (Knuth, 1992)[1].

The travel time $T^{trv}_{n,d}$ of nurse n in day d is computed by a specific constraint *traveltime*, explained in detail in "Addressing the Routing" on page 75.

The Objective Function

Ideally, in the HHC problem in Ferrara, one should find a solution that achieves the objectives described in the introduction. As it is often the case in real world applications, the objectives are stated by the users in a blurred, informal way, not clear enough to convert them into a unique objective function, and all functions one can think of are debatable. The main objectives, as stated by the user, are to (i) avoid that a patient is visited by too many different nurses (loyalty) and (ii) balance the weekly workloads of the nurses (but they also have to keep reasonable conditions for all nurses and for the NHS).

We first address the two issues separately, then we combine them.

Loyalty One way to obtain maximum loyalty is to minimize the number of nurses that visit each patient. For each patient $p \in \mathcal{S}_{patient}$ we link the array of decision variables $Nurse_p$ with the number of different values in it.[2] The *alldifferent* (Régin, 1994; van Hoeve, 2001) global constraint enforces that all the variables in an array are given different

[1] let P be a statement, $[P]$ evaluates to 1 if P is *true* and to 0 otherwise.

[2] To simplify the notation, we indicate with $Nurse_p$ the subset of the *Nurse* array corresponding to patient p.

values. The global constraint *NValue* (Bessière et al., 2006; Hebrard et al., 2011; van Hoeve, 2004) can be thought of as a soft version of *alldifferent*, and it counts the number of different values in a vector of variables. Various efficient propagation algorithms have been proposed for the *NValue* constraint; using the global constraint *NValue* is usually considered more efficient than using a set of smaller constraints (Bessière et al., 2010). By counting the different values in the set of decision variables associated to a certain patient p, we obtain the number PN_p of different nurses who take care of that patient:

$$NValue(Nurse_p, PN_p) \qquad (\forall p \in \mathcal{S}_{patient})$$

and we can compute the overall loyalty penalty LP summing up the single contributions:

$$LP = \sum_{p \in \mathcal{S}_{patient}} PN_p. \tag{3.2}$$

Balancing workloads There are various ways to achieve balanced workloads, see (Simonis, 2007). A widely used method is the minimization of the maximum workload, so a first proposal is to minimize:

$$BalanceWL' = MaxWL = \max_{n \in \mathcal{S}_{nurse}} (WL_n). \tag{3.3}$$

There are known pitfalls in adopting the minimization of the maximum (see, e.g., (Monette et al., 2007)). One simple observation is that two solutions are considered equivalent if they have the same value for the highest workload, regardless of the working conditions of all other nurses.

Consider for instance the following example, also depicted in Figure 3.4:

Example 1 *Suppose that there is a patient X that lives rather far from the hospital, and also requires a long service time (Figure 3.4); suppose that, in order to visit and service him, a nurse takes a time exactly equal to the maximum day workload MaxDayWL. The other patients, instead, live close to the hospital and have shorter service times. In such a case, one nurse will serve only patient X, while the other patients will be subdivided among the other nurses.*

In this example, the minimization of the maximum workload provides the same value for all the possible assignments: the maximum workload is fixed to the maximum day workload, and corresponds to the nurse visiting patient X.

More recent proposals to obtain balanced solutions are the *spread* (Pesant & Régin, 2005) and *deviation* (Schaus et al., 2007) constraints. These constraints link a set of decision variables with their variance or with the mean absolute deviation. In this way, it is both simple to define objective functions that minimize the variance (resp. mean absolute deviation), and efficient, thanks to the powerful global constraint propagators. In our application, we decided to experiment with the *deviation* constraint, thus minimizing the L_1 norm although, as noticed in (Schaus et al., 2009), *"These criteria are*

Figure 3.4: *Example situation: the hospital is depicted with a black square, while patients are depicted as circles. The area of each circle represents the service time required by the patient.*

not equivalent: Minimizing L_1 or L_2 does not lead to the same solutions and it is not always obvious which one to choose. In fact, this is an old and recurrent debate".

These constraints are very useful when the total workload is fixed, and the objective is only to distribute it fairly to a set of resources. Indeed, in some CP languages the sum of the variables occurring in the *spread* or *deviation* constraint is *required* to be a constant. In our case, however, the workload of the nurses depends also on the travel time, that depends on the partitioning of the patients to the nurses. This can lead to situations in which minimizing the absolute deviation of the workload provides low quality solutions, as shown in the following example.

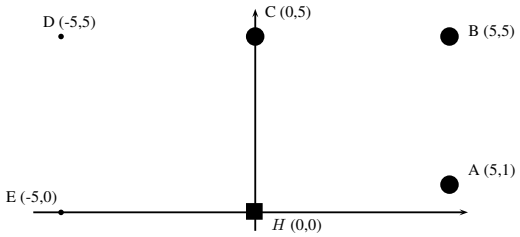


Figure 3.5: *Example situation: the hospital is depicted with a black square, while patients are depicted as circles. The area of each circle represents the service time required by the patient.*

Example 2 *We have 5 patients, named A..E, on the Cartesian plane, as depicted in Figure 3.5. The time (in minutes) required to move from a patient to another is simply given by the Euclidean distance. Patients A, B, and C have a service time of 10 minutes, while the other two have a service time of 1 minute. The hospital H is in $(0, 0)$, and there are two available nurses.*

If we minimize the maximum workload, we obtain the assignment

Nurse	*Patients*	*Travel*	*+Service=Workload*
1	*A, B*	$\sqrt{26}+4+5\sqrt{2}+$	$20 \simeq 36.17$
2	*C, D, E*	$20 +$	$12 = 32$

In this case, the absolute deviation is 4.17.

However, consider this assignment:

Nurse	*Patients*	*Travel*		*Service*	= *Workload*
1	*B, C*	$5\sqrt{2}+10$	+	20	$\simeq 37.07$
2	*A, D, E*	$\sqrt{26}+\sqrt{116}+10$	+	12	$\simeq 37.87$

The absolute deviation is only 0.8, so if we minimize the workload deviation, we would definitely prefer this second assignment. However, the second assignment has worse working conditions for both *nurses: both have a higher workload. This solution is also worse from the National Health Service viewpoint, as it is more expensive, since nurses are paid for the whole workload, and they cannot be used for more useful tasks at the hospital.*

Finally, the nurses also employ more time driving, producing pollution and traffic.

But there can be even worse situations:

Example 3 (Example 1 continued) *In the example in Figure 3.4, the nurse servicing patient X has the highest possible workload, and this cannot be reduced. So minimizing the absolute deviation amounts to maximize the workload for the other nurses! Since the total service time is fixed, the only way to increase the workload for the nurses is to increase their travel time, by providing them* the worst possible assignment *(the assignment that maximizes the total travel time).*

This is even worse than minimizing the maximum: in this example, the minimization of the maximum considers all solutions as the same; but the deviation of the workload considers as best solutions those in which each nurse is forced to travel uselessly in the whole city area, spending as much time as possible driving from a patient to another. In order to avoid this unreasonable behaviour, one possibility is to remove the travel time from balancing.

Our implementation is based on the Comet constraint programming system (Van Hentenryck & Michel, 2009). In Comet, the sum of the variables occurring in *deviation* is required to be a constant. For this reason, our objective function could not contain a term $deviation(WL)$, because the sum of the week workloads of the nurses is not fixed (it depends on the nurses' travel time, which, on its turn, depends on the actual partitioning of the patients to the nurses). We decided to remove the travel time from balancing; the absolute deviation is computed only on the service time, and we add a term containing the total travel time:

$$Balance WL'' = \underset{n}{deviation}(WT_n^{svc}) + \sum_{n,d} T_{n,d}^{trv} \tag{3.4}$$

where WT_n^{svc} is the week service time of nurse n, computed simply as

$$WT_n^{svc} = \sum_{d=1}^{N_d} T_{n,d}^{svc},$$

and

$$\underset{n}{deviation}(WT_n^{svc}) = \sum_{n=1}^{N_n} \left| WT_n^{svc} - \sum_{k=1}^{N_n} \frac{WT_k^{svc}}{N_n} \right| .$$

This solves the problem highlighted in Examples 2 and 3: since the total travel time appears outside the *deviation*, the solver will try to minimize it. On the other hand, minimizing Eq. (3.4) does not ensure that a minimum of the deviation of the workload is found, because, in general

$$\underset{n}{deviation}(WT_n^{svc}) + \sum_{n,d} T_{n,d}^{trv} \neq \underset{n}{deviation} \left(WT_n^{svc} + \sum_{n,d} T_{n,d}^{trv} \right) .$$

Combination of the objective functions. As we explained, we have two criteria to be minimized: one tries to provide solutions with a good *loyalty*, while the other tries to balance the workload of the nurses, without increasing uselessly the travel time of the nurses.

There are many ways in the literature to combine two criteria; we adopted a widely used method, namely a weighted sum of the two components:

$$minimize \left(\alpha_1 \cdot BalanceWL + \alpha_2 \cdot LP \right) . \tag{3.5}$$

α_1 and α_2 are positive real numbers that can be chosen by the user in order to reflect the current priorities adopted in the AUSL. Of course, such values can be tuned later on in order to better suit the decision maker preferences. LP is the Loyalty Penalty (see Eq. 3.2), while the term $BalanceWL$ balances the workload of the nurses and will be instantiated either to $BalanceWL'$ or to $BalanceWL''$, as will be clear soon.

We proposed two ways to balance the workload; so we will have two objective functions to experiment with.

The first is the minimization of the maximum workload (Equation 3.3), so the objective function in Eq. 3.5 becomes:

$$\begin{aligned} OF_1 =& \alpha_1 \cdot \max_{n \in \mathcal{S}_{nurse}} (WL_n) + \alpha_2 \cdot LP \\ =& \alpha_1 \cdot \max_{n \in \mathcal{S}_{nurse}} \left(\sum_{d=1}^{N_d} T_{n,d}^{svc} + T_{n,d}^{trv} \right) \\ &+ \alpha_2 \cdot LP. \end{aligned} \tag{3.6}$$

The second method is in Eq. (3.4), so the objective function OF_2 becomes:

$$\begin{aligned} OF_2 =& \alpha_1 \left(\underset{n \in \mathcal{S}_{nurse}}{deviation}(WT_n^{svc}) + \sum_{n,d} T_{n,d}^{trv} \right) \\ &+ \alpha_2 LP \\ =& \alpha_1 \left(\underset{n \in \mathcal{S}_{nurse}}{deviation}(\sum_{d=1}^{N_d} T_{n,d}^{svc}) + \sum_{n,d} T_{n,d}^{trv} \right) \\ &+ \alpha_2 LP. \end{aligned} \tag{3.7}$$

In Section 3.4, we will compare experimentally which of the two objective functions (OF_1 and OF_2) performs best in our problem.

Addressing the Routing

In order to satisfy the constraint on the maximum day workload, one should compute both the service time and the travel time of a nurse's daily shift (Eq. 3.1). On the other hand, computing the travel time of a nurse implies computing the optimal route that connects the hospital and all the patients assigned to that nurse on that day, which is a NP-hard problem in its own, known as the TSP, see Chapter 1.

We embedded a TSP solver inside a (user-defined) constraint *traveltime*, so that it can perform constraint propagation during search of the main problem (i.e., the HHC). Notice that in a given node of the search tree, some of the patients will be assigned to a nurse n, while others will still be unassigned; the constraint *traveltime* should compute the bounds of the travel time, namely the minimal travel time of the nurse (assuming that nurse n will be assigned a minimal number of patients, amongst those in the domain), and the maximum one (assuming that nurse n will be assigned all the possible patients in the domain). In order to compute the travel time $T_{n,d}^{trv}$ of a nurse n in day d, we need to provide to such constraint:

1. the patients associated to the services pat_s,
2. the matrix of distances D, providing the travel time between patient locations as well as to and from the hospital,
3. and the subset $Nurse|_{day=d}$ of the decision variables (assigning nurses to services) that correspond to the services to be provided in day d, i.e.,

$$Nurse|_{day=d} \triangleq \{Nurse_s | s \in \mathcal{S}_{serv}, day_s = d\}.$$

Note that, since to each patient is assigned a unique location, once the patient of a service is given also the location where the service must be delivered is known. As a recap, the actual parameters are:

$$T_{n,d}^{trv} = traveltime(pat_s, D, Nurse|_{day=d}).$$

Operationally, the *traveltime* constraint awakes every time one of the variables in $Nurse|_{day=d}$ is instantiated, i.e., when a nurse is assigned to a service. The constraint computes the TSP corresponding to the patients that must be visited by nurse n in day d. This provides a valid lower bound to the actual travel time, and can be used in a branch-and-bound search. When all the $Nurse|_{day=d}$ variables are ground, the cost of the TSP becomes the real travel time, and we are able to fix the value of $T^{trv}_{n,d}$ to the TSP cost. In our implementation, the constraint implements directed bound consistency from the variables in $Nurse|_{day=d}$ to the $T^{trv}_{n,d}$ variable, considering only the lower bound (as the upper bound would not perform significant pruning in our case). Even if recomputing a TSP on each instantiation may sound computationally expensive, preliminary computational results showed that this choice provides better results than heuristically inserting the new node, by way of a cheapest insertion procedure, in the tour previously computed for the other nodes. This is probably due to the fact that the TSP instances to be solved at each step are rather small, so that the TSP solver we embedded in the *traveltime* constraint is quite fast. Moreover, we use a caching mechanism to avoid recomputing TSPs that have been solved previously during the exploration of the search tree.

Regarding TSP solvers, one may either implement one, e.g., in CP (Beldiceanu & Contejean, 1994; Kaya & Hooker, 2006), or use an off-the-shelf solution. We found that implementing a solver in pure CP slowed down significantly the search (as also witnessed in the literature (Caseau & Laburthe, 1997)). The state of the art in the area is by no doubt Concorde (Applegate et al., 2015), which relies on integer linear programming with the addition of specific TSP-targeted features. However Concorde requires an ILP solver, and is oriented to large instances, while the number of patients visited by a nurse in a single day in our case is rather limited, usually far below 15. Furthermore, Concorde is not easily customizable but it can only be used as a black box, so that it would be impossible to extend our solution method in a further development in order to handle additional constraints on the services, such as time windows or precedence constraints.

Therefore, we adopted the proposed Tailored Lagrangian Metaheuristic, see Section 1.6 for a full description. This metaheuristic performs very well on small TSP instances, would allow us to add further customizations in the future if needed, and proved to be suitable for integration in our system. In the following we provide a brief description of the method.

Search Strategies

In CP, domain filtering, achieved by the imposed constraints, has to be coupled with search tree exploration in order to reach solutions. The search strategy design is responsible for the shape of the tree and the way to explore it, the aim being incorporating knowledge of the problem and its structure to rapidly drive the search towards (good) solutions.

Since the decision variables are the nurses associated to the services, we have to select a variable (a service) and assign a nurse to it. Since all nurses can provide equivalent services, the constraint model has a large number of symmetries: given a solution, a

permutation of the nurses provides another equivalent solution. We decided to break symmetries during search, with a variation of the technique proposed in (Meseguer & Torras, 2001). In particular, the first service is always assigned the first nurse; the n-th service can be either associated to one of the already employed nurses, or to the first not-yet-employed nurse. In fact, trying all the possible combinations of the not-yet-employed nurses is useless, as they can provide only symmetric solutions. Formally, if $E_{n-1} = \{Nurse_k | k \in 1..n-1\}$ is the set of nurses employed for the first $n-1$ patients, then

$$Nurse_n \in E_{n-1} \cup \{1 + \max E_{n-1}\}.$$

Given this definition of the search space, we applied several search strategies to our problem; we first defined a general purpose one (i.e., a predefined search strategy, that can be applied to any problem), then we developed one more tailored for the problem at hand.

A first, general purpose search strategy, called in the following **GSRFF** (Generic Search with Restarts and First Fail), performs a depth first search selecting the variable to assign, at each node, with the First Fail heuristic (Haralick & Elliott, 1980), i.e., it selects first the variable with the smallest domain. The value to assign is selected randomly among the ones in the domain. A widely used technique to improve the efficiency of a search strategy is using *restarts* (Kautz et al., 2002), in order to avoid that wrong decisions taken near the root can drive the search into a bad area. This technique was shown to be very effective in presence of a so-called heavy-tail behavior (Gomes et al., 1997; Hulubei & O'Sullivan, 2006). Using restarts means that after some pre-defined timeout, the search is stopped, and started again, possibly with a different timeout. In the GSRFF search strategy, we apply restarts with the optimal timeout sequence proposed by (Luby et al., 1993), as also suggested in (Sinz & Iser, 2009).

Algorithm 7 Large Neighborhood Search for the Home Health Care

An initial feasible solution x is found using the HEUR heristic based on CP.
while the timeout (10 minutes) is not reached **do**
  A part of the current solution x is relaxed:
  - undoing the assignment on a set of variables, by restoring the domain of 10% of the main decision variables, namely the $Nurse_s$ variables that represents the assignment of nurses to services.
  - fixing the remaining variables to their current value.

  The restricted problem is re-optimized using CP with a limit on the number of failures. Again, we used the HEUR heuristics obtaining a neighbor solution x', with 50 failures as a limit.
  if the solution x' is better than x **then**
    set $x \leftarrow x'$
  end if
end while

We also devised a search heuristic more tailored to the problem structure (**HEUR**).

The variable selection criterion selects first variables related to services with bigger durations, which is a sensible choice when dealing with *multiknapsack* problems. Let the selected variable be associated to service s of patient pat_s, the value selection heuristic tries to assign a nurse who is already visiting pat_s in the attempt to keep low the loyalty penalty LP (see Eq. 3.2). If more than one nurse was already assigned to that patient previously, the one with the lowest week workload WL is selected, in the attempt to keep workloads balanced.

This heuristic can also be adopted in the context of a LNS (**H+LNS**), a technique that hybridizes CP and Local Search (Shaw, 1998). The idea is to perform local search with a large neighborhood, and to explore the neighborhood with CP. Our LNS procedure is presented in Algorithm 7. The Comet programming language allows the programmer to adapt a CP model into a hybridized procedure that scales very well with the problem size, and which explores a neighborhood large enough to avoid the need for a metaheuristic.

3.4 Experiments and Results

The implementation is based on the Comet (Van Hentenryck & Michel, 2009) Constraint Programming system. All tests were performed in a computer equipped with a 3.4GHz Intel Core i7 processor and 7GB of RAM.

Search strategy comparison

The instances were provided by the AUSL 109 and represent four weeks of February 2010. In the first set of experiments, the aim is to compare the constraint model to the Hand Made Solution (HMS) adopted by the nurses in 2010. In these experiments, we used OF_1 as objective function (Eq. 3.6), considering $\alpha_1 = \alpha_2 = 1$. We show the results of the various search strategies defined in Section 3.3; all algorithms were run with a 10 minutes timeout. Each of the algorithms was executed 40 times; the results are shown in a set of box-plots.

In Figure 3.6, we show the Maximum Week Workload of the nurses obtained by the best solution found by the various algorithms within 10 minutes. The first observation is that all algorithms were able to improve significantly on the Hand-Made Solution in each instance and in each run. The generic search algorithm (GSRFF) is very competitive, but the best algorithm is always the Large Neighbourhood Search combined with our heuristic (H+LNS+OF1). Algorithm HEUR is deterministic, so each run provides exactly the same result.

Figure 3.7 shows the Loyalty Penalty (Eq. 3.2) obtained by the same algorithms in the four real instances. Considering the loyalty, the GSRFF search algorithm was not able to improve on the Hand-Made Solution. This is probably due to the fact that the Hand-Made Solution is based on the a-priori partitioning of the patients according to zones, as explained in Section 3.1, so a patient gets always the same nurse, unless the workload of that nurse exceeds the working hours (in such a case, some patient gets

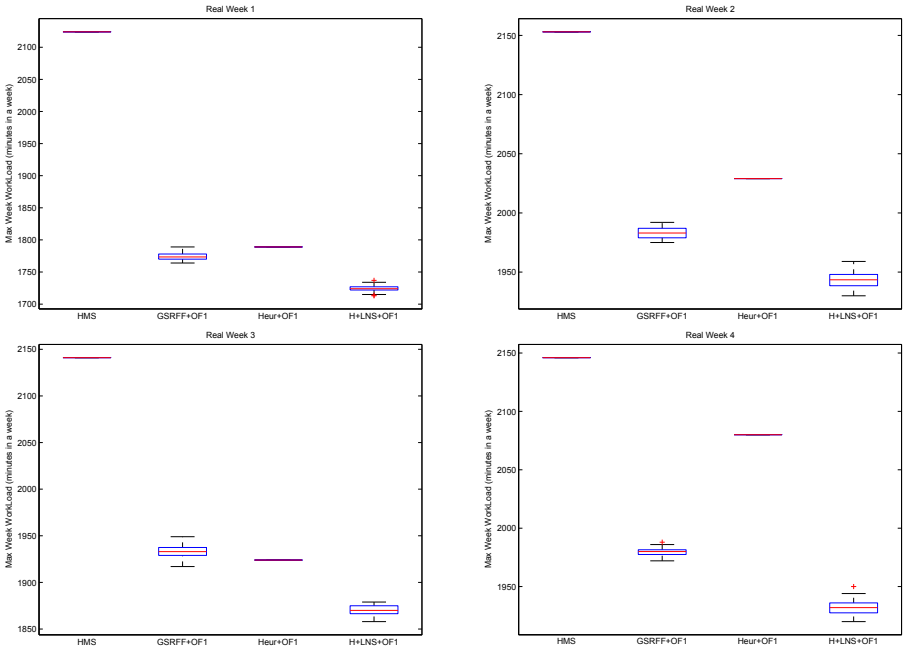


Figure 3.6: Maximum Week Workload (the lower the better) of the nurses in the four real instances, computed by the various algorithms, and compared to the Hand-Made Solution (HMS).

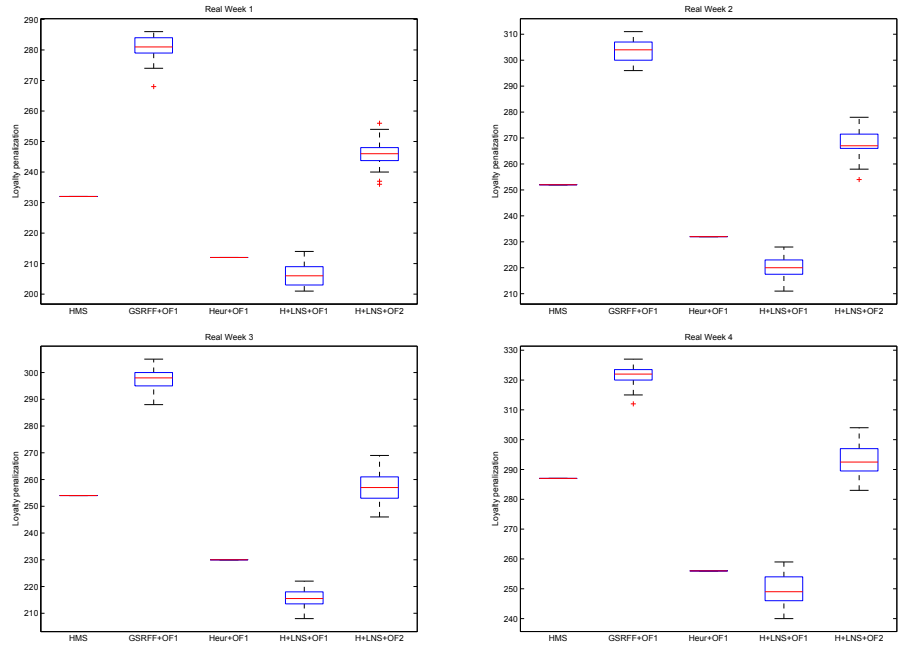


Figure 3.7: Loyalty penalization (the lower the better) obtained by the algorithms in the four real instances, compared with the Hand-Made Solution (HMS).

reassigned to other nurses). On the other hand, since the instances have very tight schedules, and the actual workload gets very close to the maximum working hours, some re-assignment takes place, and the greedy procedure followed by the nurses is not able to obtain the best possible solution. The heuristic we developed for this problem (HEUR), in fact, gets always better loyalty scores than the Hand-Made Solution, and using Large Neighbourhood Search provides a further improvement.

Sensitivity analysis

After studying the efficiency of the various search algorithms, we did a trial concerning the relative importance of the two components of the objective function OF_1. As shown in Eq. (3.6), the objective function OF_1 is the weighted sum of two components. We asked the users about reasonable values for the coefficients α_1 and α_2, but they were not able to provide values nor motivations for them, although they desired that both components were taken into consideration. When the preferences of the decision maker are totally unknown, a rigorous approach would be to compute the whole Pareto frontier (Gavanelli, 2002), that is the set of all the non dominated solutions in the space of the two criteria. However, computing the Pareto frontier is much more demanding from a computational viewpoint, and a common approach is that of a weighted sum. We chose, as tentative values for the weights, $\alpha_1 = \alpha_2 = 1$; however in the developed application these parameters can be varied by the user. We are now interested in performing some kind of sensitivity analysis to examine where our solution would be placed with respect to the Pareto front, in order to get some hints on the impact of each of the criteria and how the solution would change if slightly altering the weights coefficients. Of course, a rigorous analysis is out of reach, since the Pareto front can only be approximated when working with incomplete search methods or when using timeouts (as we do, and it is standard practice in HHC related problems, see Section 3.2).

We approximate the Pareto frontier with a classical method (Wassenhove & Gelders, 1980). The procedure exploits the fact that the domain of (at least) one of the two criteria is integer. In a minimization problem, the idea is to solve a sequence of single objective optimization problems where one criterion is optimized and the other is constrained to be less than the value of that criterion obtained at the previous solution. In our case, we kept the maximum weekly workload as the objective to be minimized and constrained the value of the loyalty component LP. As both have integer domains, the reverse would be a feasible choice too. In practice, at the first iteration the problem is solved by setting $\alpha_1 = 1$ and $\alpha_2 = 0$ in Equation (3.5). This yields the solution at the extreme right hand side in Figure 3.8. Let $LP^{*(1)}$ denote the value of the loyalty component associated to the solution found at iteration 1. At iteration $k = 2$, the same problem is solved with the addition of the constraint $LP < LP^{*(k-1)}$, so that the solution with lowest maximum weekly workload among those with lower loyalty is found. This yields the next point going leftwards along the Pareto front in Figure 3.8. Of course, in case we find a point that dominates a previously found one, the dominated is removed from the Pareto set. The procedure iterates until no feasible solution is found.

On the left side of Figure 3.8 we present the approximation of the Pareto front

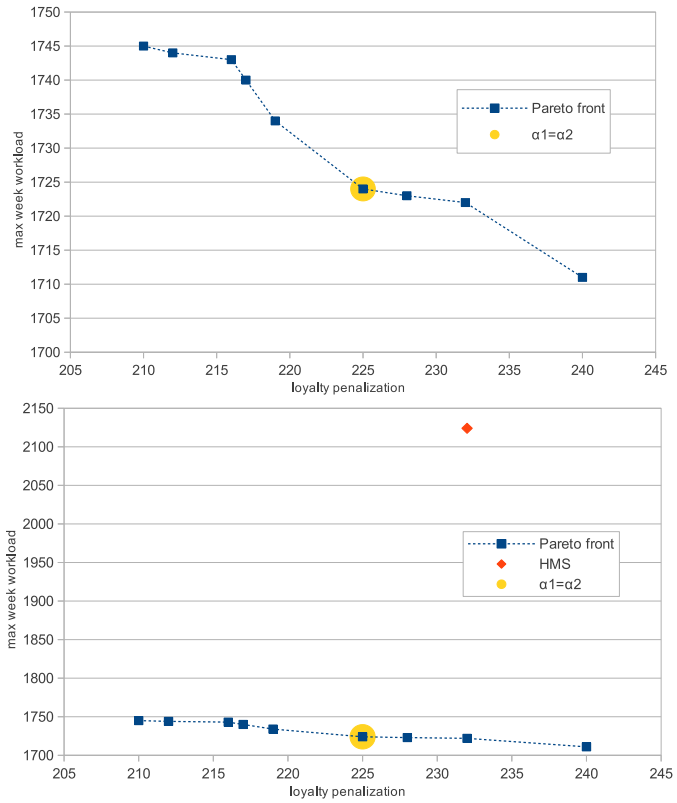


Figure 3.8: (Near)Pareto front of solutions of one weekly instance, reporting also the hand-made solution (HMS). The highlighted point is the one selected if $\alpha_1 = \alpha_2$

obtained by the above mentioned procedure using search H+LNS (Section 3.3) on one weekly instance with a 10 minutes timeout (Intel Atom Processor N450 1.66GHz and 1GB of RAM). The solution computed by setting $\alpha_1 = \alpha_2 = 1$ is highlighted. One can see that, at least for this instance, such solution is a good compromise, about in the middle of the Pareto graph; in this solution a small reduction in the maximum workload should be traded by a consistent deterioration of loyalty. The width of the Pareto graph is 30 units (it ranges from 210 to 240 in the loyalty penalization) and its height is 34 units (from 1711 to 1745 in the maximum week workload), so the choice of coefficients $\alpha_1 = \alpha_2 = 1$ does not seem unreasonable (although, of course, the graph could be different in other instances).

On the right of Figure 3.8, the (near) Pareto front is scaled to depict the hand-made solution (HMS) for the same instance; it can be noted that it is quite far from the (near) Pareto front, and that it is dominated by many points in that set.

While we are well aware of the limitations of using a scalar function to address a multi objective problem, we believe that, for our instances, the proposed method is able to reach good compromises between the two criteria.

Objective function comparison

We now evaluate the effect of using the objective function OF_2, defined in Equation (3.7), instead of OF_1, given in Equation (3.6). The objective stated by the user was a fair balancing of the workload between the nurses; this was implemented in OF_1 by adopting the minimization of the maximum workload as a balance criterion, while in OF_2 we use the absolute deviation from the mean of the service time, plus the total travel time. In Figure 3.9 we show the absolute deviation from the mean of the week workload of the various nurses. As expected, the hand-made solution is far from the optimum, and results the most unfair of the considered strategies. The algorithm adopting OF_1 gives lower deviation of Week Workload than OF_2. This is probably due to the fact that in OF_2 we minimize the deviation of the Service Time, so the Travel Time is not included in the balancing.

In order to investigate further how the objective function impacts on the distribution of the workload, in Figure 3.10 we show a comparison of the best found solutions in a typical week. The box-plot shows the distribution of the week workloads to the various nurses: each data item used in the box-plot represents the week workload of one of the nurses, obtained in the best solution found by each of the two algorithms. As we can see, using as balancing criterion the maximum week workload (as in Equation 3.3) provides a very narrow distribution of the week workload (column H+LNS+OF1 in Figure 3.10), while minimizing the absolute deviation from the mean of the service time (as in Equation 3.4) provides an unfair distribution of the workload (column H+LNS+OF2 in Figure 3.10). Again, this is due to the fact that one of the components of the workload, namely the travel time, could not be included in the *deviation*.

Another comparison between the objective functions can be seen by comparing Figures 3.11a and 3.11b. Figure 3.11a provides the absolute deviation of the Service Time (ST) in a typical instance, as provided by the proposed algorithms. We can see that through the *deviation* constraint we were able to provide the best values of absolute deviation of the service time (see column H+LNS+OF2 in Figure 3.11a). Unluckily, the same algorithm also provides the worst balancing of the week workload (Figure 3.11b), that was the real objective of the users.

Considering the loyalty criterion, from Figure 3.7 we can see that H+LNS+OF1 provides better values than H+LNS+OF2.

Finally, we consider the total travel time in Figure 3.12. The Hand-Made Solution provides very low values of travel time, as the nurses used as criterion a partitioning of the patients into contiguous zones. Also, OF_2 provides better values than OF_1, probably because only the travel time of the nurse with the highest workload is important in OF_1, while OF_2 tries to improve the travel time of all nurses.

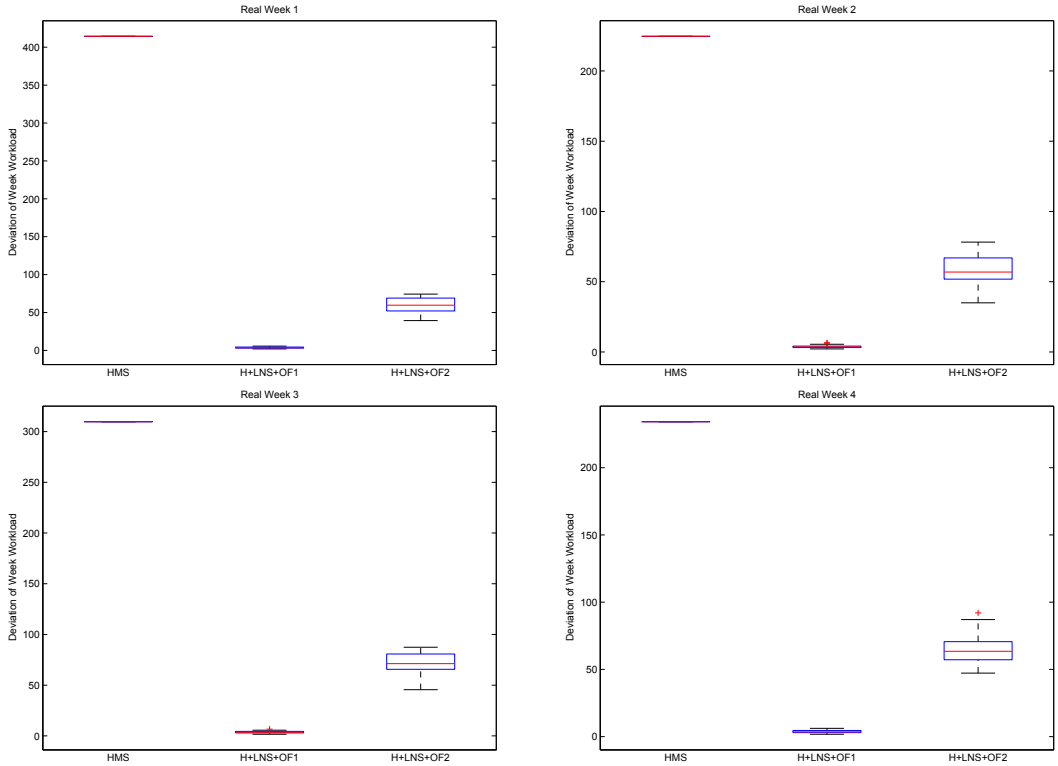


Figure 3.9: Absolute deviation from the mean of the nurses' week workload (the smaller the better) obtained with the two objective functions in the four real instances, compared with the Hand-Made Solution (HMS).

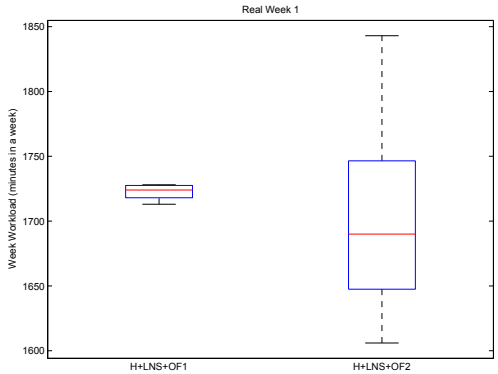
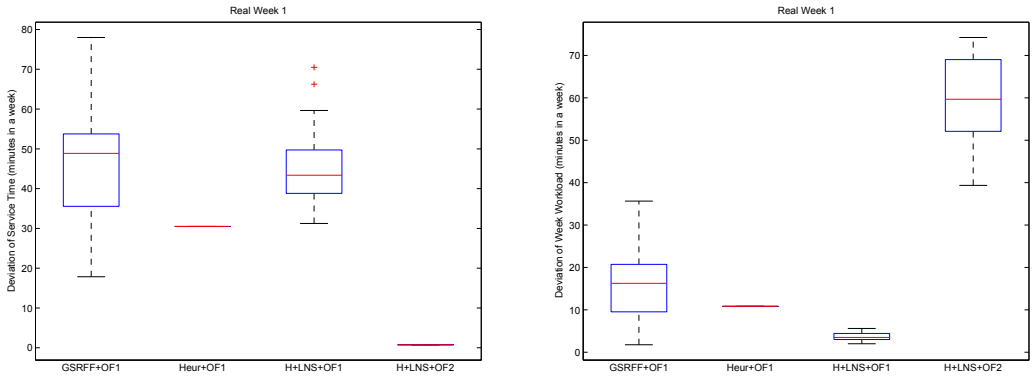


Figure 3.10: Distribution of the week workload to the various nurses in the best solutions found using the two objective functions.



(a) Absolute deviation of the service time. (b) Absolute deviation of the week workload.

Figure 3.11: Absolute deviation from the mean of the service time and the week workload obtained by the various algorithms in real week 1.

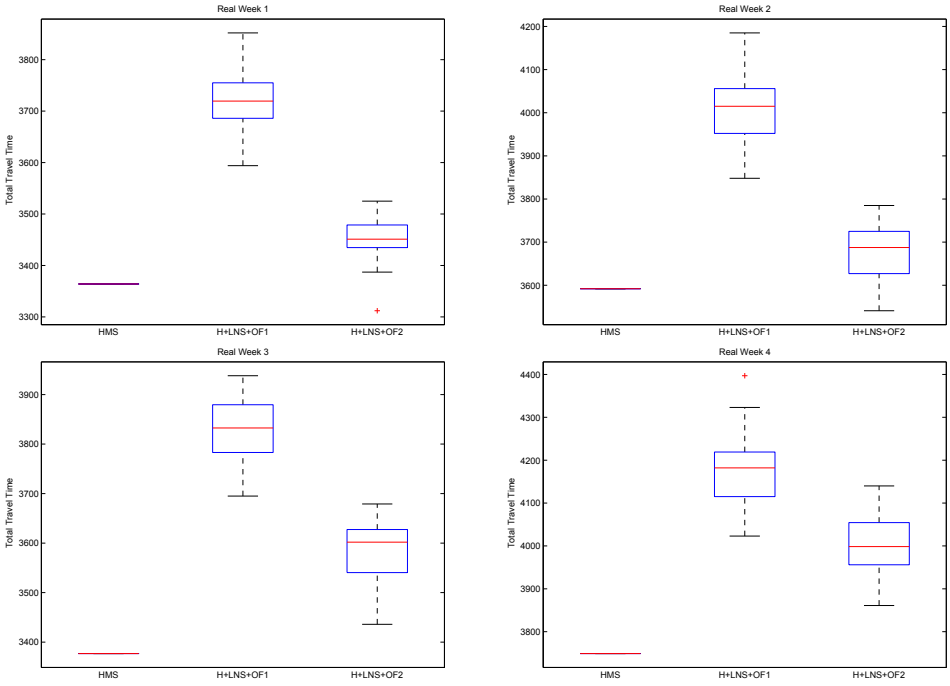


Figure 3.12: Total Travel time of the nurses (the smaller the better) obtained with the two objective functions in the four real instances, compared with the Hand-Made Solution (HMS).

TSP solver

Finally, we did some experiments to assess the efficiency of the adopted TSP solver with respect to other techniques. We considered the hand-made solution in all the four instances. Each solution of the HHC contains one TSP for each nurse for each day, for a total of 240 TSP instances (4 weeks $\times$ 12 nurses $\times$ 5 days). We solved all these TSPs with our Tailored Lagrangian Metaheuristic described in Section 1.6, with a program written for the Answer Set Programming (ASP) solver `clingo` (Gebser et al., 2011), and with a CP approach written in Comet. The total time needed to solve all the instances is the following:

- Lagrangian Relaxation: 0.485s
- ASP : 24405.22s
- Comet : 1.12s

Lagrangian Relaxation was the fastest amongst the considered technologies.

We also computed the percentage of time required by TSP solver with respect to the rest of the solution process. We obtained that, in a typical run of the HHC solution process, the LR algorithm took about 66s, out of a total running time of 698s (including CP and LR), so LR took about 9.46% of the running time. This shows that even if we changed the TSP solver, moving to a faster one (e.g., a commercial one), we could not reduce the running time by more than 10%.

Crafted instances

Although the best evaluation of an algorithm is based on real instances, in order to perform a wide and systematic evaluation of our algorithms, we generated a set of crafted instances by randomly selecting patients from the real instances. In particular, we wanted to evaluate the efficacy and efficiency of the algorithms varying two possible indicators of hardness of an instance: the total number of patients per day, and the daily total service time requested by the patients.

Since the real instances have an average of 65 patients per day with a maximum and a minimum of 82 and 51 respectively, we generated instances varying the number of patients from 50 to 80. To generate an instance, we selected at random the given number of patients; then the instances were classified into groups according to the obtained total service time. In this way, given the services on the real instances we were able to obtain 12 classes of instances, shown in Table 3.13; for each class we considered 20 instances.

In the following, we report the behavior of the various algorithms we developed, and, for the sake of comparison, we implemented an algorithm that mimics the greedy algorithm used by the nurses in the Hand Made Solution (HMS).

Note that this generation strategy does not ensure that each instance has a solution. However, it is interesting to note which algorithms were able to solve most instances, when the instance become tighter. Table 3.14 shows the number of instances that were solved for each algorithm:

Pat.	Total Service Time							
50	2700-2900	2900-3100						
60		2900-3100	3100-3300	3300-3500	3500-3700			
70				3300-3500	3500-3700	3700-3900	3900-4100	4100-4300
80							3900-4100	

Table 3.13: Classes of instances in the crafted set.

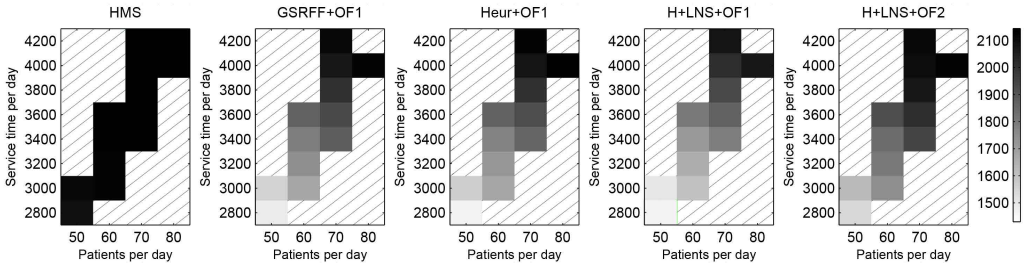
patient No.	Svc. Time	GSRFF	HEUR	HMS
50	2700-2900	20	20	20
50	2900-3100	20	20	**19**
60	2900-3100	20	20	20
60	3100-3300	20	20	20
60	3300-3500	20	20	20
60	3500-3700	20	20	20
70	3300-3500	20	20	20
70	3500-3700	20	20	20
70	3700-3900	20	20	20
70	3900-4100	20	20	20
70	4100-4300	**17**	**8**	**19**
80	3900-4100	20	**11**	20

Table 3.14: Number of solved instances in the crafted set.

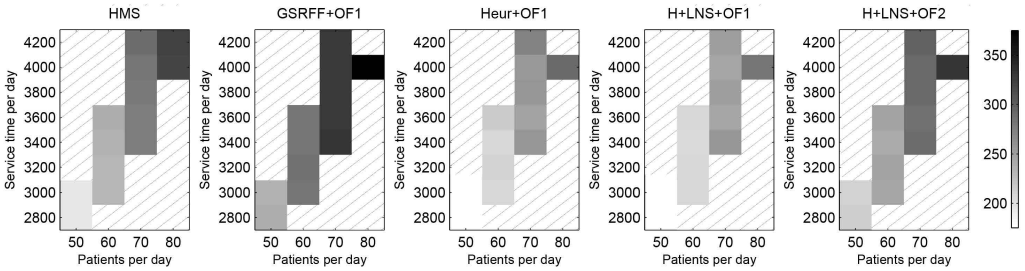
Given algorithms H+LNS+OF1, H+LNS+OF2 and HEUR solved the same number of instances, we report only one column: HEUR. The algorithms based on LNS are good to improve a solution, based on local search from an initial solution. When the instance become tighter, the heuristic does not find any initial solution, so the three algorithms were not able to find any solution. The algorithms able to find most solutions are the greedy one, that mimics the hand made solution, and the GSRFF, probably because it includes restarts, which help finding feasible solutions. However, the real instances are not that tight, so for the practical application the algorithms based on LNS are preferred. On the other hand, the HMS algorithm fails to find a solution in one small-sized instance.

Figure 3.15 represents colour maps of the 12 types of instances, where the x-axis is the number of patients per day and the y-axis is the total service time per day. The map is colored according to the average value of 10 runs for 3 randomly selected instances of each class.

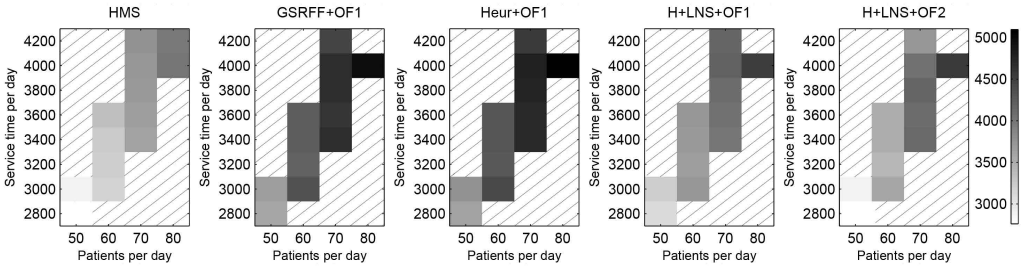
As can be seen in Figure 3.15a, all algorithms are able to improve the hand-made solution regarding the minimization of the Maximum Workload. The generic search strategy (GSRFF) is already able to improve on the hand made solution, using a tailored heuristics (HEUR) gives a further improvement, and Large Neighbourhood Search (H+LNS) improves further. Not surprisingly, using as balancing criterion the deviation of the service time (Eq. 3.4, map H+LNS+OF2 in Figure 3.15a) does not give low values



(a) Average of Max Week Workload.



(b) Average of Loyalty Penalty.



(c) Average of Total Travel Time.

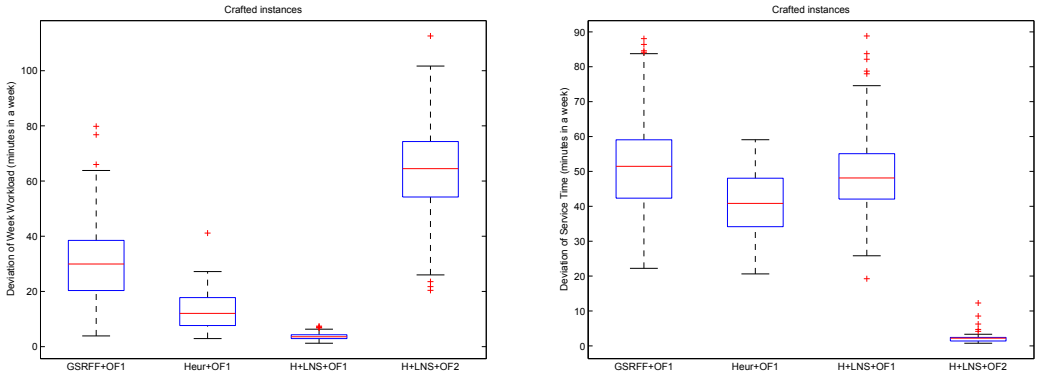
Figure 3.15: Average of the different components of the objective function in the 12 classes of instances.

for the maximum workload. Of course, the workload increases as the requested service time increases, but also the number of patients has a strong effect, probably due to the increased travel time.

Considering the *loyalty* (Figure 3.15b), we have that the GSRFF search algorithm achieves values worse than the Hand-Made Solution (although it is able to improve the sum of the two terms of Eq. (3.6) with respect to the HMS). In the Hand-Made Solution, patients are partitioned according to zones, so the HMS strategy is directed towards achieving good loyalty, since a patient is likely to have always the same nurse. However, if the workload of some nurse exceeds the maximum working hours, some patients are redistributed amongst the other nurses, so the loyalty term is not necessarily optimal and there could be still room for improvement. The other algorithms, in fact, are able to improve also the *loyalty* term of Eq. (3.6). Again, the tailored search strategies improve

significantly the loyalty. In Figure 3.15b we can also see that the loyalty depends mainly on the number of patients, and does not depend significantly on the total service time.

For the *Total Travel Time* (Figure 3.15c), we can see that the hand-made solution is able to obtain very good solutions. This is due to the fact that patients are partitioned into contiguous zones, and, in the algorithm that mimics the nurses' partitioning, we assumed that each nurse computes the optimal TSP (which is not always the case in real situations). The best algorithm is H+LNS+OF2, due to the fact that the sum of the nurses travel times appears explicitly as a term of the summation in the objective function (see Eq. 3.4), while the other algorithms strive to optimize the maximum workload, possibly overlooking the workload of the less-occupied nurses. As the intuition suggests, also the travel time depends strongly on the number of patients and weakly on the total service time (Figure 3.15c).



(a) Absolute Deviation from the mean of the Week Workload.

(b) Absolute Deviation from the mean of the Service Time.

Figure 3.16: Absolute Deviation from the mean of the Week Workload and the Service Time of the nurses. Value computed considering all the instances.

Finally, we compare the absolute deviation from the mean of the weekly workloads of the nurses. In Figure 3.16a, we report the box-plots of the results from all the instances, including both the four real instances and the 12×20 crafted ones, for a total of 244 instances. The hand-made solution is not reported because it gave values about 80 times higher, that would make the graph unreadable. There is a clear pattern showing that algorithm H+LNS+OF1 performed best. We also plot in Figure 3.16b the best values of absolute deviation of the service time; as expected we can see that algorithm H+LNS+OF2 obtains the best results. Varying the class of the instance (in the 12 classes), did not show any significant pattern, so the color maps are not reported.

3.5 Conclusions

This chapter presents a real-life application of constraint solving technologies to the Home Health Care problem for the city of Ferrara, Italy. Our proposed Tailored Lagrangian Metaheuristic has been included into a Constraint Programming framework using a search algorithm based on Large Neighborhood Search.

The objectives, as requested by the chief nurses in charge of the service, were to provide schedules with balanced workloads without increasing working hours and travel times, and to improve the service for the patients avoiding the unpleasant situation in which a patient is taken care by many different nurses. These objectives were obtained through discussions with the chief nurses responsible for the service in Ferrara.

The results showed that the obtained solutions are significantly better under these respects than the ones previously obtained by hand by the nurses. The use of CP was considered a success; we decided to use it because of the ease of its use, and flexibility with respect to other techniques, such as Integer Programming. SAT and Answer Set Programming (Brewka et al., 2011) are other interesting technologies to solve constrained optimization problems; modern solvers have sophisticated general-purpose search strategies, featuring restarts and nogood learning. CP, on the other hand, has powerful global constraints and allows the user to implement search strategies tailored for the given problem; in our experiments, custom search strategies had significantly better results than general-purpose ones; however it would be interesting to compare with a SAT or ASP solution.

The CP model has been implemented in Comet, which is more focused on the efficiency, and contains more global constraints, so we were able to propose a refined constraint model adopting the *multiknapsack* and $NValue$ constraints, which can provide enhanced constraint propagation. Furthermore, we used a new search algorithm, based on Large Neighborhood Search, that proved to be the most effective for this problem.

A deeper experimentation has been proposed with a set of crafted instances created from data of the real instances, and show the Pareto front of the two main objectives.

Different objective functions have been studied. The first one is minimizing the maximum workload and the second one is minimizing the deviation of the workload. Both objective functions consider the traveling time and the loyalty. The first objective function has reached better results.

Our proposed Tailored Lagrangian Metaheuristic has been used to calculate traveling time for each nurse separately.

- Our Tailored Lagrangian Metaheuristic ensures the partial optimality of most solutions from the routing perspective. The reason is that, since we are considering a relatively small number of patients per nurse, the proposed approach can quickly find the optimal solution to most TSP instances.

- We have compared our Tailored Lagrangian Metaheuristic respect ASP and Comet technologies, obtaining that our proposed approach was the fastest.

- We also computed the percentage of time required by the Tailored Lagrangian

Metaheuristic with respect to the rest of the solution process. We obtained that, in a typical run of the HHC solution process, it took less than 10% of the running time.

Part II:
Asymmetric Problems

Chapter 4

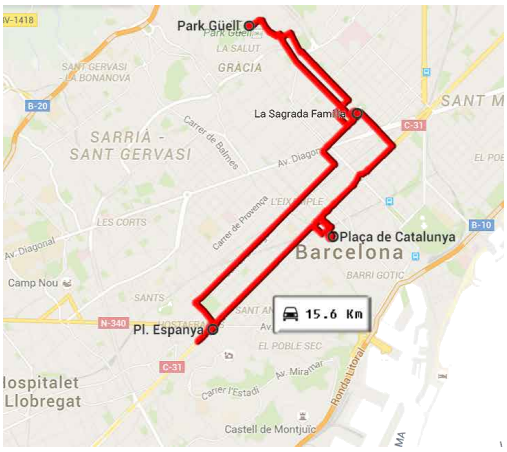
Asymmetric Traveling Salesman Problem

The Asymmetric Traveling Salesman Problem (ATSP) is a well-known NP-hard combinatorial optimization problem. The ATSP is a variation of the TSP where the cost matrix is asymmetric, i.e., the time or distance to travel between two customers is dependent to the direction. The cost matrix can be represented by travel distance, time, fuel consumption or any other cost function between two locations. The cost matrix can refer any of these terms interchangeably, but it is usually a function of the distance between pairs of locations.

As the TSP, it has numerous applications to different problems in logistics and management, see Chapter 1. In real world applications, the usage of Euclidean or orthodromic distances are no valid. Mainly given that they do not represent the urban networks, see Figure 4.1. For example, there might be a river, railroad or highway that could separate a city into zones, and vehicles are required to traverse bridges or specific pathways. In real problems, the usage of Geographic Information Systems (GIS) and geo-spatial databases are crucial. Moreover, it's worthy to distinguish that urban networks are asymmetric given that there are some one-way streets. Figures 4.1a-4.1b shows a representation of the effect of reversing the orientation of an urban route.

From the research point of view, the asymmetric TSP has been less studied than the symmetric TSP. For example, the number and size of benchmarks for the ATSP is limited, *TSPLIB: ATSP benchmark* (2008) has 19 instances, where the largest one is of 443 nodes, and they commonly have random distance matrices with no plausible connection to real applications. There are some other less-known benchmarks, but offering fewer or smaller instances. In addition, no single type of instance dominates the applications of the ATSP, thus the experimental study of asymptotic behavior is much less advanced than in the case of the TSP. In the real-world context different researches have been created their own benchmarks. For example, 12 types of random instances were generated by Cirasella et al. (2001), and in the work of Rodríguez and Ruiz (2012a), instances with 3 types of locations were generated calculating the distances with the aid of a GIS. Furthermore, some ATSP instances remain unsolved given that the number of

(a) Counterclockwise tour using asymmetric costs obtained from GIS (17.7 km).



(b) Clockwise tour using asymmetric costs obtained from GIS (15.6 km).

(c) Tour obtained using symmetric costs calculated with Euclidean distance (10.7 km).

Figure 4.1: Comparative of a tour in Barcelona according the symmetry or asymmetry of the distance cost.

researches on ATSP is much lower, and experimental studies of heuristics for the ATSP is less advanced.

Few of the algorithms designed to solve the symmetric TSP can be adapted to solve the asymmetric problem. Some authors have been transformed the asymmetric matrix into symmetric matrix, aiming to use their algorithms designed for solving the symmetric TSP. However, these transformations affect the efficiency of the methodology. For instance, the Concorde solver which was designed for solving the symmetric TSP, only works with symmetric matrices and the transformation of matrices is the only possible way of dealing with asymmetric problems. Rodríguez and Ruiz (2012a) compare the

effect of the asymmetry respect three different transformations, and conclude that the transformation to solve ATSP problems is viable, but affects drastically the efficiency of Concorde. In the work of Fischetti et al. (2003), it is compared Concorde with Branch-and-Cut methodology designed for the ATSP, and suggest that enriching the Concorde arsenal of symmetric TSP separation tools by means of ATSP-specific separation procedures would be the road to go for the effective solution of hard ATSP instances.

Given all these motivations, we pretend to adapt our Tailored Lagrangian Metaheuristic to the ATSP. In the next section, it is presented a literature review of the ATSP without considering the works which perform matrix transformations.

4.1 Problem Definition

The Asymmetric Traveling Salesman Problem is a generalization of the TSP presented in Chapter 1. Consider a complete graph, $G(I, E)$, where I and E are the set of nodes and arcs, respectively. It assumes that the cost C is an asymmetric matrix that is $c_{ij} \neq c_{ji}$ in general, and then the complete graph G is directed. The asymmetric matrix cost C can represent the travel time, travel distance, travel cost, or a combination of them.

The ATSP goal, which is the same of the TSP, consists of determining the route with minimum total traveling cost such that each customer is visited exactly once by the salesman.

A well-known Integer Linear Programming formulation of ATSP is the following:

$$\sum_{i \in I} \sum_{j \in I} c_{ij} x_{ij} \tag{4.1}$$

subject to

$$\sum_{i \in I} x_{ij} = 1 \ , \qquad \forall j \in I \tag{4.2}$$

$$\sum_{j \in I} x_{ij} = 1 \ , \qquad \forall i \in I \tag{4.3}$$

$$\sum_{i \in S} \sum_{j \in I \setminus S} x_{ij} \geq 1 \ , \qquad \forall \ S \subset I \ : \ S \neq \emptyset \tag{4.4}$$

$$x_{ij} \in \{0, 1\} \ , \qquad i, j \in I \tag{4.5}$$

In this formulation for the ATSP, $O(n^2)$ binary variables x are used. The binary variable x_{ij} indicates whether or not customer j is visited immediately after i. The objective function (4.1) aims to minimize the total cost of the route.

Every city node must be visited once and only once. From the point of view of the asymmetry, it means that every city must have one starting edge and one ending edge. Constraints (4.2) and (4.3) impose in-degree and out-degree of each vertex equal to one.

As well as the set of Subtour Elimination constraint (1.3) of the TSP, constraints (4.4) impose strong connectivity. Constraints (4.5) correspond exactly to equation (1.4) of the TSP, which impose integer variables.

Notice that the two constraints (4.2) and (4.3) could be replaced by $\sum_{i \in I} \sum_{j \in I} x_{ij} = n$, where $n = \mid I \mid$ is the number of nodes. An interesting comparative of different formulations for the ATSP problem can be found in the work of Öncan et al. (2009).

As the TSP, the ATSP is a NP-Hard problem and it can be formulated as finding a Hamiltonian circuit. The condition to ensure feasible solutions is that the directed graph G must be strongly connected, and its underlying graph must be complete.

4.2 Literature Review

The best known exact algorithms for the ATSP are based on the Branch-and-Bound method which uses the well known Assignment Problem relaxation of the ATSP, for a complete description see (D. L. Miller & Pekny, 1991) and (Fischetti et al., 2007). Carpaneto et al. (1995) presented a lowest-first Branch-and-Bound algorithm to obtain exact solution of large-scale. They generated random instances with up to 2000 nodes, they also solved real-world stacker crane problems derived from a Siemens factory in Augsburg. Pekny and Miller (1992) presented a parallel Branch-and-Bound algorithm which is a parallelization of (Carpaneto & Toth, 1980), but improved with the application of the patching heuristic developed by Karp at the root node. Turkensteen et al. (2008) considered depth-first Branch-and-Bound algorithm. Their main algorithmic contribution are the upper tolerance values of arcs in the corresponding Assignment Problem instance to determine which arcs should be excluded, which is aiming to guide the search for optimal solutions. Germs et al. (2012) developed a new tolerance-based Branch-and-Bound algorithm proposing a lower bounding strategy with the expected costs of including arcs in the solution.

As state above, the Branch-and-Bound is the best known exact method, however, it is not the most effective exact method. Fischetti et al. (2007) surveyed exact methods for the ATSP and compared computationally Branch-and-Bound and Branch-and-Cut codes. The results of this comparison proved that Branch-and-Cut is the most effective method to solve hard ATSP instances. In the work of Fischetti and Toth (1997b), the authors exploited additional classes of facet-inducing inequalities for the ATSP polytope that proved to be of crucial importance for the solution of some real-world instances. Fischetti et al. (2003) compare the Branch-and-Cut algorithm by Fischetti and Toth with Concorde, and conclude that it is generally better than Concorde. It was indicated the effectiveness of exploiting the ATSP-specific separation procedures. Moreover, this Branch-and-Cut algorithm has been adapted to ATSP with time windows (Ascheuer et al., 2001) and ATSP with precedence constraints (Ascheuer et al., 2000).

Some classical heuristics for the ATSP are Nearest Neighbor, Nearest Insertion and Multiple Fragment (often referred to as Greedy) algorithms. In addition to this classic class of tour construction heuristics, some of the latest heuristic approaches that are being studied to solve the ATSP are Lin-Kernighan heuristic, Patching heuristic, Ge-

netic algorithms, Memetic algorithms and Evolutionary algorithms. For a review and experimental analysis of heuristics see (Johnson et al., 2007).

Lin-Kernighan heuristic (LK) is a local search algorithm based on re-arranging segments of the tour. Lin and Kernighan (1973) proposed an effective heuristic for the TSP, which is based on a k-Opt procedure with k variable in each step. Some of the most important adaptations for LK heuristic to the ATSP are Kanellakis and Papadimitriou (1980) and Helsgaun (2000). On the work of Helsgaun (2000), it is described a very effective implementation of the LK, named LKH, for both symmetric and asymmetric problems.

The second class to be considered are Cycle-Patching algorithms which are based on patching together the cycles in a minimum cycle cover, it can be computed as the solution to an Assignment Problem. The Patching heuristic was proposed by Karp (1979), his heuristic has been extensively used to find an initial solution or upper bounds in Branch-and-Bound algorithms and hybrid methodologies. In the work of Glover et al. (2001), the Patching heuristic is modified joining two shortest cycles together, it tries all possible pairs using the best one. They also described an algorithm combining their modified Patching heuristics with a recursive Path Contraction heuristic.

Genetic algorithms are a population-based approach for heuristic search in optimization problems, in our case for the ATSP, Nagata and Kobayashi (1997) proposed the use of an edge assembly crossover operator for both cases, the symmetric and the asymmetric. Later, Nagata and Soler (2012) designed an effective crossover operator for the ATSP, experimental results demonstrate that their proposed Genetic algorithm outperforms state-of-the-art heuristic algorithms and, in addition, they created a new set of benchmarks for the ATSP. In the work of Xing et al. (2008), it is proposed a hybrid approach which incorporates an improved Genetic algorithm and some optimization strategies that contribute to its effectiveness, both the crossover operation and the mutation operation are improved by selecting the optimum from a set of solutions.

Some authors have improved the Genetic algorithm combining it with a local search, this approach is named Memetic algorithm, for an introduction see (Neri et al., 2012). The approach proposed by I. C. Choi et al. (2003) extends the Genetic algorithm search space by purposefully generating and including infeasible solutions in the population. Freisleben and Merz (1996) introduced a new recombination operator called the Distance Preserving Crossover and employed the 3-opt variant as a local search engine for the ATSP. Later, Merz and Freisleben (1997) improved their results by adding a variant of the 4-opt move to the search in the ATSP cases. According to Buriol et al. (2004), based on the computational results reported for this method, one can say that a good implementation must combine several essential features: (i) suitable recombination and mutation operators; (ii) a fast and effective local search algorithm; (iii) a hierarchically structured population; (iv) suitable data structures and smart codification mechanisms. In their work, all of the four key features are incorporated responsibly for an effective solution.

Evolutionary algorithms seek to solve search problems by modeling the behaviors of social insects such as bees and ants. Ant Colony Optimization is not only capable

of generating good solutions for the symmetric TSP, but also for asymmetric instances of the TSP, it was first initially proposed by Dorigo and Gambardella (1997). In the work of Andziulis et al. (2011), it is compared an Ant Colony Optimization with the Nearest Neighbor heuristic using specific real-life instances. Their conclusions suggest that the Ant Colony Optimization algorithm obtains better results, but it is slightly slower. Bai et al. (2011) proposed parallel optimization architecture in which a Branch-and-Bound and a Ant Colony Optimization are launched concurrently, and a reduction procedure takes useful information from both solver to reduce the global search space and hence speed up the rate of convergence to optimality. In Wong et al. (2013), the authors presented a generic Bee Colony Optimization framework, which computationally realizes the bee foraging behavior in a typical bee colony where bees travel across different locations to discover new food sources and perform waggle dances to recruit more bees towards newly discovered food sources. Celik and Ulker (2012) proposed a Marriage in Honey Bee Optimization which is a meta-heuristic procedure inspired by the mating and insemination process of honey bees.

A quite recent method to solve VRPs is the Variable Neighborhood Search, it was introduced for the first time by Mladenović and Hansen (1997). Burke et al. (2001) presented effective new local and Variable Neighborhood Search heuristics for the ATSP, they introduced a local search routine based on splitting the original problem into small subproblems which are then solved to optimality using an exact algorithm.

The Lagrangian Relaxation (LR) is a well studied method, for a good survey on LR for solving the ATSP see Rocha et al. (2004). Balas and Christofides (1981) proposed a Restricted Lagrangian Relaxation for the ATSP based on the Assignment Problem. They developed an LR approach by relaxing the subtour elimination constraints. In addition, it is generated upper bounds using a fast tour-building heuristic. Once the bound-strengthening techniques have been exhausted without matching the upper with the lower bound, they branch by using two different rules, according to the situation: the usual subtour breaking disjunction, and a new disjunction based on conditional bounds. In the work of Smith (1980), it is used an 1-Arborescence relaxation, in which the out-degree constraints are relaxed, and developed a Lagrangian method based on this relaxation to provide lower bounds for a Branch-and-Bound method. (Fischetti & Toth, 1992) devoted much more computational effort to improve the bound of the Assignment problem relaxation. It was used a fairly complex additive bounding procedure solving a different relaxation of the ATSP at each iteration with costs given by the residual cost from the previous iteration. They used the Restricted Lagrangian Relaxation proposed by Balas and Christofides. In Asadpour et al. (2010), it is considered the Lagrangian Relaxation of the ATSP based on the Spanning Arborescence of minimum weight. They first constructed a spanning tree with special properties. Then they find a minimum cost Eulerian augmentation of this tree, and finally, shortcut the resulting Eulerian walk.

4.3 Our proposed TLM for the asymmetric case

In this section, our Tailored Lagrangian Metaheuristic for solving the TSP is adapted to the ATSP. For this problem, the out-degree constraints are relaxed, then the dual problem obtained is an 1-Arborescence, that is, the directed analog of the 1-Tree problem. In order to adapt our TLM, Edmond's Algorithm is used to solve the dual problem, and a new heuristic to obtain a feasible solution from a dual one is proposed.

The remainder aspects about our TLM is as proposed in Algorithm 3 on page 35, specially our main contribution consists on the new parameter which is updated using the values of both UB and LB and has improved significantly algorithm's convergence on the optimal solution.

Modeling the Lagrangian Dual Problem

The Subtour Elimination constraint (4.4) and the in-degree constraint (4.2) can be disregarded solving x as an 1-Arborescence. Then, the complicated set of constraints is the out-degree constraints (4.3).

A Spanning Arborescence is the directed graph form of a rooted tree, that is, for a node r called the root, there is exactly one directed path from r to any other node. A minimum Spanning Arborescence can be calculated easily, for example, the order of Edmonds' Algorithm is $O(EI)$. Therefore, the 1-Arborescence is a Spanning Arborescence plus an additional incoming edge to the root, and thus, it has only one cycle in it.

In this proposed approach to the ATSP, LR relaxes the constraint set requiring that all customers must have one out going edge, as proposed by Smith (1980). Therefore, the Lagrangian Dual problem obtained from ATSP by taking into the objective function the inequalities 4.3 is as follows:

$$\max_{u \in \mathbb{R}^N} L(u) \tag{4.6}$$

where

$$\begin{aligned} L(u) &= \min_{x\ 1-Arborescence} \Big(\sum_{i \in I} \sum_{j \in I} x_{ij} c_{ij} + \sum_{i \in I} u_i (1 - \sum_{j \in I} x_{ij}) \Big) = \\ &= \sum_{i \in I} u_i + \min_{x\ 1-Arborescence} \Big(\sum_{i \in I} \sum_{j \in I} (c_{ij} - u_i) x_{ij} \Big) = \\ &= \sum_{i \in I} u_i + \min_{x\ 1-Arborescence} \sum_{i \in I} \sum_{j \in I} \hat{c_{ij}} x_{ij} \end{aligned} \tag{4.7}$$

where $\hat{c_{ij}} = c_{ij} - u_i$ define the dual cost.

Its subgradient γ represents the penalization of the outgoing edges being a vector of dimension n, calculated as follows:

$$\gamma = 1 - \sum_{j \in I} x_{ij} \tag{4.8}$$

The dual solution is an optimal solution of the ATSP if and only if

$$\| \gamma \|^2 = 0 \tag{4.9}$$

Edmonds' Algorithm

Considering a directed graph $G(I, E)$, where I and E are the set of nodes and arcs, respectively. Let $|I| = n$. The Spanning Arborescence problem consists on finding a rooted directed spanning tree, $G(I, S)$ where S is a subset of E of minimal cost. The rooted directed spanning tree is defined as a connected graph with $n - 1$ arcs, i.e., each node, except the root, has one incoming arc without any cycle.

Algorithm 8 Edmonds' Algorithm for the 1-Arborescence

Initialization:
  Let be the subset $S = \emptyset \subset E$.
  Discard the incoming edges to the depot.
for each node other than the depot **do**
  Select the incoming edge e with the smallest cost.
  Add this edge to the subset: $S = S \cup \{e\}$.
end for
while $G(I, S)$ contains a cycle **do**
  for each cycle C_k formed **do**
    Contract the nodes in the cycle C_k into a pseudo-node k.
    Modify the cost of each edge which enters a node j in the cycle from some node i outside the cycle according to the equation 4.10.
  end for
  for each pseudo-node k **do**
    Select the entering edge which has the smallest modified cost.
    Replace the edge which enters the same real node in S by the new selected edge.
  end for
end while

Find $e = (i, 1) \in E$ the minimal weighted edge entering to the depot with $(1, i) \notin S$.
Add this edge to the subset: $S = S \cup \{e\}$.
return S.

The main idea of Edmonds' algorithm is to find a replacing edge which has the minimum extra cost to eliminate a cycle if any. For that, a pseudo-node is created with associated cost according to the following equation representing the associated extra cost to eliminate the cycle.

$$c_{i,k} = \min_{j \in C_k} \{c_{i,j} - c_{x(j),j} + c_k\} \tag{4.10}$$

where i is a node outside the cycle, j is a node in the cycle, $c_{x(j),j}$ is the cost of the edge in the cycle which enters j and c_k is the minimum cost of all the edges entering to the

cycle, i.e., $c_k = \min_{j \in C_k}\{c_{x(j),j}\}$.

For solving the 1-Arborescence, an additional incoming edge to the root is added forming a cycle. The modified Edmonds' algorithm is shown in Algorithm 8, where the depot ($i = 1$) is the root.

The Proposed Heuristic

A heuristic to obtain a feasible solution from a dual one and to improve the UB has been proposed. The subgradient γ is related with the outgoing edges, the next two penalized situations can happen for each node:

- $\gamma^- = 1$ means that the node has not any successor, and it belongs to the source list I_S.
- $\gamma^- < 0$ means that the node has several successors, and it belongs to the pivot list I_P.

The proposed heuristic is presented in Algorithm 9, it iteratively applies a swapping movement removing an edge connected to a node from the pivot list and adding an edge from a node to the source list.

Algorithm 9 Heuristic to feasible dual solution for asymmetric cases

Data: S the obtained 1-Arborescence.
Initialize:
$I_S = \{i \in I \mid \gamma_i = 1\}$ the list of nodes without successors.
$I_P = \{i \in I \mid \gamma_i < 0\}$ the list of nodes with several successors.
Swapping Movement:
while $I_R \neq \emptyset$ **do**
Find $\{n_P, n_C, n_S\} = argmin\{c_{n_S,n_C} - c_{n_P,n_C}\}$ where $n_P \in I_P$, n_C is an immediate successor of n_P, and $n_S \in I_S$ is a successor of n_P but it is not a successor of n_P.
Remove node n_S from the list I_S.
Swap these edges within the subset: $S = S \setminus (n_P, n_C) \cup (n_S, n_C)$
Calculate the subgradient γ_{n_P} of the node n_P.
if $\gamma_{n_P} = 0$ **then**
Remove node n_P from the list I_P.
end if
end while
return S.

4.4 Computational Results

Our TLM methodology adapted for the ATSP in this chapter has been implemented in Java language. All tests have been performed on a personal laptop with an Intel Core i5 processor at 2.40GHz and 4GB RAM.

A total of 27 ATSP instances have been used to test the efficiency of the proposed approach. They have been obtained from the library TSPLIB (http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/atsp/ last updated August 6, 2008), a reference site with a large number of instances for the TSP, and related problems, from various sources and of various types.

The experiments have been conducted using the distance according to the specification included in the library, and using the next values for the parameters: 300 the number of iterations; $\delta_0 = 1.99$; $\rho = 0.95$ and $\alpha_L = 2/3$.

Table 4.2 shows results obtained for all 27 instances from the selected benchmark sets; and each of these results are compared to the best known solutions (BKS) so far. The table summarizes information regarding the best lower bound (LB, i.e, the best dual value), the percentage distance from BKS of the LB ($\%\Delta LB$), the best feasible solution found (UB), the percentage distance from BKS of the UB ($\%\Delta UB$), and the final CPU time in seconds (t_{Final}).

The methodology has found the optimal value in four instances, in these cases, the number of iteration when it stopped is shown inside brackets in the last column. Note that in two cases, the metaheuristic does not recognize the optimal value without stopping until the last iteration is reached. The table shows that the gap is kept reasonably low for most of the considered instances, however, it does not succeed for all instances being the average gap 2.43 %.

Furthermore, symmetric TSP instances are used to compare if this adaptation for the asymmetric case is as good as the original TLM for the symmetric case. Tables 4.3–4.4 provide a comparison between both proposed algorithms for 33 symmetric TSP instances. The first results correspond to the original TLM for the TSP, presented in Chapter 1. And the second results correspond to this adaptation for the ATSP.

	# Nodes	BKS	LB	$\%\Delta LB$	UB	$\%\Delta UB$	CPU_{time} (s)
br17	17	39	38.99	0.03 %	40	2.56 %	0.59
p43	43	5620	1025.32	81.76 %	5627	0.12 %	1.60
ry48p	48	14422	14288.87	0.92 %	14422	**0.00%**	1.83 (Max it)
ft53	53	6905	6905.00	0.00 %	6905	**0.00%**	1.51 (it 258)
ft70	70	38673	38651.04	0.06 %	38802	0.33 %	3.06
ftv33	34	1286	1286.00	0.00 %	1286	**0.00%**	0.28 (it 140)
ftv35	36	1473	1457.28	1.07 %	1475	0.14 %	0.61
ftv38	39	1530	1514.26	1.03 %	1532	0.13 %	0.73
ftv44	45	1613	1584.46	1.77 %	1648	2.17 %	1.04
ftv47	48	1776	1747.93	1.58 %	1810	1.91 %	1.50
ftv55	56	1608	1583.63	1.52 %	1632	1.49 %	2.10
ftv64	65	1839	1807.15	1.73 %	1858	1.03 %	2.60
ftv70	71	1950	1907.96	2.16 %	1965	0.77 %	3.58
ftv90	91	1579	1569.46	0.60 %	1579	**0.00%**	7.72 (Max it)
ftv100	101	1788	1766.78	1.19 %	1851	3.52 %	9.58
ftv110	111	1958	1922.67	1.80 %	2038	4.09 %	14.19
ftv120	121	2166	2128.67	1.72 %	2266	4.62 %	16.73
ftv130	131	2307	2284.52	0.97 %	2440	5.77 %	21.73
ftv140	141	2420	2398.15	0.90 %	2527	4.42 %	25.92
ftv150	151	2611	2589.82	0.81 %	2694	3.18 %	31.78
ftv160	161	2683	2645.72	1.39 %	2759	2.83 %	41.13
ftv170	171	2755	2713.96	1.49 %	2800	1.63 %	47.58
kro124p	100	36230	35993.03	0.65 %	36682	1.25 %	11.90
rbg323	323	1326	1206.22	9.03 %	1495	12.75 %	1464.93
rbg358	358	1163	835.82	28.13 %	1275	9.63 %	2078.44
rbg403	403	2465	1595.82	35.26 %	2484	0.77 %	888.30
rbg443	443	2720	1750.80	35.63 %	2736	0.59 %	1211.06
				7.90 %		2.43 %	

Table 4.2: Results for 27 ATSP instances.

		Adapted TLM for asymmetric cases					Original TLM for symmetric cases				
	BKS	LB	$\%\Delta LB$	UB	$\%\Delta UB$	CPU_{time} (s)	LB	$\%\Delta LB$	UB	$\%\Delta UB$	CPU_{time} (s)
eil51	426	422.38	0.85 %	444	4.23 %	2.547	422.42	0.84 %	440	3.29 %	0.226
st70	675	670.77	0.63 %	693	2.67 %	5.546	670.89	0.61 %	687	1.78 %	0.255
eil76	538	536.74	0.23 %	550	2.23 %	6.456	536.94	0.20 %	544	1.12 %	0.297
pr76	108159	105099.82	2.83 %	110380	2.05 %	6.705	105112.85	2.82 %	108584	0.39 %	0.25
rat99	1211	1205.81	0.43 %	1240	2.39 %	14.128	1205.91	0.42 %	1237	2.15 %	0.257
kroA100	21282	20928.63	1.66 %	21845	2.65 %	14.893	20930.10	1.65 %	21319	0.17 %	0.432
kroB100	22141	21821.86	1.44 %	23174	4.67 %	15.579	21830.02	1.40 %	22931	3.57 %	0.267
kroC100	20749	20464.95	1.37 %	21609	4.14 %	15.163	20467.78	1.36 %	21270	2.51 %	0.53
kroD100	21294	21136.74	0.74 %	22014	3.38 %	15.484	21138.27	0.73 %	21792	2.34 %	0.298
kroE100	22068	21795.27	1.24 %	23495	6.47 %	15.564	21798.00	1.22 %	22905	3.79 %	0.734
rd100	7910	7897.41	0.16 %	8240	4.17 %	15.385	7898.65	0.14 %	8006	1.21 %	0.499
eil101	629	627.33	0.27 %	641	1.91 %	16.071	627.35	0.26 %	629	0.00 %	0.36
lin105	14379	14368.82	0.07 %	14529	1.04 %	17.342	14370.14	0.06 %	14402	0.16 %	0.237
pr107	44303	39032.82	11.90 %	45856	3.51 %	20.189	39313.79	11.26 %	44420	0.26 %	0.379
pr124	59030	58040.04	1.68 %	61449	4.10 %	28.436	58048.18	1.66 %	60112	1.83 %	0.463
bier127	118282	117381.05	0.76 %	123862	4.72 %	31.991	117396.83	0.75 %	124502	5.26 %	1.404
ch130	6110	6072.11	0.62 %	6351	3.94 %	35.601	6074.46	0.58 %	6203	1.52 %	0.351
pr136	96772	95491.94	1.32 %	105653	9.18 %	38.368	95373.05	1.45 %	102534	5.95 %	0.905
			1.57 %		3.75 %	17.52		1.52 %		2.07 %	0.45

Table 4.3: Comparison between both proposed algorithms, the original TLM for the TSP and the adaptation for the ATSP, for 33 TSP instances.

		Adapted TLM for asymmetric cases					Original TLM for symmetric cases				
	BKS	LB	$\%\Delta LB$	UB	$\%\Delta UB$	CPU_{time} (s)	LB	$\%\Delta LB$	UB	$\%\Delta UB$	CPU_{time} (s)
ch150	6528	6487.79	0.62 %	6703	2.68 %	52.461	6488.12	0.61 %	6640	1.72 %	1.279
kroA150	26524	26280.77	0.92 %	28177	6.23 %	55.463	26280.90	0.92 %	27598	4.05 %	1.608
kroB150	26130	25716.88	1.58 %	28137	7.68 %	53.733	25725.39	1.55 %	27273	4.37 %	1.577
pr152	73682	66353.72	9.95 %	77637	5.37 %	57.241	66583.97	9.63 %	78848	7.01 %	0.219
u159	42080	41910.24	0.40 %	43702	3.85 %	63.464	41921.90	0.38 %	42636	1.32 %	1.186
rat195	2323	2290.07	1.42 %	2502	7.71 %	125.598	2290.58	1.40 %	2489	7.15 %	2.443
kroA200	29368	29045.96	1.10 %	31742	8.08 %	145.176	29056.19	1.06 %	30983	5.50 %	3.231
kroB200	29437	29152.20	0.97 %	31844	8.18 %	136.549	29158.27	0.95 %	31126	5.74 %	2.96
ts225	126643	115463.11	8.83 %	140220	10.72 %	189.093	115472.09	8.82 %	144299	13.94 %	1.145
tsp225	3919	3872.44	1.19 %	4360	11.25 %	216.722	3875.89	1.10 %	4261	8.73 %	3.23
pr226	80369	78537.49	2.28 %	84748	5.45 %	207.97	78451.35	2.39 %	84203	4.77 %	3.155
gil262	2378	2351.62	1.11 %	2515	5.76 %	341.253	2354.00	1.01 %	2473	3.99 %	5.117
pr264	49135	46190.85	5.99 %	51898	5.62 %	346.002	46236.48	5.90 %	50868	3.53 %	5.18
a280	2579	2564.48	0.56 %	2727	5.74 %	400.35	2565.40	0.53 %	2704	4.85 %	7.202
pr299	48191	47326.67	1.79 %	52052	8.01 %	524.62	47360.81	1.72 %	51053	5.94 %	7.86
			2.58 %		6.82 %	194.38		2.53 %		5.51 %	3.16

Table 4.4: (continued) Comparison between both proposed algorithms, the original TLM for the TSP and the adaptation for the ATSP, for 33 TSP instances.

Table 4.5 summarizes the results obtained in tables 4.3–4.4, the solved problems are presented ordered by size. It may be observed that the adaptation for the asymmetric case obtains almost as good as the original one. Particularly, the adaptation is not comparable in terms of computational efficiency, see Figure 4.6.

		Adapted TLM for asymmetric cases		Original TLM for symmetric cases	
Size	Problems	$\%\Delta UB$	CPU_{time} (s)	$\%\Delta UB$	CPU_{time} (s)
$n < 150$	18	3.75 %	17.52	2.07 %	0.45
$150 \leq n < 300$	15	6.82 %	194.38	5.51 %	3.16

Table 4.5: Summary of results obtained comparing both algorithms for 33 TSP instances.

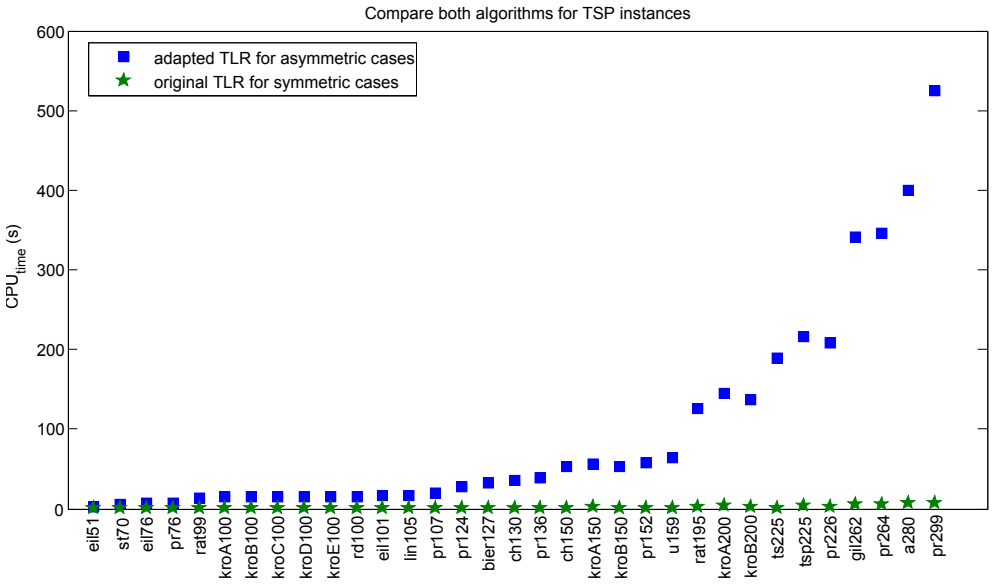


Figure 4.6: Comparison of CPU_{time} between both proposed algorithms, the original TLM for the TSP and the adaptation for the ATSP, for 33 TSP instances.

4.5 Conclusions

This chapter has presented an adaptation of our proposed Tailored Lagrangian Metaheuristic for solving the Asymmetric Traveling Salesman Problem.

Our method uses the Subgradient algorithm combined with a heuristic which is introduced to obtain a feasible solution from the dual variable providing both UB and LB, thus a posterior quality check of the solution is obtained. The motivation of our method is the common sense belief that dual solutions could be relevant to primal solutions.

The results have shown that the new parameter has improved significantly Subgradient's convergence on the optimal solution and it is showing good results in terms of the quality of the solution. However, it is not showing a good behavior in terms of computational efficiency.

Chapter 5

Asymmetric Capacitated Vehicle Routing Problem

The Vehicle Routing Problem (VRP) is a flourishing research area with clear applications to real-life distribution companies. VRPs deal with the physical distribution of goods from a central depot to customers, see for instance Toth and Vigo (2002a) and Golden et al. (2008). The best-known VRP variant is the so-called Capacitated Vehicle Routing Problem ($CVRP$). However, frequent assumption is that distance-based costs associated with traveling from one node i (customer or depot) to another node j, c_{ij}, are symmetric, i.e., $c_{ij} = c_{ji}$ for all pair of nodes. This assumption is not always reasonable in real-life scenarios. It should also be taken into account that the main goal of VRPs is to minimize distance-based costs associated with the distribution of products among customers while satisfying customers‘ demands. To contribute closing this gap between theory and practice, we propose a hybrid methodology for solving the Asymmetric Capacitated Vehicle Routing Problem (ACVRP).

In the VRP is common to consider a matrix which elements are the individual costs of traveling from one node to another. These costs can be based on several distance-related factors, such as: actual distance, time employed, fuel consumption, etc. In fact, classical benchmark instances are based on Euclidean distances between each pair of locations, which results in symmetric costs. However, this metric is just a lower bound of the real distance between two nodes connected by a transport network or highway. The real distance will depend upon the specific location of the nodes in the territory and also on the structure of the road network that communicates them. Moreover, when considering oriented networks, real distances might not have to be symmetric (Rodríguez & Ruiz, 2012b).

This chapter aims to present an efficient and relatively easy-to-implement methodology for solving the Asymmetric Capacitated Vehicle Routing Problem (ACVRP). Our approach is based on the combination of a randomized version of a classical heuristic with several local search processes specifically adapted to deal with the asymmetric nature of costs. The approach presented in this chapter was first presented in (Herrero et al., 2014).

5.1 Problem Definition

In general, VRP problems are $NP - complete$ within the combinatorial optimization field. Its constraints make it difficult to find feasible and good solutions in a reasonable computing time.

The ACVRP goal is the same of the CVRP, see Chapter 2. Solving the ACVRP consists of determining the set of minimum $k \leq K$ routes with minimum total traveling cost and such that each customer is visited exactly once by a single vehicle, each route starts and ends at the single-depot, and the total demand of the customers assigned to a route does not exceed the vehicle capacity Q. The only difference from the CVRP is the asymmetry of the distance matrix.

Let $G = (I, E)$ be a complete and directed graph, where $I = \{0, 1, \ldots, n\}$ is the set of nodes and E is the set of arcs. The nodes $i = 1, 2, \ldots, n$ correspond to customers, each with a deterministic demand $d_i \geq 0$. The node $i = 0$ is the depot with zero-demand $d_0 = 0$. Let $K > 0$ be the number of available vehicles, all of them with the same load capacity $Q > 0$. Let c_{ij} be the non-negative distance-based cost associated with the arc $(i, j) \in E$, and $c_{ii} = +\infty$, $\forall i \in I$. The distance matrix is considered asymmetric, that may be calculated using the real distances between pairs of locations (i, j), i.e., it can happen that $c_{ij} \neq c_{ji}$.

The first formulation of this problem was proposed by Laporte et al. (1986). A well-known programming formulation of the problem is presented as follows:

$$\min \sum_{i \in I} \sum_{j \in I} c_{ij} x_{ij} \tag{5.1}$$

subject to:

$$\sum_{i \in I \setminus \{j\}} x_{ij} = 1 \qquad \forall j \in I \setminus \{0\} \tag{5.2}$$

$$\sum_{j \in I \setminus \{i\}} x_{ij} = 1 \qquad \forall i \in I \setminus \{0\} \tag{5.3}$$

$$\sum_{i \in I \setminus \{0\}} x_{i0} \leq K \tag{5.4}$$

$$\sum_{i \in I \setminus \{0\}} x_{i0} = \sum_{j \in I \setminus \{0\}} x_{0j} \tag{5.5}$$

$$\sum_{i \in I \setminus \{j\}} y_{ij} + d_j = \sum_{i \in I} y_{ji} \qquad \forall j \in I \setminus \{0\} \tag{5.6}$$

$$0 \leq d_i x_{ij} \leq y_{ij} \leq (Q - d_j) x_{ij} \qquad \forall (i, j) \in E \tag{5.7}$$

$$x_{ij} \in \{0, 1\} \qquad \forall (i, j) \in E \tag{5.8}$$

$$y_{ij} \in [0, Q] \qquad \forall (i, j) \in E \tag{5.9}$$

In this formulation for the ACVRP, $O(n^2)$ binary variables x are used. The binary variable x_{ij} in (5.8) indicates whether or not customer j is visited immediately after i.

In addition, there are $O(n^2)$ variables y, defined in Equation (5.9), where y_{ij} represents the load in the truck arriving at customer j after visiting customer i in terms of units of commodity.

The objective function (5.1) minimizes the total distance-based cost of the arcs used by all routes generated. Constraints (5.2) and (5.3) require that each customer is visited exactly once, having one starting edge and one ending edge. Constraint (5.4) imposes that the number of used vehicles does not exceed the number of available vehicles. Constraint (5.5) implies that the number of vehicles leaving the depot is the same as the number of vehicles returning to it.

Constraints (5.6) state that the quantity of products y_{ij} in the truck leaving customer j plus the demand of that customer, equals the quantity of products in the truck leaving it after the service has been completed. Constraints (5.7) guarantee lower and upper bounds ensuring that: the quantity of products y_{ij} in the truck leaving customer i is equals to or greater that its demand, d_i; and the total demand served by each vehicle does not exceed the load capacity Q.

5.2 Literature Review

As described in (Laporte & Nobert, 1987) and also in (Toth & Vigo, 2001), there is a wide range of algorithms for solving the classic CVRP problem –exact algorithms, classical heuristics, and meta-heuristics. However, most techniques have been focused on solving the symmetric CVRP.

It is much less common to find articles that focus exclusively on the asymmetric CVRP. In (Laporte et al., 1986) an exact algorithm is presented using the Branch-and-Bound method where the subproblems are a modified Assignment Problem. (Fischetti et al., 1994) propose a Branch-and-Bound algorithm using a new heuristic, named AV, explicitly tailored for ACVRP, and its practical application to a real case of pharmaceutical distribution in a city of Italy.

In (Vigo, 1996), the author discusses the extension to the ACVRP of two of the most important and successful techniques for the symmetric case: the Savings algorithm of Clarke and Wright (1964), and the optimization method of Fisher and Jaikumar (1981). The author states that the solutions found using the proposed asymmetric version of the Clarke-and-Wright Saving algorithm quickly evolves to worse values as the number of customers increases, in addition to the inconvenience of the parameter combination for the parametric saving function. Furthermore, the author concluded that the new heuristic AV proposed in (Fischetti et al., 1994) is able to bluid a feasible solution better than those that can be obtained with the other algorithms from the literature.

More recently, there are two promising techniques that have been shown to work well in both cases of symmetrical and asymmetrical CVRP. The first is a general heuristic proposed by (Pisinger & Røpke, 2007) which is the result of a unified heuristic for several variants of VRP using the Adaptive Large Neighborhood Search (ALNS). The second is a Memetic Algorithm (MA) described in (Nagata, 2007). These algorithms have been selected by their performance and recognition. We have striven for a balance between

simple classical techniques and also current and state-of-the-art methods.

Moreover, (De Franceschi et al., 2006) presented a ILP-based refinement heuristic for CVRP variants, for both symmetric and asymmetric costs, and for both with/without distance constraints. They extended the refinement procedure proposed by Sarvanov and Doroshko (1981) for the TSP, and a procedure is involved to generate a large number of new sequences through the extracted nodes, as well as a more sophisticated ILP model for the reallocation of some of these sequences. In Pessoa et al. (2008), the authors proposed a robust Branch-Cut-and-Price algorithm for the ACVRP that was also shown to be effective on a number of related problems. In particular, the use of cuts defined over the extended formulation seems very promising and deserves further development.

5.3 The SR-GCWS-CS algorithm adapted for the ACVRP

Constructive Component

The proposed algorithm is based on the randomized Clarke and Wright Savings algorithm (SR-GCWS-CS), developed by Juan et al. (2010); ? (?). The Clarke and Wright Savings heuristic (CWS), presented by Clarke and Wright (1964), is one of the most commonly cited methods in the VRP literature. It uses the concept of savings associated with each arc for merging routes. At each step, the arc with the greatest savings is selected if and only if the two corresponding routes can be combined into a new feasible route and if the selected arc is composed of nodes that are directly connected with the depot. We address the AVRP without considering an extensive asymmetric saving list –i.e., a list including two directed arcs for each pair of customers. Instead we consider a weighted savings list considering just one arc for each pair of customers. Also, we consider the direction of the resulting route after each merging.

Algorithm 10 offers a high-level view of our algorithm, where lines 3, 10 and 11 are essential for asymmetric problems. Our approach starts solving the problem as proposed in the CWS heuristic –i.e.: computing a dummy solution assigning one round-trip route from the depot to each customer. Then the algorithm computes the weighted savings list using an auxiliary parameter β (see below the formulas for weighted saving, i.e., $\hat{S}_{ij}$, and CWS saving, i.e., S_{ij}). At this point the CWS heuristics is combined with a randomization process. We use a pseudo-geometric distribution to assign a selection probability to each edge in the savings list ($alpha$). Moreover, this selection probability is coherent with the weighted saving value associated with each edge, i.e., edges with higher savings will be more likely to be selected from the list than those with lower savings. Therefore, each combination of edges has a chance of being selected and merged with previously built routes. Then, a multi-start process is initiated and controlled by a time parameter ($maxTime$). At each iteration of this process, different edges are selected using the aforementioned biased probability distribution. This allows obtaining different outputs at each iteration. After merging, we improve the merged route applying two promising local search processes. At the end, we apply a general local search to the whole solution which is explained below.

Algorithm 10 SR-GCWS-CS algorithm for the Asymmetric case

Initialization:
  Compute initial dummy solution (Clarke & Wright, 1964).
  Compute weighted *savingsList* using new parameter β.
repeat until $MaxTime$ is consumed.
  Perform a biased randomization of *savingsList* using α (Juan et al., 2010).
  repeat until *savingsList* is empty.
    Extract the next edge from *savingList*.
    if both nodes are exterior, they belong to a two different routes and both routes' load can be assigned to a vehicle (load $\leq Q$).
      Merge routes without considering orientation.
      Apply Asymmetric Local Search and Cache technique.
    end if
  end repeat
  Apply splitting-based local search using *maxSplitter* (Juan et al., 2011).
  Update the best found solution.
end repeat

One important contribution of our approach is the fact that we consider a weighted savings list merging two routes without taking into account directions at this initial stage. See an example in Figure 5.1. The application of a local search will help to define the best direction. The weighted saving associated with an arc connecting customers i and j is defined as:

$$\hat{S}_{ij} = \beta * max\{Sij, Sji\} + (1 - \beta) * min\{Sij, Sji\}$$

where $\beta \in [0.5, 1]$ and $S_{ij} = c_{0i} + c_{0j} - c_{ij}$.

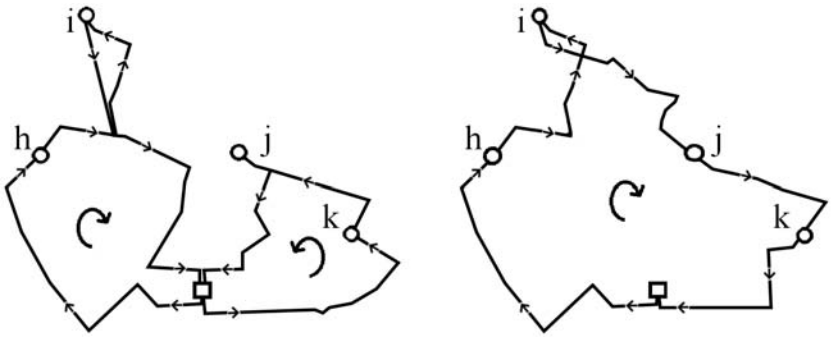


Figure 5.1: Merging routes $R_1 = \{0, h, i, 0\}$ and $R_2 = \{0, k, j, 0\}$ with different orientation into a route $R = \{0, h, i, j, k, 0\}$. The route R_2 is reversed and some edges are changed given that it is an urban area with some one-way streets.

The disregard of orientation is important given that an asymmetric savings list does not choose arcs which do not match their orientation established. For example, it rejects

the routes in Figure 5.1 since they have different orientation. This reduces the solution space and worsens the general solution; even some obtained solutions are using a greater number of vehicles.

Given that the orientations are not considered, an original local search for the asymmetric context was created exploring the near solution space with few steps. It represents another important contribution of our approach.

Asymmetric Local Searches

Once a merged route is obtained, two local searches are applied in order to explore the solution space with few steps. The first local search procedure is the so called Reversing Routes local search. This procedure intends to find an improvement in the order and orientation of the nodes. Given a merged route, we first try to sort the nodes in a more efficient way. If a route is composed by more than four nodes, then we take each four nodes –i.e., (i, j, k, l)– and try to determine if a swapping of two middle-nodes could improve the cost –i.e., (i, k, j, l). After that, we try to reverse the order in which nodes are traversed.

A second local search, originally described in (Juan et al., 2011), is focused on checking if a given set of nodes already exists in a memory but with a better order of the nodes. The basic idea of this learning mechanism is to store in a cache memory the best-known order to travel among the nodes that constitute one route. This cache is constantly updated whenever a better order with a lower cost is found for a given set of nodes. At the same time, the routes contained in this cache are re-used whenever possible to improve newly merged routes.

Finally, once all edges in the saving list have been considered, the resulting solution is improved through a Splitting local search method proposed in (Juan et al., 2011). The current solution is divided into disjoint subsets of routes; then, each of these subsets are solved applying the same methodology described before during a given number of iterations ($maxSplitter$). This tries to apply a "divide and conquer" approach since smaller instances could be easier to solve.

5.4 Computational Results

In order to validate our algorithm, we present the results of 20 AVRP instances comparing our results with those obtained using the Memetic algorithm (MA) of (Nagata, 2007) and the Adaptive Large Neighbourhood Search (ALNS) of (Pisinger & Røpke, 2007).

We have selected 20 public AVRP instances from the workbench presented in the work of Rodríguez and Ruiz (2012b), details on these instances can be found in `http://soa.iti.es/files/Instances_CVRP.7z`. They have been generated with a realistic perspective and mathematical justification. The selection was made at random among the set of medium- and large-size instances (in terms of number of nodes). They have 50 or 100 customers and are designed to employ a homogeneous fleet with the number of vehicles ranging from 2 to 7. These instances consider large demands and vehicle

capacities, as well as random location of the nodes within intra-city areas. The depot may be in the center of the area, those with 'C' in the second letter of the name, or a random position, those with 'A'. The intra-city instances were chosen given that they represent a higher asymmetry degree. Rodríguez and Ruiz (2012b) conclude that these instances affect in a statistically significant way the CPU time needed by some algorithms and deteriorate the quality of the solutions obtained.

The algorithm described in this Chapter has been implemented as a Java application. An Intel QuadCore i5 at 3.2 GHz and 4 GB RAM was used to perform all tests, which were run over Windows XP.

For the 20 AVRP instances, we have used 10 random seeds (10 replicas), an elapsed time of 1 minute ($maxTime$) for each seed, and 60 iterations for the splitting technique ($maxSplitter$). In order to perform a biased randomization of the weighted savings list, a quasi-geometric distribution with a parameter α randomly selected in the interval $[0.5, 0.10]$ was used; and the value chosen for the weighted saving was $\beta = 0.6$.

We have selected the following state-of-the-art AVRP methods in order to compare with:

- General heuristic of (Pisinger & Røpke, 2007). It is a unified heuristic that works for several variants of routing problems and that uses an Adaptive Large Neighborhood Search (ALNS).
- Memetic algorithm of (Nagata, 2007) (MA). Similar to ALNS, MA is a very powerful and recent AVRP metaheuristic.

The previous algorithms have been selected by their performance and recognition. We have striven for a balance between simple classical techniques and current and state-of-the-art methods. Algorithms MA and ALNS were run from the original C++ code which was kindly provided by their respective authors. No code modification was carried out and the methods were run according to their recommendations.

The MA algorithm of Nagata was executed with a parameter setting: $N_{pop} = 100$, $N_{ch} = 30$, 10 trials and 2 parents. Also ten runs with elapsed time of 1 minute were executed for each instance. For the ALNS, only one run was executed for each instance without time limit.

The results of these tests are summarized in table 5.2, which contains the following information for each instance: name of instance; number of nodes; number of vehicles; the best solution of 10 replicas of the MA, {1}; the ALNS solution, {2}; gap, expressed as a percentage value, between {1} and {2}; the time used for ALNS in seconds; our best solution found, OBS {3}; and gap between {1} and {3}.

Instance	# Nodes	# Vehicles	MA {1}	ALNS {2}	gap {1}-{2}	time (s)	OBS {3}	gap {1}-{3}
G-A-CAA0501	50	2	370.26	370.26	0.00 %	17.83	370.26	0.00 %
G-A-CAA0502	50	3	414.44	414.44	0.00 %	13.30	414.44	0.00 %
G-A-CAA0503	50	4	444.69	444.69	0.00 %	10.48	444.69	0.00 %
G-A-CAA0504	50	2	362.01	362.01	0.00 %	17.95	362.01	0.00 %
G-A-CAA0505	50	3	395.78	395.78	0.00 %	15.59	398.47	0.68 %
G-A-CAA1001	100	5	661.88	664.53	0.40 %	43.03	675.31	2.03 %
G-A-CAA1002	100	5	621.06	622.67	0.26 %	39.36	625.82	0.77 %
G-A-CAA1003	100	5	627.29	627.29	0.00 %	42.23	627.29	0.00 %
G-A-CAA1004	100	6	681.89	681.89	0.00 %	34.84	686.25	0.64 %
G-A-CAA1005	100	7	810.97	810.97	0.00 %	29.03	820.56	1.18 %
G-C-CAA0501	50	2	376.62	376.62	0.00 %	17.70	376.62	0.00 %
G-C-CAA0502	50	3	372.48	372.48	0.00 %	13.31	372.48	0.00 %
G-C-CAA0503	50	4	404.30	404.30	0.00 %	10.36	404.30	0.00 %
G-C-CAA0504	50	2	361.74	361.74	0.00 %	17.84	361.74	0.00 %
G-C-CAA0505	50	3	386.73	386.73	0.00 %	13.80	386.73	0.00 %
G-C-CAA1001	100	5	596.54	596.86	0.05 %	40.83	600.35	0.64 %
G-C-CAA1002	100	5	578.15	578.15	0.00 %	38.61	583.39	0.90 %
G-C-CAA1003	100	5	561.08	561.08	0.00 %	41.13	566.10	0.89 %
G-C-CAA1004	100	6	660.81	660.81	0.00 %	35.19	664.14	0.50 %
G-C-CAA1005	100	7	652.08	652.18	0.02 %	28.63	652.42	0.05 %
					0.04 %			0.41 %

Table 5.2: Comparison of results for AVRP instances.

Notice that the results obtained with our approach are quite competitive, showing an average gap of 0.41% with respect to (Nagata, 2007), which always obtains the best results. Our approach also found exactly the same solution (gap = 0.00%) for 10 out of 20 instances. Notice that all three algorithms were run using the same computing time and machine. The only difference is that our algorithm was codified in Java, which runs over a virtual machine and thus has a lower performance than a code implemented in native C/C++.

5.5 Conclusions

In this chapter, we have discussed the importance of taking into account asymmetric costs in realistic Vehicle Routing Problems. Despite the fact that real-life distances are frequently asymmetric –especially in urban transportation–, there is a lack of works considering asymmetric distances. Accordingly, we have presented a hybrid algorithm for solving the Asymmetric Capacitated Vehicle Routing Problem (ACVRP). This algorithm combines a multi-start randomized savings algorithm (SR-GCWS-CS) with two local search processes specifically adapted to the asymmetric nature of costs in real-life scenarios.

A complete set of ACVRP tests have been performed to illustrate the methodology and analyze its efficiency when compared with two state-of-the-art algorithms. The results show that our approach is able to produce competitive results for the ACVRP while, at the same time, it is much simpler to implement and requires less parameters –and fine-tuning efforts– than current state-of-the-art algorithms.

Chapter 6

Asymmetric and Heterogeneous Vehicle Routing Problem

As explained in Chapter 5, the Vehicle Routing Problem (VRP) has clear applications to real-life distribution companies. This chapter focuses on the fact that most road-transportation companies own a heterogeneous fleet of vehicles. However, most VRP-related academic articles assume the existence of a homogeneous fleet of vehicles.

The diversity in the vehicles capacity of companies might be due to the fact that different customers and locations might require different types of vehicles, e.g.: narrow roads in a city, available parking spaces, vehicle weight restrictions on certain roads, etc. Another reason for owning vehicles with distinct capacities is the natural diversity that arises when vehicle acquisitions are made over time. Accordingly, Ruiz et al. (2004) and Prive et al. (2006) state the importance of developing new vehicle routing methods considering heterogeneous fleets. Some real-life applications of heterogeneous fleets are illustrated in (Golden et al., 2002). Examples of these are: urban waste collection, residential pickups and delivery of beverages, food and newspapers delivery, etc.

Several VRP variants have been extensively studied during the last decades (Laporte, 2009). Different VRP variants incorporate different sets of realistic constraints –i. e. vehicle capacity, asymmetric cost, heterogeneous fleets, delivery time windows, service priorities, pickup and delivery options, etc. Despite the fact they are common situations in real-life scenarios, the combination of heterogeneous fleets with asymmetric costs has been rarely discussed in the existing literature.

The first goal of this chapter is to solve the Asymmetric and Heterogeneous Vehicle Routing Problem (AHVRP), for that the proposed methodology presented in Chapter 5 is adapted. Additionally, this chapter also analyzes how asymmetric routing costs vary when slight deviations from the homogeneous fleet are considered, i.e., how marginal costs/savings change when a few 'standard' vehicles in the homogeneous scenario are substituted by other vehicles with different loading capacity.

6.1 Problem Definition

This chapter is focused on a particular variant of the VRP family, called Asymmetric Heterogeneous Fleet Vehicle Routing Problem. The model used next is an adaptation of the models proposed by Baldacci et al. (2008) and Wen et al. (2010). Both assume symmetric and different transportation costs for each type of vehicle, limited number of vehicles, as well as additional fixed costs for using each type of vehicle.

The model of three sub-indexes is formulated as follows. Let $G = (I, E)$ be a complete and directed graph, where $I = \{0, 1, \ldots, n\}$ is the set of nodes and E is the set of arcs. The nodes $i = 1, 2, \ldots, n$ correspond to customers, each with a deterministic demand $d_i \geq 0$. The node $i = 0$ is the zero-demand depot, i.e., $d_0 = 0$. Let $m > 0$ be the number of different types of vehicles. For the k^{th} type of vehicles ($k = 1, 2, \ldots, m$), let $I_k = \{1, 2, \ldots, v_k\}$ be the set of vehicles of type k, with load capacity $Q_k > 0$. Let c_{ij} be the non-negative distance-based cost associated with the arc $(i, j) \in E$, and $c_{ii} = +\infty$, $\forall i \in I$. In the case of calculating the real distances between pairs of locations (i, j), the distance matrix may be asymmetric, i.e., it can happen that $c_{ij} \neq c_{ji}$.

The AHVRP goal is to find the set of minimum-cost round-trip routes, starting from and ending at the depot, which satisfy all customers' demands, visit each customer only once, and do not exceed the load capacity of each type of vehicle.

$$\min \sum_{i \in I} \sum_{j \in I} c_{ij} \sum_{k=1}^{m} x_{ijk} \tag{6.1}$$

subject to:

$$\sum_{i \in I \setminus \{0\}} x_{i0k} = \sum_{j \in I \setminus \{0\}} x_{0jk} \qquad \forall k \in \{1, \ldots, m\} \tag{6.2}$$

$$\sum_{k=1}^{m} \sum_{i \in I \setminus \{j\}} x_{ijk} = 1 \qquad \forall j \in I \setminus \{0\} \tag{6.3}$$

$$\sum_{i \in I \setminus \{h\}} x_{ihk} = \sum_{j \in I \setminus \{h\}} x_{hjk} \qquad \forall h \in I \setminus \{0\}, \forall k \in \{1, \ldots, m\} \tag{6.4}$$

$$\sum_{j \in I \setminus \{0\}} x_{0jk} \leq v_k \qquad \forall k \in \{1, \ldots, m\} \tag{6.5}$$

$$\sum_{i \in I \setminus \{j\}} y_{ij} + d_j = \sum_{i \in I \setminus \{j\}} y_{ji} \qquad \forall j \in I \setminus \{0\} \tag{6.6}$$

$$0 \leq d_i \cdot x_{ijk} \leq y_{ij} \leq (Q_k - d_j) \cdot x_{ijk} \qquad \forall (i, j) \in E, \forall k \in \{1, \ldots, m\} \tag{6.7}$$

$$x_{ijk} \in \{0, 1\} \qquad \forall (i, j) \in E, \forall k \in \{1, \ldots, m\} \tag{6.8}$$

In this formulation, for both symmetrical and asymmetrical issues, and for both homogeneous and heterogeneous fleet, $O(n^2K)$ binary variables x are used. The binary variable x_{ijk} in (6.8) indicates whether or not the arc $(i, j) \in E$ is traveled by a vehicle of type k ($k = 1, 2, \ldots, m$). In addition, there are $O(n^2)$ variables y where y_{ij} represents

the load in the truck arriving at customer j after visiting customer i in terms of units of commodity.

The objective function (6.1) minimizes the total distance-based cost of the arcs used by all m routes generated. Constraints (6.2) imply that the number of vehicles of each type leaving the depot is the same as the number of vehicles of that type returning to it. Constraints (6.3) and (6.4) require that each customer is visited exactly once, and that the same type of vehicle k arrives and leaves each h customer location respectively. Constraints (6.5) impose that the number of used vehicles of each type does not exceed the number of available vehicles of that type. Constraints (6.6) state that the quantity of products y_{ij} in the truck leaving customer j plus the demand of that customer, equals the quantity of products in the truck leaving it after the service has been completed. Constraints (6.7) guarantee lower and upper bounds ensuring that: the quantity of products y_{ij} in the truck leaving customer i is equals to or greater that its demand, d_i; and the total demand served by each vehicle k does not exceed the load capacity Q_k.

6.2 Literature Review

Many combinatorial problems consider more than one fleet of vehicles with different capacities. Since the first axis of our addressed *rich* problem is the asymmetric costs matrix, the second axis is the Heterogeneous Fleet VRP (HVRP). The HVRP has been quite studied in the literature. For instance, Baldacci et al. (2008) present a comprehensive description of some of its most important variants. On this last study, we can appreciate how the research community has addressed the HVRP in different ways. First, considering either a limited or an unlimited number of available vehicles of each type. Second, considering fixed and/or variable costs associated with the use of each type of vehicle. Each vehicle could have a fixed cost for using it in a trip as well as a variable cost that is the result of multiplying a coefficient by the distance of the assigned route, i.e., $c_{ijk_1} \neq c_{ijk_2}\ \forall i, j$ and $\forall k_1 \neq k_2$. These costs used to be associated to each type of vehicle. The combinations of the aforementioned parameters have created the main HVRP families, known as:

- H1: Fleet Size and Mix VRP with fixed and variable costs where an unlimited number of vehicles is considered for minimizing the addition of using a specific vehicle and the variable distance.
- H2: Fleet Size and Mix VRP with only fixed costs where an unlimited number of vehicles is considered for minimizing fixed cost of all used vehicles.
- H3: Fleet Size and Mix VRP with only variable costs where an unlimited number of vehicles is considered for minimizing the variable distance of all routes.
- H4: Heterogeneous Fixed Fleet VRP with only variable costs where a limited number of vehicles is used to minimizing the variable cost.
- H5: Heterogeneous Fixed Fleet VRP with fixed and variable costs where a limited number of vehicles is used to minimizing both variable and fixed costs.

Study	HVRP family	Method
Golden et al. (1984)	H0, H2	Heuristics
Taillard (1999)	H0, F2, H4	Column Generation-based heuristic
Gendreau et al. (1999)	H0, H2, H3	Tabu Search
Wassan and Osman (2002)	H0, H2, H3	Tabu Search
Renaud and Boctor (2002)	H0, H2	Heuristic-based
Lima et al. (2004)	H0, H2	Memetic algorithm
Tarantilis et al. (2004)	H0, H4	Threshold Accepting algorithms
E. Choi and Tcha (2007)	H0, H1, H2, H3	Branch-and-bound
F. Li et al. (2007)	H0, H4	Record-To-Record
Prins (2009)	H0, H1, H2, H3, H4	Memetic algorithm
Imran et al. (2009)	H0, H1, H3	Variable Neighborhood-based heuristic
Liu et al. (2009)	H0, H2	Genetic algorithm
Brandao (2009)	H0, H2, H3	Tabu Search
Euchi and Chabchoub (2010)	H0, H4	Hybrid Tabu Search
X. Li et al. (2010)	H0, H5	Adaptive Memory Programming Metaheuristic
Brandao (2011)	H0, H3, H4	Tabu Search
Penna et al. (2011)	H0, H1, H2, H3, H4, H5	Iterative Local Search with Variable Neighborhood Descent
Subramanian et al. (2012)	H0, H1, H2, H3, H4, H5	Iterative Local Search with Set Partitioning

Table 6.1: Summary of published HVRP studies.

Notice that there are others HVRP branches considering constraints like Site-Dependent, Site-Road, etc. For instance, Prins (2002) considers that each vehicle can optionally perform several trips using a savings-based heuristic but without any extra cost. The basic family (H0) consists in the natural condition of considering a heterogeneous fleet inside of the combinatorial problem.

The Rich VRP (RVRP) is a generalized variant of the VRP where several constraints, aspects or objectives functions are considered at the same time. Examples of constraints and assumptions considered in RVRPs could be: multi-depot, periodic visits to clients, open routes, multi-products, time windows, etc. Drexl (2012) compiles some current needs on the RVRP field as well as a state-of-the-art of scientific research and commercial software. In some studies like Prins (2002); Bolduc et al. (2006); Irnich (2008); Rieck and Zimmermann (2010); Oppen et al. (2010); Prescott-Gagnon et al. (2010); Vallejo et al. (2012) the variable and fixed costs are ignored when combined with other routing features. In fact, Bolduc et al. (2006) is included with other papers in a Special Issue of Rich VRP Hartl et al. (2006). Tables 6.1 and 6.2 summarize the work done in several types of HVRP. On its columns, we can appreciate the addressed HVRP families, the

applied methodology, as well as others routing constraints considered on each study. Also, the first table (6.1) presents papers exclusively dedicated to the HVRP, while the second table (6.2) shows papers related to RVRP where the HVRP is also considered together with other constraints. Most of these works just include the heterogeneous capacity of vehicles (H0) and not the related costs associated with each type of vehicle. Therefore, there is not a single way to include the HVRP feature in a RVRP.

To the best of our knowledge, there is not any published work on the AHVRP. The most approximated ones are presented in Marmion et al. (2010); Pessoa et al. (2008). In the first study, the authors analyses the sensitivity of two classical neighborhoods methods for the AHVRP. Thus, they simulate a heterogeneous fleet assigning different variable costs to each vehicle, but the capacity remains unchanged. On Pessoa et al. (2008), the authors developed a set of robust Branch-Cut-and-Price algorithms for several VRPs. Some promising experiments are presented, but changes in the capacity of fleets are not justified for the HVRP. More details about these two studies are discussed later.

Study	HVRP family	Method	Extra constraints
Bolduc et al. (2006)	H0	Heuristics	Multi-period
Irnich (2008)	H0	Local Search-Based metaheuristic	Multi-depots, and time windows
Baldacci and Mingozzi (2009)	H0, H5	MIP model with Set Partitioning	Site-dependent, and multi-depots
Rieck and Zimmermann (2010)	H0	MILP model	Time windows, and simultaneous delivery & pickup
Oppen et al. (2010)	H0	Column Generation-based	Multi-products, multiple trips, precedence and inventory constraints
Prescott-Gagnon et al. (2010)	H0	Heuristics	Multi-depots, intra-route replenishments, time windows, driver shifts and optional customers
Vallejo et al. (2012)	H0	Memory-based Approach with Clustering	Time windows, multi-depots, and multi-trip

Table 6.2: Summary of published Rich HVRP studies.

6.3 The SR-GCWS-CS algorithm adapted for the AHVRP

The algorithm we propose is an adaptation of our proposed Asymmetric SR-GCWS-CS algorithm presented in Chapter 5. The Clarke and Wright Savings heuristic (CWS) is a well known method (Clarke & Wright, 1964). Our approach combines the randomized Clarke and Wright Savings algorithm (SR-GCWS-CS), developed by Juan et al. (2010, 2011) for the CVRP, with the modified Clarke and Wright Savings heuristic, presented by Prins (2002) for the HVRP.

Constructive Component

Algorithm 11 offers a high-level view of our algorithm. Being demonstrated the competence of our algorithm 10 for the asymmetric case, for the AHVRP only lines 8–11 and 13 are modified.

Our approach starts solving the problem as proposed in the CWS heuristic, i.e., computing a dummy solution assigning one round-trip route from the depot to each customer. Then the algorithm computes the weighted savings list as proposed in Chapter 5. At this point, the CWS heuristics is combined with the randomization process proposed in (Juan et al., 2010). Then, a multi-start process is initiated and controlled by a time parameter ($maxTime$). At each iteration of this process, different edges are selected using a biased probability distribution, this allows obtaining different outputs at each iteration.

Addressing the heterogeneous fleet

Before merging, the validation of the capacity constraint in a heterogeneous fleet is addressed as a vehicle-trip assignment. For this, an effective method based on CWS is employed, (Prins, 2002). In the work of Prins, the list of vehicles and the list of routes are sorted decreasingly by load capacity and accumulated demands respectively; after that, a temporary assignment between the two lists is searched. If a successful match –including all previously routes plus the new desirable merged one– is found, then the capacity constraint is satisfied and the temporary assignment becomes final. Otherwise, the merge becomes unfeasible. If a situation arises in which the number of routes is greater than the number of vehicles, then new fictitious vehicles are assigned to the remaining routes. Notice that this vehicle assignment validation is made for each possible saving, increasing the computational operations. However, in our approach it is not considered the assumption regarding that the largest demand cannot exceed the capacity of the smallest vehicle.

After merging the routes and assigning vehicles, the Asymmetric Local Searches explained in Section 5.3 is applied. Notice that it does not search a new vehicle assignment, the previously assigned vehicle to each route remains unchanged during this process.

Finally, once all edges in the saving list have been considered, the resulting solution is improved through a Splitting local search method proposed in (Juan et al., 2011). The current solution is divided into disjoint subsets of routes together with their previously

Algorithm 11 The Asymmetric SR-GCWS-CS for Heterogeneous Fleets

```
1: Initialization:
2:   Compute initial dummy solution (Clarke & Wright, 1964).
3:   Compute weighted savingsList using β.
4: repeat until MaxTime is consumed.
5:   Perform a biased randomization of savingsList using α (Juan et al., 2010).
6:   repeat until savingsList is empty.
7:     Extract the next edge from savingList.
8:     if both nodes are exterior and they belong to a two different routes.
9:       Compute decreasing sorted list of vehicles and decreasing sorted list of
10:      routes loads (Prins, 2002).
11:      if all routes' load can be assigned to a candidate vehicle (load ≤ Q_k).
12:        Merge routes without considering orientation.
13:        Assign final vehicles.
14:        Apply Asymmetric Local Search and Cache technique.
15:      end if
16:    end if
17:  end repeat
18:  Apply splitting-based local search using maxSplitter (Juan et al., 2011).
19:  Update the best found solution.
20: end repeat
```

assigned vehicles; then, each of these subsets are solved applying the same methodology described before during a given number of iterations (*maxSplitter*). This tries to apply a "divide and conquer" approach since smaller instances could be easier to solve. So a new set of routes could be created on each partition with the previously assigned vehicles.

6.4 Experimental Design

The most commonly-used methodology to compare the performance of different algorithms for solving VRPs consists in running these algorithms over a set of well-defined benchmark instances. In the case of the CVRP or the AVRP, several benchmark sets are available through open-access websites, so that researchers worldwide can use them. Usually, these data sets contain complete information, including not just the instance inputs and the best-known value for the objective function, but also a complete description of the corresponding solution –i.e., the specific composition of each route in the best-known solution. In the case of the AHVRP, however, there is not a commonly-accepted set of instances to test algorithms, since the AHVRP has rarely been discussed so far in the literature.

For the AHVRP, (Fischetti et al., 1994) developed some preliminary experiments based on a set of real AVRP instances. Likewise, (Marmion et al., 2010) analyzed the use of a heterogeneous fleet. However, these studies have only considered the effect of

variable cost on vehicles selection, but no differences in actual vehicle capacities are considered. Also (Pessoa et al., 2008) have used these benchmarks first modifying the capacity of the original fleets and then running the experiments with homogeneous fleets.

Therefore, we propose to extend, for the AHVRP, the 20 selected ACVRP instances from http://soa.iti.es/files/Instances_CVRP.7z, used in Chapter 5. We propose to use exactly the same nodes of this ACVRP instances, including their asymmetric costs and demands, and the same number of vehicles. We then consider a heterogeneous fleet composed of 'standard' vehicles –i.e., vehicles with the capacity defined in the ACVRP instances– and 'non-standard' vehicles with modified capacities. In our opinion, this is a natural way to adapt the homogeneous-capacity instances, since it allows the decision-maker to answer sensitivity-analysis questions such as: "How would my routing costs be changed if we could employ one or two trucks with a different capacity?".

Thus, in order to test our approach, a total of twenty classical ACVRP instances were selected and adapted as 'base' AHVRP instances. For each base instance, six different fleet typologies were defined –thus, 120 different instances were considered in total. These fleet typologies are partially composed of 'standard' vehicles, each of them with capacity Q_0, but they differ in their exact composition as explained in the following general rule:

- Fleet 150-125: two 'standard' trucks are substituted by a 'large' truck (with capacity $Q_l = 150\% \cdot Q_0$) and by a 'large-medium' truck (with capacity $Q_{lm} = 125\% \cdot Q_0$), respectively.
- Fleet 125-125: two 'standard' trucks are substituted by two 'large-medium' trucks.
- Fleet 125-80: two 'standard' trucks are substituted by a 'large-medium' truck and by a 'small' truck (with capacity $Q_s = 80\% \cdot Q_0$), respectively.
- Fleet 100-100: the homogeneous case where all trucks are 'standard'.
- Fleet 90-90: two 'standard' trucks are substituted by two 'small-medium' trucks (with capacity $Q_{sm} = 90\% \cdot Q_0$).
- Fleet 90-80: two 'standard' trucks are substituted by a 'small-medium' truck and by a 'small' truck, respectively.

Notice, however, that in some cases a reduction in the fleet capacity might cause the infeasibility of the problem, i.e., the total demand to be satisfied might be greater than the total fleet capacity. In those particular cases, an additional 'standard' vehicle is added to the fleet to promote feasibility of the problem.

6.5 Computational Results

Previously the proposed approach has shown to be competitive for the AVRP, see section 5.4, now it is used for solving the AHVRP, and analyze different fleet compositions. Thus, twenty ACVRP instances were adapted as 'base' AHVRP instances. For each base instance, six different fleet typologies were defined, see section 6.4 for more details.

Instance	# Nodes	Homogeneous Case		Heterogeneous Case			
		# Veh	OBS {1}	Fleet Rule (%)	OBS {2}	number of (veh1, veh2, standard)	gap {1}-{2}
G-A-CAA0501	50	2	370.257	150-125	368.829	(1,1,0)	-0.39 %
				125-125	368.829	(1,1,0)	-0.39 %
				125-80	378.112	(1,1,0)	2.12 %
				90-90	384.199	(1,1,1)*	3.77 %
				90-80	388.652	(1,1,1)*	4.97 %
G-A-CAA0502	50	3	414.438	150-125	372.184	*(1,1,0)	-10.20 %
				125-125	383.219	*(1,1,0)	-7.53 %
				125-80	398.927	(1,1,1)	-3.74 %
				90-90	414.438	(1,1,1)	0.00 %
				90-80	414.438	(1,1,1)	0.00 %
G-A-CAA0503	50	4	444.688	150-125	404.259	*(1,1,1)	-9.09 %
				125-125	426.958	(1,1,2)	-3.99 %
				125-80	432.413	(1,1,2)	-2.76 %
				90-90	452.598	(1,1,2)	1.78 %
				90-80	459.583	(1,1,2)	3.35 %
G-A-CAA0504	50	2	362.011	150-125	363.537	(1,1,0)	0.42 %
				125-125	359.153	(1,1,0)	-0.79 %
				125-80	360.206	(1,1,0)	-0.50 %
				90-90	377.796	(1,1,1)*	4.36 %
				90-80	379.382	(1,1,1)*	4.80 %
G-A-CAA0505	50	3	398.471	150-125	378.349	*(1,1,0)	-5.05 %
				125-125	380.181	*(1,1,0)	-4.59 %
				125-80	382.431	*(1,1,0)	-4.03 %
				90-90	404.163	(1,1,1)	1.43 %
				90-80	402.230	(1,1,1)	0.94 %

Table 6.3: Experimental results with different fleet configurations for small-size instances using random locations for nodes.

Instance	# Nodes	Homogeneous Case		Heterogeneous Case			
		# Veh	OBS {1}	Fleet Rule (%)	OBS {2}	number of (veh1, veh2, standard)	gap {1}-{2}
G-C-CAA0501	50	2	376.618	150-125	367.698	(1,1,0)	-2.37 %
				125-125	367.698	(1,1,0)	-2.37 %
				125-80	367.698	(1,1,0)	-2.37 %
				90-90	384.928	(1,1,1)*	2.21 %
				90-80	384.928	(1,1,1)*	2.21 %
G-C-CAA0502	50	3	372.479	150-125	358.013	*(1,1,0)	-3.88 %
				125-125	359.318	*(1,1,0)	-3.53 %
				125-80	372.479	(1,1,1)	0.00 %
				90-90	372.479	(1,1,1)	0.00 %
				90-80	372.479	(1,1,1)	0.00 %
G-C-CAA0503	50	4	404.302	150-125	379.883	*(1,1,1)	-6.04 %
				125-125	397.431	(1,1,2)	-1.70 %
				125-80	398.024	(1,1,2)	-1.55 %
				90-90	405.117	(1,1,2)	0.20 %
				90-80	414.637	(1,1,3)*	2.56 %
G-C-CAA0504	50	2	361.741	150-125	356.353	(1,1,0)	-1.49 %
				125-125	357.220	(1,1,0)	-1.25 %
				125-80	358.817	(1,1,0)	-0.81 %
				90-90	381.993	(1,1,1)*	5.60 %
				90-80	381.993	(1,1,1)*	5.60 %
G-C-CAA0505	50	3	386.727	150-125	374.048	*(1,1,0)	-3.28 %
				125-125	374.048	*(1,1,0)	-3.28 %
				125-80	374.048	*(1,1,0)	-3.28 %
				90-90	386.727	(1,1,1)	0.00 %
				90-80	386.727	(1,1,1)	0.00 %

Table 6.4: Experimental results with different fleet configurations for small-size instances using grid locations for nodes.

Instance	# Nodes	Homogeneous Case		Heterogeneous Case			
		# Veh	OBS {1}	Fleet Rule (%)	OBS {2}	number of (veh1, veh2, standard)	gap {1}-{2}
G-A-CAA1001	100	5	675.935	150-125	634.141	*(1,1,2)	-6.18 %
				125-125	634.746	*(1,1,2)	-6.09 %
				125-80	649.345	*(1,1,2)	-3.93 %
				90-90	683.221	(1,1,3)	1.08 %
				90-80	683.221	(1,1,3)	1.08 %
G-A-CAA1002	100	5	625.820	150-125	583.819	*(1,1,2)	-6.71 %
				125-125	603.274	*(1,1,2)	-3.60 %
				125-80	612.554	(1,1,3)	-2.12 %
				90-90	626.279	(1,1,3)	0.07 %
				90-80	626.280	(1,1,3)	0.07 %
G-A-CAA1003	100	5	628.596	150-125	605.825	*(1,1,2)	-3.62 %
				125-125	620.880	*(1,1,2)	-1.23 %
				125-80	622.516	*(1,1,2)	-0.97 %
				90-90	643.889	(1,1,3)	2.43 %
				90-80	643.084	(1,1,3)	2.30 %
G-A-CAA1004	100	6	686.254	150-125	654.072	*(1,1,3)	-4.69 %
				125-125	659.945	*(1,1,3)	-3.83 %
				125-80	664.556	*(1,1,3)	-3.16 %
				90-90	689.191	(1,1,4)	0.43 %
				90-80	694.969	(1,1,4)	1.27 %
G-A-CAA1005	100	7	821.511	150-125	776.321	(1,1,5)	-5.50 %
				125-125	799.071	(1,1,5)	-2.73 %
				125-80	814.771	(1,1,5)	-0.82 %
				90-90	846.249	(1,1,6)*	3.01 %
				90-80	847.796	(1,1,6)*	3.20 %

Table 6.5: Experimental results with different fleet configurations for medium-size instances using random locations for nodes.

Instance	# Nodes	Homogeneous Case		Heterogeneous Case			
		# Veh	OBS {1}	Fleet Rule (%)	OBS {2}	number of (veh1, veh2, standard)	gap {1}-{2}
G-C-CAA1001	100	5	600.344	150-125	585.121	*(1,1,2)	-2.54 %
				125-125	586.221	*(1,1,2)	-2.35 %
				125-80	588.736	*(1,1,2)	-1.93 %
				90-90	604.696	(1,1,3)	0.72 %
				90-80	604.244	(1,1,3)	0.65 %
G-C-CAA1002	100	5	585.031	150-125	562.806	*(1,1,2)	-3.80 %
				125-125	562.656	*(1,1,2)	-3.82 %
				125-80	579.052	(1,1,3)	-1.02 %
				90-90	588.200	(1,1,3)	0.54 %
				90-80	587.204	(1,1,3)	0.37 %
G-C-CAA1003	100	5	566.005	150-125	543.784	*(1,1,2)	-3.93 %
				125-125	549.958	*(1,1,2)	-2.84 %
				125-80	552.630	*(1,1,2)	-2.36 %
				90-90	565.012	(1,1,3)	-0.18 %
				90-80	567.901	(1,1,3)	0.33 %
G-C-CAA1004	100	6	665.511	150-125	633.026	*(1,1,3)	-4.88 %
				125-125	635.761	*(1,1,3)	-4.47 %
				125-80	647.009	*(1,1,3)	-2.78 %
				90-90	667.831	(1,1,4)	0.35 %
				90-80	672.168	(1,1,4)	1.00 %
G-C-CAA1005	100	7	652.424	150-125	635.248	(1,1,5)	-2.63 %
				125-125	640.440	(1,1,5)	-1.84 %
				125-80	648.603	(1,1,5)	-0.59 %
				90-90	668.156	(1,1,6)*	2.41 %
				90-80	672.150	(1,1,6)*	3.02 %

Table 6.6: Experimental results with different fleet configurations for medium-size instances using grid locations for nodes.

Thus, 120 different instances were considered in total. For each of them, 10 random seeds (10 replicas) and an elapsed time of 1 minute were used.

The algorithm described in this chapter has been implemented as a Java application. An Intel QuadCore i5 at 3.2 GHz and 4 GB RAM was used to perform all tests, which were run over Windows XP.

Tables 6.3, 6.4, 6.5 and 6.6 contain, for each base instance, the following information: name of instance; number of nodes; number of vehicles; our best solution found for the homogeneous case, OBS {1}; different fleet rules for the heterogeneous case, each of them defining a new routing instance; our best solution found for the heterogeneous case, OBS {3}; the obtained number of vehicles for the fleet configuration; and percentage gap between the BKS for the homogeneous case and the OBS for the heterogeneous case. Instances are distributed in the tables according to their sizes (50 or 100 nodes) and the location of the nodes on territory (random or grid).

Observe that * highlights different number of vehicles. For example, on table 6.3, (1,1,1)* of the fifth row remarks that this heterogeneous solution is using one more vehicle than the homogeneous solution. It uses one vehicle of 90% of capacity, one vehicle of 80% and one 'standard' vehicle. Instead, *(1,1,0) of the sixth row remarks that this solution is using one less 'standard' vehicle.

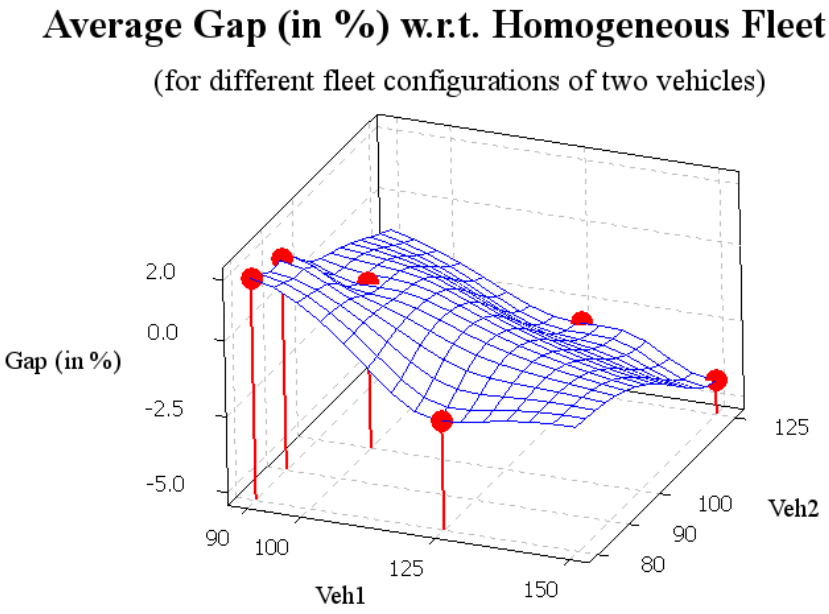


Figure 6.7: Surface Plot of Average Gap vs. Fleet Configuration

Figure 6.7 shows a 3D scatter plot representing the average gap associated with each of the 6 fleet configurations considered in this article. In other words, for each fleet rule, the twenty gaps with respect to the homogeneous OBS –one per base instance– have been averaged. From these results, it can be noticed the following:

- Just by employing two large vehicles (fleet 150-125) instead of two 'standard' ve-

hicles (fleet 100-100), it is possible to obtain noticeable costs reductions that can go up to 10% in some instances (e.g., G-A-CAA0502).

- Likewise, when using two small vehicles (fleet 90-80) instead of two 'standard' vehicles, costs can suffer an increase of about 5% for some instances (e.g., G-C-CAA0504).

Therefore, it can be concluded that routing costs can be in fact quite sensitive to small variations in the fleet configuration. This justifies the necessity for employing new approaches in real-life routing applications, i.e., algorithms which are able to deal with both asymmetric costs as well as heterogeneous fleets.

6.6 Conclusions

In this chapter, we have discussed the importance of taking into account heterogeneous fleets in realistic Vehicle Routing Problems. Despite the fact that most real-life fleets of vehicles are heterogeneous and that real-life distances are frequently asymmetric – especially in urban transportation–, there is a lack of works considering both situations simultaneously. Accordingly, we have presented a multi-start process for solving the Asymmetric Heterogeneous Vehicle Routing Problem. This algorithm combines a randomized savings algorithm (SR-GCWS-CS) with three local search processes specifically adapted to the heterogeneity of fleets in real-life scenarios.

A set of benchmarks for the AHVRP have been developed and a sensitivity analysis on the fleet composition has been performed. This experiment shows how decision-makers can benefit from our approach when deciding the actual composition of their heterogeneous fleets.

Contributions

Main contributions

The thesis has been introduced hybrid methodologies that integrates several techniques to efficiently tackle six combinatorial problems related to road transport. It starts with theoretical problems and evolves into more realistic scenarios, a study of different variants focusing on the impact that causes the asymmetry of the costs and the heterogeneity of the fleet has been presented, and a real problem of Home Health Care service in the city of Ferrara, Italy, has been solved. Below detailed contributions are presented for each chapter.

Our proposed Tailored Lagrangian Metaheuristic (TLM) method has been presented in Chapter 1, this metaheuristic method is based on the Lagrangian Relaxation technique. This scheme has been used to tackle the Traveling Salesman Problem (TSP), showing very good results both in terms of the quality of the solution and in terms of computational efficiency.

Our proposed TLM method uses the Subgradient algorithm combined with a heuristic. The motivation of our method is the common sense belief that dual solutions must obviously be relevant to primal solutions. The new introduced parameter has improved significantly algorithm's convergence on the optimal solution.

Our proposed TLM has been included into a Variable Neighborhood Search framework to tackle the Capacitated Vehicle Routing Problem (CVRP), in Chapter 2.

The presented hybrid methodology combines a randomized version of the Clarke and Wright Savings heuristic with Constraint Programming and our Tailored Lagrangian Metaheuristic to efficiently solve CVRP instances. These techniques have been embedded into a Multi-Start Variable Neighborhood Descent framework. According to the tests performed, the proposed algorithm is competitive with state-of-the-art metaheuristics.

Our proposed TLM has been used to calculate routing cost for each vehicle separately. It solves all routes in negligible times and is an efficient alternative for intra-route optimization processes and avoids defining intra-route moves. Moreover, it ensures the partial optimality of most solutions from the routing perspective. The reason is that, since we are considering a relatively small number of customers per route, the proposed approach can quickly find the optimal solution to most TSP instances.

Chapter 3 presents a real-life application of constraint solving technologies to the Home Health Care problem for the city of Ferrara, Italy. Our proposed Tailored La-

grangian Metaheuristic has been included into a Constraint Programming framework using a search algorithm based on Large Neighborhood Search.

Our proposed TLM has been used to calculate traveling time for each nurse separately. It has been compared respect ASP and Comet technologies, obtaining that our proposed TLM was the fastest, it ensures the partial optimality of most solutions from the routing perspective. The reason is that, since there are a relatively small number of patients per nurse, the proposed approach can quickly find the optimal solution to most TSP instances.

An adaptation of our TLM for the Asymmetric Traveling Salesman Problem (ATSP) is presented in Chapter 4. For this problem, the out-degree constraints are relaxed and Edmond's Algorithm is used to solve the dual problem. In this metaheuristic approach, a new heuristic to obtain a feasible solution from a dual one is proposed.

The results have shown that the new parameter has improved significantly Subgradient's convergence on the optimal solution and it is showing good results in terms of the quality of the results. However, it is not showing a good behavior in terms of computational efficiency.

A hybrid algorithm for solving the Asymmetric Capacitated Vehicle Routing Problem (ACVRP) has been presented in Chapter 5. The proposed algorithm is based on the randomized Clarke and Wright Savings algorithm (SR-GCWS), combined with two local search processes specifically adapted to the asymmetric nature of costs in real-life scenarios.

The results show that our approach is able to produce competitive results for the ACVRP while, at the same time, it is much simpler to implement and requires less parameters –and fine-tuning efforts– than current state-of-the-art algorithms.

In Chapter 6, the proposed asymmetric adaptation of the SR-GCWS algorithm, presented in Chapter 5, has been modified to tackle the Asymmetric Heterogeneous Vehicle Routing Problem (AHVRP). Our approach combines the SR-GCWS, developed by Juan et al. (2010) for the CVRP, with the modified Clarke and Wright Savings heuristic, presented by Prins (2002) for the HVRP.

A set of benchmarks for the AHVRP have been developed and a sensitivity analysis on the fleet composition has been performed. This experiment shows how decision makers can benefit from our approach when deciding the actual composition of their heterogeneous fleets.

Finally, the hybrid VNS methodology described in Chapter 2 has been registered with the following software license:

- D. Guimarans, R. Herrero, J.J. Ramos, and M.A. Piera. *ITSLogisim Simulation Suite 1.0* and *ITSLogiSim Optimization Suite 1.0*. Registered at *Universitat Autnoma de Barcelona*, 30th June 2010.

Currently, the company *Digital Aeronautics Engineering Services* is using this software license under a commercial agreement. Combined with a clustering algorithm, it is being applied to solve problems up to 22,000 customers and more than 5,000 planned routes.

Conclusions

The conclusions of this thesis can be summarized as follows:

- The Tailored Lagrangian Metaheuristic has been developed to tackle the Traveling Salesman Problem, it is based on the Lagrangian Relaxation.
- This metaheuristic is competitive in terms of both results and computational efficiency for the symmetric scenarios. However, the impact of the asymmetry of the cost in our metaheuristic has been studied, its adaptation for the asymmetric scenarios is only showing good results in terms of quality, but it is not showing a good behavior in terms of computational efficiency.
- Our proposed metaheuristic has been integrated into a hybrid methodology combine with Constraint Programming to tackle the Capacitated Vehicle Routing Problem. Our metaheuristic solves optimally all routes in negligible times.
- An application using our metaheuristic has been developed for a real scenario, the Home Health Care service in Ferrara, Italy. Our proposed metaheuristic, which solves optimally all routes consuming 10% of the computation time, has been included into a Constraint Programming framework using a search algorithm based on Large Neighborhood Search.
- The impact of the asymmetry of the cost has been studied tackling the Asymmetric version of the Capacitated Vehicle Routing Problem. The randomized Clarke and Wright Savings algorithm has been adapted combined with two local search processes specifically designed for the asymmetric nature of costs in real-life scenarios. Unlike other approaches, this adaptation has been flexible and it is competitive with state-of-the-art metaheuristics.
- This Clarke and Wright approach has been adapted to consider Heterogeneous fleets. A study of the impact that causes the heterogeneity of the fleet has been presented showing that different fleet configurations can need one more or less vehicle than the homogeneous solution, and the total cost can be reduced up to 10% or increase up to 5% for different fleet configurations.

Future Research

As it can be appreciated in Chapter 4, the proposed dual problem for the ATSP has not been able to produce results quickly, in part derived from the running time of Edmonds' algorithm. In fact, the research of the scientific community is more focused on the Lagrangian Relaxation for the ATSP based on the Assignment Problem. Adapt our Tailored Lagrangian Metaheuristic to this relaxation could be quite interesting to develop as future work.

Furthermore, it can be developed a parallel implementation of the algorithm presented in Chapter 5–6, aiming to significantly accelerate its execution speed when dealing with large-scale instances. In this context, the splitting process can be designed to run different splitting policies in a parallel computing environment sharing a centralized hash table among all threads. In addition, the entire multi-start process can be designed to run in a parallel computing environment using several distributed hash tables which communicate among themselves to share information at certain time periods.

As for future work, the following lines of research are suggested to adapt the SR-GCWS-CS to solve some variants of VRP:

- **VRP with Time Windows** Each customer is associated with a time interval and can only be served within this interval. When the next edge is selected from savingList, it must be verified whether the customer is served within its time interval before merging routes. It would be appropriate to use the well-known Insertion heuristic during the local search process.

- **Multi-Depot VRP** It integrates a combinatorial assignment problem -which customers are to be assigned to each depot- with the several CVRPs that must be solved for each customers-depot assignment, this problem is less studied variant in the literature. Before starting the current SR-GCWS-CS algorithm, each customer could be randomly assigned to a depot according to a distance-based criterion. Thus, each VRP could be solved independently using the SR-GWS algorithm, it could be implemented in parallel using independent hash tables and without considering the splitting process. Then, a new splitting-based local search could be applied to all VRPs.

- **Pickup-and-delivery VRP** Each customer is associated by two quantities, representing one demand to be delivered at the customer and another demand to be picked up and returned to the depot. When the next edge is selected from the saving list, it must be verified whether both quantities can be satisfied while the total pickup and total delivery on the merged route are not exceeding the vehicle capacity.

- **Green VRP** This variant deals with the optimization of energy consumption of transportation which may clash with the designated economic objectives. It takes into account the vehicles fuel tank capacity limitation and chooses the optimal placement of Alternative Fueling Stations visits within the tour. When the next edge is selected from the saving list, it must be verified the fuel tank capacity, if it is necessary to visit a fuel station, the routes will not merged. However, the cost of the edge could be modified in the saving list adding the cost of inserting the nearest fuel station to the resultant route.

- **Fuel Consumption Rate VRP** New objective function is considered in which the cost function is a product of the total load (including the weight of the empty vehicle) and the distance traveled minimizing fuel consumption. For this problem,

the saving list should be reordered, it would be rather select the edge of maximum saving related to the least loaded vehicles.

Publications

The work presented in this thesis has been partially published in the following journal articles:

- Rosa Herrero, Alejandro Rodríguez, José Cáceres-Cruz, and Angel Juan (2014): Solving vehicle routing problems with asymmetric costs and heterogeneous fleets. *International Journal of Advanced Operations Management*, vol. 6(1), pp. 58-80. Inderscience Enterprises Ltd. DOI: 10.1504/IJAOM.2014.059620.

- Massimiliano Cattafi, Rosa Herrero, Marco Gavanelli, Maddalena Nonato, and Federico Malucelli (2015): An Application of Constraint Solving for Home Health Care. *AI Communications*, vol. 28(2), pp. 215-237, indexed in IOS Press (IF 2015: 0.547) ISSN: 0921-7126.

- Daniel Guimarans, Rosa Herrero, Daniel Riera, Angel Juan, and Juan José Ramos (2011): Combining probabilistic algorithms, Constraint Programming and Lagrangian Relaxation to solve the Vehicle Routing Problem. *Annals of Mathematics and Artificial Intelligence*, vol. 62(3), pp. 299-315, indexed in ISI SCI (FI = 0.358, 4Q). ISSN: 1012-2443. DOI: 10.1007/s10472-011-9261-y.

- Daniel Guimarans, Rosa Herrero, Juan José Ramos, and Silvia Padrón (2011): Solving Vehicle Routing Problems Using Constraint Programming and Lagrangian Relaxation in a Metaheuristics Framework. *International Journal of Information Systems and Supply Chain Management (IJISSCM), Special Issue: Hybrid Algorithms for Solving Realistic Routing, Scheduling and Availability Problems*, vol. 4(2), pp. 61-81. ISSN: 1935-5726. DOI: 10.4018/jisscm.2011040104.

Some parts of this work have also been presented in several international conferences and published in the following related articles:

- Rosa Herrero, Juan José Ramos, and Daniel Guimarans (2009): Solving the Travelling Salesman Problem with Time Windows by Lagrangian Relaxation. In proceeding of *21st European Modelling and Simulation Symposium (EMSS)*. Tenerife, Spain. September, 2009.

- Rosa Herrero, Juan José Ramos, and Daniel Guimarans (2010): Lagrangian Metaheuristic for the Travelling Salesman Problem. In proceedings of *Operational Research Annual Conference (OR 52)*. Surrey, UK. September, 2010.

- Rosa Herrero, Daniel Guimarans, Juan José Ramos, and Silvia Padrón (2010): A Variable Neighbourhood Search Combining Constraint Programming and Lagrangian Relaxation for Solving Routing Problems. In proceedings of *2010 Summer Computer Simulation Conference (SCSC'10)*. Ottawa, Canada. July, 2010.

- Daniel Guimarans, Rosa Herrero, Daniel Riera, Angel Juan, and Juan José Ramos (2010): Combining Constraint Programming, Lagrangian Relaxation and Probabilistic Algorithms to solve the Vehicle Routing Problem. In proceedings of *17th International RCRA workshop (RCRA 2010): Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion.* Bologna, Italy. June, 2010. ISBN: 1613-0073. http://ceur-ws.org/Vol-616/paper04.pdf

- Massimiliano Cattafi, Rosa Herrero, Marco Gavanelli, Maddalena Nonato, Federico Malucelli, and Juan-José Ramos (2012): Improving quality and efficiency in home health care: an application of constraint logic programming for the Ferrara NHS unit. In Agostino Dovier and Vítor Santos Costa, editors, *Technical Communications of the 28th International Conference on Logic Programming (ICLP'12), volume 17 of Leibniz International Proceedings in Informatics (LIPICS)*, pp. 415-424, Dagstuhl, Germany, 2012. ISSN: 1868-8969. DOI: 10.4230/LIPIcs.ICLP.2012.415

- Massimiliano Cattafi, Rosa Herrero, Marco Gavanelli, Maddalena Nonato, Federico Malucelli and Juan José Ramos (2012): An Application of Constraint Solving for Home Health Care. In proceedings of *19th RCRA workshop on Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion. In association with the 12-th AI*IA Symposium on Artificial Intelligence.* Rome, Italy, 14-15 June 2012.

- José Cáceres-Cruz, Daniel Riera, Roman Buil, Angel Juan, and Rosa Herrero (2013): Multi-start Approach for Solving an Asymmetric Heterogeneous Vehicle Routing Problem in a Real Urban Context. In proceedings of *2nd International Conference on Operations Research and Enterprise Systems (INSTICC-ICORES)* February, 2013.

Copy of scientific publications, references and any other additional information will be provided upon request (Email: RHerrero.Math@gmail.com).

References

Alba, E. (2005). *Parallel metaheuristics: a new class of algorithms* (Vol. 47). John Wiley & Sons.

Alba, E., Blum, C., Asasi, P., Leon, C., & Gomez, J. A. (2009). *Optimization techniques for solving complex problems* (Vol. 76). John Wiley & Sons.

Alba, E., & Dorronsoro, B. (2008). A hybrid cellular genetic algorithm for the capacitated vehicle routing problem. In A. Abraham, C. Grosan, & W. Pedrycz (Eds.), *Engineering Evolutionary Intelligent Systems (Studies in Computational Intelligence, 82)* (pp. 379–422). Springer Berlin Heidelberg.

Alba, E., Luque, G., & Nesmachnow, S. (2013). Parallel metaheuristics: recent advances and new trends. *International Transactions in Operational Research*, *20*(1), 1–48.

Altinel, I., & Oncan, T. (2005). A new enhancement of the Clarke and Wright savings heuristic for the capacitated vehicle routing problem. *Journal of the Operational Research Society*, *56*, 954–961.

Andziulis, A., Dzemydienė, D., Steponavičius, R., & Jakovlev, S. (2011). Comparison of two heuristic approaches for solving the production scheduling problem. *Information Technology And Control*, *40*(2), 118–122.

Applegate, D., Archer, A., Gopalakrishnan, V., Lee, S., & Ramakrishnan, K. K. (2010). Optimal Content Placement for a Large-scale VoD System. In *Proceedings of the 6th International Conference Co-NEXT '10* (pp. 4:1–12). ACM.

Applegate, D., Bixby, R., Chvátal, V., & Cook, W. (2006). *The Traveling Salesman Problem: A Computational Study*. Princeton University Press.

Applegate, D., Bixby, R., Chvátal, V., & Cook, W. (2015, August). *Concorde*. Retrieved from www.tsp.gatech.edu

Apt, K., & Wallace, M. (2007). *Constraint Logic Programming using ECLiPSe*. Cambridge University Press.

Asadpour, A., Goemans, M. X., Madry, A., Gharan, S. O., & Saberi, A. (2010). An O(log n/log log n)-approximation algorithm for the asymmetric traveling salesman problem.

Ascheuer, N., Fischetti, M., & Grötschel, M. (2001). Solving the asymmetric travelling salesman problem with time windows by branch-and-cut. *Mathematical Programming*, *90*(3), 475–506.

Ascheuer, N., Jünger, M., & Reinelt, G. (2000). A branch & cut algorithm for the asymmetric traveling salesman problem with precedence constraints. *Computational*

Optimization and Applications, *17*(1), 61–84.

Bai, J., Zhu, J., Yang, G.-K., & Pan, C.-C. (2011). Collaborative optimization under a control framework for ATSP. In Y. Tan, Y. Shi, Y. Chai, & G. Wang (Eds.), *Advances in Swarm Intelligence* (pp. 355–363). Springer.

Balas, E., & Christofides, N. (1981). A restricted Lagrangean approach to the traveling salesman problem. *Mathematical Programming*, *21*(1), 19–46.

Balas, E., & Toth, P. (1985). Branch and Bound Methods. In E. Lawler, J. Lenstra, A. R. Kan, & D. Shmoys (Eds.), *The Traveling Salesman Problem* (pp. 361–401). John Wiley & Sons, Chichester.

Baldacci, R., Battarra, M., & Vigo, D. (2008). Routing a Heterogeneous Fleet of Vehicles. In B. Golden, S. Raghavan, & E. Wasil (Eds.), *The vehicle routing problem: Latest advances and new challenges* (Vol. 43, pp. 3–27). Springer US.

Baldacci, R., & Mingozzi, A. (2009). A unified exact method for solving different classes of vehicle routing problems. *Mathematical Programming*, *120*(2), 347–380.

Baldacci, R., Toth, P., & Vigo, D. (2010). Exact algorithms for routing problems under vehicle capacity constraints. *Annals of Operations Research*, *175*, 213–245.

Barahona, F., & Anbil, R. (2000). The Volume Algorithm: producing primal solutions with a subgradient method. *Mathematical Programming*, *87*(3), 385–399.

Basart, J. (2003). *Grafs: fonaments i algorismes* (Third ed.). Universitat Autonoma de Barcelona.

Begur, S. V., Miller, D. M., & Weaver, J. R. (1997). An integrated spatial DSS for scheduling and routing home-health care nurses. *Interfaces*, *27*(4), 35–48.

Beldiceanu, N., Carlsson, M., & Rampon, J.-X. (2005). *Global Constraint Catalog* (Vol. 8; Tech. Rep. No. 2005). Kista, Sweden: Swedish Institute of Computer Science (SICS). Retrieved from http://eprints.sics.se/2366/01/SICS-T--2005-08--SE.pdf

Beldiceanu, N., & Contejean, E. (1994). Introducing global constraints in CHIP. *Mathematical and Computer Modelling*, *20*(12), 97–123.

Bell, J., & McMullen, P. (2004). Ant Colony Optimization Techniques For The Vehicle Routing Problem. *Advanced Engineering Informatics*(18), 41–48.

Bellman, R. (1962). Dynamic Programming Treatment of the Travelling Salesman Problem. *Journal of the ACM*, *9*(1), 61–63.

Belloni, A., & Lucena, A. (2000). A relax and cut algorithm for the traveling salesman problem. In *17th international symposium on mathematical programming.*

Bent, R., & Van Hentenryck, P. (2004). A two-stage hybrid local search for the vehicle routing problem with time windows. *Transportation Science*, *38*(4), 515–530.

Berger, J., & Barkaoui, M. (2003). A Hybrid Genetic Algorithm for the Capacitated Vehicle Routing Problem. In *Proceedings of the International Genetic and Evolutionary Computation Conference –GECCO 2003* (pp. 646–656). Illinois, Chicago, USA: Springer-Verlag.

Bertels, S., & Fahle, T. (2006). A hybrid setup for a hybrid scenario: combining heuristics for the home health care problem. *Computers & Operations Research*, *33*(10), 2866–2890.

Bertsekas, D. P. (1999). *Nonlinear Programming.* Belmont, Massachusetts: Athena Scientific.

Bessière, C. (2006). Constraint propagation. In F. Rossi, P. van Beek, & T. Walsh (Eds.), *Handbook of constraint programming* (pp. 29–83). Elsevier.

Bessière, C., Hebrard, E., Hnich, B., zıltan, Z. K., & Walsh, T. (2006). Filtering Algorithms for the NValue Constraint. *Constraints*, *11*(4), 271–293.

Bessière, C., Katsirelos, G., Narodytska, N., Quimper, C.-G., & Walsh, T. (2010). Decomposition of the NValue Constraint. In *Principles and Practice of Constraint Programming –CP 2010* (pp. 114–128).

Bolduc, M. C., Renaud, J., & Montreuil, B. (2006, June). Synchronized routing of seasonal products through a production/distribution network. *Central European Journal of Operations Research*, *14*(2), 209–228.

Bontoux, B., Artigues, C., & Feillet, D. (2010). A memetic algorithm with a large neighborhood crossover operator for the generalized traveling salesman problem. *Computers and Operations Research*, *37*(11), 1844–1852.

Borsani, V., Matta, A., Beschi, G., & Sommaruga, F. (2006). A home care scheduling model for human resources. In *International Conference on Service Systems and Service Management* (Vol. 1, pp. 449–454). IEEE.

Boschetti, M., & Maniezzo, V. (2009). Benders decomposition, Lagrangean relaxation and metaheuristic design. *Journal of Heuristics*, *15*(3), 283–312.

Boyer, C. B. (1985). *A history of mathematics* (P. Press, Ed.). Princeton Univ.

Brandao, J. (2009). A deterministic tabu search algorithm for the fleet size and mix vehicle routing problem. *European Journal of Operational Research*, *195*(3), 716–728.

Brandao, J. (2011). A tabu search algorithm for the heterogeneous fixed fleet vehicle routing problem. *Computers & Operations Research*, *38*(1), 140–151.

Bräysy, O. (2003). A reactive variable neighborhood search for the vehicle-routing problem with time windows. *INFORMS Journal on Computing*, *15*(4), 347–368.

Brewka, G., Eiter, T., & Truszczyński, M. (2011). Answer set programming at a glance. *Communications of the ACM*, *54*(12), 92–103.

Buriol, L., França, P. M., & Moscato, P. (2004). A new memetic algorithm for the asymmetric traveling salesman problem. *Journal of Heuristics*, *10*(5), 483–506.

Burke, E. K., Cowling, P. I., & Keuthen, R. (2001). Effective local and guided variable neighbourhood search methods for the asymmetric travelling salesman problem. In *Applications of evolutionary computing* (pp. 203–212). Springer Berlin Heidelberg.

Buxey, G. (1979). The Vehicle Scheduling Problem and Monte Carlo Simulation. *Journal of Operational Research Society*, *30*, 563–573.

Caprara, A., Fischetti, M., & Toth, P. (1996). A heuristic algorithm for the set covering problem. In *Integer Programming and Combinatorial Optimization* (pp. 72–84). Springer.

Caprara, A., Fischetti, M., & Toth, P. (2002). Modeling and Solving the Train Timetabling Problem. *Operations Research*, *50*(5), 851–861.

Caprara, A., Toth, P., & Fischetti, M. (2000). Algorithms for the set covering problem.

Annals of Operations Research, *98*(1), 353–371.

Carpaneto, G., Dell'Amico, M., & Toth, P. (1995). Exact solution of large-scale, asymmetric traveling salesman problems. *ACM Transactions on Mathematical Software (TOMS)*, *21*(4), 394–409.

Carpaneto, G., & Toth, P. (1980). Some new branching and bounding criteria for the asymmetric travelling salesman problem. *Management Science*, *26*(7), 736–743.

Carrabs, F., Cordeau, J.-F., & Laporte, G. (2007). Variable neighborhood search for the pickup and delivery traveling salesman problem with lifo loading. *INFORMS Journal on Computing*, *19*(4), 618–632.

Caseau, Y., & Laburthe, F. (1997, Jul.). Solving small TSPs with constraints. In L. Naish (Ed.), *Proc. of the 14th international conference on logic programming* (pp. 316–330). The MIT Press.

Cattafi, M., Herrero, R., Gavanelli, M., Nonato, M., & Malucelli, F. (2012). Improving Quality and Efficiency in Home Health Care: an application of Constraint Logic Programming for the Ferrara NHS unit. In *Iclp (technical communications)* (pp. 415–424).

Cattafi, M., Herrero, R., Gavanelli, M., Nonato, M., & Malucelli, F. (2015). An Application of Constraint Solving for Home Health Care. *AI Communications*, *28*(2), 215–237.

Celik, Y., & Ulker, E. (2012). A marriage in honey bee optimisation approach to asymmetric traveling salesman problem. *International Journal of Innovative Computing, Information and Control*, *8*(6), 4123–4132.

Chen, M. (1996). A mathematical programming model for AGVs planning and control in manufacturing systems. *Computers & industrial engineering*, *30*(4), 647–658.

Chen, S. M., & Chien, C. Y. (2011). Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques. *Expert Systems with Applications*, *38*(12), 14439–14450.

Cheng, E., & Rich, J. L. (1998). *A home health care routing and scheduling problem* (Tech. Rep. Nos. CAAM TR98–04). Rice University.

Cheng, R., & Gen, M. (1994). Crossover on intensive search and traveling salesman problem. *Computers & Industrial Engineering*, *27*(1), 485–488.

Choi, E., & Tcha, D. W. (2007). A column generation approach to the heterogeneous fleet vehicle routing problem. *Computers & Operations Research*, *34*(7), 2080–2095.

Choi, I. C., Kim, S. I., & Kim, H. S. (2003). A genetic algorithm with a mixed region search for the asymmetric traveling salesman problem. *Computers & Operations Research*, *30*(5), 773–786.

Christofides, N. (1976). *Worst-case analysis of a new heuristic for the travelling salesman problem* (Tech. Rep. No. RR-388). Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group.

Christofides, N., Mingozzi, A., & Toth, P. (1979). The vehicle routing problem. In N. Christofides et al. (Eds.), *Combinatorial optimization* (pp. 315–338). Wiley, Chichester, U.K..

Cipra, B. (1993). Quick trips. *SIAM News*, *26*.

Cirasella, J., Johnson, D. S., McGeoch, L. A., & Zhang, W. (2001). The asymmetric traveling salesman problem: Algorithms, instance generators, and tests. In A. L. Buchsbaum & J. Snoeyink (Eds.), *Algorithm engineering and experimentation* (Vol. 2153, pp. 32–59). Springer.

Clarke, G., & Wright, J. (1964). Scheduling of Vehicles from a central Depot to a Number of Delivering Points. *Operations Research*, *12*(4), 568–581.

Cordeau, J., & Laporte, G. (2004). Tabu search heuristics for the vehicle routing problem. In C. Rego & B. Alidaee (Eds.), (pp. 145–163). Boston, MA, U.S.A.: Kluwer Academic.

Cordeau, J., Laporte, G., Savelsbergh, M., & Vigo, D. (2007). Vehicle routing. In C. Barnhart & G. Laporte (Eds.), *Handbook in operations research and management science* (Vol. 14, pp. 367–428). Amsterdam, The Netherlands: Elsevier.

Cormen, T., Leiserson, C., Rivest, R., & Stein, C. (2001). Section 23.2: The algorithms of Kruskal and Prim. In M. Press & S. E. McGraw-Hill (Eds.), *Introduction to algorithms* (pp. 567–574).

Crispim, J., & Brandao, J. (2001). Reactive tabu search and variable neighbourhood descent applied to the vehicle routing problem with backhauls. In *Micõ2001 4th metaheuristic international conference, porto, portugal.*

Dantzig, G. B., Fulkerson, R., & Johnson, S. (1954). Solution of a large-scale Traveling Salesman Problem. *Operations Research*, *2*, 393–410.

Dantzig, G. B., & Ramser, J. H. (1959). The truck dispatching problem. *Management Science*, *6*, 80–91.

Dantzig, G. B., & Wolfe, P. (1960). Decomposition principle for linear programs. *Operations Research*, *8*, 101–111.

De Franceschi, R., Fischetti, M., & Toth, P. (2006). A new ILP-based refinement heuristic for vehicle routing problems. *Mathematical Programming*, *105*(2-3), 471–499.

Deep, K., & Mebrahtu, H. (2011). Combined mutation operators of genetic algorithm for the travelling salesman problem. *International Journal of Combinatorial Optimization Problems and Informatics*, *2*(3), 1–23.

Delisle, P., Gravel, M., Krajecki, M., Gagné, C., & Price, W. L. (2005). A shared memory parallel implementation of ant colony optimization. In *Proceedings of the 6th metaheuristics international conference* (pp. 257–264).

Delisle, P., Krajecki, M., & Gravel, M. (2009). Multi-colony parallel ant colony optimization on SMP and multi-core computers. In *2009 world congress on nature&biologically inspired computing (nabic).*

Desrosiers, J., Sauvé, M., & Soumis, F. (1988). Lagrangean relaxation methods for solving the minimum fleet size multiple travelling salesman problem with time windows. *Management Science*, *34*, 1005–1022.

Dorigo, M., & Gambardella, L. (2014). Ant-Q: A reinforcement learning approach to the traveling salesman problem. In *Proceedings of ML-95, Twelfth Intern. Conf. on Machine Learning* (pp. 252–260).

Dorigo, M., & Gambardella, L. M. (1997). Ant colony system: a cooperative learning approach to the traveling salesman problem. *Evolutionary Computation, IEEE Transactions on*, *1*(1), 53–66.

Dorronsoro, B., Arias, D., Luna, F., Nebro, A. J., & Alba, E. (2007). A grid-based hybrid cellular genetic algorithm for very large scale instances of the cvrp. In *High performance computing & simulation conference (hpcs)* (pp. 759–765).

Drexl, M. (2012). Rich vehicle routing in theory and practice. *Logistics Research*, *5*(1-2), 47–63.

Edmonds, J. (1967). Optimum Branchings. *Journal of Research of the National Bureau of Standards B*, *71*, 233–240.

Edmonds, J., & Johnson, E. L. (1973). Matching, Euler tours and the Chinese postman. *Mathematical programming*, *5*(1), 88–124.

Elbenani, B., Ferland, J., & Gascon, V. (2008). Mathematical programming approach for routing home care nurses. In *Industrial Engineering and Engineering Management, IEEM 2008. IEEE International Conference on* (pp. 107–111).

Escobar, J. W., Linfati, R., Baldoquin, M. G., & Toth, P. (2014). A Granular Variable Tabu Neighborhood Search for the capacitated location-routing problem. *Transportation Research Part B: Methodological*, *67*, 344–356.

Euchi, J., & Chabchoub, H. (2010). A hybrid tabu search to solve the heterogeneous fixed fleet vehicle routing problem. *Logistics Research*, *2*(1), 3–11.

Eveborn, P., Rönnqvist, M., Einarsdóttir, H., Eklund, M., Lidén, K., & Almroth, M. (2009). Operations Research Improves quality and efficiency in Home Care. *Interfaces*, *39*(1), 18–34.

Faulin, J., Gilibert, M., Juan, A. A., Ruiz, R., & Vilajosana, X. (2008). SR-1: A simulation-based algorithm for the capacitated vehicle routing problem. *Proceedings of the 2008 Winter Simulation Conference*.

Faulin, J., & Juan, A. A. (2008). The ALGACEA-1 Method for the Capacitated Vehicle Routing Problem. *International Transactions in Operational Research*, *15*, 1–23.

Feo, T., & Resende, M. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, *6*(2), 109–133.

Fernández, P., García, L., Mayado, A., & Sanchís, J. (2000). A Real Delivery Problem Dealt with Monte Carlo Techniques. *TOP*, *8*, 57–71.

Festa, P., & Resende, M. (2009). An Annotated Bibliography of GRASP–Part I: Algorithms. *International Transactions in Operational Research*, *16*, 1–24.

Fischetti, M., Lodi, A., & Toth, P. (2003). Solving real-world ATSP instances by branch-and-cut. In *Combinatorial optimization–eureka, you shrink!* (pp. 64–77). Springer.

Fischetti, M., Lodi, A., & Toth, P. (2007). Exact methods for the asymmetric traveling salesman problem. In *The traveling salesman problem and its variations* (pp. 169–205). Springer.

Fischetti, M., Salazar, J., & Toth, P. (1997a). A branch-and-cut algorithm for the symmetric generalized Traveling Salesman Problem. *Operations Research*, *45*, 378–394.

Fischetti, M., & Toth, P. (1992). An additive bounding procedure for the asymmetric travelling salesman problem. *Mathematical Programming*, *53*(1–3), 173–197.

Fischetti, M., & Toth, P. (1997b). A polyhedral approach to the asymmetric traveling salesman problem. *Management Science*, *43*(11), 1520–1536.

Fischetti, M., Toth, P., & Vigo, D. (1994). A branch-and-bound algorithm for the capacitated vehicle routing problem on directed graphs. *Operations Research*, *42*(5), 846–859.

Fisher, M. L. (2004). The Lagrangian Relaxation Method for Solving Integer Programming Problems. *Management Science*, *50*(12S), 1861–1871.

Fisher, M. L., & Jaikumar, R. (1981). A generalized assignment heuristic for vehicle-routing. *Networks*, *11*(2), 109–124.

Fisher, M. L., Jornsten, K. O., & Madsen, O. B. G. (1997). Vehicle routing with time windows: Two optimization algorithms. *Operations research*, *45*(3), 488–492.

Focacci, F., Lodi, A., & Milano, M. (2002). A hybrid exact algorithm for the TSPTW. *INFORMS Journal on Computing*, *14*(4), 403–417.

Freisleben, B., & Merz, P. (1996). New genetic local search operators for the traveling salesman problem. In *Parallel problem solving from nature–ppsn iv* (pp. 890–899). Springer.

Fukunaga, A. S., & Korf, R. E. (2007). Bin completion algorithms for multicontainer packing, knapsack, and covering problems. *J. Artif. Int. Res.*, *28*(1), 393–429.

Garrido, P., Castro, C., & Monfroy, E. (2009). Towards a Flexible and Adaptable Hyperheuristic Approach for VRPs. In *Ic-ai* (pp. 311–317).

Gaskell, T. (1967). Bases for the vehicle fleet scheduling. *Operational Research Quarterly*, *18*, 281–295.

Gavanelli, M. (2002). An Algorithm for Multi-Criteria Optimization in CSPs. In *Ecai 2002. proceedings of the 15th european conference on artificial intelligence* (pp. 136–140). IOS Press.

Gavish, B., & Pirkul, H. (1986). Computer and Database Location in Distributed Computer Systems. *Computers, IEEE Transactions on Computers*, *35*(7), 583–590.

Gebser, M., Kaminski, R., Kaufmann, B., Ostrowski, M., Schaub, T., & Schneider, M. (2011). Potassco: The Potsdam answer set solving collection. *AI Communications*, *24*(2), 105–124.

Gen, M., & Cheng, R. (2000). *Genetic algorithms and engineering optimization* (Vol. 7). John Wiley & Sons.

Gendreau, M., Hertz, A., & Laporte, G. (1994). A Tabu Search Heuristic for the Vehicle Routing Problem. *Management Science*, *40*, 1276–1290.

Gendreau, M., Laporte, G., Musaraganyi, C., & Taillard, E. (1999). A tabu search heuristic for the heterogeneous fleet vehicle routing problem. *Computers & Operations Research*, *26*(12), 1153–1173.

Gendron, B., Khuong, P.-V., & Semet, F. (2013). *A lagrangian-based branch-and-bound algorithm for the two-level uncapacitated facility location problem with single-assignment constraints*. Montreal: Cirrelt.

Geoffrion, A. (1974). Lagrangean relaxation for integer programming. In M. Balinski (Ed.), *Approaches to integer programming* (Vol. 2, pp. 82–114). Springer Berlin Heidelberg.

Germs, R., Goldengorin, B., & Turkensteen, M. (2012). Lower tolerance-based branch and bound algorithms for the ATSP. *Computers & Operations Research*, *39*(2), 291–298.

Giuliani, M. V., Scopelliti, M., & Fornara, F. (2005). Elderly people at home: technological help in everyday activities. In *Robot and human interactive communication, 2005. roman 2005. ieee international workshop on* (pp. 365–370).

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers & operations research*, *13*(5), 533–549.

Glover, F., Gutin, G., Yeo, A., & Zverovich, A. (2001). Construction heuristics for the asymmetric TSP. *European Journal of Operational Research*, *129*(3), 555–568.

Glover, F., & Laguna, M. (1997). *Tabu search*. Kluwer Academic Publishers.

Glover, F., & Laguna, M. (2013). *Tabu search* (Second Edition ed.). Springer.

Golden, B., Assad, A., Levy, L., & Gheysens, F. (1984). The Fleet Size and Mix Vehicle Routing Problem. *Computers & Operations Research*, *11*(1), 49–66.

Golden, B., Assad, A. A., & Wasil, E. A. (2002). Routing Vehicles in the Real World: Applications in the Solid Waste, Beverage, Food, Dairy and Newspaper Industries. In P. Toth & D. Vigo (Eds.), *The vehicle routing problem* (pp. 245–286). SIAM Monographs on Discrete Mathematics and Applications.

Golden, B., Raghavan, S., & Wasil, E. A. (2008). *The Vehicle Routing Problem: Latest Advances and New Challenges* (Vol. 43). Springer.

Gomes, C. P., Selman, B., & Crato, N. (1997). Heavy-Tailed Distributions in Combinatorial Search. In *Cp* (pp. 121–135).

Grötschel, M., & Holland, O. (1991). Solution of large-scale symmetric travelling salesman problems. *Mathematical Programming*, *51*(1-3), 141–202.

Guignard, M. (2003). Lagrangean Relaxation. *Sociedad de Estadística e Investigación Operativa, Top*, *11*(2), 151–228.

Guimarans, D. (2012). *Hybrid algorithms for solving routing problems* (PhD Thesis). Universitat Autònoma de Barcelona. Departament de Telecomunicació i Enginyeria de Sistemes. Retrieved from http://hdl.handle.net/10803/96386

Guimarans, D., Herrero, R., Ramos, J., & Padrón, S. (2011a). Solving vehicle routing problems using constraint programming and lagrangean relaxation in a metaheuristics framework. *International Journal of Information Systems and Supply Chain Management*.

Guimarans, D., Herrero, R., Ramos, J. J., & Padrón, S. (2013). Management innovations for intelligent supply chains. In J. Wang (Ed.), (pp. 123–143). IGI Global.

Guimarans, D., Herrero, R., Riera, D., Juan, A., & Ramos, J. (2010). Combining Constraint Programming, Lagrangian Relaxation and Probabilistic Algorithms to Solve the Vehicle Routing Problem. In *Proceedings of the 17th rcra international workshop.* Bologna, Italy.

Guimarans, D., Herrero, R., Riera, D., Juan, A. A., & Ramos, J. J. (2011b). Combining

probabilistic algorithms, Constraint Programming and Lagrangian Relaxation to solve the Vehicle Routing Problem. *Annals of Mathematics and Artificial Intelligence*, *62*(3-4), 299–315.

Gutin, G., & Karapetyan, D. (2010). A memetic algorithm for the generalized traveling salesman problem. *Natural Computing*, *9*(1), 47–60.

Gutin, G., & Punnen, A. P. (2002). *The Traveling Salesman Problem and Its Variations* (Vol. 12). Springer Science & Business Media.

Hanada, K., & Hirayama, K. (2011). Distributed Lagrangian Relaxation Protocol for the Over-constrained Generalized Mutual Assignment Problem. In D. Kinny, J.-j. Hsu, G. Governatori, & A. Ghose (Eds.), *Agents in principle, agents in practice* (Vol. 7047, pp. 174–186). Springer Berlin Heidelberg.

Hansen, K. H., & Krarup, J. (1974). Improvements of the Held–Karp algorithm for the symmetric traveling-salesman problem. *Mathematical Programming*, *7*(1), 87–96.

Hansen, P., & Mladenović, N. (2003). *A tutorial on variable neighborhood search* (Tech. Rep. No. G-2003-46). Montreal, Canada: Groupe d'Études et de Recherche en Analyse des Décisions (GERAD). Retrieved from http://www.gerad.ca/fichiers/cahiers/G-2003-46.pdf

Hansen, P., & Mladenović, N. (2006). First vs. best improvement: An empirical study. *Discrete Applied Mathematics*, *154*(5), 802–817.

Haralick, R. M., & Elliott, G. L. (1980). Increasing Tree Search Efficiency for Constraint Satisfaction Problems. *Artif. Intell.*, *14*(3), 263–313.

Hartl, R., Hasle, G., & Janssens, G. (2006). Special issue on Rich Vehicle Routing Problems. *Central European Journal of Operations Research*, *14*(2), 103–104.

Hasle, G., & Kloster, O. (2007). Industrial vehicle routing. In G. Hasle, K. Lie, & E. Quak (Eds.), *Geometric modelling, numerical simulation, and optimization* (pp. 397–435). Berlin, Germany: Springer-Verlag.

Hebrard, E., Marx, D., O'Sullivan, B., & Razgon, I. (2011). Soft Constraints of Difference and Equality. *J. Artif. Intell. Res. (JAIR)*, *41*, 97–130.

Held, M., & Karp, R. (1970). The Traveling Salesman Problem and Minium Spanning Trees. *Operations Research*, *18*, 1138–1162.

Held, M., & Karp, R. (1971). The Traveling Salesman Problem and Minium Spanning Trees: Part II. *Mathematical Programming*, *1*, 6–25.

Helsgaun, K. (2000). An effective implementation of the Lin–Kernighan traveling salesman heuristic. *European Journal of Operational Research*, *126*(1), 106–130.

Helsgaun, K. (2009). General k-opt submoves for the Lin-Kernighan TSP heuristic. *Mathematical Programming Computation*, *1*(2-3), 119–163.

Herrero, R., Guimarans, D., Ramos, J. J., & Padrón, S. (2010a). A variable neighbourhood search combining constraint programming and lagrangian relaxation for solving routing problems. In *Proceedings of the 2010 summer computer simulation conference* (pp. 379–386).

Herrero, R., Ramos, J. J., & Guimarans, D. (2010b). Lagrangian Metaheuristic for the Travelling Salesman Problem. In *Extended abstracts of operational research conference 52.* Royal Holloway, University of London.

Herrero, R., Rodríguez, A., Cáceres-Cruz, J., & Juan, A. A. (2014). Solving vehicle routing problems with asymmetric costs and heterogeneous fleets. *International Journal of Advanced Operations Management*, *6*(1), 58–80.

Hiriart-Urruty, J., & Lemarechal, C. (1993). *Convex Analysis and Minimization Algorithms* (Vol. 1-2). Berlin, Germany: Springer Verlag.

Hoeft, J., & Palekar, U. (1997). Heuristics for the plate-cutting traveling salesman problem. *IIE Transactions (Institute of Industrial Engineers)*, *29*(9), 719–731.

Hoffman, A., & Wolfe, P. (1985). The Traveling Salesman Problem. In E. Lawler, J. Lenstra, A. R. Kan, & D. Shmoys (Eds.), (pp. 1–16). John Wiley.

Hoffman, K. L., Padberg, M., & Rinaldi, G. (2013). Traveling Salesman Problem. In *Encyclopedia of operations research and management science* (pp. 1573–1578). Springer.

Holland, J. (1975). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. Ann Arbor: University of Michigan Press.

Hooker, J. (2008). Integer Programming: Lagrangian Relaxation. In *Encyclopedia of Optimization* (pp. 1667–1673). Springer.

Hulubei, T., & O'Sullivan, B. (2006). The Impact of Search Heuristics on Heavy-Tailed Behaviour. *Constraints*, *11*(2-3), 159–178.

IBM ILOG CPLEX Optimization Studio. (2015). Retrieved from https://www.ibm.com/developerworks/downloads/ws/ilogcplex/

Imran, A., Salhi, S., & Wassan, N. A. (2009). A variable neighborhood-based heuristic for the heterogeneous fleet vehicle routing problem. *European Journal of Operational Research*, *197*(2), 509–518.

Irnich, S. (2008). A Unified Modeling and Solution Framework for Vehicle Routing and Local Search-Based Metaheuristics. *INFORMS Journal on Computing*, *20*(2), 270–287.

Jena, S. D., Cordeau, J.-F., & Gendron, B. (2014). Lagrangian Heuristics for Large-Scale Dynamic Facility Location with Generalized Modular Capacities. *INFORMS Journal on Computing*.

Jin Ai, T., & Kachitvichyanukul, V. (2009). Particle swarm optimization and two solution representations for solving the capacitated vehicle routing problem. *Computers and Industrial Engineering*, *56*, 380–387.

Johnson, D., Gutin, G., McGeoch, L. A., Yeo, A., Zhang, W., & Zverovitch, A. (2007). Experimental analysis of heuristics for the ATSP. In *The traveling salesman problem and its variations* (pp. 445–487). Springer.

Juan, A., Faulin, J., Jorba, J., Grasman, S., & Barrios, B. (2008). SR-2: A Hybrid Intelligent Algorithm for the Vehicle Routing Problem. In *Proceedings of the 8th international conference on hybrid intelligent systems* (pp. 78–83). Barcelona, Spain.

Juan, A., Faulín, J., Jorba, J., Riera, D., Masip, D., & Barrios, B. (2011). On the use of Monte Carlo simulation, cache and splitting techniques to improve the Clarke and Wright savings heuristics. *Journal of the Operational Research Society*, *62*(6),

1085–1097.

Juan, A., Faulin, J., Ruiz, R., Barrios, B., & Caballé, S. (2010). The SR-GCWS hybrid algorithm for solving the capacitated vehicle routing problem. *Applied Soft Computing*, *10*(1), 215–224.

Juan, A., Faulin, J., Ruiz, R., Barrios, B., Gilibert, M., & Vilajosana, X. (2009). Using oriented random search to provide a set of alternative solutions to the capacitated vehicle routing problem. *Operations Research and Cyber-Infrastructure*, 331–346.

Kallehauge, B., Larsen, J., & Madsen, O. B. (2006). Lagrangian duality applied to the vehicle routing problem with time windows. *Computers & Operations Research*, *33*(5), 1464–1487.

Kanellakis, P.-C., & Papadimitriou, C. H. (1980). Local search for the asymmetric traveling salesman problem. *Operations Research*, *28*(5), 1086–1099.

Karaboga, D., & Gorkemli, B. (2011). A combinatorial artificial bee colony algorithm for traveling salesman problem. In *Innovations in intelligent systems and applications (inista), 2011 international symposium on* (pp. 50–53).

Karp, R. M. (1972). Reducibility among combinatorial problems. In R. Miller & J. Thatcher (Eds.), *Complexity of computer computations* (pp. 85–103). New York: Plenum Press.

Karp, R. M. (1979). A patching algorithm for the nonsymmetric traveling-salesman problem. *SIAM Journal on Computing*, *8*(4), 561–573.

Kautz, H., Horvitz, E., Ruan, Y., Gomes, C., & Selman, B. (2002). Dynamic Restart Policies. In *Aaai/iaai* (p. 674-681).

Kaya, L. G., & Hooker, J. N. (2006). A Filter for the Circuit Constraint. In *Cp* (pp. 706–710).

Khooban, Z. (2011). Transportation. In R. Z. Farahani, S. Rezapour, & L. Kardar (Eds.), *Logistics Operations and Management* (p. 109 - 126). London: Elsevier.

Kilby, P., Prosser, P., & Shaw, P. (2000). A Comparison of Traditional and Constraint-based Heuristic Methods on VRPs with Side Constraints. *Constraints*, *5*(4), 389–414.

Kiwiel, K. C. (1996). Restricted step and Levenberg-Marquardt techniques in proximal Bundle methods for nonconvex nondifferentiable optimization. *SIAM Journal on Optimization*, *6*(1), 227–249.

Knuth, D. E. (1992, May). Two notes on notation. *Am. Math. Monthly*, *99*(5), 403–422.

Koeleman, P., Bhulai, S., & van Meersbergen, M. (2012). Optimal patient and personnel scheduling policies for care-at-home service facilities. *European Journal of Operational Research*, *219*(3), 557–563.

Kohl, N., & Madsen, O. B. G. (1997, May). An optimization algorithm for the vehicle routing problem with time windows based on lagrangian relaxation. *Operations research*, *45*(3), 395–406.

Kolata, G. (1991, March). Math Problem, Long Baffling, Slow Yields. *The New York Times*, *12*.

Korf, R. E. (2003). An Improved Algorithm for Optimal Bin Packing. In *Ijcai* (pp. 1252–1258).

Korte, B., & Vygen, J. (2012). *Combinatorial Optimization: Theory and Algorithms* (No. 21). Springer-Verlag Berlin Heidelberg.

Lanzarone, E., Matta, A., & Scaccabarozzi, G. (2010). A patient stochastic model to support human resource planning in home care. *Production Planning and Control*, *21*(1), 3–25.

Laporte, G. (1992). The traveling salesman problem: an overview of exact and approximate algorithms. *European Journal of Operational Research*, *59*(2), 231–247.

Laporte, G. (2007). What you should know about the Vehicle Routing Problem. *Naval Research Logistics*, *54*, 811–819.

Laporte, G. (2009, NOV). Fifty Years of Vehicle Routing. *Transportation Science*, *43*(4), 408–416.

Laporte, G., Mercure, H., & Nobert, Y. (1986). An exact algorithm for the asymmetrical capacitated vehicle routing problem. *Networks*, *16*(1), 33–46.

Laporte, G., Mercure, H., & Nobert, Y. (1987). Generalized travelling salesman problem through n sets of nodes: the asymmetrical case. *Discrete Applied Mathematics*, *18*(2), 185–197.

Laporte, G., & Nobert, Y. (1987). Exact algorithms for the vehicle routing problem. *Annals of Discrete Mathematics*, *31*, 147–184.

Law, A. (2007). *Simulation Modeling & Analysis*. McGraw-Hill.

Lawler, E., Lenstra, J., Rinnooy-Kan, A., & Shmoys, D. (1985). *The Traveling Salesman Problem: A Guided Tour of Combinatorial*. Wiley, New York.

Lenstra, J. K., & Kan, A. (1981). Complexity of vehicle routing and scheduling problems. *Networks*, *11*(2), 221–227.

Li, F., Golden, B., & Wasil, E. (2007). A record-to-record travel algorithm for solving the heterogeneous fleet vehicle routing problem. *Computers and Operation Research*, *34*(9), 2734–2742.

Li, X., Tian, P., & Aneja, Y. (2010). An adaptive memory programming metaheuristic for the heterogeneous fixed fleet vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review*, *46*(6), 1111–1127.

Li, Z., & Shahidehpour, M. (2003, Nov). Generation scheduling with thermal stress constraints. *Power Systems, IEEE Transactions on*, *18*(4), 1402–1409.

Lima, C., Goldbarg, M., & Goldbarg, E. (2004). A Memetic Algorithm for the Heterogeneous Fleet Vehicle Routing Problem. *Electronic Notes in Discrete Mathematics*, *18*, 171–176.

Lin, S., & Kernighan, B. (1973). An effective heuristic algorithm for the traveling salesman problem. *Operations Research*, *21*, 498–516.

Lin, S., Lee, Z., Ying, K., & Lee, C. (2009). Applying hybrid meta-heuristics for capacitated vehicle routing problem. *Expert Systems with Applications*, *2*(36), 1505–1512.

Liu, S., Huang, W., & Ma, H. (2009). An effective genetic algorithm for the fleet size and mix vehicle routing problems. *Transportation Research Part E: Logistics and Transportation Review*, *45*(3), 434–445.

Luby, M., Sinclair, A., & Zuckerman, D. (1993). Optimal speedup of Las Vegas algo-

rithms. *Inf. Process. Lett.*, *47*(4), 173–180.

Lucena, A. (2006). Lagrangian relax-and-cut algorithms. In *Handbook of Optimization in Telecommunications* (pp. 129–145). Springer.

Luh, P. B., & Hoitomt, D. J. (1993). Scheduling of manufacturing systems using the lagrangian relaxation technique. *Automatic Control, IEEE Transactions on*, *38*(7), 1066–1079.

Mao, K., ke Pan, Q., Pang, X.-F., Chai, T., & Li, J. (2014, June). Scheduling a real-world hybrid flow shop with variable processing times using Lagrangian relaxation. In *Intelligent control and automation (WCICA), 2014 11th world congress on* (pp. 3434–3439).

Marinakis, Y., Migdalas, A., & Pardalos, P. (2005). A Hybrid Genetic-GRASP Algorithm Using Lagrangean Relaxation for the Traveling Salesman Problem. *Journal of Combinatorial Optimization*, *10*(4), 311–326.

Marinakis, Y., Migdalas, A., & Pardalos, P. (2009). Multiple phase neighborhood Search-GRASP based on Lagrangean relaxation, random backtracking Lin-Kernighan and path relinking for the TSP. *Journal of Combinatorial Optimization*, *17*, 134–156.

Marmion, M.-E., Humeau, J., Jourdan, L., & Dhaenens, C. (2010, may). Comparison of neighborhoods for the HFF-AVRP. In *Computer systems and applications (aiccsa), 2010 ieee/acs international conference on* (pp. 1–7).

Mavrovouniotis, M., & Yang, S. (2013). Ant colony optimization with immigrants schemes for the dynamic travelling salesman problem with traffic factors. *Applied Soft Computing*, *13*(10), 4023–4037.

Menger, K. (1930). Das botenproblem. *Ergebnisse eines Mathematischen Kolloquiums*, *2*, 11–12.

Merz, P., & Freisleben, B. (1997). Genetic local search for the TSP: New results. In *Evolutionary Computation, 1997., IEEE International Conference on* (pp. 159–164).

Meseguer, P., & Torras, C. (2001). Exploiting symmetries within constraint satisfaction search. *Artif. Intell.*, *129*(1–2), 133-163.

Mester, D., & Bräysy, O. (2007). Active-guided evolution strategies for the large-scale capacitated vehicle routing problems. *Computers and Operations Research*, *34*, 2964–2975.

Mileo, A., Merico, D., & Bisiani, R. (2011, March). Reasoning support for risk prediction and prevention in independent living. *Theory and Practice of Logic Programming*, *11*(2-3), 361–395.

Miller, C., Tucker, A., & Zemlin, R. (1960). Integer Programming Formulations and Traveling Salesman Problems. *Journal of the ACM*, *7*(4), 326–329.

Miller, D. L., & Pekny, J. F. (1991). Exact solution of large asymmetric traveling salesman problems. *Science*, *251*(4995), 754–761.

Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, *24*(11), 1097–1100.

Monette, J., Schaus, P., Zampelli, S., Deville, Y., & Dupont, P. (2007). A CP approach to the balanced academic curriculum problem. In B. Benhamou, B. Choueiry, &

B. Hnich (Eds.), *Symcon.*

Monfroy, E., Saubion, F., & Lambert, T. (2004). On Hybridization of Local Search and Constraint Propagation. In *Iclp* (pp. 299–313).

Nagata, Y. (2007). Edge Assembly Crossover for the Capacitated Vehicle Routing Problem. In C. Cotta & J. I. van Hemert (Eds.), *Evolutionary computation in combinatorial optimization, 7th european conference, evocop* (pp. 142–153). Valencia: Springer.

Nagata, Y., & Kobayashi. (1997). Edge assembly crossover. a high-power genetic algorithm for the traveling salesman problem. In *7th international conference on genetic algorithms (icga)* (pp. 450–457).

Nagata, Y., & Kobayashi, S. (2013). A Powerful Genetic Algorithm Using Edge Assembly Crossover for the Traveling Salesman Problem. *INFORMS J. on Computing*, *25*(2), 346–363.

Nagata, Y., & Soler, D. (2012). A New Genetic Algorithm for the Asymmetric Traveling Salesman Problem. *Expert Systems with Applications*, *39*(10), 8947–8953.

Nassief, W., Contreras, I., & Asád, R. (2015). A mixed-integer programming formulation and lagrangean relaxation for the cross-dock door assignment problem. *International Journal of Production Research*, *(ahead-of-print)*, 1–15.

Nemhauser, G., & Wolsey, L. (1988). *Integer and combinatorial optimization*. New York: John Wiley & Sons.

Neri, F., Cotta, C., & Moscato, P. (2012). *Handbook of memetic algorithms* (Vol. 379). Springer.

Nezhad, A. M., Manzour, H., & Salhi, S. (2013). Lagrangian relaxation heuristics for the uncapacitated single-source multi-product facility location problem. *International Journal of Production Economics*, *145*(2), 713–723.

Nguyen, T. V., Safaei, F., Boustead, P., & Chou, C. T. (2005). Provisioning overlay distribution networks. *Computer Networks*, *49*(1), 103–118. (Networking Issue in Entertainment Computing)

Nickel, S., Schröder, M., & Steeg, J. (2012). Mid-term and short-term planning support for home health care services. *European Journal of Operational Research*, *219*(3), 574–587.

Nilsson, C. (2003). *Heuristics for the Traveling Salesman Problem* (Theoretical Computer Science Reports). Linkoping University.

Nishi, T., Ando, M., & Konishi, M. (2005, Dec). Distributed route planning for multiple mobile robots using an augmented lagrangian decomposition and coordination technique. *Robotics, IEEE Transactions on*, *21*(6), 1191-1200.

Nishi, T., Hiranaka, Y., & Inuiguchi, M. (2010). Lagrangian relaxation with cut generation for hybrid flowshop scheduling problems to minimize the total weighted tardiness. *Computers & Operations Research*, *37*(1), 189–198.

Nishi, T., Konishi, M., & Ago, M. (2007). A distributed decision making system for integrated optimization of production scheduling and distribution for aluminum production line. *Computers & Chemical Engineering*, *31*(10), 1205–1221.

Noon, C. E., & Bean, J. C. (1991). A lagrangian based approach for the asymmetric

generalized traveling salesman problem. *Operations Research*, *39*(4), 623–632.

Nowak, M. P., & Römisch, W. (2000). Stochastic lagrangian relaxation applied to power scheduling in a hydro-thermal system under uncertainty. *Annals of Operations Research*, *100*(1-4), 251–272.

Öncan, T., Altınel, İ. K., & Laporte, G. (2009). A comparative analysis of several asymmetric traveling salesman problem formulations. *Computers & Operations Research*, *36*(3), 637–654.

Oppen, J., Løkketangen, A., & Desrosiers, J. (2010). Solving a rich vehicle routing and inventory problem using column generation. *Computers & Operations Research*, *37*(7), 1308–1317.

Osaba, E., & Díaz, F. (2012). Comparison of a memetic algorithm and a tabu search algorithm for the traveling salesman problem. In *Computer Science and Information Systems (FedCSIS), 2012 Federated Conference on* (pp. 131–136).

OTLE. (2015, March). Integración de la logística en el Observatorio del Transporte y la Logística en España (OTLE). In *Jornada Anual del OTLE.* Madrid, Spain.

Padberg, M., & Rinaldi, G. (1987). Optimization of a 532-city symmetric traveling salesman problem by branch and cut. *Operations Research Letters*, *6*(1), 1–7.

Padberg, M., & Rinaldi, G. (1991). A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM review*, *33*(1), 60–100.

Papavasiliou, A., Oren, S. S., & O'Neill, R. P. (2011). Reserve requirements for wind power integration: A scenario-based stochastic programming framework. *Power Systems, IEEE Transactions on*, *26*(4), 2197–2206.

Pekny, J. F., & Miller, D. L. (1992). A parallel branch and bound algorithm for solving large asymmetric traveling salesman problems. *Mathematical programming*, *55*(1-3), 17–33.

Pekny, J. F., & Miller, D. L. (1994). A staged primal-dual algorithm for finding a minimum cost perfect two-matching in an undirected graph. *ORSA Journal on Computing*, *6*(1), 68–81.

Penna, P., Subramanian, A., & Ochi, L. (2011). An Iterated Local Search heuristic for the Heterogeneous Fleet Vehicle Routing Problem. *Journal of Heuristics*, 1-32.

Pesant, G., Gendreau, M., Potvin, J.-Y., & Rousseau, J.-M. (1998). An exact constraint logic programming algorithm for the TSP with time windows. *Transp. Sci.*.

Pesant, G., Gendreau, M., & Rousseau, J.-M. (1997). GENIUS-CP: A generic single-vehicle routing algorithm. In *Cp '97* (pp. 420–433). Springer-Verlag.

Pesant, G., & Régin, J.-C. (2005). SPREAD: A Balancing Constraint Based on Statistics. In *Cp* (pp. 460–474).

Pessoa, A., de Aragão, M. P., & Uchoa, E. (2008). Robust Branch-Cut-and-Price Algorithms for Vehicle Routing Problems. In B. Golden, S. Raghavan, & E. Wasil (Eds.), *The vehicle routing problem: Latest advances and new challenges* (Vol. 43, pp. 297–325). Springer US.

Pisinger, D., & Røpke, S. (2007). A general heuristic for vehicle routing problems. *Computers & Operations Research*, *34*(8), 2403–2435.

Posta, M., Ferland, J., & Michelon, P. (2012). An exact method with variable fixing

for solving the generalized assignment problem. *Computational Optimization and Applications*, *52*(3), 629–644.

Prescott-Gagnon, E., Desaulniers, G., & Rousseau, L. (2010, 10). Heuristics for an Oil Delivery Vehicle Routing Problem. *Les Cahiers du GERAD G–2010–63*.

Prins, C. (2002). Efficient Heuristics for the Heterogeneous Fleet Multitrip VRP with Application to a Large-Scale Real Case. *J. Math. Model. Algorithms*, 135–150.

Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers and Operations Research*, *31*, 1985–2002.

Prins, C. (2009, September). Two memetic algorithms for heterogeneous fleet vehicle routing problems. *Eng. Appl. Artif. Intell.*, *22*(6), 916–928.

Prins, C., Prodhon, C., & Roberto, W. C. (2006, 10). Two-phase method and Lagrangian relaxation to solve the Bi-Objective Set Covering Problem. *Annals of Operations Research*, *147*(1), 23–41.

Prive, J., Renaud, J., & Boctor, F. (2006). Solving a vehicle-routing problem arising in soft-drink distribution. *Journal of the Operational Research Society*, *57*, 1045–1052.

Radharamanan, R., & Choi, L. (1986). A branch and bound Algorithm for the Travelling Salesman and the Transportation Routing Problems. *Computers & Industrial Engineering*, *11*, 236–240.

Rajagopalan, S., Heragu, S. S., & Taylor, G. D. (2004). A Lagrangian relaxation approach to solving the integrated pick-up/drop-off point and AGV flowpath design problem. *Applied Mathematical Modelling*, *28*(8), 735–750.

Rasmussen, M. S., Justesen, T., Dohn, A., & Larsen, J. (2012). The Home Care Crew Scheduling Problem: Preference-based visit clustering and temporal dependencies. *European Journal of Operational Research*, *219*, 598–610.

Ray, S. S., Bandyopadhyay, S., & Pal, S. K. (2007). Genetic operators for combinatorial optimization in TSP and microarray gene ordering. *Applied intelligence*, *26*(3), 183–195.

Régin, J.-C. (1994). A Filtering Algorithm for Constraints of Difference in CSPs. In *Aaai* (pp. 362–367).

Reinelt, G. (1994). The traveling salesman: computational solutions for TSP applications. *Lecture notes in computer science*, *840*.

Renaud, J., & Boctor, F. F. (2002). A sweep-based algorithm for the fleet size and mix vehicle routing problem. *European Journal of Operational Research*, *140*(3), 618–628.

Rendl, A., Prandtstetter, M., Hiermann, G., Puchinger, J., & Raidl, G. (2012). Hybrid heuristics for multimodal homecare scheduling. In *Integration of ai and or techniques in contraint programming for combinatorial optimzation problems* (pp. 339–355). Springer.

Resende, M. (2008). Tutorials in operations research. In Z. Chen & S. Raghavan (Eds.), (pp. 295–319).

Rest, K., & Hirsch, P. (2012). A tabu search approach for daily scheduling of home health care services using multi-modal transport. In *Proc 5th int workshop freight*

transp logist, odysseus (pp. 373–376).

Rieck, J., & Zimmermann, J. (2010). A new mixed integer linear model for a rich vehicle routing problem with docking constraints. *Annals of Operations Research*, *181*(1), 337–358.

Rocha, A. M., Fernandes, E., & Soares, J. (2004). Solving asymmetric travelling salesman problems with lagrangian relaxation. *Optimization*, 1.

Rodríguez, A., & Ruiz, R. (2012a). The effect of the asymmetry of road transportation networks on the traveling salesman problem. *Computers & Operations Research*, *39*(7), 1566–1576.

Rodríguez, A., & Ruiz, R. (2012b). A study on the effect of the asymmetry on real capacitated vehicle routing problems. *Computers & Operations Research*, *39*(9), 2142–2151.

Rossi, F., van Beek, P., & Walsh, T. (2006). *Handbook of Constraint Programming.* Elsevier.

Rousseau, L. M., Gendreau, M., & Pesant, G. (2002). Using constraint-based operators to solve the vehicle routing problem with time windows. *Journal of Heuristics*, *8*, 43–58.

Ruiz, R., Maroto, C., & Alcaraz, J. (2004). A decision support system for a real vehicle routing problem. *European Journal of Operational Research*, *153*, 593–606.

Sarvanov, V., & Doroshko, N. (1981). The approximate solution of the traveling salesman problem by a local algorithm that searches neighborhoods of exponential cardinality in quadratic time. *Software: Algorithms and Programs*, *31*, 8–11.

Savelsbergh, M. (1985). Local search in routing problems with time windows. *Annals of Operations Research*, *4*(1), 285–305.

Savelsbergh, M. (1988). *Computer aided routing. Centrum voor Wiskunde en Informatica.*

Schaus, P., Deville, Y., Dupont, P., & Régin, J.-C. (2007). The Deviation Constraint. In *Cpaior* (pp. 260–274).

Schaus, P., Van Hentenryck, P., & Régin, J. C. (2009). Scalable Load Balancing in Nurse to Patient Assignment Problems. In *Cpaior* (pp. 248–262).

Schimpf, J., & Shen, K. (2012). Ecl$^{\text{i}}$ps$^{\text{e}}$ - from LP to CLP. *Theory and Practice of Logic Programming*, *12*(1-2), 127–156.

Shaw, P. (1998). Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. In (pp. 417–431).

Shaw, P. (2004). A Constraint for Bin Packing. In *Cp* (pp. 648–662).

Shmoys, D. B., & Williamson, D. P. (1990). Analyzing the Held-Karp TSP bound: A monotonicity property with application. *Information Processing Letters*, *35*(6), 281–285.

Shor, N. Z. (1985). The Subgradient Method. In *Minimization methods for non-differentiable functions* (Vol. 3, pp. 22–47). Springer Berlin Heidelberg.

Simonis, H. (2007). Models for Global Constraint Applications. *Constraints*, *12*(1), 63–92.

Sinz, C., & Iser, M. (2009). Problem-Sensitive Restart Heuristics for the DPLL Procedure. In *Sat* (pp. 356–362).

Smith, T. (1980). A LIFO implicit enumeration algorithm for the asymmetric travelling salesman problem using a one-arborescence relaxation. In *Combinatorial optimization* (pp. 108–114). Springer.

Smith, T., Meyer, T., & Thompson, G. (1990). Lower bounds for the symmetric travelling salesman problem from Lagrangean relaxations. *Discrete Applied Mathematics*, *26*(2-3), 209–217.

Smith, T., & Thompson, G. L. (1977). A LIFO implicit enumeration search algorithm for the symmetric traveling salesman problem using Held and Karp's 1-tree relaxation. *Annals of Discrete Mathematics*, *1*, 479–493.

Steeg, J., & Schröder, M. (2008). A Hybrid Approach to Solve the Periodic Home Health Care Problem. In J. Kalcsics & S. Nickel (Eds.), *Operations research proceedings 2007* (Vol. 2007, pp. 297–302). Springer Berlin Heidelberg.

Subramanian, A., Penna, P., Uchoa, E., & Ochi, L. (2012). A Hybrid Algorithm for the Heterogeneous Fleet Vehicle Routing Problem. *European Journal of Operational Research*, *221*(2), 285–295.

Taillard, E. (1993). Parallel Iterative Search Methods for Vehicle Routing Problems. *Networks*, *23*, 661–673.

Taillard, E. (1999). A heuristic column generation method for the heterogeneous fleet Vehicle Routing Problem. *RAIRO - Operations Research*, *33*(1), 1–14.

Tarantilis, C., & Kiranoudis, C. (2002). Boneroute: an Adaptative Memory-Based Method for Effective Fleet Management. *Annals of Operations Research*, *115*, 227–241.

Tarantilis, C., Kiranoudis, C., & Vassiliadis, V. (2004). A threshold accepting metaheuristic for the heterogeneous fixed fleet vehicle routing problem. *European Journal of Operational Research*, *152*(1), 148–158.

Timsjo, S. (1999). *An application of lagrangian relaxation to the traveling salesman problem* (Technical Report IMa-TOM-1999-02). Department of Mathematics and Physics, Malardalen University.

Toth, P., & Vigo, D. (2001). An overview of vehicle routing problems. In *The vehicle routing problem* (pp. 1–26). Philadelphia: SIAM - Society for Industrial and Applied Mathematics.

Toth, P., & Vigo, D. (2002a, NOV 15). Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Applied Mathematics*, *123*(1-3), 487–512.

Toth, P., & Vigo, D. (2002b). *The Vehicle Routing Problem*. SIAM.

Toth, P., & Vigo, D. (2003). The Granular Tabu Search and its Application to the Vehicle Routing Problem. *INFORMS Journal on Computing*, *15*, 333–346.

TSPLIB: ATSP benchmark. (2008). Retrieved from http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/atsp/

Turkensteen, M., Ghosh, D., Goldengorin, B., & Sierksma, G. (2008). Tolerance-based branch and bound algorithms for the ATSP. *European Journal of Operational*

Research, *189*(3), 775–788.

Uchida, A., Ito, Y., & Nakano, K. (2012). An efficient GPU implementation of ant colony optimization for the traveling salesman problem. In *Networking and computing (icnc), 2012 third international conference on* (pp. 94–102).

Vallejo, M., Vargas, P., & Corne, D. (2012, sept.). A fast approximative approach for the Vehicle Routing Problem. In *12th uk workshop on computational intelligence (ukci)* (pp. 1–8). UK.

Van Hentenryck, P., & Michel, L. (2009). *Constraint-based local search*. The MIT Press.

van Hoeve, W. J. (2001). The alldifferent Constraint: A Survey. In K. R. Apt, R. Barták, E. Monfroy, & F. Rossi (Eds.), *Proceedings of the 6th annual workshop of the ercim working group on constraints* (Vol. cs.PL/0105015).

van Hoeve, W. J. (2004). A Hyper-arc Consistency Algorithm for the Soft All different Constraint. In *Cp* (pp. 679–689).

Vigo, D. (1996, FEB 22). A heuristic algorithm for the asymmetric capacitated vehicle routing problem. *European Journal of Operational Research*, *89*(1), 108–126.

Volgenant, T., & Jonker, R. (1982). A branch and bound algorithm for the symmetric traveling salesman problem based on the 1-tree relaxation. *European Journal of Operational Research*, *9*(1), 83–89.

Voß, S. (2001). Meta-heuristics: The state of the art. In *Local search for planning and scheduling* (pp. 1–23). Springer.

Wallace, M. (2006). Hybrid Algorithms in Constraint Programming. In *Csclp* (pp. 1–32).

Wang, X., Wu, X., & Wang, Z. (2010). A Lagrangian heuristic for the urgency vehicle routing problem. In *Logistics systems and intelligent management, 2010 international conference on* (Vol. 3, pp. 1633–1637).

Wang, Y., Li, J., Gao, K., & Pan, Q. (2011). Memetic Algorithm based on Improved Inver–over operator and Lin–Kernighan local search for the Euclidean traveling salesman problem. *Computers & Mathematics with Applications*, *62*(7), 2743–2754.

Wassan, N. A., & Osman, I. H. (2002). Tabu search variants for the mix fleet vehicle routing problem. *The Journal of the Operational Research Society*, *53*(7), 768–782.

Wassenhove, L. V., & Gelders, L. (1980). Solving a bicriterion scheduling problem. *European Journal of Operational Research*, *4*(1), 42–48.

Wen, M., Clausen, J., & Larsen, J. (2010). *Rich Vehicle Routing Problems and Applications*. DTU Management.

Wong, L.-P., Khader, A. T., Al-Betar, M. A., & Tan, T.-P. (2013). Solving Asymmetric Traveling Salesman Problems using a generic Bee Colony Optimization framework with insertion local search. In *Intelligent systems design and applications (isda), 2013 13th international conference on* (pp. 20–27).

Wu, W., Lori, M., Martello, S., & Yagiura, M. (2014, Dec). Algorithms for the min-max regret generalized assignment problem with interval data. In *Industrial engineering and engineering management (ieem), 2014 ieee international conference on* (pp.

734–738).

Xing, L.-N., Chen, Y.-W., Yang, K.-W., Hou, F., Shen, X.-S., & Cai, H.-P. (2008). A hybrid approach combining an improved genetic algorithm and optimization strategies for the asymmetric traveling salesman problem. *Engineering Applications of Artificial Intelligence*, *21*(8), 1370–1380.

Yurtkuran, A., & Emel, E. (2010). A new hybrid electromagnetism-like algorithm for capacitated vehicle routing problems. *Expert Systems with Applications*, *37*(4), 3427–3433.

Zachariadis, E., & Kiranoudis, C. (2010). A strategy for reducing the computational complexity of local search-based methods for the vehicle routing problem. *Computers and Operations Research*, *37*, 2089–2105.

Zamani, R., & Lau, S. (2010). Embedding learning capability in Lagrangean relaxation: An application to the travelling salesman problem. *European Journal of Operational Research*, *201*, 82–88.

Zhao, X., Luh, P., & Wang, J. (1999). Surrogate Gradient Algorithm for Lagrangian Relaxation. *Journal of Optimization Theory and Applications*, *100*(3), 699–712.

Appendices

Appendix A

Spanish Airports Problem

GPS coordinates of the Spanish Airports problem which requires visiting once each of the 37 airports of Spain.

MAD	-3.562636	40.472023
BCN	2.080855	41.299711
PMI	2.735833	39.552776
AGP	-4.496944	36.674168
LPA	-15.384722	27.930277
ALC	-0.559583	38.283722
TFS	-16.571945	28.044167
ACE	-13.604167	28.947222
VLC	-0.481944	39.488609
IBZ	1.373056	38.86861
FUE	-13.8625	28.452778
BIO	-2.908333	43.299999
TFN	-16.341944	28.480278
GRO	2.762778	41.904999
SVQ	-5.893055	37.417221
MAH	4.219444	39.859444
SCQ	-8.416111	42.897499
MJV	-0.8125	37.773335
REU	1.166667	41.146946
XRY	-6.058889	36.745304
OVD	-6.033333	43.562778
SPC	-17.754723	28.626112
VGO	-8.63	42.234165
LEI	-2.377625	36.84446
GRX	-3.7775	37.18639
LCG	-8.379167	43.301388
SDR	-3.818889	43.42889
VLL	-4.851666	41.706112
ZAZ	-1.041667	41.666111
PNA	-1.647222	42.770557
EAS	-1.789167	43.35611
MLN	-2.954444	35.281944
VDE	-17.889444	27.810278
VIT	-2.726389	42.896946
BJZ	-6.821389	38.890278
SLM	-5.501389	40.952778
ODB	-4.848611	37.845554

Matrix distance of the Spanish Airports problem which requires visiting once each of the 37 airports of Spain.

0	483	546	430	1769	350	1821	1572	284	456	1629	319	1768	556	393	644	490	347	401	466	377	1840	464	415	370	512	330	163	251	301	352	578	1922	275	329	184	311
483	0	202	767	2183	392	2244	1973	295	277	2029	467	2193	94	809	216	893	459	82	866	701	2277	892	632	674	889	539	567	261	348	392	800	2354	428	802	648	706
546	202	0	709	2119	308	2186	1903	275	141	1957	628	2136	268	786	119	1011	376	224	829	840	2231	998	544	615	1017	695	670	394	511	566	690	2302	581	825	728	684
430	767	709	0	1418	401	1481	1216	471	569	1262	749	1431	859	150	828	767	333	695	139	752	1522	710	183	94	814	754	554	631	720	777	207	1594	705	319	488	135
1769	2183	2119	1418	0	1814	95	222	1889	1977	176	2045	90	2274	1385	2237	1767	1744	2109	1320	1909	223	1702	1574	1511	1826	2014	1810	2014	2066	2107	1443	225	2019	1457	1703	1483
350	392	308	401	1814	0	1880	1600	123	169	1654	583	1830	486	485	427	839	69	337	523	718	1923	809	243	307	863	626	510	367	496	562	407	1994	529	551	525	385
1821	2244	2186	1481	95	1880	0	306	1951	2045	266	2090	54	2335	1441	2305	1794	1810	2169	1379	1945	135	1731	1642	1575	1854	2056	1855	2068	2116	2154	1516	131	2066	1504	1747	1541
1572	1973	1903	1216	222	1600	306	0	1678	1762	56	1858	271	2065	1183	2021	1608	1530	1901	1116	1736	405	1541	1359	1299	1669	1831	1623	1813	1872	1915	1224	437	1829	1268	1518	1278
284	295	275	471	1889	123	1951	1678	0	171	1733	469	1900	389	524	387	770	176	228	575	627	1987	748	342	378	786	519	430	246	377	443	516	2062	416	550	468	420
456	277	141	569	1977	169	2045	1762	171	0	1816	608	1995	365	653	260	941	236	254	692	790	2090	920	402	475	955	666	600	370	501	563	551	2160	556	706	640	552
1629	2029	1957	1262	176	1654	266	56	1733	1816	0	1914	237	2121	1239	2075	1662	1582	1956	1172	1792	376	1596	1413	1355	1723	1887	1680	1869	1928	1971	1277	398	1885	1324	1574	1335
319	467	628	749	2045	583	2090	1858	469	608	1914	0	2036	489	699	681	460	610	409	775	249	2096	480	720	688	442	74	235	239	119	90	891	2182	54	589	343	627
1768	2193	2136	1431	90	1830	54	271	1900	1995	237	2036	0	2283	1389	2255	1740	1761	2117	1327	1890	139	1676	1593	1525	1799	2001	1800	2015	2062	2100	1469	169	2012	1450	1692	1488
556	94	268	859	2274	486	2335	2065	389	365	2121	489	2283	0	897	241	934	553	165	955	733	2366	940	726	767	925	563	622	316	374	405	894	2443	458	881	711	795
393	809	786	150	1385	485	1441	1183	524	653	1239	699	1389	897	0	904	641	425	731	77	656	1469	580	313	204	692	690	479	629	694	743	356	1547	661	180	393	103
644	216	119	828	2237	427	2305	2021	387	260	2075	681	2255	241	904	0	1091	495	275	948	908	2350	1084	662	734	1093	752	755	461	562	608	805	2421	639	936	822	801
490	893	1011	767	1767	839	1794	1608	770	941	1662	460	1740	934	641	1091	0	828	820	708	214	1776	68	845	760	60	389	341	633	563	550	968	1869	479	462	317	636
347	459	376	333	1744	69	1810	1530	176	236	1582	610	1761	553	425	495	828	0	400	458	723	1854	792	173	239	856	647	510	409	533	599	341	1925	556	506	511	327
401	82	224	695	2109	337	2169	1901	228	254	1956	409	2117	165	731	275	820	400	0	790	633	2200	816	570	604	819	478	489	187	290	342	742	2277	365	720	567	629
466	866	829	139	1320	523	1379	1116	575	692	1172	775	1327	955	77	948	708	458	790	0	732	1411	645	322	221	761	767	557	698	768	818	324	1487	737	246	470	162
377	701	840	752	1909	718	1945	1736	627	790	1792	249	1890	733	656	908	214	723	633	732	0	1937	247	783	718	192	176	214	448	359	339	932	2027	275	499	264	617
1840	2277	2231	1522	223	1923	135	405	1987	2090	376	2096	139	2366	1469	2350	1776	1854	2200	1411	1937	0	1714	1690	1616	1834	2057	1861	2089	2129	2164	1574	97	2074	1516	1752	1571
464	892	998	710	1702	809	1731	1541	748	920	1596	480	1676	940	580	1084	68	792	816	645	247	1714	0	798	708	129	414	328	631	574	570	913	1807	492	400	283	581
415	632	544	183	1574	243	1642	1359	342	402	1413	720	1593	726	313	662	845	173	570	322	783	1690	798	0	108	884	742	569	551	663	726	178	1759	669	445	536	240
370	674	615	94	1511	307	1575	1299	378	475	1355	688	1525	767	204	734	760	239	604	221	718	1616	708	108	0	802	700	511	551	648	708	214	1689	640	341	460	134
512	889	1017	814	1826	863	1854	1669	786	955	1723	442	1799	925	692	1093	60	856	819	761	192	1834	129	884	802	0	368	355	632	551	532	1012	1927	467	515	349	681
330	539	695	754	2014	626	2056	1831	519	666	1887	74	2001	563	690	752	389	647	478	767	176	2057	414	742	700	368	0	211	301	191	163	909	2145	113	564	313	626
163	567	670	554	1810	510	1855	1623	430	600	1680	235	1800	622	479	755	341	510	489	557	214	1861	328	569	511	355	211	0	307	282	304	724	1947	215	355	108	422
251	261	394	631	2014	367	2068	1813	246	370	1869	239	2015	316	629	461	633	409	187	698	448	2089	631	551	551	632	301	307	0	132	197	729	2170	188	580	394	535
301	348	511	720	2066	496	2116	1872	377	501	1928	119	2062	374	694	562	563	533	290	768	359	2129	574	663	648	551	191	282	132	0	66	839	2213	84	613	387	610
352	392	566	777	2107	562	2154	1915	443	563	1971	90	2100	405	743	608	550	599	342	818	339	2164	570	726	708	532	163	304	197	66	0	902	2249	89	650	413	663
578	800	690	207	1443	407	1516	1224	516	551	1277	891	1469	894	356	805	968	341	742	324	932	1574	913	178	214	1012	909	724	729	839	902	0	1637	841	527	674	332
1922	2354	2302	1594	225	1994	131	437	2062	2160	398	2182	169	2443	1547	2421	1869	1925	2277	1487	2027	97	1807	1759	1689	1927	2145	1947	2170	2213	2249	1637	0	2160	1600	1839	1648
275	428	581	705	2019	529	2066	1829	416	556	1885	54	2012	458	661	639	479	556	365	737	275	2074	492	669	640	467	113	215	188	84	89	841	2160	0	562	323	585
329	802	825	319	1457	551	1504	1268	550	706	1324	589	1450	881	180	936	462	506	720	246	499	1516	400	445	341	515	564	355	580	613	650	527	1600	562	0	252	206
184	648	728	488	1703	525	1747	1518	468	640	1574	343	1692	711	393	822	317	511	567	470	264	1752	283	536	460	349	313	108	394	387	413	674	1839	323	252	0	353
311	706	684	135	1483	385	1541	1278	420	552	1335	627	1488	795	103	801	636	327	629	162	617	1571	581	240	134	681	626	422	535	610	663	332	1648	585	206	353	0

Its optimal solution with value 7223 is {1, 28, 36, 23, 17, 26, 21, 27, 12, 34, 31, 30, 29, 19, 2, 14, 16, 3, 10, 9, 6, 18, 24, 32, 4, 25, 37, 15, 20, 8, 11, 5, 13, 7, 33, 35, 1}.

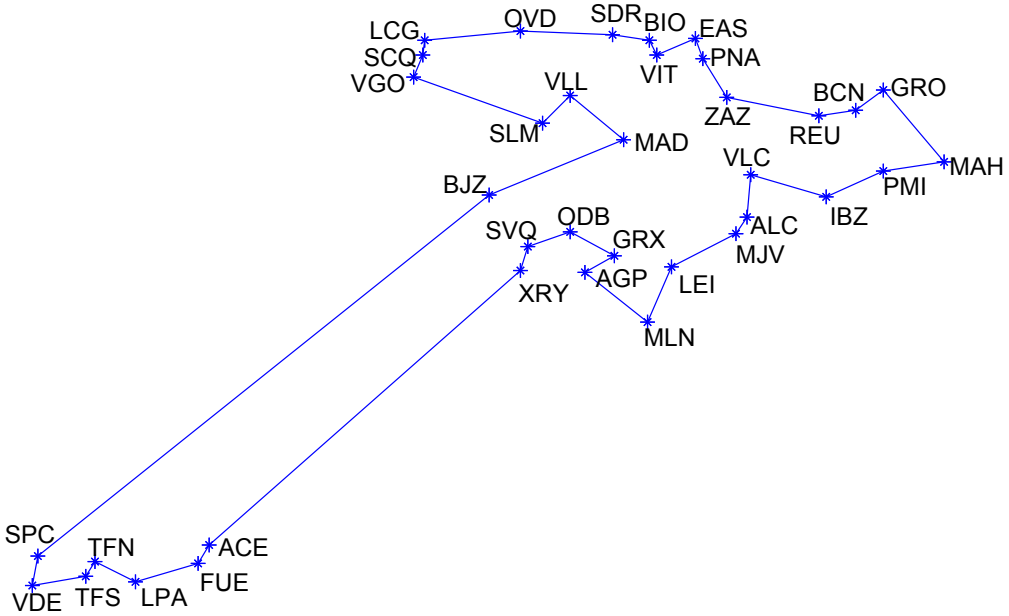


Figure: Optimal route of the Spanish Airport problem.