FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

A metaheuristic for the capacity-pricing problem in the car rental business

Luís Filipe Ferreira Soares

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Prof. Beatriz Brito Oliveira

July 6, 2020

Resumo

Atualmente, nas empresas de aluguer de automóveis, as decisões de capacidade e preço são definidas pela elevada interdependência e flexibilidade. Isso faz com que seja imperativo abordar essas decisões de maneira unificada, pois o impacto dessa abordagem é substancial. Esse tipo de questões são compostas por decisões de estratégias de *pricing* e gestão de frota. Esta dissertação procura abordar essas questões ao desenvolver uma metaheurística capaz de auxiliar no planeamento de um período de vendas para uma empresa de aluguer de automóveis. Os resultados obtidos devem ser comparados com os da literatura.

Neste trabalho, é inicialmente realizada uma revisão da literatura sobre o negócio de aluguer de automóveis, bem como sobre o estudo de metaheurísticas. Na primeira fase da revisão de literatura, o principal objetivo é entender o negócio de aluguer de automóveis e as suas características. Também é traçado um paralelo com outros setores que possuem problemas semelhantes, com o intuito de descobrir se já são usados métodos que possam ser relevantes para este trabalho. A segunda fase da revisão concentra-se nas metaheurísticas com o objetivo de perceber o que são, como funcionam, como são classificadas e onde podem ser aplicadas. Finalmente, a revisão termina com uma reflexão sobre o uso de metaheurísticas no negócio de aluguer de automóveis.

O problema específico da empresa de aluguer de automóveis em questão é então apresentado e representado num modelo matemático. Todas as decisões sobre gestão de frota e estratégias de *pricing* para os diferentes tipos de reservas de aluguer são descritas, bem como as políticas comerciais em vigor.

Após a exposição do problema, é realizada uma comparação das metaheurísticas estudadas para decidir qual a mais adequada para o problema em questão. Uma metaheurística baseada em Variable Neighborhood Search (VNS) é então proposta como uma solução. As estruturas de vizinhança são discutidas e definidas, bem como o procedimento para inicializar a solução e o método para escapar dos ótimos locais.

Os resultados finais mostraram que a abordagem escolhida não é completamente validada, já que o algoritmo proposto não foi capaz de produzir os resultados esperados. O método demonstrou dificuldades em encontrar soluções admissíveis e a escapar dos ótimos locais. Contudo, foi possível demonstrar a importância da selecção das metaheurísticas adequadas e do desenvolvimento das vizinhanças para a melhor adequação ao problema, em especial em problemas complexos e integrados como o abordado.

Abstract

Nowadays, in car rental companies, the capacity and pricing choices are defined by high interdependece and flexibility. This makes it so that it is imperative to address these types of choices in a unified manner, as the impact this approach has is substantial. These matters are comprised by decisions on pricing planning and on fleet definition, re-positioning and deployment. This dissertation aims to tackle these matters by designing a metaheuristic capable of aiding the planning of a selling season for a car rental company. The results obtained are meant to be compared to the ones in the literature.

In this work, a literature review is performed on the car rental business as well as on the study of metaheuristics. In the first phase of the review, the main goal is to understand the car rental business and its intricacies. It is also drawn a parallel to other sectors with similar issues in order to find out if there are already employed methods that could be relevant to this work. The second phase of the review focuses on metaheuristics with the purpose of learning what they are, how they work, how they are classified and where they can be applied. Finally the review ends with a reflection on the use of metaheuristics in regards to the car rental business.

The specific problem of a car rental company at hand is then presented and depicted in a mathematical model. All the decisions regarding fleet management and pricing strategies for the different rental requests are well described, as well as business policies in place.

After the problem statement, a comparison of the studied metaheuristics is conducted to then decide which one is the most adequate for the problem at hand. As a result, a Variable Neighborhood Search (VNS) metaheuristic is proposed as a solution. The neighborhood structures are discussed and defined, as well as the procedure to initialize the solution and the method to escape the local optima.

The final results showed that this approach is not completely validated as the proposed algorithm was not able to produce the expected results. The method displayed weaknesses in finding feasible solutions and escaping local optima. However, it was possible to show the importance of selecting the appropriate metaheuristics and the development of the neighborhoods so that they are the most adequate to the problem, especially in complex and integrated problems like the one tackled.

Agradecimentos

Em primeiro lugar quero deixar o meu sincero agradecimento à orientadora deste trabalho, Prof. Dr. Beatriz Oliveira. Obrigado por ter confiado em mim, por me ter ajudado incondicionalmente, pela motivação e pela paciência ao longo deste projeto.

Quero também agradecer à minha família pois sei que sempre fizeram de tudo para me proporcionar a oportunidade de chegar a esta posição. Devo-lhes tudo e nunca teria superado estes 5 anos sem o seu apoio.

Por fim agradecer a todos os meus amigos, sem eles seria muito mais difícil.

Luís Soares

“Prediction is very difficult,
especially about the future”

Niels Bohr

Contents

List of Figures

List of Tables

Abbreviations and Symbols

ACO	Ant Colony Optimization
ALSP	Adaptive Local Search Procedure
CaRS	Car Renter Salesman Problem
EA	Evolutionary Algorithm
FSP	Flow Shop Problem
GA	Genetic Algorithm
GVNS	General Variable Neighborhood Search
PNLP	Probabilistic Non-Linear Programming
PSO	Particle Swarm Optimization
SA	Simulated Annealing
TS	Tabu Search
TSP	Traveling Salesman Problem
VND	Variable Neighborhood Descent
VNS	Variable Neighborhood Search

Chapter 1

Introduction

The capacity-pricing problem in car rental has increasingly been stepping in the spotlight as companies began investing in advanced decision-support tools for these critical issues. When planning a sales period, a company must decide the number and type of vehicles needed in its fleet in order to meet demand. The demand for rental vehicles is particularly price-sensitive and therefore capacity and pricing decisions are closely linked. In addition, as the products are rented, the capacity "returns". This creates an association between capacity, fleet mobilization and other tools that allow the company to meet demand, such as upgrades, transferring vehicles between locations or the temporary leasing of additional vehicles.

The impact of solving this complex problem on a company's profit has already been estimated and evaluated, but when real-world problems are taken into account, the size and complexity of the problem makes existing methods slow and inadequate to provide solutions within a reasonable time.

Therefore, as one of the possible approaches, the main objective of this dissertation is then to select, design and develop an efficient metaheuristic that provides similar or better results than the ones obtained in the literature, namely in Oliveira et al. [15] where the problem is presented and modeled, and where the approach used to solve real-sized instances was a matheuristic that hybridized a genetic algorithm with mathematical models.

1.1 Structure of the Dissertation

Apart from the introduction, this dissertation contains four more chapters. In Chapter 2, a literature review of subjects that concern similar problems is presented. The problem description is detailed in Chapter 3, including its mathematical formulation. Chapter 4 details the development of the chosen metaheuristic and Chapter 5 presents its results. Finally, Chapter 6 states the conclusions and proposes future work.

Chapter 2

Literature Review

The literature review for this project is sectioned in three topics:

- The car rental business, to better understand its procedures and issues that might need to be taken into account later focusing, however, on the capacity-pricing problem. This topic is also expanded to other types of businesses that have similar characteristics;
- The study of metaheuristics, in order to identify and compare the best ones to apply to the problem at hand;
- And the junction of both topics, that is, the application of metaheuristics to the problem of capacity-pricing in the car rental business.

2.1 Introduction

The overall goals of this chapter are to establish how the car rental business works, the significance of its procedures, some of the issues that it faces, and then move on to the analysis of the more predominantly used metaheuristics in the literature. The core of this chapter is on defining these different methodologies that might serve a purpose in this field so as to later compare them and identify the appropriate approach for development.

2.2 Car Rental Business

Car rental is a growing market. In fact, its growth trajectory has been steady since 2010 and is forecasted to continue. From 2016 to 2021, the global car rental industry is expected to grow 5.6%, due to increasing tourism activities, the globalization of operations, and the global rise of income levels [14].

The recent global pandemic had a major impact on these activities due to the sudden and immense drop in tourism. The travel bans to stem the spread of Covid-19 have hit the car rental industry harder than most. In the United States, Avis, Hertz and Enterprise have asked the Treasury Department to include their industry in federal plans to rescue travel companies. On the other side of the Atlantic, France has underwritten loans worth 225 million euros requested by Europcar to maintain the continuity of its car rental business [21]. These new struggles make it all the more urgent to find good analytic decision support methods that guarantee financial sustainability.

Fleet management is one of the most recognized and addressed problems by car rental companies given that profitability is highly dependent on the fleet and all decisions that have to do with it. So, to deal with this issue, car rental companies must consider several factors:

- Their network design and the distribution of the fleet throughout it;
- The fleet definition and utilization, as car rental companies must make decisions on what types of vehicles and how many of which type they need, as well as strive to have every vehicle occupied all the time;
- The booking of requests and assigning vehicles to them;
- The uncertainty that derives from elements like selling/acquiring vehicles and, most importantly, varying demand.

Logistics management in the car rental business involves short-term decisions about the transportation and deployment of cars with regard to optimizing fleet utilization while maintaining a high service level. Fink and Reiners [5] model and solve this problem by means of minimum cost network flow optimization under consideration of essential practical needs such as multi-period planning, a country-wide network, customized transportation relations, fleeting and defleeting, and car groups with partial substitutability.

On the topics of fleet definition and distribution, Li and Tao [11] develop a two-stage dynamic programming model, in which the vehicle transfer policy is determined in the second stage and the optimal fleet size in the first stage, and they propose an heuristic solution for the fleet size problem.

Regarding the booking of incoming rental requests, Li and Pang [10] tackled this matter by considering dynamic booking control for a single-station car rental revenue management problem. As a solution, it was proposed a decomposition approach by developing two heuristics that outperform the more commonly used probabilistic non-linear programming (PNLP) heuristic in most of other instances.

The forecast of demand is also one of the key topics in logistic problems in car rental. Forecast is the basis for decision making. The car rental industry is different from airline or hotel industry. The difficulty of demand forecast lies in how to consider customers' behavior, such as the high rate of no-show, returning cars remotely, and the uncertainty of length-of-rent (which determines the date a car is returned) [26].

The capacity of car rental businesses is dynamic since the number of cars available might dynamically change due to the dynamic return time of the customers and the possibility of acquiring vehicles through leasing. Thus, pricing can become an issue. Masruroh et al. [12] propose optimal pricing models (non-linear programming with constraints) considering the customer segments using the car rental business as a case study.

In light of the works just presented, it is crucial to note that the problem addressed in this dissertation is different as it integrates fleet management decisions (e.g. aquisitions, transfers, etc.) for numerous stations and vehicle groups over time with pricing decisions, making all the more complex.

2.2.1 Similar Sectors

A myriad of sectors and industries have been using quantitative approaches to optimize or improve their fleet management processes. From maritime transportation to humanitarian aid, the need to efficiently manage a fleet of vehicles is extended across strategic levels and business functions [14]. In this section, there is an attempt to understand what type of research has been developed in the fleet management context in different sectors that show parallels to car rental.

The comparison to the airline industry is very common in the car rental literature. In terms of operational fleet decisions, there is indeed a significant amount of research in this field. Sherali et al. [22] provides a detailed review on assignment in the airline industry regarding fleet, tail and crew assignment, among other issues. As a particular question of this sector, tail assignment is the problem of assigning specific aircraft to flights, producing a fully operational, robust schedule which fulfills operational constraints [8]. The tail assignment problem can be seen as parallel to the problem of assigning accepted booking requests to vehicles in car rental.

The truckload carrying industry displays several fleet management problems that can be compared to the car rental industry. Powell [17] reviewed optimization models and algorithms for problems such as the assignment of drivers to pending loads or the distribution of vehicles among locations and dynamically moving them to meet new demand, the latter being a common practice in car rental [14].

Chakroborty et al. [1] address both issues of optimal fleet size distribution, which is also a problem present in car rental, and scheduling for a transit system. A simple binary coded Genetic Algorithm (GA) based approach to the optimization problem is proposed as the use of GA allows a more efficient formulation of the problem then traditional optimization methods, which are unable to give optimal solutions to even simple versions of the problem in question.

2.3 Metaheuristics

When dealing with large-scale optimization problems, exact methods may not be practical since they might require too many computational resources and take too long to reach the optimal solution. Metaheuristics are then used to provide satisfactory solutions within a reasonable amount of time, having applications in many areas like machine learning or planning and production problems.

Talbi [24] states that, in designing a metaheuristic, it should be taken into account the exploration of the search space (diversification) and the exploitation of the best solutions (intensification). This intensification means analyzing promising regions more intensely to find better solutions until eventually finding the local optima. This process can be described as searching for a peak on a mountain range while walking in a dense fog [13]. When you find this peak it might not be the highest in the mountain range, being then the “local optima". Finding better solutions requires visiting non-explored regions to assure that all regions of the search space are evenly explored and eventually discover the global optima: a solution that is as good or better than all other feasible solutions. Figure 2.1 illustrates the described difference between local and global optima.

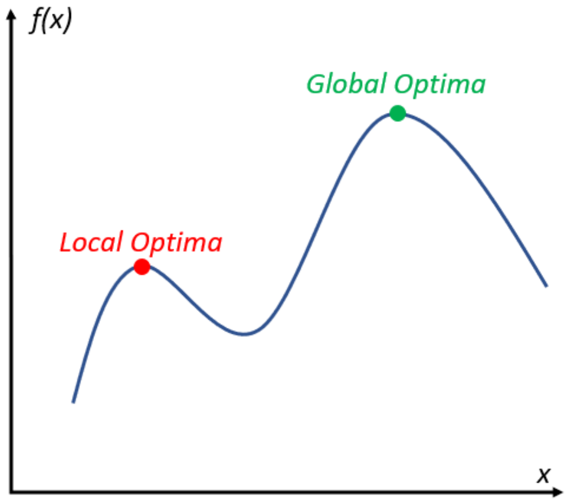

Figure 2.1: Local Optima versus Global Optima in a maximization problem

There are many classification criteria for metaheuristics. The ones stated in Talbi [24] are as follows (see Figure 2.2):

- Population Based or Single-Solution Based: Single-solution based algorithms (e.g., Variable Neighborhood Search) manipulate and transform a single solution during the search, whereas in population-based algorithms a whole set of solutions is transformed together in a evolution process, such as Genetic Algorithms.
- Nature inspired or Nonnature inspired: Many metaheuristics are inspired by natural processes from biology, physics or social sciences (e.g., Ant Colony Optimization, Particle Swarm Optimization).
- Memory or Memoryless: In memoryless metaheuristic algorithms no information extracted dynamically is used during the search. On the other hand, some metaheuristics use a memory that contains some information extracted during the search. For instance, short-term and long-term memories in Tabu Search.
- Stochastic or Deterministic: A deterministic metaheuristic solves an optimization problem by making deterministic decisions(e.g., Iterated Local Search). In stochastic metaheuristics, some random rules are applied during the search, like in Simulated Annealing.
- Iterative or Constructive: Iterative algorithms start with a complete solution and transform it at each iteration using some search operators. Greedy algorithms start from an empty

solution, and, at each step, a decision variable of the problem is assigned until a complete solution is obtained.

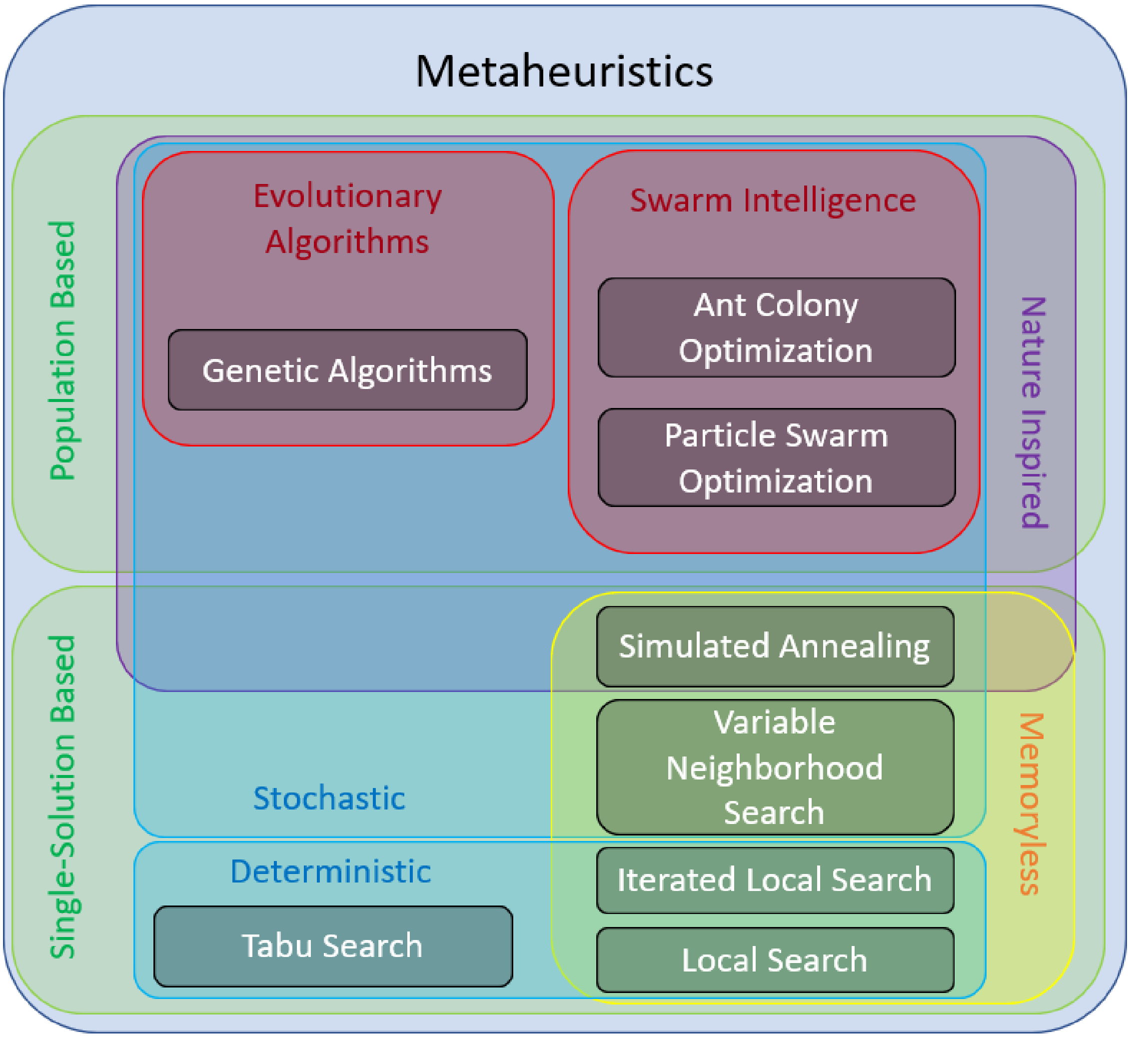

Figure 2.2: Classification of the metaheuristics analyzed in this report

The main differentiation made in this thesis is between Population and Single-Solution Based metaheuristics since these two families have complementary characteristics: single-solution based metaheuristics are exploitation oriented; they have the power to intensify the search in local regions. Population-based metaheuristics are exploration oriented; they allow a better diversification in the whole search space. In fact the algorithms belonging to each family of metaheuristics share many search mechanisms [24].

2.3.1 Single-Solution Based

Single-solution based metaheuristics improve a single solution and can be viewed as search trajectories performed by iterative procedures through the search space of the problem [24]. The definition of the neighborhood is a required common step for the design of any of these metaheuristics and is described by Michalewicz and Fogel [13] as the region of the search space that's “near" some particular point in that space, given that this nearness can be defined in many different ways.

The Traveling Salesman Problem (TSP) can be taken as an example. This is a problem where the objective is to, given a list of cities and the distances between each pair of cities, find the shortest possible route that visits each city and returns to the origin city. Therefore, a solution to this problem can be represented as an ordered sequence of the cities to visit. As for a neighborhood structure, it could be, for instance, the set of possible exchanges between two elements of the sequence, as portrayed in Figure 2.3.

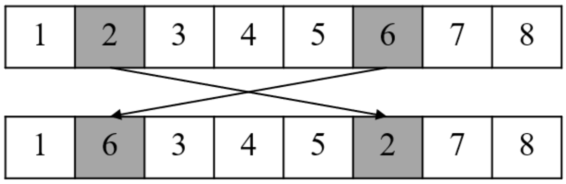

Figure 2.3: Exchange operator on a TSP problem

2.3.1.1 Local Search

This is the simplest method and the basis for most single-solution based metaheuristics. According to Michalewicz and Fogel [13], this procedure can be explained in four steps: starting with a given initial solution and evaluating its merit; applying a transformation to the current solution generating a new one; if the new solution is better than the current one then exchange them, otherwise discard it; repeat the last two steps until all replacement options are worse than the current solution.

The selection of the next incumbent neighbor has to be taken into account when designing a local search algorithm and there are various strategies that can be applied [24]:

- Best Improvement (steepest descent): the entire neighborhood is evaluated and the neighbor that improves the most the objective function is chosen;
- First Improvement (first descent): the first improving neighbor evaluated and that is better than the current solution is chosen;
- Random Selection: a random improving neighbor is chosen.

The main disadvantage of these algorithms is that they tend to converge towards the local optima. To avoid this, there are four different families of approaches:

- Accepting nonimproving neighbors (Simulated Annealing and Tabu Search - which will be described later in this section, along with other mentioned metaheuristics - are popular representatives of this class of algorithms);

- Starting with different initial solutions (used in algorithms like Iterated Local Search and Greedy Randomized Adaptive Search Procedure);
- Changing the neighborhood structure (used in Variable Neighborhood Search strategies);
- Changing the objective function or perturbing the input data (used in algorithms like Guided Local Search);

Simulated Annealing, Tabu Search, Iterated Local Search and Variable Neighborhood Search are some of the more commonly employed metaheuristics using the Local Search procedure.

Simulated Annealing Simulated Annealing is a stochastic and memoryless algorithm that allows the degradation of a solution under some conditions in order to escape the local optima. It simulates the energy changes in a system subjected to a cooling process until it converges to an equilibrium state [24].

This algorithm is summarized as starting with an initial solution and with each iteration a random neighbor is generated; moves that improve the cost function are always accepted, otherwise the neighbor is selected with a given probability that usually follows the Boltzmann distribution. A control parameter T (temperature) is used to determine the probability of accepting a nonimproving solutions. The current solution and the best one found since the beginning of the search are stored. The main design issues of this metaheuristic are related to the acceptance probability function and the cooling schedule, which sets the temperature at each step of the algorithm.

Van Laarhoven et al. [25] utilize a SA approach for finding the minimum makespan in a job shop scheduling problem.

Tabu Search Michalewicz and Fogel [13] explain that the main concept behind this algorithm is to use a memory that forces the search to analyze new areas of the search space. Some solutions that have been examined are memorized and then become tabu points. This way these points are avoided in the making of decisions about selecting the next solution. Tabu Search can also be described as a steep Local Search algorithm that accepts nonimproving solutions to escape from the local optima when all neighbors are nonimproving solutions [24].

The whole neighborhood is explored in a deterministic manner and when a better neighbor is found it replaces the current solution. At the point when a local optima is reached the best noninproving solution is chosen. This can generate cycles so, in order to avoid that from happening, neighbors that have been visited before are discarded and a list of solutions or moves recently applied is managed and updated each iteration - the aforementioned tabu list. However this list can sometimes be too restrictive. So, for some conditions denominated *aspiration criteria*, tabu solution can be accepted.

Gendreau et al. [6] developed a tabu search heuristic for the vehicle routing problem with capacity and route length restrictions.

Iterated Local Search The quality of the local optima obtained by a local search method depends on the initial solution. As we can generate local optima with high variability, Iterated Local Search may be used to improve the quality of successive local optima as, after a local search is applied to an initial solution, a perturbation of the obtained local optima is carried out at each iteration. Finally, a local search is applied to the perturbed solution. The generated solution is accepted as the new current solution under some conditions. This process iterates until a given stopping criteria [24].

Stützle and Stützle [23] consider the application of ILS to the permutation flow shop problem (FSP), a strongly studied problem in machine scheduling.

Variable Neighborhood Search The main objective of Variable Neighborhood Search (VNS) is to delve into a set of predefined neighborhoods to provide a better solution, exploring either randomly or methodically in order to escape from the local optima and move towards the global optima.

VNS is a stochastic algorithm in which, first, a set of neighborhood structures N_k (k = 1, ... , n) are defined. Then, each iteration of the algorithm is composed of three steps: shaking, local search, and move. At each iteration, an initial solution is shaken from the current neighborhood N_k. For instance, a solution x' is generated randomly in the current neighborhood $N_k(x)$. A local search procedure is applied to the solution x' to generate the solution x''. The current solution is replaced by the new local optima x'' if and only if a better solution has been found. The same search procedure is thus restarted from the solution x'' in the first neighborhood N_1. If no better solution is found, the algorithm moves to the next neighborhood N_{k+1}, randomly generates a new solution in this neighborhood, and attempts to improve it. It is also important to notice that cycling is possible (i.e., $x'' = x$) [24].

Expanding on VNS, Variable Neighborhood Descent (VND) is a deterministic version of it that has two reoccurring phases: descent to find local optima (by combining the steepest and first descent search methods) and perturbation to get out of the valley. By replacing the simple local search procedure of the basic VNS algorithm with the VND algorithm a different is created called General Variable Neighborhood Search (GVNS).

VNS is a very capable metaheuristic which is commonly be applied to, for instance, routing problems. This can be seen in the work developed in Hemmelmayr et al. [9].

2.3.2 Population Based

Talbi [24] presents population-based metaheuristics as an iterative improvement in a set of solutions. Firstly, the population is initialized, then a new population of solutions is generated and finally this new population is integrated into the current one using selection procedures. This process stops once a given stopping criteria is met. Evolutionary Algorithms and Swarm Intelligence methods are some of the procedures in this category and are described in the next subsections.

2.3.2.1 Evolutionary Algorithms

Evolutionary algorithms are the most studied population based metaheuristics predicated on the notion of competition and representing the stimulation of species. An objective function associates a fitness value with every individual indicating its suitability to the problem. At each step, individuals are selected to form the parents, following the selection paradigm in which individuals with better fitness are selected with a higher probability. Then, selected individuals are reproduced using variation operators (e.g., crossover, mutation) to generate new offsprings. Finally, a replacement scheme is applied to determine which individuals of the population will survive from the offsprings and the parents. This iteration represents a generation. They have been highly successful in continuous or combinatorial optimization, system modeling and identification, planning and control, engineering design, data mining and machine learning [24].

Genetic Algorithms This is one of the most popular classes of Evolutionary Algorithms and is usually related to the use of chromosomes to represent the solutions, these being arrays of either binary values or real values between 0 and 1. A Genetic Algorithm applies crossover operators to two solutions and mutation operators that randomly modify the individual contents or randomly generate new solutions to promote diversity. These algorithms use a probabilistic selection and the replacement is generational, as the parents are replaced by the offsprings. GAs have many applications, one of them being for flowshop scheduling, as Reeves [18] presents.

2.3.2.2 Swarm Intelligence

The algorithms in this class are inspired by the collective behavior of species such as ants, bees and birds. They are mainly characterized by the cooperation of the particles by an indirect communication medium and their movements in the decision space [24]. Two of the most known/used algorithms in this class are presented in this section.

Ant Colony Optimization As the name suggests, the idea of these algorithms is to imitate the behavior of real ants. An ant colony can find the shortest path between two points by creating a pheromone trails between these two points for the ants to follow. The larger the amount of pheromone on a particular path, the larger the probability that the ants will select the path. Then, the solution construction is done in a probabilistic way by artificial ants that are considered as stochastic procedures and the target optimization problem can be seen as a decision graph where an ant will construct a path. This iterative process considers the pheromone trails (memory of the characteristics of good generated solutions) and the problem-dependent heuristic information. Parpinelli et al. [16] apply an ACO algorithm as a data mining tool, with the goal of extracting classification rules from data.

Particle Swarm Optimization In Talbi [24], particle swarm optimization algorithms are described as stochastic metaheuristics that mimic the behavior of natural organisms like bird flocking and fish schooling. A swarm consists of N particles in a D-dimensional search space, with each particle i being a candidate solution represented by the vector x_i and successively adjusting its position toward the global optimum. Optimization uses the cooperation between particles. The success of some particles will influence the behavior of others. PSO has many applications involving artificial neural networks, as stated in Eberhart and Shi [4].

2.4 Metaheuristics Implementation in Car Rental

The specific use of metaheuristics as an approach to tackle the characteristic problems of the car rental business is extremely scarce. A few instances can be found , however, they are mostly adaptations of the Traveling Salesman Problem to the car rental sector: the Traveling Car Renter Salesman Problem (CaRS). Both Rios et al. [20] and da Silva and Ochi [3] propose the hybrid use of an Evolutionary Algorithm with a Adaptive Local Search Procedure, EA+ALSP, and obtained positive results. In this problem, a customer (car renter) desires to visit some places (cities) using a vehicle while minimizing his expenses towards the car rental company. Since this tackles a different perspective of the situation and doesn't address issues like pricing and fleet management, it is not the desired approach for the problem considered in this dissertation.

Nevertheless, as also mentioned before, metaheuristics are commonly applied to individual areas that are or can be related to the car rental business, as presented in Table 2.1, based on Gogna and Tayal [7].

Table 2.1: Metaheuristics applications

	Previous Applications
SA	Job shop scheduling Vehicle routing in transport and logistics management
TS	Resource allocation in industry Artificial Intelligence: pattern recognition, data mining
ILS	Job shop scheduling Flow shop scheduling Clustering Problems
VNS	Location problems Data mining Graph problems Time tabling Scheduling Vehicle routing Arc routing
GA	Financial planning, stock predictions
ACO	Scheduling: flow shop scheduling, process planning
PSO	System simulation and identification Decision making and planning

Chapter 3

Problem Statement

In this chapter, the problem at hand is presented at first, followed by its detailed representation in a mathematical model, explaining every index, parameter, variable, constraint and the objective function. The goal of using the mathematical model is to fully describe and understand the business constraints, goals, and decisions that are incorporated in the capacity-pricing problem in car rental.

3.1 Introduction

This work deals with the integration of two of the main decisions that car rentals face: determining the capacity of their fleet – which includes decisions on acquisition modes and timings (longer term buying and selling, or shorter term leasing), as well as fleet deployment between locations, in order to meet demand – and defining the price of the variety of rentals that are requested [15]. Thus, the primary purpose of this work is to design and develop a metaheuristic capable of improving a company's profitability by integrating the capacity with the pricing decisions since profitable utilization of the fleet is the main priority of the company presented below.

Since in this dissertation the approach taken and results in Oliveira et al. [15] will serve as a baseline comparison, it is important to consider the same context, setting and problem details. The problem tackled in this work concerns a case inspired by a car rental company needing support in planning a selling season (1-3 months). This planning entails deciding on the acquisition and fleet capacity plan as well as the overall pricing strategy, which will be described in detail with the Mathematical Model.

The company has approximately 40 rental stations in Portugal, divided into four regions that serve as "units of location" when tackling the tactical/strategic problems. All regions share the same fleet, which comprises roughly 10,000 vehicles and is split in owned and leased, as well as divided in up to 5 vehicle groups (e.g., compact, minivan, SUV, etc.), depending on the selling season. The purchase of owned vehicles is planned with a certain advance and these vehicles are kept as a part of the fleet for the entire season. On the other hand, leased vehicles are used to help with peaks in demand and are only kept for their corresponding leasing period. Depending on the selling season, this company can deal with 450–2500 different rental types which are priced individually and characterized by start and end locations, as well as dates and required vehicle group.

In order to meet demand, the company regularly performs "empty transfers", which consists in moving a vehicle from one location with less demand to another with more requests, either by truck or using a driver from the staff. Another practice used by this company to meet demand is to offer upgrades (i.e., offering a more valued vehicle group for the same price as the requested group) when the requested group is not available so that they can maximize the utilization of the fleet. However, upgrades are usually avoided and used only as a last resource.

3.2 Mathematical Model

The mathematical model used to fully define the problem is the one proposed by Oliveira et al. [15] in order to make the new metaheuristic approach fully comparable. In this model, the objective is to maximize the company's profit, which is the difference between the revenues obtained with the rentals fulfilled and the costs of leasing/acquiring fleet, performing the "empty transfers" and maintaining the owned fleet, as well as a penalization factor for upgrades.

This model sets four different main groups of constraints:

- Stock calculating constraints, where the stock of vehicles of each group in each time period and station is computed;
- Capacity/Demand constraints, that limit the number of rentals fulfilled and empty transfers realized;
- Business constraints, where the limitations regarding possible upgrades and available purchase budget are established;
- Other auxiliary constraints.

3.2.1 Indices and Parameters

$t, t' = \{0, ..., T\}$	Index for the set $\mathscr{T}$ of time periods, where $t = 0$ represents the initial conditions of the time horizon (season) and "overlaps" with $t = T$ for the previous season
$g, g1, g2 = \{1, ..., G\}$	Index for the set $\mathscr{G}$ of vehicle groups
$s, s1, s2, c = \{1, ..., S\}$	Indices for the set $\mathscr{S}$ of rental locations
$r = \{1, ..., R\}$	Index for the set $\mathscr{R}$ of rental types (characterized by check-out and check-in location and time period, and vehicle group requested)
$sout_r$	Check-out location of rental type r
sin_r	Check-in location of rental type r

$dout_r$	Check-out time period of rental type r
din_r	Check-in time period of rental type r
gr_r	Vehicle group requested by rental type r
$a = \{0, ..., A\}$	Index for the set $\mathscr{A}$ of antecedences allowed (number of time periods between the rental request and the start of the rental), where $a = 0$ represents a "walk-in" customer
$p = \{1, ..., P\}$	Index for the set $\mathscr{P}$ of price levels allowed
PRI_{pg}	Pecuniary value associated with price level p for vehicle group g (for example, for group $g = 2$, price level $p = 1$ has a pecuniary value of $PRI_{1,2} = 20$ EUR)
DEM_{rap}	Demand for rental type r, at price level p, with antecedence a
COS_g	Buying cost of a vehicle of group g. The value considered is the net cost: purchase gross cost minus salvage value derived from its sale after one year
LEA_g	Leasing cost (per time unit) of a vehicle of group g
OWN_g	Ownership cost (per time unit) of a vehicle of group g
LP_g	Leasing period for a vehicle of group g
PYU	Penalty charged for each upgrade
UPG_{g1g2}	Whether a vehicle of group $g1$ can be upgraded to a vehicle of group $g2$ $(= 1)$ or not $(= 0)$
TT_{s1s2}	Transfer time from location $s1$ to location $s2$
TC_{gs1s2}	Transfer cost of a vehicle of group g from location $s1$ to location $s2$
BUD	Total budget for the purchase of vehicles
M	Big-M large enough coefficient

3.2.1.1 Other sets:

$\mathscr{R}^{g^-}$	Rental types that do not require group g
$\mathscr{R}^{in}_{st}$	Rental types whose check-in is at location s at time $\in [t-1, t[$
$\mathscr{R}^{out}_{st}$	Rental types whose check-out is at location s at time $\in [t-1, t[$
$\mathscr{R}^{use}_{t}$	Rental types that require a vehicle to be in use at t (i.e., $dout < t \wedge din \geq t$)

3.2.1.2 Inputs from previous seasons (previous decision periods):

INX^{O}_{gs} Initial number of owned (O) vehicles of group g located at s, at the beginning of the season ($t = 0$)

$ONY^{L/O}_{gts}$ Number of owned (O) or leased (L) vehicles of group g on on-going empty transportation (previously decided), being transferred to location s, arriving at time t

$ONU^{L/O}_{gts}$ Number of owned (O) or leased (L) vehicles of group g on on-going rentals (previously decided), being returned to location s at time t

3.2.2 Decision Variables

w^{O}_{gs} Number of vehicles of group g acquired for the owned fleet available at $t = 0$ in location s

w^{L}_{gts} Number of vehicles of group g acquired by leasing to be available at time t in location s

q_{rap} $= 1$ if price level p is charged for rental type r with antecedence a; $= 0$ otherwise

$x^{L/O}_{gts}$ Number (stock) of leased (L) or owned (O)vehicles of group g located at s at time t

$y^{L/O}_{s1s2gt}$ Number of leased (L) or owned (O) vehicles of group g empty transferred at time t from location $s1$ to location $s2$

$u^{L/O}_{rag}$ Number of fulfilled rentals requested as rental type r with antecedence a that are served by a leased (L) or owned (O) vehicle of group g

$f^{L/O}_{gt}$ Auxiliary variable: total leased (L) or owned (O) fleet of group g at time t

3.2.3 Objective Function

Eq. (3.1) represents the objective function of the model, which aims to maximize the profit of the company. As presented before, this function is the difference between the revenue earned from the fulfilled rentals (3.1a) (number of rentals fulfilled times the priced charged for each one) and all the costs: purchasing/leasing the fleet (3.1b), maintaining the owned fleet (3.1c), performing the empty transfers (3.1d) and a penalty factor for offering upgrades (3.1e). The last member of this calculation is needed so that these upgrades are avoided and only used as a last resort.

$$\max \quad \sum_{r=1}^{R}\sum_{a=1}^{A}\left(\left(\sum_{g=1}^{G} u^{L}_{rag} + u^{O}_{rag}\right)\sum_{p=1}^{P} q_{rap} PRI_{p,gr_r}\right) \tag{3.1a}$$

$$- \sum_{g=1}^{G}\left(\sum_{s=1}^{S} w^{O}_{gs}\right) COS_g - \sum_{g=1}^{G}\left(\sum_{t=1}^{T} f^{L}_{gt}\right) LEA_g \tag{3.1b}$$

$$- \sum_{g=1}^{G}\left(\sum_{t=1}^{T} f^{O}_{gt}\right) OWN_g \tag{3.1c}$$

$$- \sum_{s1=1}^{S}\sum_{s2=1}^{S}\sum_{g=1}^{G}\left(\sum_{t=1}^{T}\left(y^{L}_{s1s2gt} + y^{O}_{s1s2gt}\right)\right) TC_{gs1s2} \tag{3.1d}$$

$$-\sum_{g=1}^{G}\sum_{r\in\mathscr{R}^{g^-}}\sum_{a=1}^{A}\left(u^L_{rag}+u^O_{rag}\right)PYU \tag{3.1e}$$

3.2.4 Constraints

Eqs. (3.2)-(3.6) are the aforementioned *Stock Calculating Constraints*. These represent the stock of available vehicles of a certain group, in a specific location, at a specific time. Eq. (3.2) expresses the stock of owned vehicles of each group, in each location, for each time period except the initial one. The stock is equal to the one of the previous period, increased by expected arrivals from rentals and empty transfers that started on previous seasons (parameters) and by the arrival of vehicles that were being employed in rentals that started this season and were meanwhile returned to this specific location, decreased by the vehicles that were meanwhile occupied by rentals that started in this location, increased also by the vehicles that were being empty-transferred from other locations and have meanwhile arrived, and finally decreased by the vehicles that were transferred to other locations.

Eqs. (3.3) and (3.4) represent an identical situation but applied to the leased fleet. The main difference of this type of fleet is that acquisitions may occur throughout the season. Thus, in these equations, the stock is increased with the corresponding leasing acquisitions and then removed from the fleet after the leasing period (*LP*) is over, as it can be seen in (3.4).

The stock of owned and leased fleet for the beginning of the season is specified, respectively, by Eqs. (3.5) and (3.6).

Capacity/Demand Constraints: Eq.(3.7) expresses that, at a given location and time period, the number of rentals fulfilled and the empty transfers that start at that location and time are limited by the stock of available cars. Eq. (3.8) also ensures that the number of rentals fulfilled is limited by the demand for the rental type, at the chosen price level.

The *Business-related Constraints* are Eqs. (3.9) and (3.10). The former represents the upgrading policies, i. e. what groups of vehicles can each group be upgraded to, and the latter limits the number of purchased vehicles in each time period by the total available budget.

Other Constraints: Eq. (3.11) makes sure that only one price level is chosen per rental type and antecedence. Eq. (3.12) represents the previously presented auxiliary variable $f^{L/O}_{gt}$. And finally, Eq. (3.13) delineates the domain of the decision variables.

$$\begin{aligned}\textbf{s.t. } x^O_{gts} &= x^O_{g,t-1,s}+ONY^O_{gts}+ONU^O_{gts}\\ &+\sum_{r\in\mathscr{R}^{in}_{s,t}}\sum_{a=1}^{A}u^O_{r,a,g}-\sum_{r\in\mathscr{R}^{out}_{s,t}}\sum_{a=1}^{A}u^O_{r,a,g}\\ &+\sum_{c=1}^{S}y^O_{c,s,g,t-TT_{cs}-1}-\sum_{c=1}^{S}y^O_{s,c,g,t-1} \qquad \forall g,t>0,s\end{aligned} \tag{3.2}$$

$$x^L_{gts} = x^L_{g,t-1,s}+ONY^L_{gts}+ONU^L_{gts}$$

$$+\sum_{r\in\mathscr{R}_{s,t}^{in}}\sum_{a=1}^{A}u_{r,a,g}^{L}-\sum_{r\in\mathscr{R}_{s,t}^{out}}\sum_{a=1}^{A}u_{r,a,g}^{L}$$

$$+\sum_{c=1}^{S}y_{c,s,g,t-TT_{cs}-1}^{L}-\sum_{c=1}^{S}y_{s,c,g,t-1}^{L}$$

$$+w_{g,t-1,s}^{L} \qquad \forall g, 0<t\leq LP_{g}, s \tag{3.3}$$

$$x_{gts}^{L}=x_{g,t-1,s}^{L}+ONY_{gts}^{L}+ONU_{gts}^{L}$$

$$+\sum_{r\in\mathscr{R}_{s,t}^{in}}\sum_{a=1}^{A}u_{r,a,g}^{L}-\sum_{r\in\mathscr{R}_{s,t}^{out}}\sum_{a=1}^{A}u_{r,a,g}^{L}$$

$$+\sum_{c=1}^{S}y_{c,s,g,t-TT_{cs}-1}^{L}-\sum_{c=1}^{S}y_{s,c,g,t-1}^{L}$$

$$+w_{g,t-1,s}^{L}-w_{g,t-LP_{g}-1,s}^{L} \qquad \forall g, t>LP_{g}, s \tag{3.4}$$

$$x_{g0s}^{O}=INX_{gs}^{O}+w_{gs}^{O} \qquad \forall g,s \tag{3.5}$$

$$x_{g0s}^{L}=0 \qquad \forall g,s \tag{3.6}$$

$$\sum_{r\in\mathscr{R}_{st}^{out}}\sum_{a=1}^{A}u_{rag}^{L/O}+\sum_{c=1}^{S}y_{scgt}^{L/O}\leq x_{gts}^{L/O} \qquad \forall g,t,s \tag{3.7}$$

$$\sum_{g=1}^{G}\left(u_{rag}^{L}+u_{rag}^{O}\right)\leq DEM_{rap}+M(1-q_{rap}) \qquad \forall r,a,p \tag{3.8}$$

$$\sum_{a=1}^{A}\left(u_{rag}^{L}+u_{rag}^{O}\right)\leq UPG_{gr_{r},g}\times M \qquad \forall r,g \tag{3.9}$$

$$\sum_{s=1}^{S}\sum_{g=1}^{G}w_{gs}^{O}COS_{g}\leq BUD \tag{3.10}$$

$$\sum_{p=1}^{P}q_{rap}=1 \qquad \forall r,a \tag{3.11}$$

$$f_{gt}^{L/O}=\sum_{s=1}^{S}x_{gts}^{L/O}+\sum_{r\in\mathscr{R}_{t}^{use}}\sum_{a=1}^{A}u_{r,a,g}^{L/O}$$

$$+\sum_{s1=1}^{S}\sum_{s2=1}^{S}\sum_{t'=\max\{0,t-TT_{s1s2}\}}^{t-1}y_{s1,s2,g,t'}^{L/O} \qquad \forall g,t \tag{3.12}$$

$$q_{rap}\in\{0,1\} \qquad \forall r,a,p$$

$$w_{gts}^{L}\in Z_{0}^{+} \qquad \forall g,t,s$$

$$
\begin{aligned}
w^{O}_{gs} &\in Z^{+}_{0} && \forall g, s \\
y^{L/O}_{s1s2gt} &\in Z^{+}_{0} && \forall s1, s2, g, t \\
x^{L/O}_{gts} &\in Z^{+}_{0} && \forall g, t, s \\
u^{L/O}_{rag} &\in Z^{+}_{0} && \forall r, a, g \\
f^{L/O}_{gt} &\in Z^{+}_{0} && \forall g, t
\end{aligned}
\tag{3.13}
$$

In conclusion, the problem at hand integrates very interconnected decisions of pricing and capacity. The main goal then is to maximize the company's profit in a way that promotes sustainability by maximizing the fleet's occupancy and the prices charged. It is also of note topics like upgrades (and corresponding constraints) carry some extra complications and intricacies to the problem and the business standpoint.

Chapter 4

Solution Method

In this chapter, the proposed solution for the problem introduced in the previous chapter is presented. First, the choice of the metaheuristic to employ is explained, based on the comparison of the ones studied. Then, the solution method development is presented in detail.

4.1 Metaheuristic Selection

As already stated, the purpose of this dissertation is to develop a metaheuristic capable of solving the problem presented in the previous section which is, as already mentioned, very complex. The amount of distinct and intricate decisions and the different interdependences between variables make it so that drawing parallels to other types of problems is not so simple. Therefore, it is key to analyze and compare the studied methodologies in order to bridge those gaps and make an informed decision on which ones could provide the desired results. With that in mind, each one of the metaheuristics presented in Chapter 2 is evaluated in regards to its implementation difficulty and suitability to the problem's characteristics, either based on personal assessment or on previous uses found in the literature (see Table 4.1).

Regarding the complexity of the implementation, the population based metaheuristics (like GA and PSO) were estimated to most likely be the hardest to realize for this problem. On the other hand, algorithms like SA or ILS should not require such a high effort.

The problem's characteristics are divided in three subjects: Pricing, Fleet Management and Demand. On the latter, various works were found using the GA, PSO or SA algorithms, although VNS shows promise. The same can be said for these methods regarding Pricing problems as, judging from papers found on this issue, it is possible to apply them. On Fleet Management, all the presented metaheuristics exhibit capabilities to tackle different topics in this subject matter.

Considering this, the conclusion reached is that VNS should be the most all-around adequate method for this problem, taking also into account that the SA algorithm could also be a good second option.

Table 4.1: Metaheuristics comparison

	Method's Difficulty	Problem's Characteristics			Discussion
		Pricing	Fleet Management	Demand	
SA	Low	Yes	Vehicle scheduling Vehicle routing Fleet Sizing	Can integrate demand	Good option, but maybe not the most efficient. Easy implementation
TS	Medium	No	Fleet re-positioning Vehicle routing Fleet Sizing	No	Should give good results. Sensitive balance: being effective versus minimizing computational time and memory
ILS	Low	No	Vehicle scheduling Vehicle routing	No	Probably not the most efficient. Easy implementation
VNS	Low/Medium	Yes	Fleet re-positioning Vehicle routing Fleet Sizing Vehicle Scheduling	Maybe	Good option, should give good results, efficient. Implementation should not be too hard
GA	Medium/High	Yes	Fleet Deployment Vehicle routing	Various Applications and integrations	Should provide good results, but maybe too close to the approach done in the baseline comparison. Implementation might present have some issues
ACO	Medium	No	Vehicle scheduling Vehicle routing	No	Maybe not the most suitable option. Implementation might have some issues
PSO	Medium/High	Yes	Vehicle scheduling Vehicle routing	Various Applications and integrations	Maybe not the most suitable option Implementation might present some issues

4.2 Solution Method Specification

This section is focused on detailing the steps taken in constructing a solution capable of solving the problem presented. These steps include the generating the initial solution, defining the local search procedure and outlining the VNS method used as a way to escape the local optima.

4.2.1 Preliminary Testing

Before moving into the development of the method intended to solve the problem, there was a simple preliminary test done on a Traveling Salesman Problem (TSP) as a way to practice and validate the thought process behind the VNS implementation.

As explained before, given a collection of cities and the cost of travel between each pair of them, TSP aims to find the cheapest way of visiting all of the cities and returning to your starting point. In the standard version, the travel costs are symmetric in the sense that traveling from city X to city Y costs just as much as traveling from Y to X, but there are also asymmetric versions of this problem [2].

The solution for this problem was structured as an ordered set of the visited cities and was initialized at random. In the built up algorithm, the VNS algorithm was fed two sets of neighborhood structures for the local search procedure to travel through: the first one was defined by swapping the position in the ordered solution of two cities chosen at random, whereas the second one was simply inverting the order of the entire solution.

The experiment was made with an instance from the TSPLIB library [19], consisted of 29 cities with distances between them ranging from 32 to 386, and a known optimal cost value of 1610. A very quick experiment with a stopping criteria of only 3000 iterations shows that the algorithm converges to a value close to the optimal result over time (see Figure 4.1). Further analysis suggests that the performance of the method is highly dependent on a good initial solution, which reveals the importance of this matter moving forward to the main problem tackled in this dissertation.

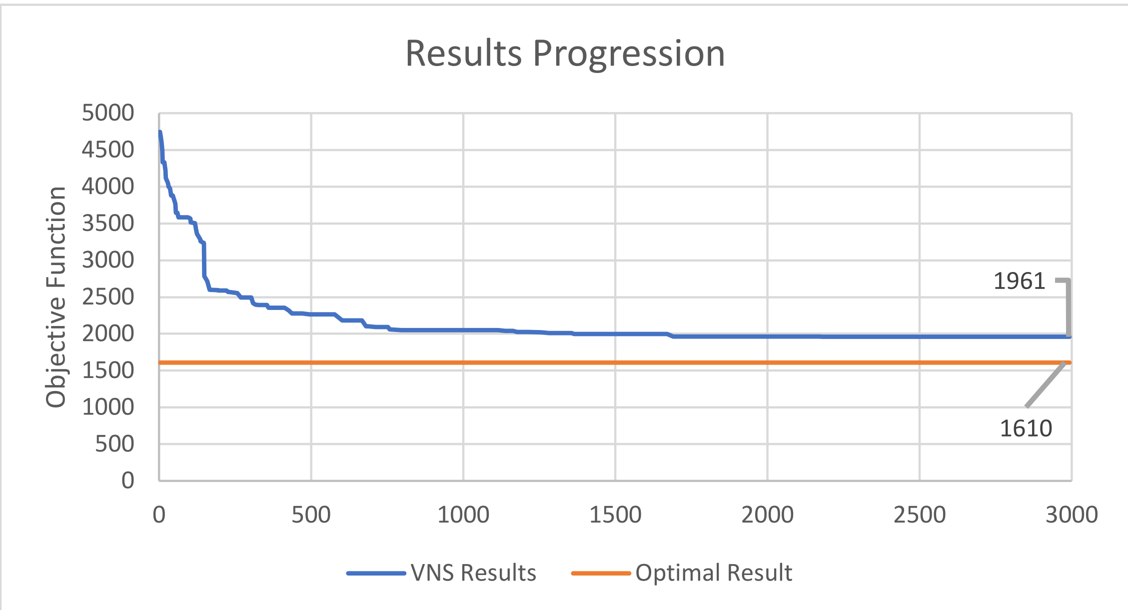

Figure 4.1: Temporal progression of VNS applied to the TSP (3000 iterations)

4.2.2 Obtaining an Initial Solution

As previously mentioned, the results of this work are to be compared to the ones in the literature. Thus, having a good initial solution can facilitate the algorithm being more efficient and effective. With that in mind, a constructive heuristic was developed to generate an initial solution for the VNS algorithm to iterate on. It should also be referenced that a completely zeroed out solution is a viable second option. The same can be said about a solution chosen at random, given that it is proven feasible in the context of the problem.

The heuristic starts off, in a somewhat greedy move, by choosing the highest price level p for each antecedence of each rental type r, establishing then the correspondent demand. The next step is to determine the number of acquired vehicles in the beginning of the season. This is done by adding all the vehicles possible within the budget cap BUD, while spreading them evenly throughout all the locations s and groups of vehicles g, not considering, however, the demand in each location. Then it iterates through all the rental types r (sorted by ascending starting date) and each of its antecedences a to fulfill all of the demand possible with the owned fleet. When this is not possible, vehicles acquired (of the requested group g) by leasing are added to the fleet and allocated to meet the remainder of demand. It also takes into account that when leased vehicles finish their rentals in a different locations then the ones they started in, they must return to it. To sustain this logic, in these cases, vehicles perform an "empty-transfer" back to the starting point. With each rental fulfilled or "empty transfer" is performed, the number of total and stocked vehicles is updated in the locations s and time periods t that the rental types r make reference to. A detailed depiction of this heuristic is presented in Algorithm 1.

4.2.3 Local Search

Having obtained an initial solution, the need now arises to improve by resorting to the chosen metaheuristic. To do so, the neighborhoods to which the local search procedure is applied to need to be set.

By examining the decision variables inherent to the mathematical model it is evident that both $x_{gts}^{L/O}$ (available stock of owned and leased vehicles) and $f_{gt}^{L/O}$ (total fleet of owned and leased vehicles) are a result of others. Thus, the movements that allow moving from a solution to its neighbor are based on applying changes to the remaining variables and calculating these two afterwards. The initial thought process was to set different neighborhoods by applying movements (see Table 4.2) to each variable individually.

However, the nature of the model and these variables is of interconnection and interdependence. It is very difficult to apply a movement to a single variable and ensure the feasibility of the overall solution. For instance, for a certain rental type r with antecedence a, the demand set by the price level q_{rap} chosen is 15 and the number of fulfilled rentals $u_{rag}^{L/O}$ is also 15. If the move of going up a price level is applied to the variable q_{rap}, then the demand will go down to 10, and since now the number of fulfilled rentals is greater then the demand the solution becomes unfeasible.

Algorithm 1 Constructive Heuristic – Initial Solution

```
Repeat
    Add a vehicle of each group to each location
    Update stock of owned vehicles of respective group, in respective location for all time periods
Until All budget is spent
For All rental types
    For All antecedences
        Choose highest price level and set demand accordingly
        If Demand ≤ No. of owned vehicles in stock at respective location and time period
            Fulfill all demand with owned vehicles of requested group
            Update stock of owned vehicles of respective group, in respective location for rental period and periods after return
        Else
            Fulfill possible demand with owned vehicles of requested group
            Update stock of owned vehicles of respective group, in respective location for rental period and periods after return
            If Rental time + Transfer Time ≤ Leasing Period
                If Starting location = Final location
                    Fulfill the rest of the demand with leased vehicles of requested group
                    Update stock of leased vehicles of respective group, in respective location for periods after return until leasing period ends
                Else
                    Fulfill the rest of the demand with leased vehicles of requested group
                    Update stock of leased vehicles of respective group, in respective location for periods after return
                    Perform an empty transfer with these vehicles back to the starting location just before the leasing period ends
```

Algorithm 1: Constructive Heuristic – Initial Solution

With this in mind, it was then settled that the best course of action was to define one neighborhood that moves the whole solution. This neighborhood is defined by every combination of the following movements:

- Go up/down a price level;
- Add/remove one vehicle to/from the owned fleet;
- Add/remove one vehicle to/from the leased fleet;
- Increase/decrease by one the number of fulfilled rentals served by a owned vehicle;
- Increase/decrease by one the number of fulfilled rentals served by a leased vehicle;
- Increase/decrease by one the number of owned vehicles empty transferred;

Table 4.2: Possible neighborhood defining movements for each variable

	Possible movements
q_{rap}	Go up/down 1 or 2 price level(s)
w^O_{gs}	Add/remove 1 or 2 vehicle(s) to/from the owned fleet Swap the number of vehicles between 2 or 3 locations Swap the number of vehicles between 2 or 3 groups
w^L_{gts}	Add/remove 1 or 2 vehicle(s) to/from the leased fleet Swap the number of vehicles between 2 or 3 locations Advance/delay by 1 or 2 time period(s)
$y^{L/O}_{s1s2gt}$	Increase/decrease by 1 or 2 the number of owned/leased vehicles empty transferred Change final location Advance/delay by 1 or 2 time period(s)
$u^{L/O}_{rag}$	Increase/decrease by 1 or 2 the number of fulfilled rentals served by a owned/leased vehicle Change group

- Increase/decrease by one the number of leased vehicles empty transferred.

The local search procedure travels through this neighborhood by iterating through the rental types r and antecedences a, using these to specify the values in the indices for the decision variables accordingly. This algorithm selects the neighbor following a first improvement logic, meaning that the local search returns the first improving and feasible neighbor that is better than the current solution.

4.2.4 Escaping Local Optima

As already acknowledged before, local search procedures are neighborhood-based: they apply some changes to the solution (which may be more or less significant) and evaluate if improvements occurred with these changes. Depending on the location of the solution in the solution space, local search is an intuitive manner of achieving significant improvements in the problem's objective function. However, these methods can only take us so far. In fact, because the search space is usually filled with valleys, searching within a solution's neighborhood leads the algorithm to be stuck in local optima. It is, therefore, important that the algorithm is able to search other portions of the solution space in order to avoid sub-optimality as far as possible.

It is with that purpose that neighborhood metaheuristics exist, namely VNS. This procedure intends to escape local optima and explore additional sections of the solution space by introducing structured neighborhoods and a *shaking* phase. Algorithm 2 provides a view of VNS, slightly adjusted to this problem.

In the *shaking* phase of the algorithm, the neighbor solution on which the local search is applied to is chosen at random. For this specific case, that is achieved by randomizing the rental type r, the antecedence a and the changes made to each variable, within the ones made possible in the specified neighborhood. Take as an example the variable q_{rap} (which determines the price

Algorithm 2 Variable Neighborhood Search Algorithm

Input: Neighborhood structure N for shaking and initial solution x_0 previously generated
$x = x_0$
Repeat
    Shaking: pick a random solution x' from the neighborhood $N(x)$ of x
    If $f(x') > f(x)$
        $x = x'$
    x'' = local search(x')
    If $f(x'') > f(x)$
        $x = x''$
Until Time limit reached
Output: Best found solution

Algorithm 2: Variable Neighborhood Search Algorithm

level charged). For the randomly chosen rental type r with random antecedence a, it is randomly selected if the price level goes up or down a level.

Chapter 5

Computational Experiments and Results

The purpose of this chapter is to present the results to the computational experiments performed (and how these were done) on the method described in section 4.

5.1 Instance Description

For the purpose of the computational experiments of this work, a baseline instance was adopted to evaluate the developed algorithm. This instance encompasses 8 rental types, 3 vehicle groups, each one with 4 price levels. The number of locations is equal to 2 and 4 antecedence levels were considered. The planning horizon consisted of 12 time periods in the total season. The levels of demand on the rentals varied between 0 and 40 per time period (depending on the antecedence and price levels).

The algorithm was implemented in a C++ environment, making use of the Standard Template Library (STL) whenever possible. All computational experiments were conducted on an Intel Core i5-6300U CPU @ 2.40GHz, running the Windows 10 operating system.

5.2 Results Obtained

The implementation of the previously specified constructive heuristic was successful, as the developed algorithm was able to obtain a feasible solution in a negligible amount of time. However, preliminary results obtained from the subsequent steps of the method showed that, during the *shaking* phase of VNS, the method was not capable of finding a feasible solution within the neighborhood from which to apply the local search procedure. This suggests that the movements that were designed to escape local optima are rather restrictive in terms of its feasibility.

Taking into account these challenges, further algorithm tuning was performed. The first attempt consisted in removing some of the random aspects in the *shake* procedure. Thus, the movements made by each variable were fixed in advance, but the random factor was still maintained in choosing the rental type r and the antecedence a. This attempt turned out unsuccessful, with very little solution improvement. Even though it was now capable of finding a feasible solution to proceed to the local search algorithm, the only better and feasible solution that was found in this search was the initial solution, making the algorithm dive into a loop in which the initial solution was consecutively obtained. Figure 5.1 illustrates the algorithm's temporal progression with the mentioned adjustments in place. As it can be seen, even though some feasible solutions can be found as a result of the shaking phase, the local search procedure is never capable of finding a better solution then the initial one.

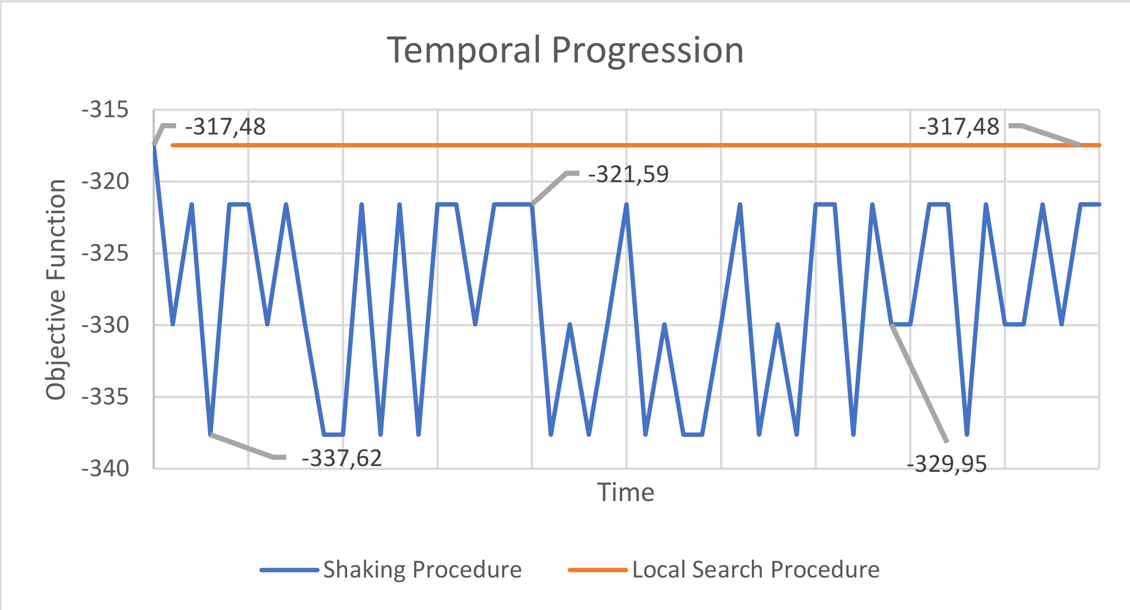

Figure 5.1: Temporal progression of the algorithm

As it can be seen, some feasible solutions can be found as a result of the shaking phase, all of them being worse than the initial one. However, the local search procedure is never capable of finding a better solution than the one initially generated. Therefore, the algorithm always returns that same result value and the solution never improves (orange line).

As a second attempt, given that, in the context of the problem, not accepting/performing any rentals is a feasible (although non-economically viable) solution, the algorithm was fed a zeroed-out initial solution. However, this proved to not even improve on the first attempt as it again wasn't able to find a random feasible solution in the neighborhood. Thus, it was not even capable of executing the local search procedure.

In sum, the difficulties that were encountered suggest that the problem is more constrained than initially perceived, due to the feasibility problems of the solutions. This fact may be attributed to the large spectrum of interdependent decisions that are considered in this problem, that is based in a real-life case-study. These interdependences probably fragment the solution space in a way that might trigger the need for more robust application of the heuristic. Due to this fragmentation, the neighborhood structure chosen might not have been the most adequate given its limitations. It is then possible that by adding moves as some of the ones presented in Table 4.2, for instance, to the tested neighborhoods, and therefore increasing their sizes, that a fix to these issues could be found. Another course of action could be implementing different successive neighborhoods to the VNS method instead of just one. These could preferably be in increasing degrees of complexity to accommodate the intricacies of the problem and facilitate the escape from local optima.

Chapter 6

Conclusions and Future Work

The conclusions attained in this chapter are based on the results presented in the previous chapter, as well as the objectives delineated in the beginning of this work.

The main goal outlined for this project since its start, was to construct and apply a metaheuristic to a problem integrating capacity and pricing, based on a real-life case-study on a car rental company, so that a direct comparison could be made with the results achieved in the literature, namely in Oliveira et al. [15]. With these objectives in mind, a method based on the Variable Neighborhood Search metaheuristic was implemented , as well as a constructive heuristic designed to create an initial solution for the problem. The latter was a successful implementation as the mentioned heuristic achieved a feasible solution in an efficient manner.

On the other hand, the development of the VNS metaheuristic revealed some complications. Inabilities to find feasible solutions in the solution space in the set neighborhood and difficulty in escaping local optima impeded its progress. These suggest that the designed neighborhood probably is not the most adequate for the complex interdependences that characterize the problem at hand and that somewhat restrict the feasible solutions domain.

6.1 Future Work

For future work, it would be necessary to address the discovered complications in order to improve the algorithm's performance.

Regarding the issue of escaping local optima, improvements can be made by diversifying the movements made either by adjusting the size of the neighborhoods or even altering the typology of these movements. Some of the unused movements considered in Table 4.2 could serve as possible alterations to apply.

There are also possibilities to revamp the algorithm as a way to deal with the problems that derive from the feasibility question. One of them is trying to extend the search space to the non-feasible domain, and also applying penalties. Another possible course of action would be implementing procedures that, when one of the variables is changed, set up and calculate other variables according to their interdependences and the restrictions inherent to the problem.

References

[1] Chakroborty, P., Deb, K., and Sharma, R. K. (2001). Optimal fleet size distribution and scheduling of transit systems using genetic algorithms. *Transportation Planning and Technology*, 24(3):209–225.

[2] Cook, W. (2015). Traveling salesman problem - university of waterloo. http://www.math.uwaterloo.ca/tsp/problem/index.html.

[3] da Silva, A. R. V. and Ochi, L. S. (2016). An efficient hybrid algorithm for the Traveling Car Renter Problem. *Expert Systems with Applications*, 64:132–140.

[4] Eberhart, R. C. and Shi, Y. (2001). Particle swarm optimization: Developments, applications and resources. In *Proceedings of the IEEE Conference on Evolutionary Computation, ICEC*, volume 1, pages 81–86.

[5] Fink, A. and Reiners, T. (2006). Modeling and solving the short-term car rental logistics problem. *Transportation Research Part E: Logistics and Transportation Review*, 42(4):272–292.

[6] Gendreau, M., Hertz, A., and Laporte, G. (1994). Tabu search heuristic for the vehicle routing problem. *Management Science*, 40(10):1276–1290.

[7] Gogna, A. and Tayal, A. (2013). Metaheuristics: review and application. *Journal of Experimental & Theoretical Artificial Intelligence*, 25(4):503–526.

[8] Grönkvist, M. and Kjerrström, J. (2005). Tail Assignment in Practice. pages 166–173. Springer, Berlin, Heidelberg.

[9] Hemmelmayr, V. C., Doerner, K. F., and Hartl, R. F. (2009). A variable neighborhood search heuristic for periodic routing problems. *European Journal of Operational Research*, 195(3):791–802.

[10] Li, D. and Pang, Z. (2017). Dynamic booking control for car rental revenue management: A decomposition approach. *European Journal of Operational Research*, 256(3):850–867.

[11] Li, Z. and Tao, F. (2010). On determining optimal fleet size and vehicle transfer policy for a car rental company. *Computers and Operations Research*.

[12] Masruroh, N. A., Tjakra, V. P., and Ratinghayu, R. R. (2017). Optimal pricing considering customer categories: Case on car rental industries. In *2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, volume 2017-Decem, pages 65–69. IEEE.

[13] Michalewicz, Z. and Fogel, D. B. (2004). *How to Solve It: Modern Heuristics*. Springer Berlin Heidelberg, Berlin, Heidelberg.

[14] Oliveira, B. B., Carravilla, M. A., and Oliveira, J. F. (2017). Fleet and revenue management in car rental companies: A literature review and an integrated conceptual framework. *Omega*, 71:11–26.

[15] Oliveira, B. B., Carravilla, M. A., and Oliveira, J. F. (2018). Integrating pricing and capacity decisions in car rental: A matheuristic approach. *Operations Research Perspectives*, 5(October):334–356.

[16] Parpinelli, R. S., Lopes, H. S., and Freitas, A. A. (2002). Data mining with an ant colony optimization algorithm. *IEEE Transactions on Evolutionary Computation*, 6(4):321–332.

[17] Powell, W. (1991). Optimization Models and Algorithms: An Emerging Technology for the Motor Carrier Industry. *IEEE Transactions on Vehicular Technology*, 40(1):68–80.

[18] Reeves, C. R. (1995). A genetic algorithm for flowshop sequencing. *Computers and Operations Research*, 22(1):5–13.

[19] Reinelt, G. (2020). Tsplib - library of sample instances for the tsp. http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsplib.html.

[20] Rios, B. H. O., Goldbarg, E. F. G., and Goldbarg, M. C. (2017). A Hybrid Metaheuristic for the Traveling Car Renter Salesman Problem. In *2017 Brazilian Conference on Intelligent Systems (BRACIS)*, volume 2018-Janua, pages 276–281. IEEE.

[21] Serres, P. (2020). How COVID-19 impacts car rental industry. https://www.globalfleet.com/en/leasing-and-rental/smart-mobility/global/features/how-covid-19-impacts-car-rental-industry.

[22] Sherali, H. D., Bish, E. K., and Zhu, X. (2006). Airline fleet assignment concepts, models, and algorithms. *European Journal of Operational Research*.

[23] Stützle, T. and Stützle, T. (1998). Applying iterated local search to the permutation flow shop problem.

[24] Talbi, E.-G. (2009). *Metaheuristics*. John Wiley & Sons, Inc., Hoboken, NJ, USA.

[25] Van Laarhoven, P. J., Aarts, E. H., and Lenstra, J. K. (1992). Job shop scheduling by simulated annealing. *Operations Research*, 40(1):113–125.

[26] Yazao Yang, Wenzhou Jin, and Xiaoni Hao (2008). Car rental logistics problem: A review of literature. In *2008 IEEE International Conference on Service Operations and Logistics, and Informatics*, volume 2, pages 2815–2819. IEEE.