Car rental logistics: a metaheuristic and a matheuristic for the vehicle-reservation assignment problem

Maria Beatriz Brito Oliveira

Master's Dissertation

Supervisor: Prof. Maria Antónia Carravilla

Faculdade de Engenharia da Universidade do Porto
Mestrado Integrado em Engenharia Industrial e Gestão

2013-07-03

Abstract

Car rental companies face a critical problem related to the empty reposition of their vehicles, which comprise significant and avoidable costs and increase their environmental impact due to the fuel consumption and CO_2 emission. Especially when dealing with special types of vehicles whose number of units is small, the company is forced to empty transfer them between rental stations in order to meet reservation requirements concerning available time and location. This dissertation describes and structures this problem, considering it within the operational vehicle-reservation assignment framework. The main objective is to develop a tool that assigns reservations to special vehicles, maximizing the company's profit whilst minimizing the impacts of these empty transfers.

A *metaheuristic* and a *matheuristic* were developed to tackle this problem. The *metaheuristic* developed is based on a GRASP algorithm. The greediness of the randomized constructive heuristic is based on several criteria with different importance degrees, such as reservation profit and time proximity, and vehicle idle time. Two local search procedures were developed. The move that defines the neighbourhood structure is in both cases based on the swap of pairs of allocated reservations, yet the approach to the selection of the new incumbent solution differs. Real instances were used to test this method.

Furthermore, this problem was formulated as a network-flow model. As the exact solution of real-sized instances is generally deemed impractical due to the processing time required, a *matheuristic* was developed to solve it. It comprises a relax-and-fix-based heuristic procedure, which includes a mechanism that enables and controls the changes between consecutive iterations, based on the local branching paradigm. Random instances based on real data were generated to test this method.

Both approaches were able to improve the company's profit and reduce the empty transfer time and consequent environmental impact, when comparing with the current procedures. The *matheuristic* was able to achieve significantly better results and tackle larger instances while the *metaheuristic* was able to allocate more reservations than the company's current procedure. Another major advantage of the utilization of these tools is the re-allocation of two qualified and experienced employees to other value-adding tasks.

Resumo

O reposicionamento de veículos em vazio é um problema crítico que traz às empresas de *rent-a-car* custos significativos e evitáveis, assim como um impacto ambiental negativo devido ao consumo de combustível e emissões de CO_2. Isto acontece especialmente no caso de tipos especiais de veículos que a empresa detém em menores quantidades, pois, para ser capaz de responder aos pedidos dos clientes, é obrigada a reposicioná-los entre estações. Nesta dissertação, esta questão é enquadrada no problema de afetação de reservas a veículos. O principal objetivo deste trabalho é, assim, desenvolver uma ferramenta que efetue esta afetação, maximizando o lucro total da empresa através da minimização dos custos destas reposições em vazio.

Para resolver este problema, uma *metaheurística* e uma *matheurística* foram desenvolvidas. A *metaheurística* é baseada num algoritmo GRASP. Neste, a heurística construtiva aleatorizada é "gulosa" e baseia a graduação dos elementos a inserir em vários critérios, como o lucro e a proximidade temporal das reservas e o tempo de paragem dos veículos. Duas abordagens à pesquisa local foram desenvolvidas, sendo a vizinhança em ambos os casos determinada por trocas entre reservas alocadas. A diferença entre as abordagens encontra-se na seleção da solução incumbente. Instâncias reais foram utilizadas para testar este método.

Este problema foi ainda formulado como um modelo de fluxo em rede. Como a resolução exata de instâncias de tamanho realista é normalmente limitada pelo tempo de corrida, tornando-se muitas vezes impossível, uma *matheurística* foi desenvolvida. Este método compreende uma heurística baseada no método *relax-and-fix*. Compreende ainda uma restrição, baseada no método de *local branching*, que permite modificar parte da solução entre iterações consecutivas de forma controlada. Instâncias aleatórias baseadas em dados reais foram geradas e utilizadas para testar este método.

Tanto a *metaheurística* como a *matheurística* melhoraram o lucro da empresa, comparando com os procedimentos atuais utilizados. Ao mesmo tempo, o tempo gasto em reposicionamentos foi também reduzido e, consecutivamente, o impacto ambiental da empresa diminuiu. A *matheurística* conseguiu resultados consideravelmente melhores e foi capaz de trabalhar com instâncias maiores, enquanto que a *metaheurística* conseguiu alocar mais reservas que os procedimentos atuais da empresa. Outra vantagem da utilização deste métodos é a realocação de dois funcionários qualificados e experientes, que podem desta forma ser encarregues de tarefas de maior valor acrescentado.

Acknowledgements

Firstly and above all, I would like to thank my supervisor, Prof. Maria Antónia Carravilla for the possibility to be a part of this project. Her confidence in me and support were the main drivers for my personal and professional growth throughout this semester at the Operations Research Lab. Above all, I am very grateful for her friendship and for the opportunity to work and learn from her.

I would also like to thank Prof. Franklina Toledo, from ICMC (Instituto de Ciências Matemáticas e de Computação), of the University of São Paulo, Brazil, whose friendly monitoring of my advances and retreats resulted in crucial insights and advices that were steer-defining for this dissertation. As for other contributors to this project, I am also grateful for the expertise and attention of Prof. José Fernando Oliveira. Moreover, I would like to thank Dr. Delfina Acácio who kindly enabled the access to the data needed.

Above all, I ought to thank all the people in the OR Lab: Prof. Miguel Gomes, Prof. Bernardo Almada Lobo, Elsa Silva, Diana Lopez, Teresa Bianchi de Aguiar, Luís Guimarães, Gonçalo Figueira, Pedro Rocha, Pedro Amorim, Marcos Furlan, Márcio Belo, Sam Heshmati, and Cleber Rocco. They have become major role models for me. Besides their brilliant minds and absurd devotion to this wonderful thing that they do, they are a real team and extremely supportive of each other. I am deeply grateful for their warm welcome and for the friends they have become. They are all a major part of this dissertation, thanks to the essential technical help they have generously and willingly provided me.

I would also like to thank all my big, messy, amazing family, but especially my mother, father and sister, for their unconditional support and for helping me become who I am today. Mãe, Pai, please know that I am very proud when someone says I am terribly alike either one of you. Thank you for teaching and always reminding me what is truly important. Above all, thank you for always being the kind of person I strive to become. Mimi, thank you for the companionship of a lifetime – you have always brought me balance and perspective. I am very grateful for our friendship; you have taught me much, even if you do not realise it. Finally, I want to thank Bernardo, who is also a major part of this family, for all that we have grown together over the last years. Your constant support, care and joy have given me a strength and drive I had not realised I have.

"Come sarebbe bello il mondo se ci fosse una regola per girare nei labirinti."

'How beautiful the world would be
if there was a procedure for moving through labyrinths."

Umberto Eco, in *Il Nome della Rosa*

Contents

Acronyms

DSS	Decisions Support System
GRASP	Greedy Randomized Adaptive Search Procedure
LB	Local Branching
LP	Linear Program
MDVSP	Multiple Depot Vehicle Scheduling Problem
PIP	Pure Integer Programming Problem
MIP	Mixed Integer Programming Problem
RCL	Restricted Candidate List
RF	Relax-and-Fix

List of Figures

List of Tables

Chapter 1

Introduction

The car rental business is becoming heavily dependent on operational efficiency. As holding costs of assets have been growing faster than the price level, it is important to assure optimal utilization of resources (Fink and Reiners, 2006). At the same time, the importance of the car rental industry within the transportation sector is growing significantly due to several factors, such as the increase of air passenger traffic. Moreover, there is an increasing need of efficient and flexible decision support software for the management of fleets, customer relations and other key activities, in order to achieve competitive advantages. (ITM, 2013)

At the same time, as environmental awareness globally awakes, the sustainability within this business area is emerging as a competitive advantage, as recent sustainable travel policies are leading travel managers to search for ground transportation suppliers based on their environmental programs or measures (Global Business Travel Association, 2012 as cited in Auto Rental News (2012)). The car rental company described in this dissertation, specifically, has developed a 5 year plan to reduce the environmental impact in 20%, namely as far as CO_2 emissions are concerned. In fact, the field of Green Logistics, which presents specific concerns about vehicle routing and scheduling, has been increasingly studied. Sbihi and Eglese (2007) state that Green Logistics considers the logistic problems within a sustainable framework, both in terms of environmental and social impact. The authors emphasize the need to specifically measure the environmental benefits of the optimization of routes; nonetheless, the reduction in the travel distance is said to be in itself valuable, as it leads to a reduction on fuel consumption.

Car rental companies face a critical problem related to the *empty reposition* of their vehicles, which comprise significant and avoidable financial and environmental impacts. Especially when dealing with special types of vehicles whose number of units is small, the company is forced to empty reposition them between rental stations in order to meet reservation requirements concerning available time and location. The main objective of this dissertation is to develop efficient tools to assign reservations to special vehicles, maximizing the company's profit whilst minimizing the impacts of these empty transfers.

The short-term logistic problem in car rental companies has been described and structured by Pachon et al. (2003). The authors propose a three sequential stage framework, which comprises *pool segmentation* - clustering the rental locations in geographically and demand-correlated pools, *strategic fleet planning* - setting the number of vehicles for each pool, and *tactical fleet planning* - determining the number of vehicles that should be available at each station, in each period of time. The vehicle-reservation assignment problem consists on allocating reservations to specific vehicles and can be considered as a sub-problem of the tactical fleet planning stage.

The problem addressed in this dissertation concerns a Portuguese car rental company, whose fleet is divided in two categories - the *free sale vehicles* and the *special vehicles*, each comprehending several rental groups. The main focus is on the *special vehicles* that either belong to expensive lines or have distinct characteristics (for example, minivans and off-road vehicles). Therefore, the company has a smaller number of vehicles belonging to each rental group within this category. The problem described contemplates them since they are often not located in the required stations leading to the need to perform costly and polluting empty repositions.

Two approaches were drawn in order to tackle this problem. The first methodology selected was a *metaheuristic* based on a GRASP (Greedy Randomized Adaptive Search Procedures) algorithm. This is an iterative technique that is based on two sequential phases: the construction of a solution based on a randomized greedy heuristic, and an improvement phase based on local search. This methodology was chosen due to its intuitive structure and relatively simple implementation. Considering real instances, the tool developed based on the GRASP algorithm was able to assign reservations to vehicles whilst improving the company's profit up to 12%, also reducing the time spent in empty repositions. Nevertheless, this method revealed some flaws, mainly related to the solving time for larger instances.

Therefore, a new approach was drawn, based on the exact formulation of this problem with a network-flow model. As the exact solution of real-sized instances was deemed to be impractical or even impossible, a *matheuristic* was developed to solve the model. This is a relax-and-fix-based heuristic procedure that divides the variables in sub-sets and progressively rearranges them in different sub-problems throughout the planning horizon. The relaxation and fixation of different sets is also coordinated with a mechanism based on the local branching paradigm that enables and controls modifications through the iterations. In order to test this approach, real-sized random instances were generated based on data retrieved from the company's database. The *matheuristic* was able to improve the company's profit in 33% and reduce the time spent in empty repositions.

The structure of this dissertation is organized as follows. Chapter 2 describes the problem tackled in detail, as well as the preparation work that supported the development of both methodologies, as far as data handling and study of processes are concerned. In order to properly frame the problem and proposed solution methods, Chapter 3 presents a literature review on empty flow management within the car rental and similar industries. Chapters 4 and 5 describe, respectively, the developed *metaheuristic* and *matheuristic*. Their design, implementation, and results are herein presented. Finally, in Chapter 6 the main conclusions are drawn and future enhancements to the work developed are discussed.

Chapter 2

The vehicle-reservation assignment problem on a car rental company

The vehicle-reservation assignment problem is a relevant yet not often studied sub-problem within the framework of the car rental logistics. In this chapter, a description of the problem is presented as well as the procedures for the data gathering, specification and handling. As for the description, a summarized overview of the car rental industry is displayed, as motivation for the study of this problem. Then, a more detailed statement of the problem is presented, as well as a preview on the current procedures used by the company to face it. As for the data, the methods used for gathering and handling it are presented, as well as the main procedures that enable its transformation into valuable and useful information.

2.1 Problem description

Car rental companies currently rely on operational efficiency in order to be able to compete in an increasingly demanding market. Being able to maximize the profit by an enhanced scheduling of reservations to specific vehicles is especially urgent for special types of vehicles whose number of units is small, in order to reduce the need to reposition them between rental stations in order to meet reservations requirements concerning available time and location.

2.1.1 Statement of the vehicle-reservation assignment problem

The car rental business model is based on making the vehicle desired by the customer available when and where the customer requests it. Therefore, it involves managing a numerous and heterogeneous fleet, in the attempt of acceding every request with minimum costs. The car rental company studied in this work has over 40 stations spread over Portugal (including the archipelagos of Azores and Madeira) where customers may pick up and deliver the rented vehicles.

The company's fleet may be essentially divided in two major categories: the *free sale* vehicles and the *special* vehicles. The *free sale* vehicles are widely and permanently available in every rental station, enabling the immediate confirmation of reservations; moreover, their assignment to a vehicle can be made on the fly by the station staff without significant profit deterioration. The *special* vehicles, however, either belong to expensive lines or have distinct characteristics (for example, minivans and off-road vehicles), thus the company has a smaller number of vehicles belonging to each group within this category. These vehicles are often not located in the required stations leading to the need to perform empty repositions.

Each reservation has the following characteristics: the date and station in which the customer wants to pick up the vehicle, the date and station in which the customer wants to deliver it, and the profit of the reservation (the revenue attained deprived of the regular operational costs). The vehicles are characterized by their current occupation; as they are currently fulfilling a certain reservation, each vehicle will be available when and where that reservation in progress ends. The costs of the empty transfers from one station to another as well as the time this transferences require are also parameters of this problem. The main objective is to maximize total profit: the profit of the assigned reservations deprived of the costs of the empty transfers. The main constraints of the problem are related to the availability of the vehicles on the moment considered.

Since the customers have specific requirements, the company should provide them with exactly what was requested. Yet when that is not possible, it is a common practice in this sector to offer the customer a vehicle from a better group for the same value (*upgrade*). When that option is not available, the company offers the customer the possibility to rent a vehicle from a worse group with a price discount (*downgrade*). The possibility to upgrade or downgrade the reservations is beneficial for the company as far as service level is concerned, although *downgrades* lead to the fulfilment of reservations for a minor profit.

The inclusion of this issue in the solving method increases the dimension of the analysis (as more vehicles are available for the same number of reservations), introducing additional constraints as for which groups can be upgraded or downgraded to which groups. Hence, reservations need to be characterized by the group (type of vehicle) required by the customer and vehicles should be associated with a specific group. As for the cost of these decisions, it is considered on this framework that the company does not incur on any additional cost by allocating a better vehicle than requested. Since the vehicles are available, it is better to seize the possible profit of the reservation than not fulfilling it and thus dissatisfying a customer. Nevertheless, a revenue decrease is usual when allocating a worse vehicle than requested. As to summarize, it is commonly a company's policy to avoid upgrading a reservation when not necessary and to execute downgrades only if no other option is possible. Both adjustments require the specific authorization of the customer; nevertheless, the upgrade is virtually always accepted.

Another lateral issue addressed is related with the time when a certain vehicle ceases to be available to the company (due, for example, to the sale of the vehicle) – the block date. This date is a characteristic of each vehicle and it conveys another constraint to the problem: the vehicles may not have any reservations allocated to them from that date onwards. Moreover, the vehicle

may be unavailable only throughout a specific time period, due to scheduled maintenance, for example; these periods are called *impros*.

It is also important to know the priority status of the reservation, as far as confirmation to the customer is concerned. As to control customer satisfaction, the company prefers the allocation of confirmed reservations over non-confirmed ones.

2.1.2 Existing procedures on the studied company

Currently, the company receives all reservations through a single call-center. As far as special vehicles are concerned, the allocation process is handled twice a day by two qualified and specialized employees. They use software that visually displays the vehicles (the ending time and location of the reservation currently being fulfilled) and the reservations (ending and starting times and locations) while enabling them to try different allocation configurations and chose the definite one. These employees are well trained and experienced in this task and follow an informal set of rules. Nevertheless, considering the amount of data tackled, the opportunities to improve this manual solution are perceptible and extensive. Moreover, the employees do not have access to information regarding the profitability of each reservation. One of the main advantages of the proposed methods could be the assignment of these two qualified employees to other value-adding responsibilities, besides the improvement of the allocation plan which implies a cost reduction and possibly a increase on the number of attended reservations. In the following section, a more detailed explanation of the decision process followed by the employees is presented.

2.2 Data handling

The data considered in this dissertation was retrieved from the database of the studied company. Data gathering, specification and handling preceded the application of the solution methods. This step was fundamental since it involves the linkage of the methods to the database system used by the company, which is a commercial fleet management software commonly employed in the sector. This software manages all information as far as fleet and reservations are concerned and is the source of the data used by the algorithms proposed. Nevertheless, some treatment of the data was required, since it was often not found in the necessary format when retrieved from the company's database and it also often lacked integrity. This section describes the data requirements, as well as the processes used to gather the necessary information and develop it in order to become valuable data.

2.2.1 Data specification and gathering

Reservations

As far as reservations are concerned, it is necessary to establish for each one: the starting date and time, the starting station, the returning date and time, the returning station, the profit of

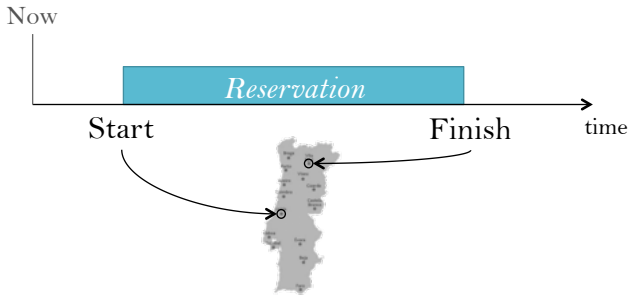


Figure 2.1: Representation of the start and end of a reservation, in terms of time and location.

the reservation, the group required, and the confirmation status. Figure 2.1 exemplifies the main characteristics of a reservation.

A simple date handling procedure (described later in this chapter) was applied to the system starting and returning dates. The status of a non-assigned reservation can take the following values: reservation not confirmed to the customer, reservation confirmed to the customer but not allocated to a vehicle yet, and confirmation pending on the customer. Due to the similar treatment and hierarchy level, confirmations pending on the customer are treated as non-confirmed ones. Therefore, the status of a confirmation takes the format of a binary parameter (confirmed or not confirmed).

Vehicles

As far as vehicles are concerned, the following data is needed: the station where it will return to in the end of the current reservation, the date and time of said return, its group, and its block date, if existent.

Some data requirements related to the vehicles, namely the station where it will return to in the end of the current reservation and the contracted date and time of return, can also be characteristics of a reservation – one that has already started and is currently taking place. In order to calculate the date and return station of the vehicles, one needs to access other information in the database, such as the vehicles status. There are three possible statuses: *free* – without an associated reservation –, *rented* – with an associated reservation –, and *impro* – not ready to be rented (for maintenance reasons, for example). *Impros* can be regarded as a special type of reservation that has to be allocated to a specific vehicle. The difference between an *impro* and a blocked vehicle is that the *impro* status has a finishing date and station whilst the blocking date inhibits the rental permanently for the future.

Furthermore, information about which groups may be upgraded or downgraded to which groups has likewise been provided by the company.

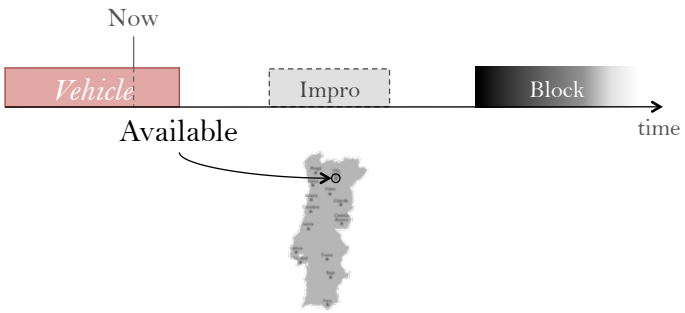


Figure 2.2: Representation of vehicle availability, defined by its current reservation and future stoppages.

Stations

Considering the rental stations, the data requirements are related to the empty transfers: the time and the cost of these transfers between each pair of stations. This data is previously determined by the company, based on their previous experience. Even though the actual carrying cost and time depend on several factors, mainly on the decision whether to transport the vehicle alone (with one driver) or on a car transporter truck, these values were considered to be good estimates of the average operations. Figure 2.3 exemplifies the need for the empty reposition of one vehicle, displaying the transfer time as well as its origin and destination.

2.2.2 Groundwork for the solution methods

This section describes how certain types of data were transformed to a non-ambiguous and more functional format. Moreover, three recognized major integrity deficiencies of the data retrieved from the system are presented, as well as the procedure followed to rebuild said data, enabling its use.

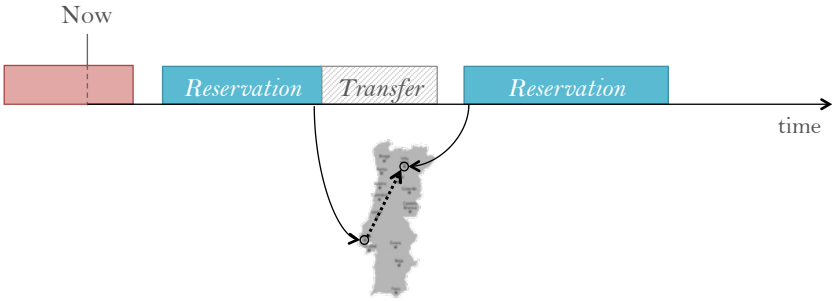


Figure 2.3: Representation of an empty reposition, which consists on the transfer of a vehicle between two stations, comporting costs and increasing travel time and its impacts.

Date and time handling

As above-mentioned, the main information source used was the company's database. The fields stored as dates were retrieved in a format that stored year, month, day, hour and minute as a fractional number. In order to transform this information into a format independent of the version of the data handling platform used, a reference date is established (specifically the current date when the application is initiated). The calculations used to achieve a proper input to the algorithm are presented on Equation 2.1, using the example of the starting date of a certain reservation i. It was also necessary to specify a time unit in hours (one hour, for example). In the system, every date will thus be represented by the number of time units that took place between its actual system date and the reference date. In order to obtain an integer value, so as to facilitate the data transference from the data handling platform to the solver, the latter information is also rounded to the specified time unit.

$$starting\ date_i = (rounded\ system\ date - rounded\ reference\ date)\frac{24}{time\ units\ [hours]} \tag{2.1}$$

The rounding of the system date may lead to a problem on the specific situation of reservations whose starting and ending time are rounded to the same time. This may happen for short span reservations (comparing to the time unit chosen) and lead to a null reservation duration, which has obvious implications on the algorithm and its relation to reality. As to prevent this, it was decided to set the value of the dates in this situation to one time unit.

Lack of integrity of the data retrieved

There were three major problems concerning the data retrieved from the company's database. As for the profit of reservations, the value was sometimes declared to be null. This was considered to be a mistake since it seemed reasonable to assume that a profit-oriented company avoids non-profitable activities directly linked to its business purpose. Even though the profit or fee for each reservation may depend on a manifold of factors other than the reservation duration, such as insurance level or extra services like ski racks or baby seats (Carroll and Grimes, 1995), in order to obtain an estimation of the missing values, it was assumed that the major cost drivers were the group of the vehicle and the number of days of the reservation. Therefore, for each group, the profit per day of each reservation is plotted and a suitable trend is chosen. Based on the observation of substantial historical data, the logarithmic trend was chosen as the best fit to this correlation and its parameters were computed for each group.

Nevertheless, in most instances it is possible to observe a few outliers in the trend, which are often a result of typing errors. Therefore, the procedure developed to correct the data offers the user the possibility to define the tolerance range and visualize the outliers. The user may then easily evaluate the possible typing mistakes and require their correction. The outliers are thus removed in order to compute a better fitted trend and then replaced by the baseline values whereas

the missing values are calculated according to the re-calculated trend. This was considered to be a simple yet realistic approach, able to restore corrupt data with reasonable reliability.

As for the initial conditions of the vehicle, in a few situations it was impossible to calculate the returning date and station of a vehicle, since the contract or *impro* number registered failed to correspond to the ones registered in the system. Since this was only observed for some instances, it was decided to substitute the system date by the reference date (mentioned above as part of the date handling procedure) as it seems to be a reasonable approximation to the reality (it is equivalent to say that the vehicle is free now).

The third problem occurred since some stations stored in the system in fields such as return station of a reservation or a contract failed to have a match in the transfer cost and time matrices provided by the company. So as to enable the use of this data, a new fictional station was added to the matrices, whose parameters were set to the average of the values found for other stations. The non-matching stations were then replaced by this average station.

Mimicry heuristic

In order to start solving this problem, it was necessary to recognize and model the two functioning modes of the allocation decision process. On the one hand, the company needs to be able to rapidly establish whether a certain reservation may or not be allocated, considering the current vehicle schedule, in order to promptly answer the client on the phone – the *online mode*. On the other hand, a more robust method is needed to improve the vehicle schedule built during the day – the *batch mode*. The latter must consider that every confirmed reservation should be necessarily allocated to some vehicle.

Considering the functioning modes, it is possible to classify the existing procedure in the company as an *online-only* mode. In fact, the employees attempt to allocate each reservation individually, not being able to enhance the global vehicle schedule as a batch. Thus, it is possible to determine that the main inputs for this decision are the single reservation to be allocated and the global vehicle schedule.

The decision process currently followed also depends on the perception that, when there are numerous possible vehicles in which to allocate a certain reservation, the employee may not be able to visually apprehend the occupation of all of them, being limited by the size of the computer screen. Therefore, another important input is the number of vehicles the employee may visualize at the same time – the size of the window screen, which was perceived as being a total of forty vehicles.

This decision process is structured and presented in Figure 2.4. During the actual process, the vehicles are presented in alphabetic order of their license plate. The employee visualizes each vehicle occupation, knowing the starting and returning stations and dates of each reservation allocated to it. The first attempt is to allocate the reservation as the last assignment of one of the visible vehicles, in the best possible position. As there is no information regarding the reservation profit on the screen, there is an attempt to reduce the costs by avoiding empty transfers and using as first criterion the coincidence between the returning station of the last reservation and the starting

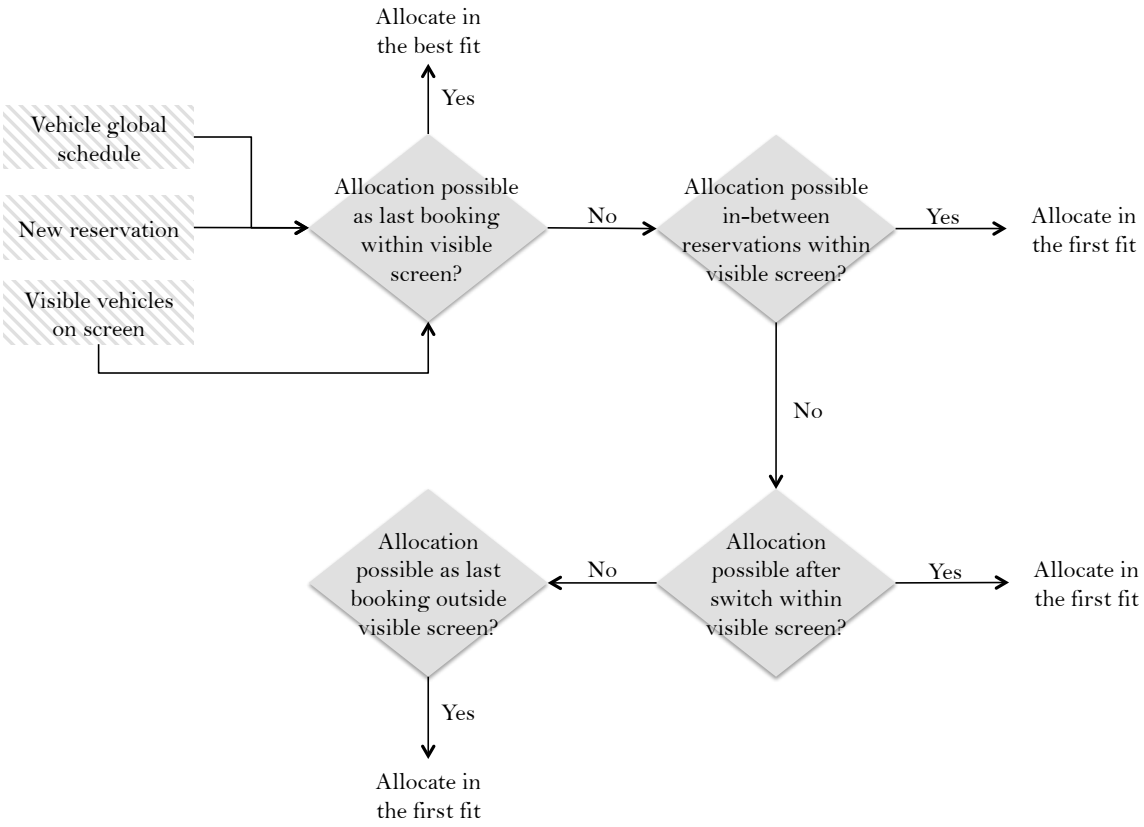


Figure 2.4: Decision process of the current vehicle allocation procedure.

station of the reservation to assign. The second criterion used is the idle time of the vehicle caused by the hypothetical allocation, as the company prefers not having vehicles stopped for more time than necessary. If this method fails to provide a possible solution, there is an attempt to allocate the reservation in-between the already allocated reservations, in the first possible fit.

When this procedure is not able to find a feasible solution, a simple switch move within the allocated reservations is tried. For each vehicle, it is verified whether the reservation to be allocated overlaps only one other reservation. If this is the case, it may be possible to allocate the overlapped reservation to another vehicle and insert the unassigned reservation in its previous place. If the feasibility conditions are fulfilled, the simple switch is completed in the first possible fit.

All the previously discussed procedures relate to the visible vehicles on screen. The last alternative procedure is the allocation as the last assignment of a vehicle outside the visible screen; as employees scroll down, if a possible fit is found the reservation is allocated.

Chapter 3

Literature review

The car rental logistics problem tackled in this work has not been often studied by the academic community. Nevertheless, some important contributions in this area ought to be mentioned. In this chapter, a general review on empty flow management is presented in order to understand the significance of this problem and how it is being studied and tackled. As to properly frame the problem approached, the freight transportation sector (and some general-transportation works) are studied in order to review the main problems and methodologies used to solve it, emphasizing the ones based on, or including, mathematical modelling. Although the airline industry does not tackle the empty reposition of vehicles, it is studied as to understand the methodologies behind aircraft assignment to flight legs, similar to the *vehicle-job assignment* attempted in this work.

As the car rental is not a traditionally studied industry, the experiences in other related transportation sectors become extremely relevant. As for fleet assignment, a very interesting parallel can be drawn with the airline industry, where useful contributions can be found. At the end of each section or subsection in this chapter, some general comments are posed, discussing insights and conclusions drawn.

3.1 Empty flow management

As previously mentioned, the car rental logistics short-term problem has not been frequently addressed on the transportation logistics literature. Although some important contributions can be found within this business sector, it is inevitable to search insights within other more traditionally studied vehicle allocation problems, which in some aspects are analogue to the one here in study, such as empty railcars and maritime containers repositioning. The truckload freight sector, despite analogue in many ways to the car rental industry and often studied within the short-term logistics framework, is herein not specifically reviewed since the approaches used are generally parallel to the ones described for railcar freight and due to the fact that most of the recent works in this field focus on the stochastic-driven intricacies of the problem studied.

3.1.1 On the car rental industry

Yang, Jin and Hao (2008) propose a review on the study of the logistic problems of the car rental industry, mainly as far as the short-term tactical fleet deployment between stations is concerned. The lack of studies within this field is recognized and a parallel with other transportation areas is drawn; the main contributions within this specific sector are analysed as well. The issues recognized as key are the fleet assignment problem and demand forecasting. Finally, amongst other conclusions, the authors pinpoint the lack of contributions related to the specific assignment or deployment of vehicles to customers.

Amongst the most relevant work in this field, one must highlight the contributions of Pachon et al. (2003), who structure the fleet planning process in three sequential phases: pool segmentation, strategic fleet planning, and tactical fleet planning. The first phase consists on clustering the rental locations of the car rental company in geographically and demand-correlated pools; the different rental stations within a pool share the same fleet, whose number of vehicles is determined in the second phase. The third phase consists on determining the number of vehicles that should be available at each station, in each period of time; the empty repositioning problem is herein considered. Pachon, Iakovou and Ip (2006) model these three phases and propose solution methodologies considering the hierarchical structure of the decision-making process. In Pachon et al. (2003), the researchers focus on the third phase, modelling the daily decisions related to the needed car transfers between stations of a certain pool in order to maximize its total expected profit. This approach assumes that all transfers can be done overnight and models the daily demand for each station with independent continuous random variables. The possibility to upgrade vehicles is not addressed. In order to solve this problem, rendered difficult by its stochastic elements, the authors propose a hierarchical approach and divide the tactical fleet planning phase in two sub-problems. Firstly, the optimal inventories at each station at the beginning of each day are found (fleet deployment sub-problem); secondly, the best car transfer policy between stations to achieve said inventories is determined (transportation sub-problem). Finally, a heuristic that approximates the values obtained with the decomposition of the problem to the optimal solution is presented, as well as the results of its application on a realistic case (considering a pool with six rental stations). It is important to note that the pragmatic output of this approach is the number of vehicles that need to be empty transferred between each pair of stations, as well as the cost it involves. Some useful insights can also be obtained when considering the extensions to the model proposed by the authors, referent to the enrichment of the objective function by the inclusion of unmet demand and unutilised fleet costs, and minimum service level constraints; the influence of price elasticity on the demand, not related to the deterministic approach proposed in this dissertation, is also modelled. It is possible to conclude that the specific allocation of reservations to the available vehicles, the main object of the work here in question, which is not addressed by the authors, could be considered as the third sub-problem of the tactical/operational fleet planning – the next step in the optimization of the company's procedures.

It is also of the utmost importance to refer Fink and Reiners (2006), as they propose a comprehensive and thorough approach to the car rental short-term logistics problem within the tactical fleet planning phase. It includes the fleeting process (enclosure of new cars bought from a car manufacturer to the fleet) and defleeting process (sale of old cars of the fleet to a reseller), as well as the car transfers between stations within a pool. The authors present a DSS whose main output is the optimization of the inventory levels at each station at the beginning of the time period considered. This system has four main steps: the input data is retrieved, namely as far as vehicle availability (current location, current use, and future unavailability due to maintenance) and information about already posed reservations are concerned; then, the demand, which is considered as a stochastic element, is forecasted (as *walk-in* customers that do not have a previous reservation are concerned); the supply and demand of cars between stations is balanced using a minimum cost time-space network model; and, finally, the obtained results are validated by the means of simulation. With a time-space network approach, the combination of specific periods of time (i.e. periods of one or half day) and space (i.e. rental stations and fleeting/defleeting depots) are represented as nodes. At the same time, upgrading possibilities are considered, both single (group A can be upgraded to group B and group B can be upgraded to group C) and double (group A can be double upgraded to group C). Therefore, in this formulation, the nodes represent a combination of specific periods of time, stations/depots, and vehicle groups. The arcs in this network represent the flow variables stating the number of cars allocated to the different transfer alternatives. The modelling of the problem is based on a rolling planning horizon of one week and the results are re-optimized every night. The authors argue that the specific assignment of cars to customers should be handled by the station staff as flexibility is required to tackle the uncertainties and delays of the process. The output of this optimization model – the representation of how cars should be deployed – is then evaluated by means of simulation. The developed DSS was successfully applied to a real case of a specific pool of a car rental company.

Li and Tao (2010) approached the car rental logistics problem with a two-stage dynamic programming model, in which the optimal fleet size and the optimal vehicle transfer policy is defined, in the case of a local car rental company that serves two cities. The authors distinguish single-trip customers (whose starting city is different from the ending one) from round-trip customers (that pick up and deliver the car to the same city) as to justify the unbalance of vehicles that requires the empty vehicle transfers, optimized on the second phase of the model. At the same time, the company studied is pondering the extension of the fleet and the authors model this decision in the first phase. The authors consider the impact of lost sales in the objective function. A heuristic to solve the first part of the model (fleet sizing) is also presented.

Comments

All researchers are unanimous in stating that a good fleet utilization and customer service are crucial to car rental companies. There are not yet many researchers dedicated to this specific area, although some sub-problems are analogue to other areas that are traditionally studied. Most studies dedicated to the tactical planning decisions (where the empty repositioning concerns are included)

focus on the vehicle deployment between stations (this problem will be henceforth referred to as *fleet deployment*) and consider two related sub-problems: the decision on the inventory levels of the fleet in each station at each time period, and the decision on the transfer policy of the cars between stations so as to achieve these levels. There are also a few attempts to link these decisions with the upper-level pool *fleet sizing* decisions. Nonetheless, to the best of our knowledge, within the car rental logistics optimization framework, the lower-level *vehicle-job assignment* tactical or operational problem has not yet been approached.

As for the stochastic nature of these problems defended on some of the works analysed, one must consider that the approach herein developed was built bearing in mind the case of *special vehicles*. The company believes, based on past experience, that these are heavily dependent on anticipated reservations. As far as uncertainties as delays are concerned, the attempt to realistically tackle them was implemented in the data pre-processing phase, as dates and times are altered by a delay parameter dependent on the type of client and calculated based on the company's experience.

As to summarize and clarify, the logistic problems further addressed in this document will be classified as: *fleet sizing* – deciding how many vehicles should be available between a group of interconnected locations; *fleet deployment* – related to the shorter-term decision of how many vehicles should be empty repositioned between stations at specific points in time as to meet the future demand requirements; and *vehicle-job assignment* – deciding which vehicle should fulfil each request, reservation, customer order, or job, considering (or not) the costs of the empty transfers.

3.1.2 On other transportation sectors

Dejax and Crainic (1987) highlighted the significance of the empty flows for any mode of freight transportation, and proposed a taxonomy for the related problems and models existent in the literature. The specific assignment of orders or customers to vehicles within the empty flow framework is not addressed by the authors since the focus of this work is on the allocation to stations or depots of vehicles or groups of vehicles with similar characteristics and availability conditions (*fleet deployment*). The authors recognize the impact empty flows have on logistic systems, namely on their operational and economic performance, as they generate costs and no revenue, and alert to the significance they assume on the freight transportation sector. The taxonomy proposed states two levels of classification. Firstly, the nature of the problem is analysed. The authors distinguish between *policy* models that cover strategic, medium/long-term problems (including fleet sizing and depot location design), and *operational* models that tackle short-term problems (including empty vehicle inventory control and dispatching of loaded and empty vehicles). Then, two types of criteria are proposed, based on the characteristics of the problem and the resolution methodology. Finally, the authors suggest that the empty flow problem should be tackled within an integrated approach with the loaded flow decisions.

In spite of the possible need for specific requirements related to the transportation sector recognized by Dejax and Crainic (1987), some authors focused on providing a global approach, extendible to several of those sectors. Beaujon and Turnquist (1991) focus on the link between *fleet sizing* and *fleet deployment* decisions, and their combined impact on the performance of

transportation systems. Although providing a model intended for the generality of transportation systems, the authors refer the historical main role of railcar freight research as far as these issues are concerned, as well as the developments related to container freight transportation or car rental. As aforementioned, the main focus of this work was to have a direct impact on the investment decisions, by means of the optimization of the *fleet deployment* process. The approach used was classified as dynamic and stochastic, since uncertainty was acknowledged both in the system performance and the forecasting of the demands. The mathematical model proposed aimed to define not only the allocation of vehicles, at each time and location, to loaded and empty movements and vehicle inventories, but also to calculate the optimal number of vehicles within the pool. As it is assumed that all the demand must be met, backordering was deemed acceptable for unmet requests. The model is formulated as to maximize the total profit, considering the stochastic nature of travel time and demand, as well as different costs for moving an empty or a loaded vehicle, costs of ownership per vehicle, holding costs per period and location, and penalty costs for unmet demand per period. In order to tackle this otherwise unsolvable formulation, the problem was represented with an expected value formulation and an approximation was developed based on a minimum cost time-space network with a non-linear objective function. Experiments were conducted to test the procedure chosen to solve the approximation, whose results were encouraging; no real instances were used.

As it is possible to conclude from the research reviewed so far, the network formulation of the tactical problems is a tool increasingly used within the freight transportation sector. Using the designation of Service Network Design, Crainic (2000) reviews the use of this type of formulation in this area and the methodologies used to solve it. Moreover, a taxonomy to classify service network design problems and formulations is proposed. The author indeed supports that tactical planning processes, namely the ones referent to the optimal allocation and utilization of resources, are vital for freight transportation companies and can be particularly difficult to solve. That is seen as the reason for the considerable use of this approach. As for the deterministic dynamic service network design that may be of interest for this dissertation, the author denotes the use of the aforementioned time-space networks.

Song and Earl (2008) propose an integrated model which encompasses *fleet sizing* and empty vehicle repositioning policy decisions for two-depot systems. The authors consider the uncertainty in the length of the arrival and transfer processes. The approach presented considers two stages, based on the different natures of the decisions tackled: strategic (*fleet sizing*) and operational (empty vehicle repositioning). Analytical expressions for the optimal cost function are also established. In this case, it is assumed vehicle leasing allows all demand to be met. Although this assumption is reasonable for most of the transportation problems that can be adapted to the system described (for example, truck movements) and even to the rental of regular cars, it is felt that this assumption collides with the main characteristic of special rental vehicles: uniqueness. Nevertheless, the authors state that the model proposed can be generalised for networks of depots and that it is effective for hub-and-spoke systems.

Railcar freight

Dejax and Crainic (1987) highlight the fact that railcar freight transportation has provided useful insights for several industries, as far as empty flows are concerned, although the existence of mode-specific requirements that are reflected on the models is recognized. In this section, some important works within this area that have the potential to inspire research within the car rental industry are presented. The major differences and similarities between the two sectors are then discussed, as well as the distinct yet complementary outline between the approach here proposed and the major trends as far as empty flows are concerned.

Yaghini and Khandaghabadi (2013) tackle the problem of the empty repositioning of the railcar freight industry within the framework of their main focus, the *fleet sizing* problem, integrating with it the *fleet deployment* decisions. This approach is multi-periodic and deterministic as well as dynamic. The authors use time periods of one day; all demand must be met during the planning horizon yet unmet demand is allowed to be served in the next time periods. Moreover, a constant-sized homogeneous fleet is considered. As for the mathematical modelling, the decision variables used are the number of railcars transferred, empty or loaded, between stations, as well as the vehicle inventory levels in each station at the end of a day. The objective function aims to maximize total profit; the costs considered include: railcars ownership cost, loaded and empty cars transfer cost, costs for holding empty railcars at stations; and penalty costs for delays. The solution approach proposed by the authors consisted of a hybrid metaheuristic that combines genetic and simulated annealing algorithms. Through computational tests, the quality of the metaheuristic was considered acceptable. The algorithm developed was also used for solving the real case of the Iran Railways network, considering fifty major stations.

A very interesting viewpoint on empty flows in the railcar freight sector can also be found in the case study presented by Spieckermann and Voß (1995). The authors tackle this problem through the *vehicle-job assignment* framework, proposing a dispatching heuristic based on production planning procedures. In this case study, based on a German company that manages a fleet of 3000 railcars, the assignment of jobs to machines in an industrial environment is compared with the assignment of customer orders to railcars, comparing the empty transfers to set-ups, which cause costs and no revenue. The case considered a heterogeneous fleet yet no substitution between different types; therefore, the problem may be divided in independent homogeneous fleet problems. The approach taken, similar to others in this sector, enforced all demand to be dispatched yet allowed delays in said fulfilment. Moreover, a multi-objective formulation was used, attempting to minimize total tardiness as well as the costs of the empty movements. This real application was considered important for the company, mainly as a good launch in the use of computerized dispatching algorithms that take into account the impact of empty movements.

Bojović (2002) approaches the empty transfer problem with a framework similar to the one used by Beaujon and Turnquist (1991): the *fleet sizing* and *fleet deployment* problems are tackled from a dynamic and stochastic viewpoint, considering a homogeneous fleet. The author considers holding costs for empty cars stopped in a station, car shortage costs, loaded and empty movement

costs, and ownership costs. The problem was modelled as a time-space network; however, the solution approach developed was based on optimal control theory and was shown to be very good as far as computational efficiency is concerned.

A specific problem within rail transportation is studied by Sherali and Tuncbilek (1997), related to the shipping of automobiles via railroad. The main focus is on the *fleet sizing* problem, yet as this work reflects the concerns of the company that manages the empty repositioning of the multilevel railcars, the *fleet deployment* problem is considered in an integrated manner, as seen in aforementioned works. This problem differs in the sense that the pool of railcars used is the result of the contributions of several automobile manufacturers and the decision of how the railcar acquisition should be distributed amongst them is also on focus. Moreover, in this problem, the origin locations are always distinct from the destination ones. The approach taken is deterministic and dynamic, and the problem is formulated as a time-space network flow model, whose main objective is to minimize the fleet size, satisfying the whole demand. The authors present an interesting heuristic to solve this model: an overall problem solution is iteratively constructed as sub-problems defined for overlapping time segments are consecutively solved. This heuristic was able to achieve the optimum for every test run. The authors also proposed a simpler model that calculates the fleet size by using information of a specific period on the year. This model, calibrated by the more complex one, was able to achieve near-optimum results with less effort; the calibrated simpler model was also shown to be helpful on conducting analyses of different "what-if" scenarios.

This same problem is again addressed by Sherali and Suharko (1998) who developed a DSS that accommodates several extensions to the problem. Empty repositions are performed once a day. In order to incorporate the trade-off between large fleet costs and the high levels of unmet demand, up to three days of tardiness are allowed. Moreover, prioritization of different demand locations is taken into consideration. In fact, each automobile manufacturer grades its locations in terms of priority. The prioritization of all the demand locations is obtained by assigning different costs on the objective function to their respective flows, with increasingly higher and wider penalty intervals for different levels. These costs are furthermore adjusted to make the priorities between manufacturers fairer (directly proportional to the railcars they have provided to the pool), and balance the priority list of each manufacturer. The uncertainty of travel times and demand levels is also considered by the inclusion of stochastic elements in the model. This was formulated as a time-space network model and proved functional for the computational tests run.

Comments

The majority of the studied papers that tackled the empty flow problem within the railcar freight sector focus on the *fleet sizing* and *fleet deployment* problems without considering the *vehicle-job assignment* operational decisions. Therefore, the discretization of time on considerably large time periods (such as days or half-days) is reasonable and desirable. In fact, if the desired output is the number of vehicles to reposition between stations and the actual physical repositioning is designed to be made in specific periods of time, this is a reasonable approach. When looking at the empty

transfer problem specifically from the specific assignment viewpoint, one sees the need to face time as a continuous variable or one discretized in smaller periods (such as hours or half-hours) as the empty movements occur in-between loaded movements and not in bundle, at a specific point in time. Moreover, as *fleet sizing* is considered, the ownership or holding costs become relevant for the problem formulation.

The railcar freight sector is here in study due to its strong presence on the empty flow management literature and pressing similarities with the car rental business: in both cases one must manage a fleet of vehicles that should be assigned to specific orders; these orders are mainly characterized by a due date (and starting date), their origin and destination; due to unbalances of supply and demand in each location and time period, the need to empty reposition the vehicles must be faced. Nevertheless, several differences can be pinpointed between the two sectors that influence the way problems are formulated or approached. Firstly, in railcar freight, different policies are taken as far as backordering is concerned; nonetheless, generally the models state that all demand must be met and therefore allow the delay on delivering. In car rental, this is not plausible as a customer is usually strict with the dates the car is needed (it is not reasonable for the company to ask the customer to wait one day for the car – he will go to another company). One other somewhat minor difference is that the revenue in railcar freight tends to be more undifferentiated, as it depends mainly on origin and destination; in car rental, several other factors may influence the fee. At the same time, the existence of a heterogeneous fleet with partial or total substitution between different types is more common in car rental than in railcar freight transportation; therefore, the homogeneous fleet assumption is usually unrealistic in one sector while it can be reasonable within the other.

Container freight

The empty flows are a concern to the container freight sector, mainly as far as the inland operations of maritime shipping companies are concerned. Crainic, Gendreau and Dejax (1993) structure the empty container allocation in this situation and propose models to deterministic and stochastic situations, as well as single and multi-commodity procedures (with substitution between container types). The authors stress the importance of managing the empty movements and tackle this issue considering the actual demand for empty containers as coming from exporting customers and storage managers, and anticipating future demand. The system analysed comprehends: harbour depots (as direct loading and unloading areas for ships, they provide and demand empty containers), inland warehouses, customers that request empty containers (exporting customers), customers that supply empty containers (importing customers), a pool of extra empty containers (e.g. rented or borrowed), and the possible and reasonable transportation links between these. The deterministic models presented were based on specifying the origin of the empty containers that would fulfil each request, which is mainly characterized by its destination (the respective customer, or the harbour depot) and requested time. The authors propose also a time-space network representation of the problem, both for single and multi-commodity situations.

Within the framework of land operations of maritime shipping companies, Bandeira, Becker and Borenstein (2009) justify the importance of managing empty container movements with the unbalance of international trade; in fact, some areas perform mainly export activities while others do the opposite. This work, in contrast with the previous one, focuses on the integration of the empty and full container distribution planning problems, highlighting the issue of reutilization of empty containers. The main output is the development of a DSS based on a heuristic procedure. The modelling of this problem is of the utmost interest as it has close links to the *vehicle-job assignment* here proposed for the car rental industry. The conceptual model developed, based on the work of Crainic, Gendreau and Dejax (1993), represents the movements of containers, empty or full, from their unloading on a sea port until they are available again for other customers, considering their circulation through intermediate storage points (e.g. harbour depot and inland warehouse). The authors then formulate this problem as a Multiple Depot Vehicle Scheduling Problem. In this, a set of container loads are considered (units of cargo that must be transported each by one container), as well as a set of depots with a specific number of containers stored. Each container load is characterized by its origin, destination and requesting time (due date). The transportation time between origin and destination are known in advance, as well as the empty transfer time between the destination of a load and the origin of the following one. A pair of container loads are said to be compatible if the loads can be transported by the same container at the same time, and for each compatible pair of loads there is a cost incurred if the second load is transported right after the first one. Considering that each load must is served by exactly one container, the number of containers that leave each depot should not surpass the inventory there available. The objective is to minimize the cost of assigning loads to containers, considering transportation, storage and handling costs of empty and full containers; in fact, this can be seen as a sort of *origin depot/job assignment* problem. The problem is modelled as a network where container loads and depots are the nodes and the decision variables represent the flow of the service originated from a specific depot to two consecutive container loads. This approach is extremely significant for the problem considered in this dissertation as it attempts to allocate resources to requests specifically, integrating empty and full movements.

Due to the complexity of a network generated by a realistic case, an alternative non-optimal resolution methodology was presented in which some extensions are considered. This methodology is based on the decomposition of the problem in two sub-problems and their iterative alternated resolution providing feedback to each other. The first model attempts to discover the optimal number of empty and full transfers between nodes (warehouses, harbours, supply customers and demand customers) that minimizes the cost of operations for a specific time period (*static model*), yet it does not take into account transportation times. The inputs received (from the second model) concern demand, and current empty and full container positioning. The second model (*dynamic model*) receives the outputs of the first one and updates demand and supply on future time periods (possibly with the addition of new container load requests) while deciding on the dispatching of empty containers from several nodes (considering backordering); this model aims to expand the first model solution in a time schedule and due to its complexity it is solved by heuristics. The

DSS which enclosed this approach showed through computational tests good potential to be used by logistic companies.

Comments

The papers reviewed that focused on the empty flow problem concerning container freight, namely the inland logistic operations for maritime shipping companies, can be extremely useful when one is trying to understand the car rental short-term logistic problem. The two sectors can be compared, since the request for containers can be compared with the reservation of cars as both generate the need for the empty transfer of the vehicles considered (cars and containers). Some differences exist and in many ways the container freight sector is closer to the railcar freight sector. For example, backordering is reasonable and the most customer-relevant time characteristic of the request is their due date – in car rental, the starting date is also critical. Nevertheless, in this sector, a need for specifically determine the empty or empty and full movements within a network was highlighted. Moreover, solutions adaptable to the car rental industry are presented, such as the formulation of the problem as a Multiple Depot Vehicle Scheduling Problem (MDVSP). In fact, this formulation can be adapted to describe the specific *vehicle-job assignment problem* within the car rental industry. Dell'Amico, Matteo and Toth (1993) describe the MDVSP as the "optimal definition of vehicle duties" through the assignment of a set of time- and location-bounded trips to vehicles which are distributed amongst a set of depots. A pair of trips is considered compatible if the same vehicle can serve them sequentially; each pair of compatible trips is associated with a cost, incurred if this sequence is actually fulfilled by one vehicle. There is also a cost associated to the start of the duty (set of sequential trips) of a vehicle (located within a certain depot) with a certain trip; one must also note that, understandably, a duty must start in a depot. The assignment must be found in a way that every trip is served by exactly one vehicle, within a feasible duty; moreover, each vehicle must return to the depot in the end of the duty and the number of vehicles exiting one depot cannot be superior to the number located there in the beginning. As for the objective function, the number of vehicles used or the sum of the duty costs is minimized. It is important to note that an unused vehicle does not bear any additional costs. In this formulation, all vehicles are considered to be identical. In the problem studied in this dissertation, even considering that no upgrades or downgrades are allowed, the vehicles differ due to their initial conditions.

Therefore, one can adapt this formulation to the car rental problem studied stating that the set of trips is represented by the set of reservations on hand, and each vehicle available is located at its own depot. One can thus assign different costs and availability constraints to the start of the duties, representing the different initial conditions. As to cope with the real system in study, the obligation of fulfilling every reservation must be relaxed; one can state that each trip must be served by one vehicle, or not be served at all. Consequently, the profit of each reservation served must be considered; minimizing the total duty costs, the profit value can be added (as a negative value) to the cost of fulfilling each reservation, whether it occurs in the start or during a duty. As for the requirement of the return of the vehicle to the depot, it is not consonant with the car rental framework. Therefore, one can assign a null cost to this trip and treat it as purely conceptual.

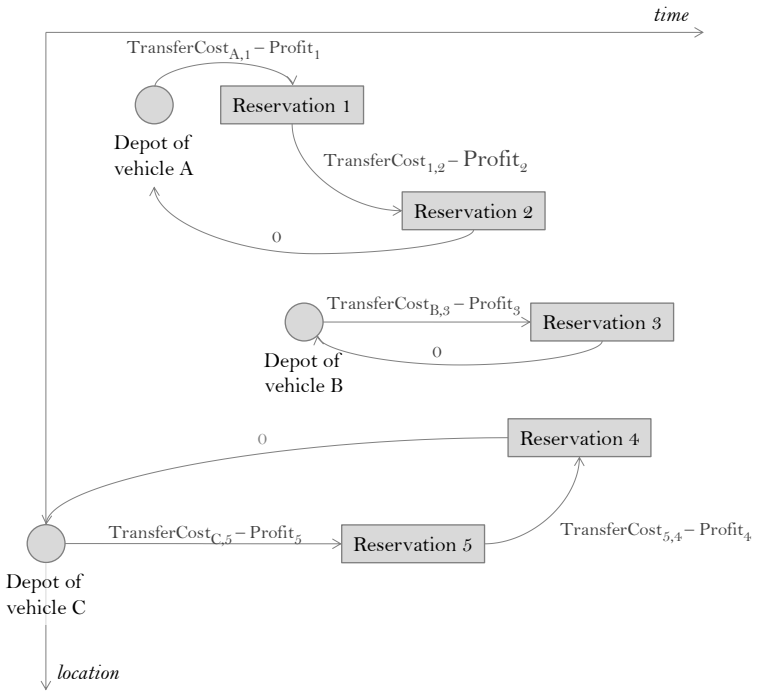


Figure 3.1: Illustration of the adaptation of the MDVSP to the vehicle-reservation assignment problem.

Figure 3.1 exemplifies the adaptation of MDVSP to the car rental problem in question. In this, there is an attempt to demonstrate how the unique characteristic of the car rental problem – the initial occupation of vehicles – is treated. The fact that their availability will happen only at a specific time is considered by not allowing arrows to have the direction of right to left (except when considering the return to the depot); thus, each vehicle can only serve trips/reservations that are located at their right, in Figure 3.1. The different locations where the vehicles will be available are considered both in the transfer times and in the transfer costs between the depots and the first trip of the duty.

3.2 Airline industry viewpoint on fleet assignment

Even though the airline industry is not particularly concerned with empty movements, as their unbearable costs force the flight design to be balanced, some important insights can be drawn from it, as it was one of the first industries to start studying the fleet assignment problem based on revenue management strategies ("finding the optimal trade-off between average price paid and capacity utilization"). (Yang, Jin and Hao, 2008)

In fact, the fleet assignment problem in the airline industry is analogue to the *vehicle-job assignment* problem tackled in this work for the car rental sector; the goal is to assign the aircraft types (based on its capacity and other characteristics) to flight legs and build a cyclic schedule.

Abara (1989) models this as a network flow problem in which feasible turns (a possible sequence of flight legs associated with an aircraft type) are represented; in this network, the flight legs are the nodes and the arcs that link them represent the aircrafts that serve them, which may not surpass the total number of aircrafts of each type. Each flight leg is characterized by departure and arrival times and airports; the compatibility of each pair of flight legs (the possibility that one is served immediately after the other by one aircraft) is dependent on the arrival time of the first and departure time of the second. The main objective is to maximize the total profit of the sequence of flight legs associated with a certain aircraft type, considering the revenue and operating costs (cost per aircraft needed and cost of operating in a certain station), which are both dependent on the conjugation of aircraft type and flight leg. Some other industry-specific constraints were modelled as well. The model presented in the article was told to be significantly affecting the decision-making processes at American Airlines.

Subramanian et al. (1994) present the case of Delta Air Lines response to this same problem (minor sector-specific differences can be recognized). The *Coldstart* model is a mixed-integer linear program whose output is the assignment of flight legs to fleet types, minimizing costs, and assuming repetitive schedules of one day. This models differs from the one proposed by Abara (1989) as far as its formulation is concerned: the decision variables in this case are not the indication if sequences of two flight legs are associated with an aircraft type but if one flight leg is associated with an aircraft type, as well as the number of aircrafts from a certain type grounded on a certain airport at a specific time period (a similar formulation to this problem is also used by Hane et al. (1995)). This different formulation of the decision variables affected the formulation of the constraints and allowed the authors to expand them to more sector-specific restrictions. The company was reported to be "pleased with both the cost savings and the revenue generation from the model"; nonetheless, the authors highlighted the difficulty in measuring actual improvement when comparing the results of the model with the ones obtained with previous procedures used by the company.

Comments

The aircraft assignment problem studied in these papers is very similar to the *vehicle-job assignment* problem tackled in the car rental industry. Although the main concern (the empty transfers) is extraneous to it, the scheduling efforts focusing on time- and location-bounded specific jobs by a fleet of vehicles are very similar. In the car rental framework, one can say that a specific car is equivalent to an aircraft type with only one vehicle available – in fact, each car has specific location and time availability characteristics, like an aircraft type – and a reservation is equivalent to a flight leg. The fact that the airline industry aims to build repetitive plans (per day or week) while the car rental sector faces a more dynamic and ever changing request list does not hinder the operational and short-term analogy presented.

Chapter 4

Metaheuristic approach

The first priority when tackling the problem faced by the car rental company was to obtain good, profitable allocation plans. Nevertheless, it was known that these plans, good for the time being, would probably cease to be so due to the dynamic characteristics of the problem. Due to the need of quickly obtaining a good solution and also due to the fact that understanding the reasoning behind the solution applied was of the utmost importance for its acceptance and use in the company's daily life, a *metaheuristic* was drawn to solve the problem.

4.1 Solution method

Blum and Roli (2003) describe *metaheuristics* as approximate algorithms that manage the integrated use of basic heuristics at a high, strategic level, so that the solution space is explored more efficiently and effectively. These strategic procedures drive the search process in order to achieve a very good solution within a reasonable time and, although they are abstract and not problem-specific, they can be enhanced by the use of problem-specific knowledge. The authors distinguish two basic heuristics used: constructive methods, which generate solutions without a starting point by the sequential addition of components until the solution is complete, and local search methods, which start with an initial solution and try to replace the current solution with a better one that belongs to a neighbourhood. A neighbourhood is a set of solutions that were built following a common rule and thus share a neighbourhood structure. Exhaustive local search procedures enable reaching the local optimum: the best solution within a neighbour solution space.

Metaheuristics can be classified as population-based or as single point search. The former term relates to *metaheuristics* that describe a trajectory within the search space by working on a single solution. The latter, on the contrary, refers to procedures that work on the evolution of a set of points in the search space. Furthermore, *metaheuristics* can be classified based on the use of the objective function. Dynamic, unlike static, objective function *metaheuristics* change this function in order to diversify and escape from local minima. Moreover, one can distinguish the

existence of one or various neighbourhood structures. In most *metaheuristics*, the neighbourhood structure used is the same, thus the differences between neighbour solutions are based on a single set of arrangement and assembly rules. The multiplicity of neighbourhood structures is another diversification tool that can be used. (Blum and Roli, 2003)

The *metaheuristic* chosen to solve this problem, as far as the *batch* mode is concerned, was GRASP (Greedy Randomized Adaptive Search Procedures), first introduced by Feo and Resende (1995). GRASP is an iterative technique that is based on two sequential phases: the construction of a solution based on a randomized greedy heuristic, and the local search, which applies small adjustments to the solution provided by the first phase in order to achieve some improvement. Each GRASP iteration comprises these two phases and originates a feasible solution; throughout the iterations, the best solution found is kept.

The randomization of the greedy heuristic is based on the ranking of the elements by a myopic (greedy) criterion; the first α elements form the restricted candidate list (RCL). This list is updated at each iteration as the criterion score may change throughout the construction; therefore, this constructive heuristic is called dynamic. The next element to be allocated to the solution "in construction" is chosen randomly from the RCL list. Therefore, the α parameter controls the greediness of the heuristic; in fact, setting its value to *one* eliminates the randomness of the construction. The value of α can remain constant throughout the GRASP iterations or it can react to the evolution of the algorithm. GRASP is a single point search (trajectory method), and, in its basic formulation, a single neighbourhood structure *metaheuristic* . It was chosen due to its intuitive structure and relatively simple implementation. This chapter describes the GRASP procedure developed and the main results of its implementation.

4.2 GRASP algorithm developed

The GRASP procedure developed is briefly presented and summarized in Algortithm 1. The solution is represented as a vector of vehicles (a schedule) in which each vehicle has specific reservations assigned to it, also forming a vector. Throughout this section, the constructive heuristic drawn and two local search approaches will be presented in detail.

The option to enable the α parameter to react to the solution quality was not used in this algorithm. Computational experiments were attempted using this approach and revealed that it was not able to improve significantly the efficiency and efficacy of the simpler constant-α approach, which was thus chosen.

4.2.1 Vehicle allocation constructive heuristic

The constructive heuristic drawn, which aims to assign reservations to vehicles, is based on the ranking of both reservations and vehicles. The reservation rank is based primarily on the status of the confirmation - confirmed reservations rank higher than non-confirmed ones. The second criterion is the proximity of the starting date of the reservation. In order to ensure feasibility and the fulfilment of as many reservations as possible, the earlier the reservation starts, the higher it

Algorithm 1 Pseudo-code for the GRASP algorithm.

```
number of iterations: nI
set of seed numbers for the randomized heuristic: S
RCL parameter: α
function GRASP(S, nI, α)
    iteration number: it ← 0
    best solution: bestSol ← ∅
    for it = 0 → nI do
        current solution: currSol ← ∅
        current objective function value: currOF ← 0
        currSol ← CONSTRUCHEURISTIC(S_it, α)
        currSol ← LOCALSEARCH(currSol)
        if currSol is better than bestSol then
            bestSol ← currSol
        end if
    end for
    return bestSol
end function
```

ranks. Thirdly, and as to contribute to the ultimate objective, the most profitable the reservation, the higher it ranks.

Subsequently, for each reservation, starting with the one ranking higher, the vehicles which could be available on the date and location required are listed, forming the RCL. These can be easily identified as the ones that fulfil equation 4.1 and belong to the same group or, if allowed, a compatible upgrade or downgrade group.

$$\begin{aligned} &\textit{returning date}_{\textit{vehicle}} + \textit{transfer duration}_{\textit{return station}_{\textit{vehicle}}\textit{starting station}_{\textit{reservation}}} \\ &\qquad < \textit{starting date}_{\textit{reservation}} \end{aligned} \tag{4.1}$$

The vehicles that fulfil equation 4.1 are then ranked from the lowest to the highest transfer cost between the returning station of their current reservation and the starting station of the reservation considered. If there is a tie, the vehicles are ranked in ascending order of idle time – the time span between being available in the required location and the start of the reservation, as described in equation 4.2. Figure 4.1 depicts these two equations.

$$\begin{aligned} \textit{idle time} = \textit{starting date}_{\textit{reservation}} - \\ (\textit{returning date}_{\textit{vehicle}} + \textit{transfer duration}_{\textit{return station}_{\textit{vehicle}}\textit{starting station}_{\textit{reservation}}}) \end{aligned} \tag{4.2}$$

The constructive heuristic presented is summarized in Algortithm 2. Once again, the solution is represented as a vector of vehicles (a schedule) in which each vehicle has specific reservations assigned to it.

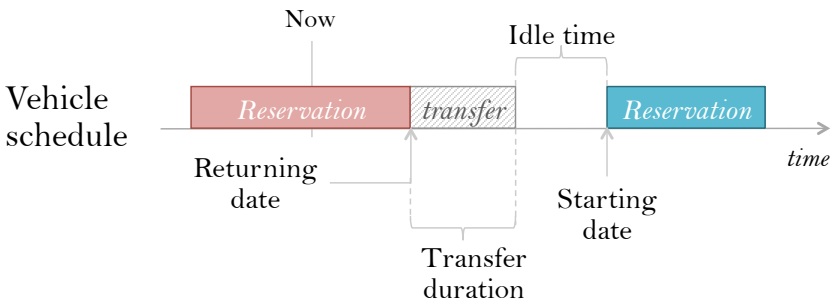
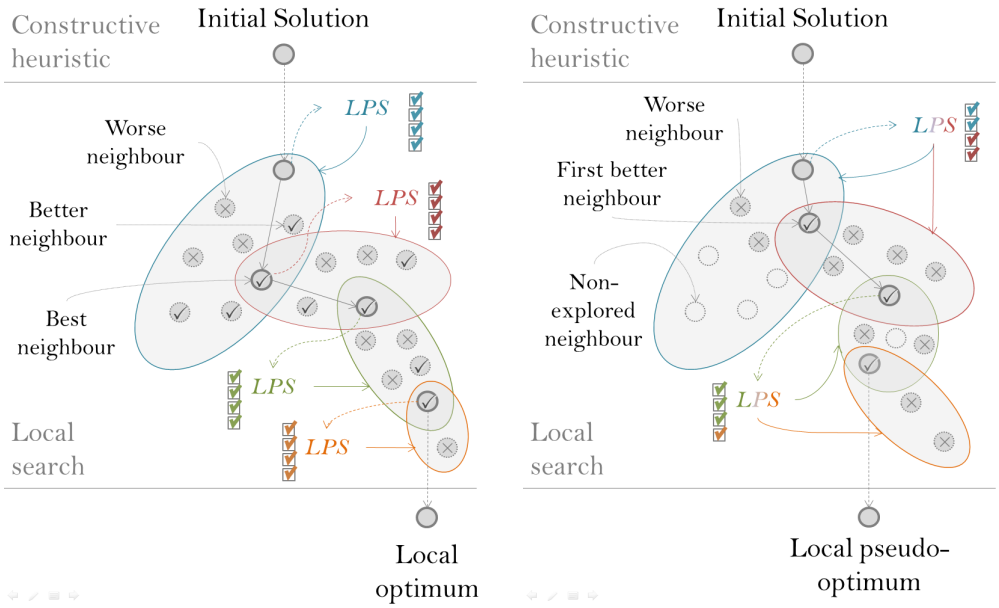


Figure 4.1: Graphical representation of the elements of the equations 4.1 and 4.2.

Algorithm 2 Pseudo-code for the constructive heuristic.

```
set of R reservations: Res
set of V vehicles with no reservations assigned: Vehic
function CONSTRUCTHEURISTIC(seed, α)
    order Res giving preference to: higher priority > earlier starting date > greater profit
    solution: VehicSched ← Vehic
    for all r ∈ R do
        set of compatible vehicles: CV_r ← ∅
        random index: index ← 0
        CV_r ← FINDCOMPATIBLEVEHICLES(Res_r, VehicSched)
        if |CV_r| ≠ 0 then
            CV_r ← ORDERCOMPATIBLEVEHICLES(CV_r)
            index ← RANDOMNUMBER(seed, α)
            ASSIGNRES(Res_r, index, VehicSched)
        end if
    end for
end function
```



(a) Best-improvement approach

(b) First-improvement approach

Figure 4.2: Representation of the two local search approaches with different neighbour selection strategies.

4.2.2 Local search approaches

Two different approaches were designed and tested as far as the local search routine is concerned. The move that defines the neighbourhood structure is in both cases based on the swap of pairs of allocated reservations, yet the approach to the selection of the new incumbent solution differs. The first explores the whole neighbourhood selecting the best improvement whilst the second may be described as a first-improvement approach. Figure 4.2 illustrates the main differences between these approaches, namely as far as the neighbourhood generation is concerned.

Best-improvement approach This approach generates an all-encompassing neighbourhood structure as each neighbour embodies the incumbent solution modified by one specific swap (previously listed on a *LPS* that stores all the swaps possible within the incumbent solution), and every feasible swap generates a neighbour. In order to choose the best possible improvement, all neighbours are evaluated. If some improvement in the objective function is possible, the best neighbour becomes the incumbent solution and a new *LPS* is constructed based on it. A new iteration is run; the algorithm stops when no neighbour is able to improve the objective function. This local search approach is summarized in Algortithm 3. Once again, the solution is represented as a vector of vehicles in which each vehicle has specific reservations assigned to it.

First-improvement approach In this approach, the local search is also initialized with the selection and listing of the possible swapping pairs within the initial solution (the *LPS*). Each listed pair originates a neighbour - the incumbent solution modified by the swap. The neighbours are

Algorithm 3 Pseudo-code for the best-improvement local search.

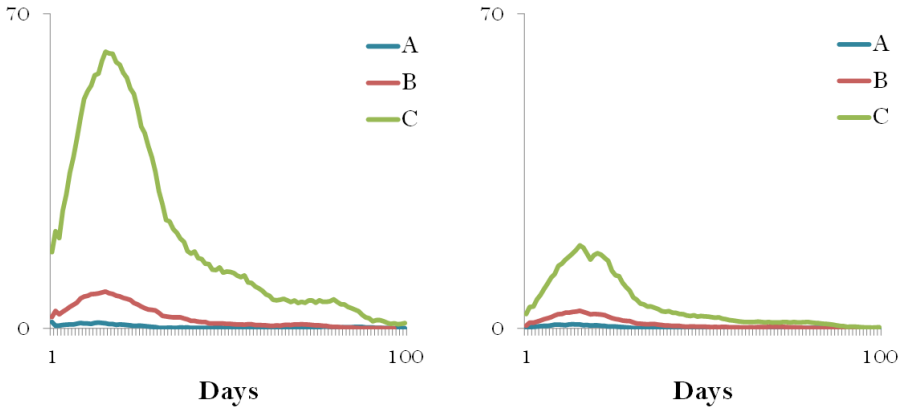
```
function LOCALSEARCH(iniSol)
    incumbent solution: incSol ← iniSol
    best neighbour: bestNeigh ← iniSol
    overall improvement: oImpr ← 0
    iteration improvement: itImpr ← 1
    local search iteration number: n ← 0
    list of possible swaps: LPS ← ∅
    while itImpr > 0 do
        LPS ← GENERATELPS(incSol)
        for all s ∈ LPS do
            current neighbour: currNeigh ← incSol
            currNeigh ←DOSWAP(LPS_s, currNeigh)
            if currNeigh is better than bestNeigh then
                bestNeigh ← currNeigh
            end if
        end for
        itImpr ← Value(bestNeigh) − Value(incSol)
        if itImpr > 0 then
            incSol ← bestNeigh
        end if
        LPS ← ∅
    end while
    oImpr ← Value(incSol) − Value(iniSol)
    return incSol
end function
```

only explored until one is found that improves the objective function. In fact, the listed pairs are swapped within each best neighbour that is found and when this happens a new neighbourhood structure is generated. Nevertheless, unlike the previous approach, a new *LPS* based on the new base solution for the neighbourhood construction is not generated. Instead, the algorithm continues to try to swap the pairs listed on the first *LPS* but now within this base solution. Note that since the listed swaps were selected within a different solution a new feasibility check must be run. Once again, the first neighbour that is able to improve the objective function is selected as the base solution for the neighbourhood generation. The procedure described is repeated until all swaps in *LPS* have been attempted. When an *LPS* has been completely explored, and while it is possible to achieve an improvement, a new *LPS* is generated from the incumbent solution and the process is repeated. It is important to understand that this approach was developed with the objective of obtaining a good, swift routine, which explored the neighbourhood structures in depth rather than in width. This local search approach is summarized in Algortithm 4.

Algorithm 4 Pseudo-code for the first-improvement local search.

```
function LOCALSEARCH(iniSol)
    incumbent solution: incSol ← iniSol
    better neighbour: bettNeigh ← iniSol
    overall improvement: oImpr ← 0
    iteration improvement: itImpr ← 1
    local search iteration number: n ← 0
    list of possible swaps: LPS ← ∅
    while itImpr > 0 do
        LPS ← GENERATELPS(incSol)
        for all s ∈ LPS do
            current neighbour: currNeigh ← bettNeigh
            if swap is feasible then currNeigh ←DOSWAP(LPS_s, currNeigh)
                if currNeigh is better than bettNeigh then
                    bettNeigh ← currNeigh
                end if
            end if
        end for
        itImpr ← Value(bettNeigh) − Value(incSol)
        if itImpr > 0 then
            incSol ← bettNeigh
        end if
        LPS ← ∅
    end while
    oImpr ← Value(incSol) − Value(iniSol)
    return incSol
end function
```

As showed by the Algorithms 3 and 4, the main difference between the two approaches is, in fact, related to the definition of the current neighbour. Please note that although the better neighbour is indeed a kind of incumbent solution since it bases the generation of a new neighbourhood,



(a) Upgrades/downgrades not allowed (b) Upgrades/downgrades allowed

Figure 4.3: Representation of the increasing difficulty of instances A, B, and C.

the term *incumbent* is here used as the best solution found after exhaustively exploring an *LPS*. Therefore, the difference comes from the fact that while in the best-improvement approach the neighbour is always created by applying a swap to the incumbent solution, in the first-improvement approach the neighbour is created by applying a swap to the better solution found so far. This is the moment where the first-improvement nature is realized: if a swap is being applied to one previous neighbour, a new neighbourhood is being created. In fact, this local search moves quickly to new neighbourhoods, not exploring them thoroughly. One must consider that a neighbourhood is defined by the application of different swaps to the same solution. Therefore, if the base solution changes, (the one in which each swap is attempted in order to create a different neighbour), the neighbourhood changes, thus resulting in a first-fit-based procedure. Nevertheless, this approach is different from the traditional first-fit-based procedures since a new *LPS* is not created every time a new neighbourhood is started but only when the current *LPS* is exhausted.

4.3 Computational tests and results

In this section, the main characteristics of the computational tests run, as well as of the instances used, are presented. Furthermore, the results obtained are discussed.

4.3.1 Real instances

The data used to test this approach was retrieved from the company's database shortly before the beginning of the highest season in the car rental business in Portugal, the period between the middle of July and the middle of August. Therefore, the instances used reflect the busiest and most demanding time period faced by the company as far as tactical planning is concerned. The data includes also reservations concerning time periods further in the future, already on the system log.

Three instances were selected, each concerning a different vehicle group within the special vehicle fleet. In order to understand the capabilities of this method, their selection was based on

their distinct levels of difficulty, characterized by a daily metric of requests per capacity. For each day, the number of reservations that started, ended or were taking place was calculated, as well as the number of vehicles available whose block or *impro* date did not hinder its availability during the specific day considered. Figure 4.3 illustrates the evolution of said metric for instances A, B and C for the first 100 days of planning. This metric is able to portrait the double-faced notion of instance difficulty. On the one hand, the difficulty to solve an instance is directly proportional to the amount of data (reservations and vehicles). In this case, it is likewise important to consider the relationship between requests (reservations) and capacity to respond to said requests (vehicles). If there is an excess of capacity, as there is no cost associated with the use of a specific vehicle, it is easy to simply assign one reservation per vehicle - the difficulty is based on the association between consecutive reservations served by a specific vehicle. Therefore, the higher the values of the aforementioned metric achieved by an instance, the higher the difficulty. On the other hand, it is important to consider the scattering of the reservations throughout the time. Considering that the starting and ending dates of the reservations are inflexible, if many reservations are concentrated in a specific period, the achievement of a good solution will be hampered. As a result, an instance that shows a flat evolution of the aforementioned metric is simpler to solve than an instance whose evolution displays peaks and troughs.

Graphics 4.3a and 4.3b should be understood as autonomous means to compare *instances* and not complementary means to compare the *upgrading/no-upgrading situations*. In fact, by allowing upgrades and/or downgrades, the overall values of the ratio decrease deeply; essentially, the number of reservations is constant whilst the number of available vehicles understandably increases. Nevertheless, the amount of data increases significantly when comparing with the previous situation.

As a result, it is possible to classify comparatively the selected instances as easy (A), average (B), and difficult (C), for both upgrade/downgrade-allowing and -not-allowing situations. These instances were solved using three routines: the mimicry heuristic presented on Section 2.2 was used as a means to simulate the results currently obtained by the company; then, the GRASP algorithm was run using the first-improvement local search as well as the best-improvement one.

Upgrades and downgrades

Each instance was solved once, considering that no upgrades or downgrades were allowed, and once again, considering the possibility to upgrade or downgrade to the certified groups. In the latter situation, a new input was needed. As the company favours the allocation of reservations to the requested group over upgrading or downgrading situations, the vehicles that could be used belonging to auxiliary groups should already be occupied with the reservations of their own group currently on the database. The allocation plans to the auxiliary groups were therefore obtained using data retrieved from the company's system at the same time as the instances considered, and constructed using the greedy vehicle allocation constructive heuristic. Those were later supplied to both the mimicry heuristic and the GRASP procedures as pre-established data, as to provide

them with identical initial conditions thus leading to a more exact evaluation and comparison of the results obtained.

4.3.2 Tests

Considering the two variations of each of the three instances (allowing for upgrade/downgrade vehicles or not) and the two approaches to the swap local search, twelve different GRASP variants were tested. The algorithms described were developed in a VBA platform, using Microsoft Office Excel as the input/output interface. Each GRASP variant was run for 15 iterations and the α parameter that limits the RCL was set to be 25%. A standard personal computer was used, with an INTEL i7 2.70 GHz CPU and 8 GB installed memory.

4.3.3 Results

The general results may be found in Table 4.1, as far as the improvement between the results of the mimicry heuristic and the GRASP algorithm is concerned.

It was possible to verify that the *metaheuristic* approach lead to better results when the difficulty of the instance increased. In fact, for the easy instance (A), the increase on the profit of the company when compared to the values obtained by the mimicry heuristic was virtually non-existent, both considering and not considering upgrades. As for the average instance (B), when considering that no upgrades or downgrades to other groups were possible, both local search routines were able to increase the company's profit in 10,7%. When considering upgrading and downgrading vehicles, the increase was of 5,5% of the profit. The difficult instance (C), solved by both local search routines, whilst not considering upgrades, lead to an increase of over 8,5% of the profit. When considering these auxiliary vehicles, both routines were able to increase the results of the company by 12,1%. These increasing values of improvement were expected, as the current procedure used by the company, although extremely refined by the experience and knowledge of the operators, meets the limits of the human ability to apprehend large amounts of data and thus tackle big combinatorial problems.

For every instance and upgrading situation, the swift first-improvement local search solved the problem faster; in fact, the time was perceived to be proportional to the amount of vehicles to assign, increasing when considering upgrades and when solving instances with more vehicles available. For every case, nevertheless, the algorithms were run in an operationally acceptable time, considering that this *batch* approach to the problem is designed to be run during the night. For most cases, the swift local search was able to match the results of the best-improvement routine. Although for instance C, considering upgrades, a worst result was obtained using the first-improvement heuristic, this difference represented only 0,02% of the profit. For the cases considered, the contribution of the local search to the overall improvement was between 0,5% and 2%, decreasing with the difficulty of the instance.

It should also be noticed that for the average instance (B) and for the upgrade-allowing difficult instance (C), this approach was also able to allocate new reservations that the mimicry heuristic was not able to insert in the global vehicle schedule.

Table 4.1: Results of the different variants of the *metaheuristic* - comparison with the company's current procedures.

			As for GRASP iterations				**Global**		
			average	**std dev**	**worst**	**best**	**↓ empty transfer time**	**↑ new reserv**	**time (min)**
A	**No Up**	**BI**	0,1%	0,2%	-0,2%	0,5%	22%	0	6
A	**No Up**	**FI**	0,0%	0,2%	-0,3%	0,5%	22%	0	1
A	**Up**	**BI**	-0,1%	0,2%	-0,4%	0,3%	22%	0	59
A	**Up**	**FI**	-0,1%	0,2%	-0,5%	0,3%	22%	0	7
B	**No Up**	**BI**	10,5%	0,4%	10,0%	10,7%	35%	23	2
B	**No Up**	**FI**	10,5%	0,4%	10,0%	10,7%	35%	23	1
B	**Up**	**BI**	5,4%	0,1%	5,3%	5,5%	31%	23	91
B	**Up**	**FI**	5,4%	0,1%	5,3%	5,5%	29%	23	19
C	**No Up**	**BI**	8,4%	0,1%	8,3%	8,5%	48%	12	2
C	**No Up**	**FI**	8,4%	0,1%	8,3%	8,5%	46%	12	1
C	**Up**	**BI**	12,0%	0,1%	11,8%	12,1%	35%	0	149
C	**Up**	**FI**	11,9%	0,1%	11,8%	12,1%	34%	0	47

Note: Up - Allowing upgrades/downgrades; No Up - Not allowing upgrades/downgrades; BI - Best-improvement; FI - First-improvement

In fact, this approach brought improvement when considering the most difficult instances and groups, representing a significant financial impact for the company. Moreover, it was also able to fulfil more reservations, increasing the service level. Furthermore, it significantly reduced the time spent on empty transfers and, consequently, the environmental impact. Another major advantage is the re-allocation of two qualified and experienced employees to other value-adding tasks, namely within the strategic rather than tactical planning level.

Nevertheless, it is still possible to improve the approach to this problem. One of the main characteristics of this problem is the extreme inflexibility of the starting and finishing times of the reservations. As mentioned before, if there are many reservations concentrated in a specific time period, the problem becomes even more rigid and the solutions more difficult to improve by this method, since small adjustments made to a specific solution lead often to infeasible results. In fact, it is the constructive nature of the solving method that leads to the major improvement and not the small adjustments brought by the local search procedures. One can also conclude that, although the run times were deemed acceptable, bigger instances cannot be solved in reasonable time. Furthermore, a reflection upon the approach to the upgrades and downgrades must be made. The approach taken in order to depict the interdependency between the reservations and vehicle of different rental groups can indeed be enhanced. In it, a "main" group is considered and the previously planned schedules of the *auxiliary groups* are accounted for when deciding their assistance to the "main" group. Nevertheless, in order to obtain better results, rental groups that are linked by

upgrading and downgrading possibilities should be solved together. In that way, the results would not be hindered by the order chosen to plan the schedules of the rental groups.

Chapter 5

Matheuristic approach

As concluded in the previous chapter, the *metaheuristic* approach, despite bringing significant improvements, could be enhanced in order to tackle the problem in a less myopic way and also in order to be able to solve bigger instances within a reasonable time. Therefore, a new optimization- and mathematical modelling-based approach was attempted, as far as the *batch* mode is concerned.

However, solving real-sized combinatorial problems as the one here described with an exact method may prove to be impractical or even impossible due to the processing time required. Pochet and Wolsey (2006) state that the use of the exact method of *branch-and-bound*, presented on Section 5.2.1 and used by most of the solvers available, requires in theory a number of iterations that is exponential in the number of variables. Therefore, heuristic procedures that aim to achieve not the optimal solution but a good solution in an adequate time are again needed. Moreover, memory constraints hinder the solution of real-sized instances in a solver, as the number of variables easily overpasses reasonable figures. A procedure that combines heuristics and exact mathematical programming techniques (a *matheuristic*) and strategically decomposes the problem is thus proposed.

5.1 Exact formulation of the problem

The vehicle-reservation problem here described can be formulated as a Mixed Integer Linear Program (MIP). MIPs involve integer and continuous variables and linear constraints. In fact, as our goal is to optimize a linear constrained problem whose variables are integer and binary, it can be seen as a special kind of MIP - a Pure Integer Program (PIP) - with only binary variables.

The model here presented aims to represent the problem described as a network whose nodes are the reservations to be fulfilled. Therefore, the flows between the nodes can be displayed in tiers, each representing the journey or schedule of a specific vehicle. The cost of these arcs is referent to the repositioning (if needed) of the vehicle between consecutive reservations. Figure 5.1 aims

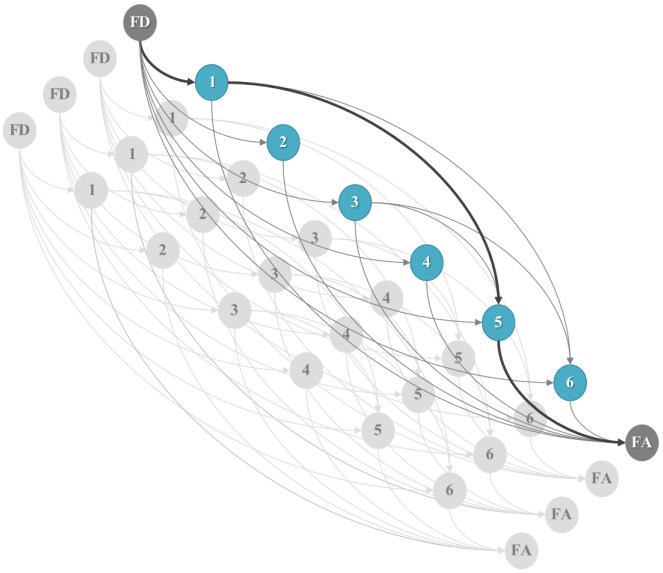


Figure 5.1: Tier representation of the network model.

to graphically represent an example of this network; an example of a feasible vehicle schedule is highlighted.

5.1.1 Nomenclature and parameters

The appropriate nomenclature regarding the characteristics of each reservation within a set of reservations $\mathscr{R}$ is herein presented. Two additional nodes are included in $\mathscr{R}$: the *fictitious departure node* 0 from where all vehicles start their journeys and the *fictitious arrival node* $R+1$ to where all vehicles return. Both nodes are represented in Figure 5.1. As far as reservations are concerned, one must take into account the following characteristics, considering $r \in \mathscr{R}$:

so_r station-out of reservation r (starting station);
do_r date-out of reservation r (starting date and time);
si_r station-in of reservation r (ending station);
di_r date-in of reservation r (ending station);
pft_r profit of reservation r;
gr_r vehicle group requested on reservation r;

$$st_r = \begin{cases} 1 & \text{if the status of reservation } r \text{ is “already confirmed to the client”,} \\ 0 & \text{on the opposite case;} \end{cases}$$

$$dga_r = \begin{cases} 1 & \text{if reservation } r \text{ accepts } \textit{downgrades}, \\ 0 & \text{on the opposite case.} \end{cases}$$

Furthermore, a set of vehicles $\mathscr{V}$ is considered and their main characteristics are as follows, considering $v \in \{1,\ldots,V\}$:

csi_v station-in of vehicle v (initial station where it is available);
cdi_v date-in of vehicle v (initial date and time when it is available);
gv_v group of vehicle v;
bd_v block date of vehicle v.

As it was discussed in Chapter 2, the limited unavailability periods that exist for some vehicles, the *impros*, can be seen as "reservations" that must be fulfilled by some specific vehicles, as they are time- and space-delimited. Therefore, in this model, the *impros* are registered as reservations with a null profit. An additional parameter will link the *impros* to their respective vehicles.

$$imp_r^v = \begin{cases} 1 & \text{if reservation } r \text{ is an } \textit{impro} \text{ of vehicle } v, \\ 0 & \text{on the opposite case.} \end{cases}$$

The other two important parameters concern the time and cost of the transfer between each pair of stations s_1, s_2:

cet_{s_1,s_2} cost of an empty transfer between stations s_1 and s_2;
tet_{s_1,s_2} time of an empty transfer between stations s_1 and s_2.

Another essential information regards the upgrading and downgrading possibilities between the requested group and the group of the assigned vehicle. That information is stored in three parameters:

$$s_{gr_r,gv_v} = \begin{cases} 1 & \text{if the requested group } gr_r \text{ is the same as the vehicle group } gv_v \ (gr_r{=}gv_v), \\ 0 & \text{on the opposite case;} \end{cases}$$

$$u_{gr_r,gv_v} = \begin{cases} 1 & \text{if the requested group } gr_r \text{ can be upgraded to group } gv_v, \\ 0 & \text{on the opposite case;} \end{cases}$$

$$d_{gr_r,gv_v} = \begin{cases} 1 & \text{if the requested group } gr_r \text{ can be downgraded to group } gv_v, \\ 0 & \text{on the opposite case.} \end{cases}$$

By definition, the values assigned to these parameters must fulfil the condition that for all combinations of groups gr_r, gr_v only one of the parameters s, u, and d may, at maximum, have the value one.

5.1.2 MIP Model

Decision variables

The decision variables of this model are binary variables that aim to describe the flow of each vehicle within its reservation schedule.

$$x_{ir}^v = \begin{cases} 1 & \text{if vehicle } v \text{ fulfills reservation } r \text{ after reservation } i, \\ 0 & \text{on the opposite case;} \end{cases}$$

The MIP model proposed is thus represented by Equations 5.1 to 5.12.:

$$\max \sum_{v\in\mathscr{V}} \left(\sum_{i=1}^{R}\sum_{r=1}^{R+1}(pft_r - cet_{si_i,so_r})x_{ir}^v + \sum_{r=1}^{R+1}(pft_r - cet_{csi_v,so_r})x_{0r}^v \right) s_{gr_r,gv_v} + \sum_{v\in\mathscr{V}} \left(\sum_{i=1}^{R}\sum_{r=1}^{R+1}(pft_r - cet_{si_i,so_r} - 1)x_{ir}^v + \sum_{r=1}^{R+1}(pft_r - cet_{csi_v,so_r} - 1)x_{0r}^v \right) u_{gr_r,gv_v} + \sum_{v\in\mathscr{V}}\sum_{i=0}^{R}\sum_{r=1}^{R+1} x_{ir}^v d_{gr_r,gv_v} \tag{5.1}$$

Subject to:

$$\sum_{r\in\mathscr{R}} x_{0r}^v \leq 1 \quad ,\forall v\in\mathscr{V} \tag{5.2}$$

$$\sum_{v\in\mathscr{V}}\sum_{i\in\mathscr{R}} x_{ir}^v \leq 1 \quad ,\forall r\in\mathscr{R} \tag{5.3}$$

$$\sum_{i\in\mathscr{R}} x_{iu}^v - \sum_{r\in\mathscr{R}} x_{ur}^v = 0 \quad ,\forall v\in\mathscr{V},\forall u\in\mathscr{R}\setminus\{0,R+1\} \tag{5.4}$$

$$(-di_i - tet_{si_i,so_r} + do_r)x_{ir}^v \geq 0 \quad ,\forall v\in\mathscr{V},\forall i,r\in\mathscr{R} \tag{5.5}$$

$$x_{ir}^v \leq s_{gr_r,gv_v} + u_{gr_r,gv_v} + d_{gr_r,gv_v} \quad ,\forall v\in\mathscr{V},\forall i,r\in\mathscr{R} \tag{5.6}$$

$$\sum_{i\in\mathscr{R}}\sum_{v\in\mathscr{V}} x_{ir}^v d_{gr_r,gv_v} \leq dga_r \quad ,\forall r\in\mathscr{R} \tag{5.7}$$

$$x_{0R+1}^v = 0 \quad ,\forall v\in\mathscr{V} \tag{5.8}$$

$$(di_i - bd_v)x_{iR+1}^v \leq 0 \quad ,\forall v\in\mathscr{V},\forall i\in\mathscr{R} \tag{5.9}$$

$$\sum_{i\in\mathscr{R}} x_{ir}^v \geq imp_r^v \quad ,\forall v\in\mathscr{V},\forall r\in\mathscr{R} \tag{5.10}$$

$$\sum_{v\in\mathscr{V}}\sum_{i\in\mathscr{R}} x_{ir}^v \geq st_r \quad ,\forall r\in\mathscr{R} \tag{5.11}$$

$$x_{ir}^v \in \{0,1\} \quad ,\forall r\in\mathscr{R} \tag{5.12}$$

Objective Function

The main goal of this model is to maximize the company's profit. Therefore, the objective function (5.1) represents the difference between the profit of each fulfilled reservation and the empty transfer cost incurred to make the vehicle ready to fulfil it. Nevertheless, in order to incorporate the characteristics of the problem related with the upgrading and downgrading possibilities, some artificial costs and profits are considered.

The first part of this function is related with vehicle-reservation assignments within the same rental group (without upgrades or downgrades). Here, the profit of each reservation is decreased by the cost of the empty transfer required to fulfil it. The first reservation to be fulfilled by each vehicle (the reservation that follows the fictitious departure node) is, however, accounted in a different way because it must reflect the initial station of each vehicle and the corresponding transfer cost. In fact, as far as the movements *from* the fictitious departure node are concerned, one can recognize that they represent real transfers and thus real costs - the repositioning of each vehicle from where

it will be available (station-in *csi*) to the starting station (station-out *so*) of its first reservation. Please note that these costs could not be given as a parameter $ctsfr_{si_{Fictitious}so_{FirstReserv}}$ because this is not a constant value. For each real destination $so_{FirstReserv}$, the origin $si_{Fictitious}$ depends on the availability characteristics of each vehicle, although there is only one fictitious departure node. Therefore, it was decided to assign a null value to the transfer costs from the fictitious station and include these departing costs in an extra segment added to the objective function. There was the alternative to use V different fictitious departure nodes, where the cost of repositioning the vehicles in order to make them available for their first reservation would be constant and therefore read as a parameter. However, in order to facilitate the understanding of the model and diminish the number of variables, the previously discussed option was chosen.

The second part of this function is related to the assignments that represent upgrades. In these, an approach similar to the first part is taken. Nevertheless, in order to use upgrades only when needed, these assignments have an additional cost of 1 monetary unit. This is an artificial penalization since the company does not incur in additional costs when performing an upgrade.

As for downgrades, their utilization is strictly restricted as they should be an absolute *last resource*. In order to accomplish this, the profit of the downgraded reservations is artificially altered. In fact, the model assigns to downgrades a marginal global profit of 1 monetary unit.

Constraints

As this is a network flow model, some general flow constraints must be considered. Firstly, one must define that the flow of each tier starts by at most one reservation (5.2). That is to say that each vehicle schedule "leaves" the fictitious departure node maximum once, and each reservation can only be fulfilled maximum once as well (5.3). Moreover, in order to maintain the flow of the schedules of each vehicle, if a certain reservation is assigned to it there must be a previous and a following reservation (5.4) (except for the fictitious nodes).

The specific constraints of this model are related to the availability of the vehicles on the requested time. Therefore, it must be defined that a reservation can only be fulfilled by a certain vehicle if its starting time is greater or equal to the ending time of the previous reservation plus the repositioning time needed (5.5). An alternative to this equation would be to only define the x_{ir}^{k} variables that verify this inequality, not considering the rest on the formulation. Nevertheless, nowadays commercial solvers eliminate quite effectively these variables.

It is also important to specify that a reservation can only be assigned to a vehicle of the same or compatible groups (5.6). Furthermore, a reservation can only be assigned to a downgrade group if the client specifically accepts that possibility (5.7).

An additional constraint is needed to avoid that a schedule is built connecting directly the fictitious nodes (5.8). Due to the penalization of the donwgrading option in the objective function, if this was to happen this schedule of zero real reservations would bring a profit of 1 monetary unit to the company.

As far as vehicle unavailability is concerned, it is important to assure that the last reservation of each vehicle does not surpass its block date (5.9), and that the *impro* "reservations" are assigned

to their respective vehicles (5.10). Moreover, all reservations that have been confirmed to the customer must be fulfilled (5.11).

Finally, all decision variables are declared as binaries (5.12).

5.2 Solution methods

In this section, the solution methods used or adapted in the *matheuristic* proposed are described. Firstly, a brief approach to the exact resolution of MIP problems with *branch-and-bound* is presented, as to enable the understanding of the basic procedures behind the MIP solver used and their effect on the efficiency and efficacy of the algorithm developed. Another exact method, local branching, is introduced as to explain a component of the *matheuristic* proposed. Afterwards, a basic introduction to *matheuristics* is presented and specific *matheuristics* used in the algorithm developed are described.

5.2.1 Exact methods

Branch-and-bound is the general solution algorithm used in exact MIP resolutions. It is based on solving a sequence of linear programs. The procedure is initialized by solving the linear relaxation of the original MIP. The linear relaxation consists on a linear program (LP) similar to the original MIP without the integrality constraint on the integer variables, thus becoming easier to solve. This is the first node of the branch-and-bound tree. The fractional value obtained is used to divide the linear program in two and generating two nodes: on the first, a constraint is added, stating that the solution value must be lesser or equal to the fractional value obtained rounded down to the nearest integer; on the second, the constraint added states that it must be greater or equal to the same value rounded up. This is called the *branching step*. These two LPs are then solved and the same procedure is applied to their solutions. The main goal is to look for the best integer solution found so far on the list of linear programs addressed. The selection of the next node to solve can be based in different criteria. This is applied to every node until all branches are pruned, i.e. all the nodes or formulations were explored. (Pochet and Wolsey, 2006)

Local Branching (LB) is a method proposed by Fischetti and Lodi (2003) to solve large MIP problems. This method, which is exact in nature, aims to strategically choose solution spaces that the MIP solver should tackle on a tactical level, thus enabling the use of generic MIP solvers as a black-box tool. Therefore, the paradigm behind this method is structured on two levels: on the higher level, a branching tree that defines solution subspaces (neighbourhoods) to be explored is constructed; on the lower level, the subspaces defined are explored by a generic MIP solver. The LB strategic, high-level tree is initialized by adding to the MIP problem the *local branching constraint*, the branching criterion.

In order to further understand this concept, please consider a generic minimization MIP, whose decision variables $x_j, j \in B$ are binaries. Given a feasible solution $\bar{x}$, let $\bar{S}$ be the set of the variables whose value is 1 in this solution. For this generic MIP, equation 5.13 displays the *local branching constraint*. The positive parameter k defines the k-OPT neighbourhood of $\bar{x}$ as the set of feasible

solutions satisfying this constraint. In fact, the left-hand side of the constraint represents the number of binary variables that change their value.

$$\Delta(x,\bar{x}) = \sum_{j\in\bar{S}}(1-x_j) + \sum_{j\in B\setminus\bar{S}} x_j \leq k \tag{5.13}$$

Two solution subspaces (nodes of the LB tree) are then defined by the disjunction $\Delta(x,\bar{x}) \leq k$ **or** $\Delta(x,\bar{x}) \geq k+1$. The MIP solver is then used, on a tactical level, to solve the former node (left-branch node). If an improved solution $\bar{x}^2$ is found, the latter node (right-branch node) is not solved yet branched again using a new disjunction: $\Delta(x,\bar{x}^2) \leq k$ **or** $\Delta(x,\bar{x}^2) \geq k+1$. Please note that when solving the new left-branch node stands for the original MIP problem "truncated" by two local branching constraints: $\Delta(x,\bar{x}) \geq k+1$ and $\Delta(x,\bar{x}^2) \leq k$. This node will also be solved by the MIP solver and the process is repeated until the solver is not able to find an improved solution. When this happens, the right-branch node is not branched again but solved as well; with this, the strategic LB tree is completed.

This method is indeed exact (in its basic formulation here described) since it does not ignore solution subspaces but guides the MIP solver on the exploration of the most promising neighbourhoods first, so as to enhance its powerful resolution tools, which are commonly based on a *branch-and-bound* approach. The authors also propose other enhancements to the method based on time limits for solving the nodes, and some diversification strategies.

5.2.2 *Matheuristics*

The definition of *matheuristics* is still "under construction" within the academic community. In some book chapters, such as Caserta and Voß (2009) or Boschetti et al. (2009), *matheuristics* are described as procedures that combine mathematical programming techniques with *metaheuristic* procedures. Nevertheless, more recent studies claim that this term should refer to the mix or interoperation of mathematical programming techniques with (general) heuristic procedures, thus encompassing methods such as *relax-and-fix* and *fix-and-optimize*. In fact, a recent journal of EURO (The Association of European Operational Research Societies), specifically dedicated to *matheuristics*, deems this broader definition as the currently accepted in the scientific community (EURO, 2013).

Relax-and-fix (RF) is described by Pochet and Wolsey (2006) as a primal construction heuristic. This procedure is based on the decomposition of the set of integer variables in a number of disjoint sets of decreasing importance. For example, if the variables concern the units produced each day by a certain production facility, they can be grouped by weeks and it is easy to understand that the closer the week the more important the set of variables indexed to it. The problem is first solved by relaxing the integrality constraint of the variables that belong to every subset except the first ones. The number of subsets herein considered is a parameter of the algorithm, as well as the criterion to divide the set of integer variables. For the next iteration, the problem is solved maintaining the integrality constraint for the following subsets, relaxing it for the subsets that follow those, and fixing the values found in the previous iteration for the the ones before. This process

is repeated until all the subsets are solved maintaining the integrality constraint. It is important to note that some overlapping may be set to occur, i.e. the subsets may be solved maintaining the integrality constraint more than once. The main advantage of this approach is the decomposition of a model in smaller sub-problems, probably easier to solve, although the certainty of reaching the global optimum is lost. Therefore, it is necessary to decide, in every problem, how to partition the variables in order to construct sub-sets that enhance the ability of the algorithm to achieve values as close as possible to the optimum. (Pochet and Wolsey, 2006)

Cherri, Toledo and Carravilla (2013) used an interesting *matheuristic* for a nesting problem based on the decomposition framework previously presented, although not involving relaxation of variables. This problem consists on cutting convex and non-convex pieces from an object with a fixed height and an infinite length, minimizing the total length needed to position different types of pieces with varying demand. In order to solve the MIP model formulated based on this problem in a reasonable computing time, the *matheuristic* proposed divides the pieces to allocate in four sets: fixed pieces, positioned pieces, free pieces, and waiting pieces. In the beginning, some pieces are free and the others are waiting to be included on the sub-problem. At each iteration, a specific number of pieces is added and a subMIP is solved for the first three sets. In the following iterations, the pieces will flow through the sets and progressively become fixed. The difference between positioned and free pieces is that the positioned pieces have already been tackled in a previous subMIP and only part of the set is allowed to change value in the solution. That is achieved by applying to this set of pieces a constraint based on the *local branching constraint* (see equation 5.13) that creates an upper bound on the number of allowed different values between two solutions.

It is possible to compare some elements of the nesting problem approached by Cherri, Toledo and Carravilla (2013) to the vehicle-reservation assignment problem herein considered. The pieces and their arrangement within a rectangular board can be compared to the reservations and their allocation within the vehicle schedule. The main difference is related once again to the fact that the "geometrical distribution" of the reservations within the vehicle schedule can only vary in heigh (vehicle considered) and not in width (time), as they are characterized by their starting and finishing date. Nevertheless, some interesting insights can be formulated from the analysis of the *matheuristic* proposed for the nesting problem and this can be adapted and extended to solve the mathematical model proposed.

5.3 *Matheuristic* algorithm proposed

It is important to consider that some special characteristics of the vehicle-reservation assignment problem in the car rental industry favour the mix and adaptation of different methods in order to develop an efficient algorithm. In fact, this assignment problem is highly constrained by the fact that the reservations (the entities to allocate) have rigid starting and finishing dates. At the same time, the allocation of a reservation to a vehicle influences the allocation of other reservations to this vehicle due to the empty transfers. Another major restraining characteristic

of the problem is its size. Therefore, it is important to control the number of variables tackled at a time. Nevertheless, one must remember that decisions related to different time spans influence each other.

The proposed *matheuristic* is a "time-based" relax-and-fix procedure. At each iteration, a slot of a time span, i.e. a partition of the whole horizon, is solved. As the reservations in each instance are the ones retrieved from the company's system at a specific moment, each instance usually comprises a high number of reservations for the closest time periods and this number decreases until the end of the planning horizon (determined by the last reservation in the system). It was hence set that each slot should comprise a constant number of reservations and not a constant time length. Therefore, the slots of the time span are are narrower in the first, busiest periods and become progressively wider.

Adapting and extending the procedure utilized by Cherri, Toledo and Carravilla (2013), the objects to allocate (in this case, the reservations of an instance) are divided in five sets: fixed reservations (Φ), assigned reservations (Δ), integer reservations (Υ), relaxed reservations (Θ), and waiting reservations (Ω). In each iteration, the assigned, the integer, and the relaxed reservations are tackled. As it was said, the number of reservations in each of the three sets tackled in each iteration (β) is constant. The reservations flow through the sets ordered by proximity of the starting date, starting as waiting reservations and then becoming progressively relaxed, integer, assigned, and finally fixed.

The fixed reservations (Φ) are the ones already definitely assigned to a vehicle and currently not considered on the sub-problem. The assigned reservations (Δ) are the ones that were previously considered in the method as integers and temporarily assigned to a vehicle; as to control modifications, only part of this set can be re-assigned. The integer reservations (Υ) are the ones which are solved keeping the integrality constraint whilst the relaxed ones (Θ) are solved relaxing this constraint. The waiting reservations (Ω) are the ones which are located further in the future and are not yet considered in the sub-problem. The flow of reservations through the sets is represented on Figure 5.2 and can also be seen on the pseudo-code for the proposed *matheuristic* presented on Algorithm 5.

In each iteration, a new constraint, inspired on the local branching paradigm, is added to the model of the sub-problem to be solved (5.16). This constraint uses the α parameter, calculated after the resolution of the sub-problem of the previous iteration, in order to limit the number of changes on the assigned set of variables (Δ) (which was previously the integer set (Υ)). This adaptation of the constraint proposed by Cherri, Toledo and Carravilla (2013) is important to achieve better global solutions.

Considering $\bar{x}$ the value of the decision variables in the solution of the previous sub-problem, one can define sets $B1$ and $B0$ as:

$$B1 = \{x_{ir}^{v} | r \in \Delta \wedge \bar{x}_{ir}^{v} = 1\} \tag{5.14}$$

$$B0 = \{x_{ir}^{v} | r \in \Delta \wedge \bar{x}_{ir}^{v} = 0\} \tag{5.15}$$

Algorithm 5 Pseudo-code for the proposed *matheuristic*.

function MAINFUNCTION($dataFile, \beta, timeLimit$)
  $IterSolution, GlobalSolution \leftarrow \emptyset$
  $\Phi, \Delta \leftarrow \emptyset$
  $\Upsilon \leftarrow$ first β reservations
  $\Theta \leftarrow$ second β reservations
  $\Omega \leftarrow$ rest of reservations
  $it, \alpha \leftarrow 0$
  repeat
    Adapt vehicle initial conditions (if needed)
    Fix variables (if needed)
    Generate model ($\alpha, \Delta \cup \Upsilon \cup \Theta$)
    Solve model ($timeLimit$)
    if $model_{status} = optimal$ or $feasible$ **then**
      $IterSolution \leftarrow$ solution
      $\alpha \leftarrow$ CALCULATEALPHA
    end if
    $\Phi \leftarrow \Delta \leftarrow \Upsilon \leftarrow \Theta \leftarrow$ next β reservations from Ω
    $GlobalSolution \leftarrow GlobalSolution + IterSolution$
    $it \leftarrow it + 1$
  until $\Theta = \emptyset$
  return $GlobalSolution$
end function

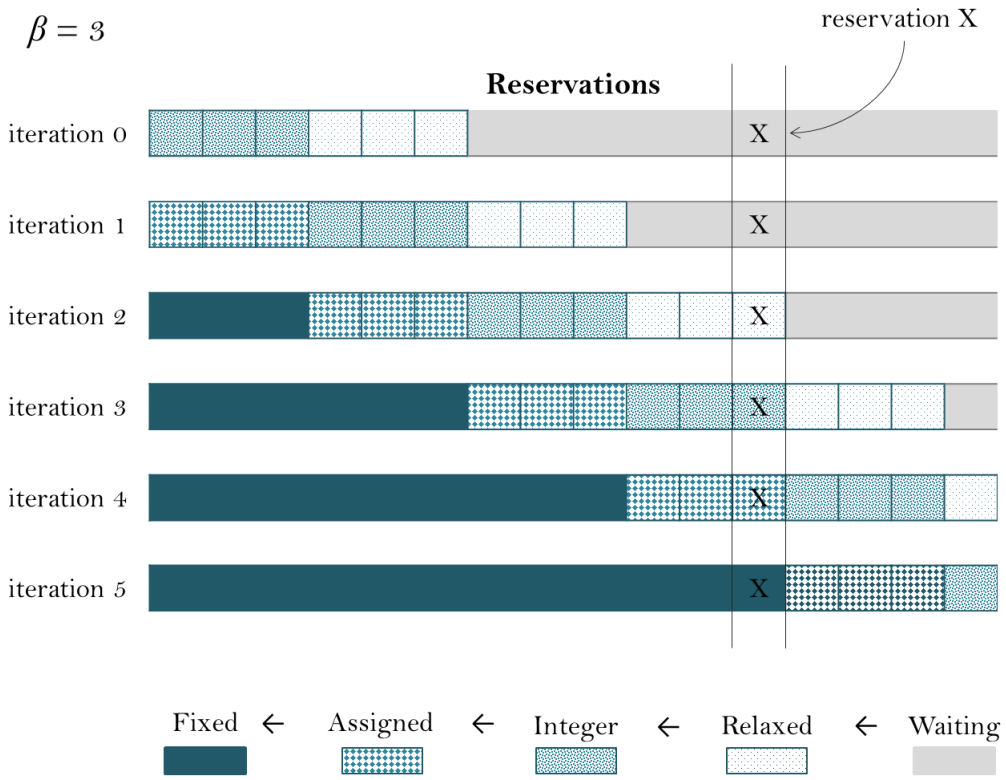


Figure 5.2: Visual representation of the reservation flow through the variable sets in different iterations.

The constraint is thus defined as:

$$\sum_{x_{ir}^k \in B1} (1 - x_{ir}^v) + \sum_{x_{ir}^k \in B0} x_{ir}^v \leq \alpha \tag{5.16}$$

This constraint states that the reservations that were previously temporarily assigned under the integrality constraint should remain as they were previously assigned, except for a specific number that is allowed to change. The solution of the previous iteration is represented by $\bar{x}$. The α parameter is the upper limit on the number of changes allowed on the variables representing this "positioning". At the end of each iteration, the parameter α is calculated; it represents the number of reservations that, at that point, belong to the integer set (Υ) and are assigned to some vehicle (5.17). Note that in the following iteration these reservations will belong to the assigned set (Δ).

$$\alpha = \sum_{v \in \mathscr{V}} \sum_{i \in \mathscr{R}} \sum_{r \in \Upsilon} x_{ir}^v \tag{5.17}$$

This allows the algorithm to correct previous assignments based on their effect on current reservations, previously not considered, in a swift and controlled way. In fact, the relation between reservations assigned to the same vehicle is extremely important within the vehicle-reservation assignment framework. The set of relaxed reservations to be solved (Θ) was added in relation to the method applied by Cherri, Toledo and Carravilla (2013) due to this issue. In this problem, due

to the empty transfers, this set allows the algorithm to acknowledge the influence that the present (integer) decisions have on the near-future time spans. The decision of "ignoring" a set of reservations (the waiting set Ω) instead of relaxing these variables as well was based on the fact that some reservations have so distant "time locations" that the connection and influence between them is virtually null.

Another idiosyncrasy of this problem relates to the unique initial conditions of each vehicle - due to their current occupation, each one will be available on a distinct time and location. Therefore, when *fixing* the variables on Φ, one needs to re-calculate the initial conditions of the involved vehicles, for the next time span.

Furthermore, a time limit is provided for the resolution of each sub-problem. As this algorithm does not guarantee the optimality of the global solution, it would not be useful to spend excessive computing time and effort in order to reach a local optimum, which could moreover be altered in the following iterations. If a feasible solution is obtained within the time limit, this solution is kept as the (local) optimal solution would be. Due to the characteristics of this problem and to preliminary computational experiments, this was deemed to be reasonable.

Throughout the preliminary computational results, in line with the implementation of this time limit, the number of reservations to be added to each set β was allowed to vary accordingly. If the optimal solution was not reached within the time limit, β was reduced (a lower bound for its value was also set). Nevertheless, this brought two significant drawbacks. On the one hand, due to the dynamics of the sets of variables linked through the different iterations, if β was increased, some variables would loose the "overlapped" integer solution, as they would move up to the fixed set (Φ) straight from the integer (Υ) set, for example. On the other hand, the run time was deeply increased without significant improvement of the global objective value; once again, the sub-problems only allow to reach the local optimum, possibly altered on the following iterations. Due to these conclusions, this variant was not considered on the proposed algorithm.

5.4 Computational tests and results

In this section, the main characteristics of the computational tests run, as well as of the instances used, are presented. Furthermore, the results obtained are discussed.

5.4.1 Generation of random reality-based instances

In order to understand in depth the opportunities for improvement brought by this new approach, it was felt the need to run a statistically relevant number of instances and compare them with the aforementioned mimicry heuristic (see Section 2.2). Nevertheless, the uniqueness of the data collected and the recognized interdependency between parameters lead to the need for exhaustive and time-consuming studies of statistical distributions and correlations. Therefore, in order to quickly obtain random data which could represent with reliability the characteristics of real data, the following procedure was designed. Firstly, the maximum and minimum number of

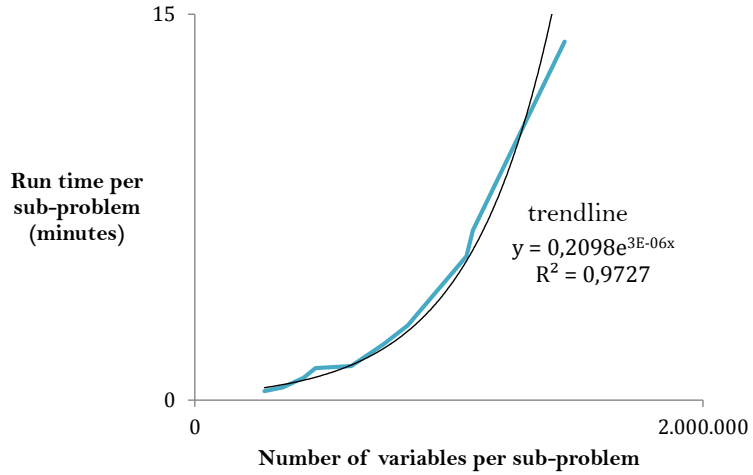


Figure 5.3: Plot of the preliminary experimental results: the time consumed by the algorithm grows exponentially with the number of variables in each sub-problem.

reservations and vehicles desired per instance was set, based on the structure of real instances. Afterwards, these values were randomly generated for each instance. Then, reservations and vehicles were randomly selected from a list with every reservation and vehicle retrieved from the database. For every reservation and vehicle, the characteristics represented on the random instances were the real ones. It was thus possible to obtain instances with reality-based characteristics, such as a peaked profile of starting reservations or a realistic use of each rental station.

Preliminary computational experiments conducted on a personal computer similar to the ones available on the company showed that the time consumed on the resolution of each instance by the proposed method was directly related to the number of variables of each sub-problem (see Figure 5.3). From this, it was possible to define (5.18), which relates the global run time with the β parameter and the number of vehicles and reservations. On the other hand, it is possible to understand that the greater the β parameter is, the closer to the resolution of the global model the procedure gets. Due to the practical application of this model, this trade-off was tackled by setting a time limit of 2.5 hours for the overall procedure and retrieving the β parameter from (5.18). The actual value used is the rounded value. In fact, the equation formulation is an approximation whose goal is to guide the run of the global procedure. The number of variables per sub-problem used is indeed true for the middle iterations, yet it may not be so for the first and last ones. Nevertheless, this has proven to be an effective approximation and was thus utilized within this framework.

$$\begin{aligned} &\textit{global run time} = \textit{sub-problem run time} \times \textit{number of sub-problems} \\ &= \left(0,2098 \times e^{3{,}00E^{-6} \times (3\beta^2)(\textit{num vehic}+2)}\right) \times \frac{\textit{num reserv}}{\beta} \end{aligned} \tag{5.18}$$

The main characteristics of the instances are presented in Table 5.1. These instances were not used to test the metaheuristic approach described in Chapter 4 due to its implementation a VBA

platform and overall structure and dynamics; memory and time constraints deemed it impossible or impractical. The smallest random instance generated was indeed approximately the same size as the biggest instance treated on Chapter 4.

Table 5.1: Main characteristics of the instances.

Instance	number of reservations	number of vehicles	number of *impros*	number of interconnected rental groups	β
1	1149	39	5	1	58
2	2683	30	1	5	58
3	2664	28	3	5	61
4	2379	26	0	3	65
5	1265	36	2	2	60
6	1707	27	0	4	68
7	1238	34	0	4	62
8	1555	26	1	1	70
9	1491	37	0	4	58
10	2440	25	4	5	66
11	2136	27	0	3	65
12	1287	31	4	2	66
13	2521	26	3	3	64
14	2363	35	2	4	55
15	2684	26	0	4	63
16	2696	29	2	4	59
17	1758	39	1	5	54
18	2012	27	1	1	66
19	2228	27	0	3	64
20	1976	39	0	3	53

Moreover, it is important to note that the approach to the upgrade and downgrade possibilities has been enhanced when compared to the one described on Chapter 4. In fact, all rental groups considered on an instance are tackled on the same hierarchical level, considering, if existent, more than one up/downgrade direction.

5.4.2 Tests

The algorithm described was developed in C++/IBM ILOG Concert Technology and was run on a HP Z820 Workstation computer with 128 GB of RAM memory, and with 2 CPUs (Xeon E5-2687W 8C 3.10 20MB 1600) with 16 threads each. The MIP Solver used was CPLEX 12.4.0.0. The time limit set for the resolution of each sub-problem was 5 minutes.

5.4.3 Results

The general results may be found in Table 5.2. The improvement they represent when compared with the company's current procedures is defined in Table 5.3.

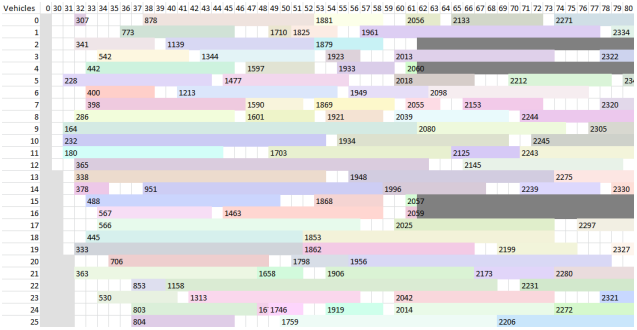
It is possible to observe that the profit has increased in average 33% when compared to the "hand-made" plans currently drawn by the company. For every instance, in fact, the improvement

Table 5.2: Results of the developed *matheuristic*.

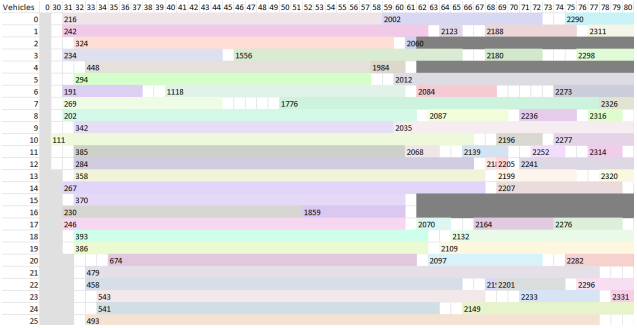
Instance	profit (obj function)	allocated reserv	empty transfer hours	run time [sec]
1	291.219	222	1984	314
2	272.822	218	1715	521
3	307.344	221	1664	597
4	255.208	203	1824	540
5	321.569	262	1710	718
6	237.004	198	1723	418
7	247.569	199	1997	444
8	272.140	196	1633	365
9	263.548	207	1966	474
10	240.097	203	1976	638
11	278.634	207	1777	581
12	260.340	202	1526	464
13	254.532	217	1606	704
14	329.680	256	2236	721
15	262.147	185	1688	731
16	319.811	231	2249	861
17	339.137	254	2066	468
18	266.758	200	1786	449
19	244.401	205	1796	484
20	345.103	280	2091	606

Table 5.3: Improvements brought by the developed *matheuristic* when comparing with the company's current procedures.

Instance	↑ profit	↑ allocated reserv	↓ empty transfer hours
1	51%	-7%	31%
2	32%	-10%	33%
3	27%	-13%	27%
4	38%	-11%	25%
5	35%	-9%	37%
6	34%	-7%	21%
7	27%	-12%	12%
8	31%	-8%	-49%
9	32%	-17%	34%
10	30%	-2%	7%
11	52%	-5%	29%
12	30%	-13%	19%
13	27%	-11%	28%
14	31%	-11%	7%
15	33%	-10%	37%
16	26%	-13%	17%
17	30%	-13%	28%
18	27%	-9%	6%
19	30%	-7%	25%
20	31%	-12%	34%
Average	**33%**	**-10%**	**20%**



(a) The company's current procedure allocates smaller reservations



(b) The proposed algorithm favours bigger, more profitable reservations

Figure 5.4: Extracts from the resulting vehicle schedule when solving an instance using different methods.

attained in terms of profit was very significant. Nevertheless, one can observe that the number of allocated reservations decreased in average 10%, decreasing the company's service level. In fact, in order to maximize the total profit, the algorithm proposed favoured the allocation of longer, more profitable reservations, thus minimizing as well the transfers. The manual procedure, on the opposite, is myopic, as it attempts to allocate the reservations as they appear on the system and it does not attempt to rearrange them on a frequent basis. Therefore, it tends to allocate reservations considering feasibility and not profitability. In fact, the average duration of the allocated reservations increased 20% when comparing the algorithm developed to the manual procedure. Figure 5.4 shows two extracts of solutions obtained for the same instance using these two methods where is possible to visually confirm this difference.

It is also important to notice the decrease on the number of hours spent in empty repositions. In fact, for most of the instances these travel times were significantly reduced, resulting in an average reduction of 20%. This is a powerful indicator that the CO_2 emissions have been significantly reduced, as desired by the company, which was expected since the empty transfers have a significant impact on the objective function.

The average run time for these instances was of 555 seconds (approximately 9 minutes). Note that this global run time is considerably smaller than the 2.5 hours initially projected because these tests were run in a computer with significantly more computational power than the ones available on the company.

Chapter 6

Conclusions and future work

This dissertation presented the problem faced by a Portuguese car rental company in managing the assignment of reservations to available special vehicles. The main objective was to maximize the total profit of the company, whilst reducing empty vehicle repositioning transfers between rental stations. A *metaheuristic* based on a GRASP procedure was developed to tackle this problem. Moreover, a network-flow model that assigns reservations to vehicles considering an heterogeneous fleet with interdependency between groups and customer authorization to downgrade, different reservation priority statuses, and two common types of vehicle unavailability was proposed. In order to solve this model on a real situation - with realistic instances and computational effort available - a *matheuristic* was proposed, based on a relax-and-fix framework and comprising a control mechanism based on local branching.

Both approaches were tested and were able to improve the company's profit, comparing to its current procedures. Although both methods were by nature heuristics and thus retrieved solutions whose optimality was not guaranteed, the *matheuristic* was able to achieve significantly better results within a reasonable time, and tackle bigger instances. Nevertheless, the development of the *metaheuristic* was very significant throughout the development of this project. In fact, it provided major insights related to the problem structure and dynamics as well as data properties. For instance, it was possible to acknowledge that this problem reacted better to greedy (randomized) construction than to incremental improvements, mainly due to the inflexibility of its data. This turned out to be relevant information when selecting the most appropriate method to solve the exact formulation of the problem.

The two approaches had different results as far as service level is concerned. While the *metaheuristic* was able to allocate more reservations than the company's current procedure, the *matheuristic* was less myopic and, in order to maximize the total profit, favoured the allocation of longer, more profitable reservations, thus minimizing as well the transfer costs. If the company considers this to be a major drawback, a new multi-objective formulation to the model should be arranged so that different trade-offs between profitability and service level could be studied. An

alternative to this new formulation could be the inclusion of a minimum service level constraint. Nevertheless, this would be better tackled on a more strategic level, mainly concerning fleet sizing decisions.

Besides the increase on profit, another major advantage of the utilization of these tools is the re-allocation of two qualified and experienced employees to other value-adding tasks, namely within the strategic rather than tactical planning level. Moreover, a significant reduction of the empty transfer times was achieved thus contributing to the company's desire to reduce its environmental impact. Furthermore, these tools may also provide the company with insights related to the strategic fleet sizing problem, as they can be used as a simulation tool to study vehicle buying/selling decisions (fleet sizing).

The company would benefit to integrate these tools with a more strategic analysis of their stations. A clustering model for the rental stations could be developed to tackle the fleet sizing and distribution, treating it as a stochastic process. If the vehicles were distributed between the clusters (pools), the assignment problem could become more tight and possibly easier to solve.

It would also be interesting to tackle the issue raised by Sbihi and Eglese (2007), related to the measuring of the environmental impact. The authors state that travel time is a better estimate of the degree of pollution caused, as this can be reduced by travelling for shorter times (and at better speeds). Therefore, another relevant future work would be to define a more precise measure of the environmental impact of the empty transfers and attempt to further include it on the objective(s) considered.

Furthermore, although this deterministic approach by itself was able to bring significant improvement to the company, uncertainties such as delays or *walk-in* customers could be incorporated on further stochastic developments of the model.

References

Abara, J. (1989), 'Applying integer linear programming to the fleet assignment problem', *Interfaces* **19**(4), 20–28.

Auto Rental News (2012), 'Global Business Travel Association Releases Sustainability Survey', *[online]* **Accessed**(4 March 2013).
URL: *www.autorentalnews.com/news/print/story/2012/07/global-business-travel-association-releases-sustainability-survey.aspx*

Bandeira, D. L., Becker, J. a. L. and Borenstein, D. (2009), 'A DSS for integrated distribution of empty and full containers', *Decision Support Systems* **47**, 383–397.

Beaujon, G. J. and Turnquist, M. A. (1991), 'A model for fleet sizing and vehicle allocation', *Transportation Science* **25**(1), 19–45.

Blum, C. and Roli, A. (2003), 'Metaheuristics in combinatorial optimization: overview and conceptual comparison', *ACM Comput. Surv.* **35**(3), 268–308.

Bojović, N. (2002), 'A general system theory approach to rail freight car fleet sizing', *European Journal of Operational Research* **136**, 136–172.

Boschetti, M. A., Maniezzo, V., Roffilli, M. and Röhler, A. B. (2009), Matheuristics: optimization, simulation and control, *in* M. J. Blesa, C. Blum, L. D. Gaspero, A. Roli, M. Sampels and A. Schaerf, eds, 'Hybrid metaheuristics', Springer Berlin Heidelberg, pp. 171–177.

Carroll, W. J. and Grimes, R. C. (1995), 'Evolutionary Change in Product Management: Experiences in the Car Rental Industry', *Interfaces* **25**(5), 84–104.

Caserta, M. and Voß, S. (2009), Metaheuristics: intelligent problem solving, *in* V. Maniezzo, T. Stützle and S. Voß, eds, 'Matheuristics: hybridizing metaheuristics and mathematical programming', 1 edn, Spinger, chapter 1, pp. 1–37.

Cherri, L. H., Toledo, F. M. B. and Carravilla, M. A. (2013), Uma math-heurística para o problema de corte de peças irregulares, *in* 'XLV SBPO – Simpósio Brasileiro de Pesquisa Operacional'.

Crainic, T. G. (2000), 'Service network design in freight transportation', *European Journal of Operational Research* **122**(2), 272–288.

Crainic, T. G., Gendreau, M. and Dejax, P. (1993), 'Dynamic and stochastic models for the allocation of empty containers', *Operations Research* **41**(1), 102–126.

Dejax, P. J. and Crainic, T. G. (1987), 'A review of empty flows and fleet management models in freight transportation', *Transportation Science* **21**(4), 227–247.

Dell'Amico, M., Matteo, F. and Toth, P. (1993), 'Heuristic algorithms for the multiple depot vehicle scheduling problem', *Management Science* **39**(1), 115–126.

EURO (2013), 'Matheuristics', *[online] EURO - The Association of the European Operational Research Societies* **Accessed**(13 June 2013).
URL: *http://www.euro-online.org/web/pages/1527/matheuristics*

Feo, T. A. and Resende, M. G. C. (1995), 'Greedy Randomized Adaptive Search Procedures', *Journal of Global Optimization* **6**, 109–133.

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.

Fischetti, M. and Lodi, A. (2003), 'Local branching', *Mathematical Programming* **98**(1-3), 23–47.

Hane, C. A., Barnhart, C., Johnson, E. L., Marsten, R. E., Nemhauser, G. L. and Sigismondi, G. (1995), 'The fleet assignment problem: Solving a large-scale integer program', *Mathematical Programming* **70**, 211–232.

ITM (2013), 'Car rental industry - an overview', *[online] The Institute of Transportation Management* **Accessed**(14 June 2013).
URL: *http://www.itmworld.com/subcategory_detail.php?iResearchId=9258&iCategoryId=62*

Li, Z. and Tao, F. (2010), 'On determining optimal fleet size and vehicle transfer policy for a car rental company', *Computers & Operations Research* **37**(2), 341–350.

Pachon, J., Iakovou, E. and Ip, C. (2006), 'Vehicle fleet planning in the car rental industry', *Journal of Revenue and Pricing Management* **5**(3), 221–236.

Pachon, J., Iakovou, E., Ip, C. and Aboudi, R. (2003), 'A synthesis of tactical fleet planning models for the car rental industry', *IEE Transactions* **35**(9), 907–916.

Pochet, Y. and Wolsey, L. A. (2006), *Production planning by mixed integer programming*, Springer Science+Business Media, Inc, New York.

Sbihi, A. and Eglese, R. W. (2007), 'Combinatorial optimization and Green Logistics', *4OR-Q J Oper Res* **5**(2), 99–116.

Sherali, H. D. and Suharko, A. B. (1998), 'A tactical decision support system for empty railcar management', *Transportation Science* **32**(4), 306–329.

Sherali, H. D. and Tuncbilek, C. H. (1997), 'Static and dynamic time-space strategic models and algorithms for multi-level rail-car fleet management', *Management Science* **43**(2), 235–250.

Song, D.-P. and Earl, C. F. (2008), 'Optimal empty vehicle repositioning and fleet-sizing for two-depot service systems', *European Journal of Operational Research* **185**(2), 760–777.

Spieckermann, S. and Voß, S. (1995), 'A case study in empty railcar distribution', *European Journal of Operational Research* **87**, 586–598.

Subramanian, R., Scheff, R. P., Quillinan, J. D., Wiper, D. S. and Marsten, R. E. (1994), 'Coldstart: Fleet Assignment at Delta Air Lines', *Interfaces* **24**(1), 104–120.

Yaghini, M. and Khandaghabadi, Z. (2013), 'A hybrid metaheuristic algorithm for dynamic rail car fleet sizing problem', *Applied Mathematical Modelling* **37**(6), 4127–4138.

Yang, Y., Jin, W. and Hao, X. (2008), 'Car rental logistics problem: a review of literature', *IEEE International Conference on Service Operations and Logistics, and Informatics. IEEE/SOLI 2008* **2**, 2815–2819.