UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

ALBERTO FRANCISCO KUMMER NETO

A study on the home care routing and scheduling problem

Thesis presented in partial fulfillment of the requirements for the degree of Doctor of Computer Science

Advisor: Prof[a]. Dr[a]. Luciana Salete Buriol

Porto Alegre
September 2021

CIP — CATALOGING-IN-PUBLICATION

Kummer Neto, Alberto Francisco

A study on the home care routing and scheduling problem / Alberto Francisco Kummer Neto. – Porto Alegre: PPGC da UFRGS, 2021.

188 f.: il.

Thesis (Ph.D.) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2021. Advisor: Luciana Salete Buriol.

1. Home health care problem. 2. Vehicle routing problem. 3. Time-window. 4. Route inter-dependency constraints. I. Buriol, Luciana Salete. II. Título.

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
Reitor: Prof. Carlos André Bulhões
Vice-Reitora: Prof[a]. Patricia Pranke
Pró-Reitor de Pós-Graduação: Prof. Celso Giannetti Loureiro Chaves
Diretora do Instituto de Informática: Prof[a]. Carla Maria Dal Sasso Freitas
Coordenadora do PPGC: Prof[a]. Luciana Salete Buriol
Bibliotecária-chefe do Instituto de Informática: Beatriz Regina Bastos Haro

“Don’t always consider all your options. Don’t necessarily go for the outcome that seems best every time. Make a mess on occasion. Travel light. Let things wait. Trust your instincts and don’t think too long. Relax. Toss a coin. Forgive, but don’t forget. To thine own self be true. ”

— Brian Christian & Tom Griffiths

ACKNOWLEDGMENTS

Some so many people make this work real. First, I would like to thank my parents for the gift of life and all their support. Without their help and effort, I certainly would never have come this far.

Secondly, I want to thank my advisor Luciana Buriol for her support during my research, especially when I got lost or stuck in some sharp corner. I also could not leave out my thanks to Professor Olinto Araújo for introducing operational research in 2013. Thanks for all the help since then.

I also owe a huge thank you to the Informatics Institute at the Federal University of Rio Grande do Sul; The support by the staff was nothing less than excellent. I also leave here a big shout-out for all my research fellows of Lab 207. All the moments we share are memorable! Thank you Alex Gliesch, Artur Brum, Carlo Sartori, Gabriel Langeloh, Henrique Becker, Luísa Santos, Marcelo Friske, Tadeu Zubaran, Victória Simonetti, and Wesley Kaiser. Also, thanks to Thiago Silveira for all our friendship from 2010.

Lastly, I would like to thank the Coordination for the Improvement of Higher Education Personnel (CAPES) for my scholarship and for supporting my research. Thank you for still believing and helping so many researchers in such a moment of uncertainty for Brazilian science.

ABSTRACT

This thesis approaches the home health care problem, focusing on the routing problems surrounding such systems. These problems are especially important due to the world-wide tendency of increased life expectancy and, consequently, global aging. In Brazil, such a system is key regarding the government's objective of implementing the so-called de-hospitalization. Home health care services positively impacts the patients' mental well-being and comfort and helps prevent contamination by common pathogens of hospital environments. Such a characteristic is also very desirable in scenarios like the current COVID-19 pandemic. The problem we study consists of developing routes for every caregiver in the problem while scheduling the patients' visits by such caregivers. Such routes must be crafted while observing the working regulations, minimizing costs, and maximizing the service levels and satisfaction of both caregivers and patients. Some additional constraints are set for patients requiring multiple visits by caregivers with distinct qualifications. We propose three techniques to solve the problem, and we study several strategies for obtaining stronger lower bounds for such a routing problem. The first solution method consists of a fix-and-optimize *matheuristic* that employs a mixed integer programming solver to iteratively optimize routes of pairs of caregivers. Due to problems to scale the matheuristic in larger instances, we also propose two meta-heuristic algorithms. The first meta-heuristic consists of a biased random-key genetic algorithm, which allows us to indirectly explore the problem's solution space. The third technique–our second meta-heuristic proposal–extends our genetic algorithm with additional components to improve algorithms' intensification and diversification capabilities. To obtain strong lower bounds, we propose several scenarios for employing a MIP solver, and we describe a technique based on combinatorial lower bounds of the problem. Results for a literature dataset indicate that the proposed techniques are effective and efficient compared to previous methods. We also propose a methodology for generating new realistic instances for a home health care system. We introduce a new dataset, and we provide both lower and upper bounds for these new instances. We also report computational results for our best-performing meta-heuristic in these test cases.

Keywords: Home health care problem. vehicle routing problem. time-window. route inter-dependency constraints.

Um estudo sobre o problema de atendimento de saúde domiciliar

RESUMO

Esta tese aborda o problema de atendimento de saúde domiciliar, e foca nos problemas de roteamento ao redor destes sistemas. Esses problemas são especialmente importantes por conta da tendência mundial do aumento da expectativa de vida humana, e consequentemente, o envelhecimento global. Para o governo do Brasil, esse tipo de sistema é muito importante, contribuíndo com a implementação de políticas de desospitalização. Serviços de atendimento domiciliar impactam positivamente no bem estar mental e conforto dos pacientes, e ajudam a prevenir contaminação por patógenos comuns a ambientes hospitalares assim como na atual pandemia de COVID-19. O problema que estudamos consiste no desenvolvimento de rotas para cada médico do problema, e simultaneamente definir o agendamento das visitas. Essas rotas devem ser construídas com observância às regras trabalhistas vigentes, minimizando custos e maximizando o nível dos serviços e satisfação de ambos médicos e pacientes. Algumas restrições adicionais são impostas quando um paciente requer múltiplas visitas por médicos de especialidades distintas. Propuseram-se três técnicas de resolução do problema, e estudaram-se diversas estratégias para obtenção de limites inferiores fortes para tal problema de roteamento. O primeiro método consiste em uma matheurística *fix-and-optimize* que usa um resolvedor de programação inteira para otimizar iterativamente, rotas de pares de médicos. Por conta de dificuldades de escalabilidade, também propuseram-se duas meta-heurísticas. A primeira meta-heurística consiste em um algoritmo genético com chaves aleatórias viciadas, que permite explorar indiretamente o espaço de soluções do problema. A terceira técnica de resolução estende o algoritmo genético através de componentes adicionais de intensificação e diversificação. Para obter limitantes inferiores fortes, propuseram-se diversos cenários de aplicação de um resolvedor MIP, e descreveu-se uma técnica de limitantes inferiores combinatórios. Resultados experimentais para instâncias da literatura indicam que as técnicas propostas são efetivas e eficientes quando comparadas com métodos anteriores. Também propôs-se uma metodologia para geração de novas instâncias, e propuseram-se limitantes inferiores e superiores para estas. Também reportaram-se seus resultados computacionais frente ao método de solução mais eficiente proposto.

Palavras-chave: problema de atendimento de saúde domiciliar, problema de roteamento de veículos, janelas de tempo, restrições de inter-dependência entre rotas.

LIST OF ABBREVIATIONS AND ACRONYMS

AAC	Automatic Algorithm Configuration
API	Application Programming Interface
BB	Branch-and-Bound
BC	Branch-and-Cut
BCP	Branch-and-Cut-and-Price
BRKGA	Biased Random Key Genetic Algorithm
BRKGA-MP-IPR	Multi-population Multi-Parent Biased Random Key Genetic Algorithm with Implicit Path-Relinking
CG	Column Generation
CLB	Combinatorial Lower Bound
F&O	Fix-and-Optimize
GA	Genetic Algorithm
HHC	Home Health Care
HHCP	Home Health Care Problem
HHCRSP	Home Health Care Routing and Scheduling Problem
HCP	Home Care Problem
IPR	Implicit Path Relinking
IBGE	Instituto Brasileiro de Geografia e Estatística (*Brazilian Institute of Geography and Statistics*)
LB	Lower Bound
mTSP	Multiple Traveling Salesman Problem
mTSPTW	Multiple Traveling Salesman Problem with Time Windows
MILP	Mixed Integer Linear Programming
MIP	Mixed-Integer Programming
MP	Multi-Parent

MWIS	Maximum Weighted Independent Set Problem
OR	Operations Research
RKGA	Random Key Genetic Algorithm
RMP	Relaxed Master Problem
SP	Set Partition Problem
STL	Step Time Limit
TSP	Traveling Salesman Problem
TW	Time Windows
VRP	Vehicle Routing Problem
VRPTW	Vehicle Routing Problem with Time Windows
VRPSolver	Vehicle Routing Problem Solver (c.f. Pessoa et al. (2020))

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

As the global population ages, the demand for healthcare services has a growth tendency. With the aging, we expect an increasing number of people with various health problems that are more common in older than in younger people–cancer, fractured hips, strokes, dementia, for example–and many of these people will have comorbidities and require clinical assistance for several years (RECHEL et al., 2013). Consequently, these increasing demands directly impact the need for healthcare and social care services, consuming many hospitalization-related resources. Consequently, we can expect a significant allocation of the health system budget to services targeted mainly to older people, which reduces health resources available to the rest of the population.

The global aging wave is rooted in the significant societal and scientific development observed in the last century, mainly by the advances in agriculture, medicine, and sanitation practice (BASHIR; CHABROL; CAUX, 2012). Despite progressing more slowly than developed countries, we can also verify this aging tendency in developing countries. In the case of Brazil, we observed only in the last decades the shifting of the average life expectancy from 38 years to 50+ years (NASRI, 2008; MIRANDA; MENDES; SILVA, 2016). According to the last census, the Brazilian population of 2010 corresponded to 14M people above 65 years old–7.32% of the total population. The expectation is to see these numbers grow to 51M people by 2050–almost 25% of the population (IBGE, 2019). A similar tendency is observed in the United States of America, as well as several European countries (LANDERS et al., 2016; GENET et al., 2012; EMILIANO; TELHADA; CARVALHO, 2017).

Such a subtle increase in healthcare demands can easily lead to a collapse of the traditional healthcare systems. For this reason, health agencies are introducing alternatives to refrain from the massive increment of hospitalizations resources. Nursing homes were conceived as one of the first attempts to mitigate the increasing number of hospital beds, but the availability of such services provided by different countries varies substantially and is unrelated to the age structure of their respective populations (MCKEE, 2004). In some cases, there is also a socioeconomic barrier to accessing such services. In Brazil, for example, nursing home facilities are only available through private healthcare providers; therefore, not all families can afford it. As an alternative, the Brazilian public health agency is implementing the so-called program *Melhor em Casa–better in home* in free translation–to assist the elderly and some other people in the home care format.

This home care format for healthcare-providing is not exactly new. As Rosenfeld and Russell (2012) observed, there is a global tendency to move from hospital-based long-term care providing or nursing homes to services performed at patient's site. According to the authors, the percentage of older people across European countries is not always comparable, but the percentage of public health budget destined for home care services is around 1% to 5% in most cases–i.e. all European countries apply some of their budgets in the form of some home care services (GENET et al., 2012). In Canada, mostly long-term care for the elderly is provided in nursing homes, but there is a growing number private-for-profit home care agencies (MARCHILDON; ALLIN, 2021).

In the last two decades, home care systems have been introduced in several countries to broaden access to healthcare services (GRIECO; UTLEY; CROWE, 2020). In addition to healthcare services, some countries also offer the so-called social care through the home care format. These combined services allow a greater level of independence for older people by offering professional assistance on day-to-day activities such as meal preparation and house cleaning. Thus, home social care has an important impact on its users' quality of life and allows them to stay living at their homes and family instead of moving to nurse homes and similar facilities (CARELLO; LANZARONE, 2014; GOMES; RAMOS, 2019). Furthermore, there is a number of people that can benefit from these social and healthcare services; Patients with some disability and patients with some stable health condition are also eligible to home care services (HULSHOF et al., 2012).

1.1 Home care problems

This section aims to introduce a simplified version of the problem targeted in our study to motivate the reader and set the grounds of what type of problem we are interested in solving. In the context of this thesis, a home care problem (HCP) consists of a generalization of the vehicle routing problem with time-windows (VRPTW) with some additional constraints. More specifically, we focus our discussion on a more concrete variant of the HCP arriving in the context of home hospitalization, known as the home health care routing and scheduling problem–HHCRSP (CHENG; RICH, 1998; MANKOWSKA; MEISEL; BIERWIRTH, 2014). By such generalization, both problems share the same theoretical difficulty to be solved on its decision version that asks whether a solution exists with a total travel time of D units (KARP, 1972; GAREY; JOHNSON, 1979). Since the VRPTW has already been proved to belong to the $\mathcal{NP}$-hard class of problems (TAILLARD et al.,

1997), HCPs also belong to the $\mathcal{NP}$-hard complexity class (STEEG; SCHRÖDER, 2008). For a more detailed description of our target problem, c.f. Section 2.

An HCP consists of scheduling the visit of a set of patients $\mathcal{C}$ by caregivers from the set $\mathcal{V}$. Each patient has a time-window in which they can be visited. A square matrix M contains the travel times between all the locations, i.e., the patients and the depot node. The objective is to find one route for each caregiver, starting and finishing at the depot, minimizing the travel time of the caregivers while visiting all patients. As the reader can note, patients and caregivers of HCP terminology map to nodes and vehicles of a VRPTW. Despite simple, such a definition of HCP covers most of the problem introduced by Cheng and Rich (1998), which is considered one of the first publications of home care literature approaching the HCP as a vehicle routing problem (GRIECO; UTLEY; CROWE, 2020). Figure 1.1 introduces a solved example of this problem. The depot node is placed in the middle of the region, with five patients scattered around it. The patients have the time-windows defined in ranges with the tilde symbol '~'. This solution contains two caregivers, depicted by the routes in blue and red.

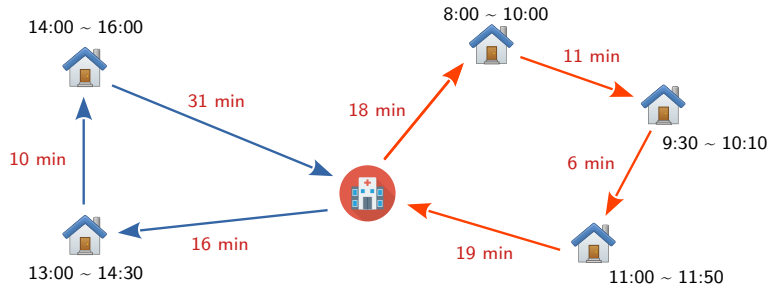


Figure 1.1 – Example of a HCP comprising two caregivers and five patients.
Source: the author.

Similar to the VRPTW, it is not easy to find optimal solutions to an HCP (GENDREAU; POTVIN et al., 2010). Depending upon the problem's constraints, the problem becomes so much structured that even finding a feasible solution can be very challenging to state-of-art solution methods (POLNIK; RICCARDI; AKARTUNAL, 2020). To the best of our knowledge, the standard approaches applying state-of-art mixed-integer programming (MIP) solvers can only solve problems spanning up to one week of planning and considering only a few dozen patients and a small number of caregivers. The most successful exact approaches for solving medium-sized problems employ complex and tailored methods such as branch-and-price (BREDSTROM; RONNQVIST, 2007; RASMUSSEN et al., 2012; GRENOUILLEAU et al., 2019). Due to the difficulty of providing solutions on larger test cases–mostly by a prohibitive consumption of computational resources–several authors

propose heuristic and *matheuristics* solving methods for handling larger instances of the problem (LASFARGEAS; GAGNÉ; SIOUD, 2019; KUMMER; BURIOL; ARAÚJO, 2019; LIU; TAO; XIE, 2019).

1.2 Motivation

In general, HCPs tend to be hard to solve to optimality, and the variety of problems arriving in the home care makes this research very rich and interesting. Furthermore, the global interest in such problems has a growth tendency due to the aging population around the globe. In addition to our interest of tackling complex yet a important problem, we also motivate our research in the real home health care problem arriving in Brazil. To the best of our knowledge, the Brazilian authorities do not provide any computational infrastructure to help manage home care system of the *Melhor em casa* program (Ministério da Saúde, 2013). More specifically, in Porto Alegre–capital of Rio Grande do Sul state, 19 teams are working at the home health care program, and the planning of the system is entirely manual. Furthermore, there are already some papers in the literature applying optimization methods to solve real cases of HCPs, and there is a consistent report of gains when practicing such methodology, both in terms of operational costs and fairness among the caregivers (EVEBORN; FLISBERG; RÖNNQVIST, 2006; TRAUTSAMWIESER; HIRSCH, 2011).

1.3 Research goal and contributions

The research goal of this work are twofold. First, we would like to advance the state-of-art of solution methods for HCPs, generating high-quality solutions and, at the same time, keeping the computational resources consumption (memory and processing times) at reasonable values. We can then compare our results with the previous ones from the literature and hopefully, we can devise some general knowledge from such experiments that can be applied to other similar problems. Secondly, we would like to provide enough evidence that an optimization algorithm can benefit *Melhor em Casa's* program (Ministério da Saúde, 2013).

Regarding the contributions of our research, we provide new stronger lower bounds for all the 70 instances from a benchmark dataset from the literature. Furthermore, we propose a new matheuristic and two new meta-heuristic approaches for solving a literature's home health care problem. The fix-and-optimize matheuristic (F&O) employs a model from the literature to devise a MIP-based local search algorithm (CHEN, 2015). With this matheuristic, we find new best-known solutions to several instances of a benchmark dataset of the literature, but unfortunately, the F&O seems to be only effective when solving instances with up to a hundred patients due to some characteristics of the problem. The proposed meta-heuristics consists of two variations of a biased random-key genetic algorithm. The first applies the original algorithm proposed by Gonçalves and Resende (2011), while the second introduces a revised version of the first meta-heuristic, augmented with the intensification components proposed by Andrade et al. (2021). With this later meta-heuristic, we obtain new best-known solutions than both the literature and the previous results of our F&O matheuristic.

Due to the difficult data access and the strike of COVID-19 in Porto Alegre, we approach the validation phase of the proposed solution methods through a new instance dataset that considers the characteristics of the real problem. These new instances consider the typical range of visit duration of the real problem among the simulated data. This new dataset also uses the `ovig` tool introduced by Sartori and Buriol (2020) to generate realistic travel times within the metropolitan region. Section 9.3 presents other important characteristics of the real problem take into consideration while generating new instances.

We are enthusiasts of the open-source initiative, and we think that science is something we build together. Thus, another minor contribution is that all codes for solution methods and instance generation are freely available on the internet. We also made available a repository that contains all the log files generated during our computational experiments, and some scripts to help parsing data and perform statistical measurements and comparisons. We did our best organizing these materials, and they are available in the following repositories.

- The source code for the MIP model is available at <https://github.com/afkummer/hhcrsp-mip-lower-bounds-2021>. These codes correspond to the algorithms proposed in Sections 6.1 and 6.2 and the computational results of Section 10.2;

- The model for combinatorial experiments with VRPSolver is available at <https://github.com/afkummer/hhcrsp-clb-lower-bounds-2021>. This implementation corresponds to the algorithm of Section 6.3, and computational results of Section 10.3;
- The source code for the fix-and-optimize is available at <https://github.com/afkummer/sbpo2019-fix-and-optimize>. These codes correspond to the matheuristic we propose in Chapter 7, and the computational results of Section 10.4;
- The repository <https://github.com/afkummer/gecco2020-brkga> contains our implementation of the BRKGA meta-heuristic described in Chapter 8 and tested in Section 10.5;
- The implementation of the more sophisticated BRKGA-MP-IPR is available at <https://github.com/afkummer/brkga-mp-ipr-hhcrsp-2021>. This implementation corresponds to the extended BRKGA algorithm of Section 8.4 and the reported results in Sections 10.6;
- The source code of the proposed instance generator proposed in Section 9.3 is available at <https://github.com/afkummer/ovig>. All the generated instances and logs regarding our instance hardness measurement are available at <https://github.com/afkummer/hhcrsp-dataset-2021>;
- Finally, the repository at <https://github.com/afkummer/phd-thesis-results-2021> contains most of the logs for all the experiments reported in this thesis, with detailed data regarding the extensive runs.

1.4 Published works

A great part of the methods and results presented in this thesis were originally published in the following works.

- Our work Kummer, Buriol and Araújo (2019) corresponds to the fix-and-optimize algorithm of Chapter 7. This paper was presented in the LI *Brazilian Operational Research Symposium* (SBPO) in 2019. Our work also received the *Roberto Diéguez Galvão Prize* for the best symposium paper of 2019;

- The paper Kummer, Buriol and Araújo (2020) contains our original proposal of the biased random-key genetic algorithm for the home health care routing and scheduling problem. This work was presented in *The Genetic and Evolutionary Computation Conference 2020* (*GECCO'*20);
- We presented a summary of our results and ideas regarding home health care problems at EURO 2021, focusing on the difficulties arriving in such routing problems with route inter-dependencies and the outcomes when applying the fix-and-optimize and the BRKGA algorithm in solving them;
- We submitted our results with the BRKGA-MP-IPR in March 2021, and we are still awaiting the response from the paper reviewers. We expect to complete the first round of revision by October-November.

In addition to these published works, we also have a work-in-progress with algorithms of 6 and our new instances of Section 9.3.

1.5 Thesis structure

The thesis is organized across 11 chapters and six appendices. Chapter 2 presents an overview of the literature regarding the home health care problem. Chapter 3 discusses the thorough planning of home health care systems through strategical, tactical, and operational planning. Chapter 4 deepens the literature on operational planning, and Chapter 5 defines the operational problem we study in this thesis. Chapters 6, 7, and 8 present the method we propose for obtaining high-quality lower and upper bounds. Chapter 9 presents the available datasets from the literature, and we discuss the characteristics we consider to devise our new realistic benchmark dataset. Chapter 10 presents the computational results for the proposed methods, and we conduct a systematic analysis against the best-known solutions from the literature. We also introduce in Chapter 10 the new instances that compose our dataset, and we discuss some details regarding the solutions produced by the most effective heuristic we proposed. We draw general conclusions in Chapter 11, and we highlight the undergoing studies and other future works.

2 LITERATURE OVERVIEW

At first glance, a newcomer researcher on home care systems may think the problem consists solely of some logistic problem. This chapter aims to illustrate the variety of decision problems arriving in the home care context and discuss home care as a way of providing services at the patient site. It also presents how the literature organized and discussed the planning level, holding significant similarities to, e.g., supply chain problem (MELO; NICKEL; SALDANHA-DA-GAMA, 2009; HULSHOF et al., 2012).

2.1 Home care as a way of providing on-site services

Home care was initially conceived as a model for providing healthcare services to people of some geographical region (FERNANDEZ et al., 1974). However, any service performed at the patient location can be operated in the home care fashion, similarly to a delivery service, but with all the "processing" done at the patient site. When targeting the aging population, a plethora of services can be offered in such home care mode, focusing on improving patients' quality of life, but it is also worth mentioning that such services can also benefit other individuals rather than only the elderly. For example, people with disabilities or some limitations can also benefit from such in-site services (BREDSTRÖM; RÖNNQVIST, 2008).

Concerning this thesis, we highlight two important materializations of home care services, and we focus on the one that better fits the necessities of the home care system of the *Melhor em casa* program. Carello and Lanzarone (2014) and Gomes and Ramos (2019) defined these two materializations of home care systems as home health care and home social care.

Home health care is the most intuitive case of home care services. In such systems, eligible patients receive medical treatment at their homes rather than hospitals (CHENG; RICH, 1998). This approach has motivation in the savings of hospital costs and positively impacts patient service level by reducing the stress regarding the displacement and stay at the hospital (LANDERS et al., 2016). Especially to pandemic scenarios like the current COVID-19 strike, such home health care services also help reduce the patient exposure to the Coronavirus, acting as a physical barrier to prevent contamination by pathogens (BASHIR; CHABROL; CAUX, 2012; LASFARGEAS; GAGNÉ; SIOUD, 2019). Despite that, it is essential to identify if the patient may suffer additional risk when moving to the

home health care system, e.g., by the initial screening at hospital identify that their health condition allows them to follow up their medical care at their home (HULSHOF et al., 2012; GRIECO; UTLEY; CROWE, 2020).

According to Gomes and Ramos (2019), home health care providers usually consider a subset of healthcare services offered in such home care modality–usually ones with great demands and low risk of performing away from the hospitals. This choice of healthcare services directly impacts the staffing of such a system, simply because the composition of the teams needs to consider the personnel qualification w.r.t. the healthcare services covered. We can say that a home health care (HHC) system considers a set of *heterogeneous caregivers*, in the sense that each caregiver qualifies to perform only a subset of the services offered by the provider (MANKOWSKA; MEISEL; BIERWIRTH, 2014). This feature also applies to several task workforce problems (CASTILLO-SALAZAR; LANDA-SILVA; QU, 2016).

Systems with single demands per patient and single qualification per caregiver allow modeling and solving each service type as independent problems (DE ANGELIS, 1998). Otherwise, if the patient requires multiple visits for distinct services, or the caregivers have more than one qualification or "degrees of qualification," the entire problem must be modeled and solved as a whole, considering all the service types offered by the HHC provider (TRAUTSAMWIESER; HIRSCH, 2011). Furthermore, if a patient can request more than one service type, typically, those multiple visits need to be made by distinct caregivers. Additional requirements such as simultaneous visits or some precedence rule may apply in those multiple visit cases (MANKOWSKA; MEISEL; BIERWIRTH, 2014; LIU; TAO; XIE, 2019).

Home social care (HSC) is the second important materialization of home care services, and focuses on assisting with day-to-day activities of their users such as bathing and house cleaning, and targets people with some lack of autonomy, but not necessarily sick or just elderly individuals (GOMES; RAMOS, 2019). Similar to HHC, the home social care provider may also restrict the service types offered, but differently than in HHC, the caregivers are usually *homogeneous*, meaning that they are qualified to perform any of the services offered by the provider (GERSHON et al., 2008). Often this means that one caregiver can perform a number of services requested by some patient, instead of the multiple visits approach of HHC. Nevertheless, there is still some cases where more than one caregiver are required to fulfill some visit, e.g., when a patient has overweight and can not be handled safely by just one carer (EVEBORN; FLISBERG; RÖNNQVIST, 2006).

The amount of work done by each caregiver is also very distinctive between HHC and HSC. As the services offered by the HSC provider span several everyday activities, each caregiver's amount of work can be very substantial, even when visiting a single patient. For this reason, providers often require periodic breaks to the caregivers after servicing some number of patients (LIU; YUAN; JIANG, 2017; GOMES; RAMOS, 2019). Furthermore, the number of persons requiring assistance is huge, and the HSC providers frequently have to manage some waiting list of individuals applying to home social care services (GOMES; RAMOS, 2019). Depending upon the provider's capacity, the same issue also arises in the HHC context (TRAUTSAMWIESER; HIRSCH, 2011).

Among other characteristics, the planning horizon of HSC services usually spans several days of operations and, in those cases, the provider had to consider the loyalty (or continuity of care) aspect of the patient scheduling since constantly changes the caregiver designed for some patients is in the short and medium-term, disruptive regarding the service levels (CARELLO; LANZARONE, 2014; GOMES; RAMOS, 2019). This feature also arises in medium and long-term HHC contexts (NICKEL; SCHRÖDER; STEEG, 2012).

As one general conclusion, the availability of home social and health care services opens the possibility of older people staying at their homes instead of moving to nursing homes (HULSHOF et al., 2012). Naturally, the optimization opportunities arriving on combining both HHC and HSC into a single optimization problem are valuable (GOMES; RAMOS, 2019). From a computational perspective, both problems comprise some routing and scheduling problems, thus sharing some optimization aspects. Nonetheless, real HHC and HSC differ substantially depending on the provider's objective and working regulations, thus requiring tailed models and solution methods (CISSÉ et al., 2017; FIKAR; HIRSCH, 2017; GRIECO; UTLEY; CROWE, 2020).

2.2 Overview of the planning process of home care problems

The home health care system's design and management is a complex task that comprises several planning phases, each one taking responsibility for some of the aspects of implementing such services, tackling both decisions regarding the implementation of the patient care as well as organizational decisions (AIANE; EL-AMRAOUI; MESGHOUNI, 2016). At an initial scanning, one may think that the HHC literature solely comprises rich variants of some classical vehicle routing problems (GRIECO; UTLEY; CROWE,

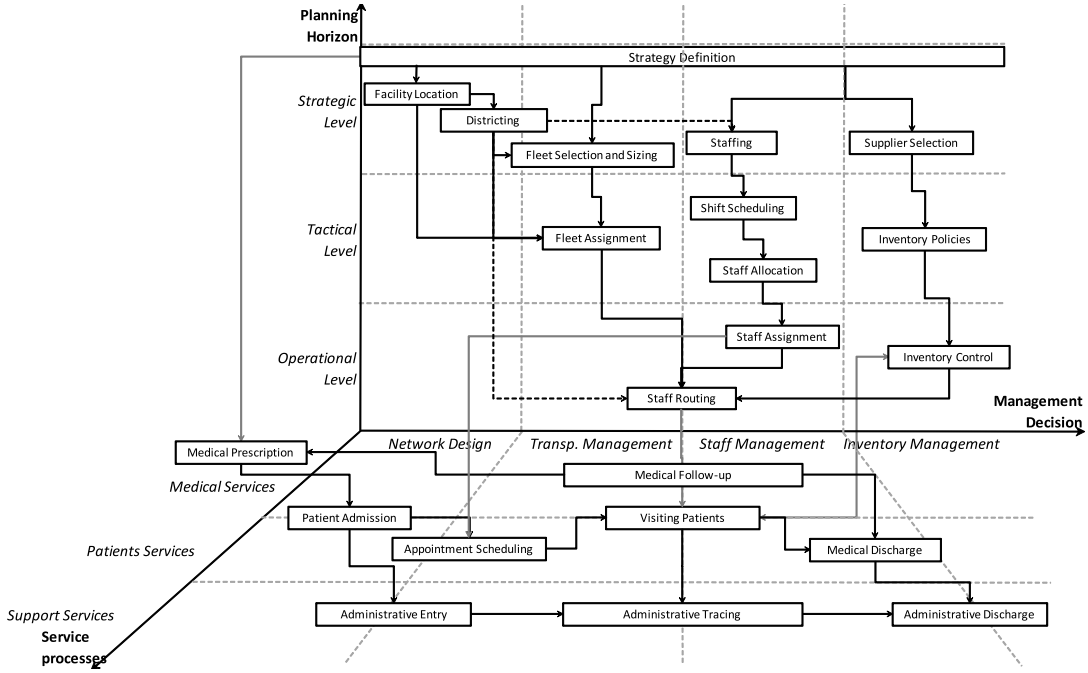
2020), but planning managing HHC has several other steps prior the operational planning. As pointed by Bashir, Chabrol and Caux (2012), the planning of HHC comprises at least three decision levels. For example, the HHC providers need first to define the covered region to place the central office–or depot. Such a decision must take care, for example, of the city's topology and the demographical distribution of the population (BENZARTI; SAHIN; DALLERY, 2013). After that, a staff management step must be taken to manage how the caregivers are allocated to each district, considering the types of service covered and estimating the number of patients to be served (BORSANI et al., 2006). With all these–and several others–distinct problems arousing into HHC, some authors propose discussing home health care as a framework to better embarking all the related problems (GUTIÉRREZ; VIDAL, 2013; HULSHOF et al., 2012; GRIECO; UTLEY; CROWE, 2020).

As it resembles a lot of the supply chain from management science and operations research, most taxonomy and in-depth analysis publications discuss the HHC in strategic, tactical, and operational levels (BALLOU; SRIVASTAVA, 2007; BASHIR; CHABROL; CAUX, 2012; GRIECO; UTLEY; CROWE, 2020). Such a multi-level approach establishes a coherent pipeline to place all the decision problems arousing on HHC. Strategic planning tackles long-term decisions that must hold for long periods due to the costs relative to change. Similarly, tactical and operational levels tackle middle-time and short-time decisions, ranging from some months to a few days or weeks, respectively. Note that the decision made in one planning level impacts the others. This way, poor choices in one planning level often can not be circumvented on the others (GUTIÉRREZ; VIDAL, 2013). Besides that, one could give valuable feedback to the other to further optimize the HHC services over time. In contrast, most authors disregard the tactical and strategic planning phases, and most of the effort is put into the operational planning of HHC (GUTIÉRREZ; VIDAL, 2008; GRIECO; UTLEY; CROWE, 2020).

Due to the number of decision arriving on HHC, the relation between the decision levels can be somewhat hard to understand. For this reason, several authors structure the HHC through a pipeline or flowcharts to better understand and to highlight the interaction across the decision problems arriving in the home care context. For example, Gutiérrez and Vidal (2013) proposed a pipeline that distributes and connects such decisions according to the planning horizon, the role of the management decision, and the role of the service process, as depicted in Figure 2.1. For example, the strategic decisions of *districting* and *facility location* (placement of the operation center) relates to management decisions on the

network design, and has an impact on the *patient admission*, as indicated by the gray dashed lines. Grieco, Utley and Crowe (2020) used a similar approach to define a flowchart of input data, decisions, and output data regarding the HHC framework. They also highlighted the possibilities of combining or rearranging the decisions from distinct planning levels to better cope with the demands and other third-party services, e.g., informal care (GOMES; RAMOS, 2019).

Figure 2.1 – Summary of planning levels and activities on home health care context.



Source: Gutiérrez and Vidal (2013).

3 MULTI-STAGE PLANNING OF HOME HEALTH CARE SERVICES

This chapter discusses some of the decision problems arriving in each of the three planning levels of a home health care system, highlighting how they impact the daily operation of a home health care provider. Section 3.1 discusses long-term decisions that involve high costs, e.g., the operations center placement. Section 3.2 discusses medium-term decisions that could be reworked from time to time, e.g., the hiring of new healthcare professionals. Section 3.3 focuses on short-term decisions, e.g., the scheduling of the visits to the patients.

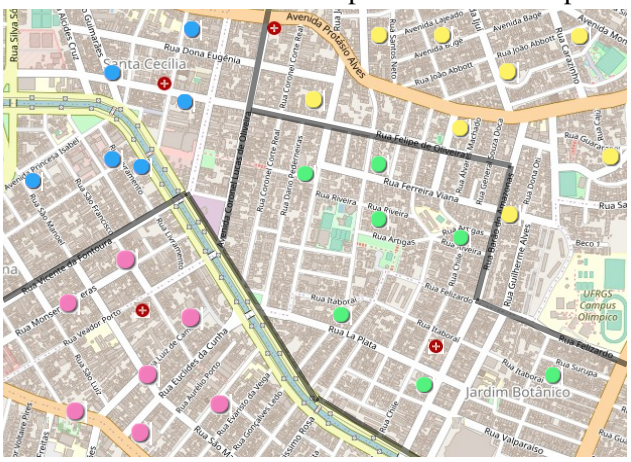
3.1 Strategical planning and long-term decisions

Strategic decisions usually endure for long periods due to the costs related to changing those decisions. A typical example is the placement of the operations center–or depot. Before deciding upon its location, the providers typically partition the covered geographical region into small and more manageable districts (HULSHOF et al., 2012). Some benefits include an increased level of service due to shorter response time, thanks to the restrainment of travel times (LAHRICHI et al., 2006; BENZARTI; SAHIN; DALLERY, 2013). According to Benzarti, Sahin and Dallery (2013), districting the covered region also helps balance the districts' workload. It increases the bonding between patients and the staff due to the small number of caregivers per district. Those secondary benefits also impact the level of service to both patients and professionals. Figure 3.1 depicts an example of a districting problem applied to some boroughs of Porto Alegre, Brazil.

The granularity of districting also helps handle the subsequent planning steps of HHC, but having too many districts can incur higher costs due to the placement of several operation centers and the larger number of vehicles required (HULSHOF et al., 2012). In contrast, having fewer districts may compromise patients' attendance with a more constrained visiting time-window, making the planning of the operational level challenging (BREDSTRÖM; RÖNNQVIST, 2008). This way, the home health care context's districting problem must consider such trade-offs.

The districting problem in the HHC context has its own set of requirements, and several of them are more similar to the police district design problems than other publications in the literature approaching the placement of hospitals and other healthcare centers (BLAIS; LAPIERRE; LAPORTE, 2003). While a hospital's placement has the

Figure 3.1 – Example of districting for a region of Porto Alegre. The black lines delimit each district, and the red markers indicate the placement of the operation centers.



Source: the author.

accessibility requirements as the most important characteristic, the district in HHC must first obey the boundaries between boroughs, thus avoiding congested accesses inside a single district. Additionally, Blais, Lapierre and Laporte (2003) highlight the importance of being diligent in keeping connected to all locations inside a district while observing that no district should be enclaved inside another. The authors also argue that it is better to have a combination of low and high-density districts than having several districts with medium density since the total travel time and amount of demands are not necessarily proportional to the district size. Other authors consider lower and upper bounds in the number of operation centers installed, which also set bounds to the number of districts to generate (PEZZELLA; BONANNO; NICOLETTI, 1981; DE ANGELIS, 1998). Others advise a two-step approach in which the districting is decided first, followed by the placement of operation centers (HULSHOF et al., 2012). Furthermore, fewer works discuss the facility location applied to the HHC than the traditional healthcare systems (GUTIÉRREZ; VIDAL, 2013; GRIECO; UTLEY; CROWE, 2020).

The placement of the operation centers is often guided by the geographical structures available in each district, considering, for example, the minimization of a metric of average travel distance between all accessible locations within a district (DE ANGELIS, 1998). Another metric can approach the district's populational density to enhance the operation center's placement (HULSHOF et al., 2012). Blais, Lapierre and Laporte (2003) approach such metrics to solve the combined districting and facility location problems. Computational results indicated that a tabu search meta-heuristic provided improved solutions compared to the manual ones by 17%, considering one operation year of a local community health clinic in Montreal. Additionally, the standard deviation between

the number of patients per district also benefits from such an optimized approach, and consequently, a better work balance in the optimized solutions than in the manual ones.

In addition to the framework proposed by Gutiérrez and Vidal (2013), some authors discuss the importance of selecting the covered health service types before all the other planning, because such decision directly impacts the volume of the patients expected, in the number of districts, and capacity of central offices regarding the staff size (HULSHOF et al., 2012; GRIECO; UTLEY; CROWE, 2020). The offered service types also impact the medical supply needed to operate the services. Furthermore, the eligibility to move a patient from traditional healthcare services to a home care modality is of concern to the HHC provider (GRIECO; UTLEY; CROWE, 2020). One usual policy is to keep the number of patients served to enforce balance and prevent overutilization of the staff (DE ANGELIS, 1998). In other cases, the eligibility of a new patient is subject to the feasibility of serving their needs, for example, in the case of a patient that needs short self-life drugs for cancer treatment (CHAHED et al., 2009). In some cases, offering partial care in a hospital combined with partial care at home can be interesting (HULSHOF et al., 2012). If necessary, hiring temporary personnel can be a low-cost alternative to cope with unexpected changes in the service requests or cases of staff shortages or emergencies (HULSHOF et al., 2012; GUTIÉRREZ; VIDAL, 2013).

The literature shows that frequently the HHC is subordinated to the traditional health system. Due to this close relationship with the traditional health care institutions, frequently, the decisions regarding strategic-level HHC have already been taken by others. Thus, most HHC publications consider a fixed districting, and the placement of operation centers is often set in fixed locations–sometimes within a hospital. Similarly, often the hospital shared its staff with the HHC teams, thus dismissing hiring and other planning steps (GUTIÉRREZ; VIDAL, 2013). These facts justify the few publications about the strategic planning level of home health care services and the larger number of publications on tactical and operational levels (DASKIN; DEAN, 2005).

3.2 Tactical planning and medium-term decisions

According to the propositions of Bashir, Chabrol and Caux (2012) and Gutiérrez and Vidal (2013), the next tier of HHC management consists of tactical planning activities regarding staff employment, according to the decisions taken on the strategical planning. The tactical planning objective meets the required staff qualification levels and handles

patient admission control and inventory bookkeeping. In general, the home health care provider decides about the staff size by observing the estimative demand for the service types offered. The staff is then scheduled in working shifts according to the regulations, then allocating the scheduled staff to the districts, and consequently, to the patients (ERNST et al., 2004).

Similar to strategical planning, few publications on staff scheduling and staff allocation are in the HHC literature. Despite that, the few authors discussing these decisions approach the underlying problem of balancing the personnel's workload while maximizing the number of patients served along with the time and reducing the patients' waiting times (DE ANGELIS, 1998). As the latter is subject to demand fluctuations, some authors suggest the estimation of team sizes based on the current demands, plus the district demographical data like age and other epidemiological markers (LANZARONE; MATTA; SAHIN, 2012). Others suggest accounting of how frequent the visits to some types of service types. Thus some medical specialties may have more caregivers than others (HULSHOF et al., 2012; GRIECO; UTLEY; CROWE, 2020). Another important aspect of tactical planning to keep the staff members' workload and the service levels is the patient admission policies. Bad decisions about this aspect can strongly affect the perceived service levels for both the staff and the patients (BORSANI et al., 2006).

Staff mobility is also taken into account during tactical planning. In most cases, each staff member has a dedicated vehicle, and the fleet is usually homogeneous (FIKAR; HIRSCH, 2017; CISSÉ et al., 2017). Some other authors consider other traveling modes that match the districts' populational density and the circulation structures, such as public transportation and the availability of walking and cycling areas (FIKAR; HIRSCH, 2015; CASTILLO-SALAZAR; LANDA-SILVA; QU, 2016). Using different transportation modal may lead to savings in the long term at the expanse of longer travel times and caregiver capacity to carry equipment and medical supplies.

More recently, Fikar and Hirsch (2018) discussed the benefits of having a sharing scheme to cope with difficulties in high urbanized environments, such as finding parking locations near-patient homes. The authors compare the standard transportation approach (one vehicle for each staff member), car-sharing, and trip-sharing. On trip-sharing, the author also studied the effects of dismissing some trips by allowing short walks from one patient to another when these are close enough. The computational results indicated that both car and trip sharing approaches reduced the number of vehicles needed to operate the services. However, there is a tradeoff on the travel times observed. Trip sharing leads

to almost the double travel times of the other ones while achieving reductions up to 88% in the number of vehicles utilized. On the other hand, car-sharing reduced the number of vehicles up to 42% more than the standard approach but kept comparable travel times.

3.3 Operational planning and short-term decisions

Due to the planning frequency and associated costs, staff routing and scheduling problems are the most expensive operational-level planning problems on HHC (FIKAR; HIRSCH, 2017; CISSÉ et al., 2017). Additionally, operational planning can dictate the assessment and intake of new patients and the medical supply and equipment (CHAHED et al., 2009). Most of the literature on operational planning of HHC approaches a single district's routing problem, with all vehicles, staff members, and patients' demands are known a priori (EVEBORN; FLISBERG; RÖNNQVIST, 2006; CISSÉ et al., 2017). Furthermore, most authors propose solving both the staff assignment and routing together: with each caregiver route set, the visit schedule is established according to caregivers' arrival time on each patient of their route. For this reason, some authors refer to such combined problems as the Home Health Care Routing and Scheduling Problem–or HHCRSP for short (MANKOWSKA; MEISEL; BIERWIRTH, 2014; GRENOUILLEAU et al., 2019).

The simplest variants of the HHCRSP disregard any additional information except the availability of patients through time-windows. E.g., the paper of Cheng and Rich (1998) is considered the first publication of such an approach (CISSÉ et al., 2017). The authors consider a staff composed of full-time and part-time nurses, the time-window for each patient, and some other side constraints regarding the nurses' working time. The compatibility between the staff qualification levels and patient service type requests is modeled implicitly. All patients must be visited within their time-windows, to minimize the overtime of full-time nurses and the number of working part-time nurses. The objective function also considers the equity of working time for full-time nurses. Such concerns are further approached in more recent publications, e.g. Eveborn, Flisberg and Rönnqvist (2006), Trautsamwieser and Hirsch (2011).

Cheng and Rich (1998) proposed a mixed-integer program and a standard two-phase construct-then-improve meta-heuristic. The authors solved instances of up to four nurses and ten patients using the CPLEX solver. The heuristic was tested on larger problems, consisting of up to 900 patients and 294 nurses. Besides that, the authors

focused on discussing results around the smaller instances to which CPLEX could find optimal solutions.

After the work of Cheng and Rich (1998), other papers tackled the HHCRSP by considering several other realistic characteristics. Additionally, the planning horizon has become a significant criterion for distinguishing the publications. For example, there are solution methods for problems with specific planning horizons, ranging from a single day to several weeks, to name a few (CISSÉ et al., 2017; FIKAR; HIRSCH, 2017). The following sections discuss the most frequent features arriving on HHRSP. Furthermore, other related operational level decisions arriving on HHC require discussion, e.g., coordinating other home care services such as home social care (GRIECO; UTLEY; CROWE, 2020). Additionally, the operational level often needs to cope with last-minute changes in the planning, e.g., due to an unpredictable change in the visit duration or traffic congestion (TRAUTSAMWIESER; GRONALT; HIRSCH, 2011; SHI; BOUDOUH; GRUNDER, 2019). Nickel, Schröder and Steeg (2012) discuss this subject through a "master schedule," where the last-minute operational changes are incorporated as the planning needs adaptation.

3.4 Integrated planning

As the reader may notice, most literature focuses on single problems arriving in the HHC context. From our perspective, the researchers try to isolate the problem of interest to limit its complexity, so they can focus on "solving one thing and solve it well." Despite that, published results from other research areas indicate significant improvements when solving such integrated problems, e.g., solving vehicle scheduling problems integrated with crew scheduling problems for public transportation systems (MESQUITA; PAIAS, 2008), and airline planning (PAPADAKOS, 2009).

To the best of our knowledge, virtually all the works from the operational planning of HHC already approach an integrated problem that solves both the patient scheduling and the caregiver routing altogether (BERTELS; FAHLE, 2006). For example, Bredström and Rönnqvist (2008) studied a problem that the patient scheduling comprises additional constraints to model the patient-caregiver assignment preferences. Furthermore, the routing of caregivers already defines the visit time of the scheduled patient. Mankowska, Meisel and Bierwirth (2014) studied an even more integrated problem, in which patient scheduling is tied to the routing decision by a set of route inter-dependency constraints.

According to Fikar and Hirsch (2017), some authors also study problems considering several transportation choices for the caregivers, such as public transportation, by bicycle or even walking, in the cases where the travel times are short enough. More recently, Fikar and Hirsch (2018) proposed solution methods for an integrated problem that combines the caregiver routing, the patient scheduling–both from the operational planning level–and the vehicle assignment for the caregivers–from the tactical planning level (GUTIÉRREZ; VIDAL, 2008; GRIECO; UTLEY; CROWE, 2020). According to their text, such work was motivated by the bad traffic conditions of densely populated regions and the difficulty of finding parking locations near the patient's homes. They propose three variations of the problem, and their results indicated that vehicle sharing approaches could significantly reduce the number of vehicles required to operate the system compared with the traditional approach of one vehicle per caregiver.

Even the solution methods designed to solve operational-level problems, such as the HHCRSP, can be employed in planning other problems, even from other planning levels. For example, the HHCRSP can be used to study the scenarios in which the provider hires additional professionals, allowing the analysis of how these simulated scenarios can impact, e.g., routing costs and the service levels. Others can study "stress" scenarios with a subtle increase of some service types or simulate cases where the visit duration exceeds the originally planned time. One can refer to this approach as a kind of *manual integration* approach, and this study of scenarios can motivate a change of other decisions made earlier, e.g., in tactical and strategical planning levels.

4 RELATED WORKS

This chapter presents some of the literature results on the operational planning of a HHC system. Section 4.1 briefly discusses the differences in HHC systems according to the planning horizon considered. Section 4.2 introduces several works from the literature approaching the single-day problem, describing details about the time windows constraints typically considered (Subsection 4.2.1) and the so-called route inter-dependency constraints approached by several authors (Subsection 4.2.2).

4.1 Planning horizon of home care problems

The literature of operational planning in HHC problems is rich and contains diverse optimization models based on mixed-integer linear programming (MIP) models, heuristics and hybrid solution methods (GRIECO; UTLEY; CROWE, 2020). Many publications consider solving the single-day (also called short-term, or single-period) HHCRSP, which resembles the traditional VRPTW (AKJIRATIKARL; YENRADEE; DRAKE, 2007; FIKAR; HIRSCH, 2017). Other authors solve several days of operation (long-term HHCRSP), consisting of attending to all the patient demands according to the patients' home health care plan (GRENOUILLEAU et al., 2019; GRIECO; UTLEY; CROWE, 2020).

The solution methods for short-term, medium-term, and long-term HHC problems vary significantly (GRIECO; UTLEY; CROWE, 2020). Due to the nature of the HHC services, often the provider prescribes periodic visits in the patient home care plan, which need to be fulfilled within a medium-term, like a week or so, so it is becoming more common to find publications approaching longer planning HHC problems (NICKEL; SCHRÖDER; STEEG, 2012; GRIECO; UTLEY; CROWE, 2020). Furthermore, the in-advance planning of medium-term and long-term HHC allows the provider to communicate the visit plan to the patient several days beforehand. A larger planning horizon also has an advantage w.r.t. the equity on the amount of work among the caregivers and the planning of allocation of medical equipment, but all these advantages came at the expanse of solving larger problems. Additionally, additional constraints and objectives arrive in such problems, e.g., the continuity of care, that enforces the assignment of the same caregivers to the same patients over the plan and increases the patient-caregiver bonding, which positively impacts the service level for the patients (FIKAR; HIRSCH, 2017).

Despite being a less common instance, the single-day HHCRSP still has its value. In some sense, such a problem can be seen as a subproblem of the multi-period home health care problem. To better understand its importance, consider solving the multi-period problem using a two approach, in which the first step consists of choosing the patients of the day. With that, the second step consists of solving independent single-period HHCRSPs. For this reason, this thesis focuses on solving these daily problems since the proposed solution methods can be adapted back to other problems comprising several days of planning (LASFARGEAS; GAGNÉ; SIOUD, 2019). Furthermore, the rest of this chapter focuses on discussing related problems to the single-day HHC, highlighting some of the important works from the literature.

4.2 Single-day home health care problem

For historical reasons, Fernandez et al. (1974) is often referred to as the first work discussing a single-day home health care problem (FIKAR; HIRSCH, 2017). Classical optimization algorithms such as simplex were not widespread, so the standard solution strategies for these problems, at the date, were primarily based on analytical and statistical methods. Thus, some other authors often referred to Cheng and Rich (1998) as a starting point of the research in HHC, frequently because they were the first authors to proposed standard MIP-based solution methods for such a problem. Their work modeled the problem as a multi-depot vehicle routing problem with time windows, minimizing domain-specific quality metrics such as overtime and working hours of part-time nurses. In addition to the MIP model, the authors also introduced a two-phase heuristic that builds an initial solution that is further optimized in an improvement phase. Both model and heuristic were tested against small and large random instances, considering 294 nurses and 900 patients. Due to difficulties in applying the MIP model, the authors could only compare the exact and the heuristic solution methods small test cases comprising four nurses and ten patients.

Akjiratikarl, Yenradee and Drake (2007) were the first to establish the relationship between the home care problem and the Vehicle Routing Problem with Time Windows, defining the former as an extension of the VRPTW. Using a two-phase particle swarm meta-heuristic (EBERHART; KENNEDY, 1995), the authors reported savings of up to 30% of the caregivers' travel distances compared to the manual solutions used in practice by a UK consortium. Their dataset consists of five real instances from five days of operation of the consortium. Each instance has 12 caregivers to serve 100 requests by 50 clients.

Bredstrom and Ronnqvist (2007) proposed a branch-and-price algorithm (BCP) for the *combined VRPTW with synchronization constraints*. The authors discussed the application of such an algorithm to solve a home health care problem with multiple visits in some patients, modeled as simultaneous attendances. The authors disregarded multiple service types and motivated the application on cases in which a single carer can not handle the patient alone, e.g., in an overweight patient. They considered a set-partitioning master problem with additional constraints for the operations synchronization. Patients' time-windows were taken into consideration only in the pricing problem. The authors proposed three branching rules to exploits the solution structure implied by the synchronized visits. They tested the algorithm against test cases comprising 20 patients and four vehicles, up to 80 patients and 16 vehicles. The authors also studied the effects of three TW length variations in the convergence of the solution method. Such instances were generated according to real data, considering 10% of multiple-visit nodes. The algorithm was implemented AMPL, and they solved both master and pricing problems with CPLEX 10.0 on an Intel Xeon 2.67 GHz computer with 2 GB of memory. 44 out of all 60 instances were solved to optimality within 1 hour of runtime, but the authors report a significant overhead on the running time due to implementation language choice.

Bredström and Rönnqvist (2008) extended the previous work from 2007 by including precedence constraints in the problem, altogether with the simultaneous attendance constraints from their previous work. Due to the unavailability of a benchmark dataset comprising all the features of the problem they wanted to solve, they approached the same instance dataset from their previous work, augmented with test cases comprising fixed visit times and absent time-window cases. CPLEX was once again used to solve the problem. From the 80 instances tested, the solver found a feasible solution to 75 and proved the optimality of 33 within a time limit of one hour. To solve the test cases in shorter computational times, the authors also proposed a MIP-based heuristic that starts from the linear programming relaxation of the problem, which iteratively adds some of the integrality constraints of the decision variables and tries to find a feasible solution applying CPLEX in the reduced problem. The authors reported computational results considering three distinct objective functions: the first minimizes the travel time of the caregivers; the second maximizes the preferences of caregivers and patients; And the third objective function minimizes the maximum difference of workload between all the caregivers, measured in terms of service duration of the patients associated to each caregiver.

Dohn, Kolind and Clausen (2009) approached a similar problem to HHCRSP called manpower allocation problem with job-teaming constraints modeled as synchronization constraints. The authors proposed a branch-and-price-and-cut (BCP) algorithm to solve the problem. Clients' time-window were considered on the pricing, and synchronization constraints were treated implicitly during the branching phase. The authors implemented their algorithm using the *COIN-OR BCP* library, and the tests were run on a 2.70 GHz AMD machine with 2 GB of memory. The authors generated instances with 12 teams and 120 tasks up to 20 teams and 300 tasks, based on the real data from the daily operation of two large European airports. Six out of 12 instances were solved to optimality in one hour of processing. The other instances either reached the time limit of ten hours of processing or run out-of-memory due to the branching tree's size. Additionally, the authors highlighted the importance of using initial feasible solutions to speedup the algorithm from hours to few seconds in some tested instances compared with the "standard approach" of starting the column generation from a high-cost artificial solution.

Kergosien, Lenté and Billaut (2009) proposed a MIP for a variant of HHCRSP in which caregivers depart and return to their own homes at the end of operations instead of starting and finishing their routes at the operations center. In their version of the problem, each caregiver has a maximum working time, and the authors considered simultaneous attendance of patients involving the synchronization of up to three distinct caregivers. The authors also considered the case of multiple visits with non-overlapping visits, and additional cuts were proposed to exploit these operation synchronization requirements. The model was tested on artificially generated instances from 10, 20, 30, and 40 service requests. Two hundred instances were generated for each instance size, and all instances with ten requests were solved to optimality. Approximately 75% of instances with 20 requests were also solved to optimality. The authors reported feasible solutions to all the other instances tested, with optimality gaps ranging from 7.0% up to 18.2% within a time limit of 10 minutes of processing. The model produced solutions with a smaller gap of 5.8% to 16.1% with these additional cuts. Besides that, their largest test cases were considerably small compared to other datasets from the literature, so further experiments were still needed to check the effectiveness of the proposed cuts in solving such larger benchmark datasets.

Trautsamwieser and Hirsch (2011) approached a very realistic variation of the HHCRSP that considers several new requirements and preferences settings regarding service types and caregiver-patient compatibility, and several practical soft constraints and

regulatory rules. The authors tested both mixed-integer programming (MIP) and variable neighborhood search (VNS) based meta-heuristic for solving the problem (MLADENOVIĆ; HANSEN, 1997). Due to prohibitive computational resources requirements such as processing time and memory consumption, the MIP model was only applied to the small instances of up to 20 patient types and four caregivers. On larger test cases, the authors produced an initial solution through a constructive heuristic, further improved by the VNS meta-heuristic. The authors successfully solved large instances of up to 420 clients and 75 nurses, comprising 512 tasks. It is worth mentioning that the meta-heuristic successfully found the optimal solutions as the MIP, w.r.t. the test cases where the exact solver was successful.

Trautsamwieser, Gronalt and Hirsch (2011) further extended their previous work to solve the HHCRSP in the presence of natural disasters. Such variation includes the problem with multiple depots, which are represented by caregiver homes. More precisely, caregivers can start and finish their routes either in a central office or in their respective homes. Additionally, some caregivers operate in the open route fashion; thus, it is not required to consider travel times at the start and the end of some routes. All the constraints from the previous work apply. The authors generated 40 artificial instances with 30 jobs and six nurses, up to 100 jobs, and 20 nurses. XPRESS solver was capable of solving to optimality the smaller instances within 12 hours of processing. They also tested with real instances from one urban and two rural regions of Austria, comprising up to 411 patients (512 jobs) and 75 nurses. A VNS meta-heuristic was employed when solving such larger test cases, requiring up to 68 minutes of processing to finish the search on the largest instances. Motivated by the flood of several European countries in 2002, the authors tested their VNS against instances that simulated the disaster scenarios in which some streets got congested as a consequence of the destruction of some others. These tests were conducted using real data from 2002, considering one urban and two rural Austria regions. On the largest test case in a rural area, with 504 service requests of 72 nurses, the algorithm finished after 26 minutes. Conversely, the VNS took almost 66 minutes while solving the urban region's case with 140 service requests and 13 nurses. The authors justified this increase in processing time due to the large solution space induced by the larger number of links between patients in the urban center.

Rasmussen et al. (2012) focused on operations synchronization arising in HHC problems, as we further discuss in Section 4.2.2. The authors proposed a BCP algorithm with the operations synchronization constraints explicitly modeled on the master problem.

Branching rules based on the patient time-window, synchronization constraints, and the integer variables are proposed, inspired by works from the VRPTW with transshipment (DREXL, 2012). The authors used the COIN-OR BCP library to solve instances of up to 15 caregivers and 150 patients with the branch-and-price algorithm. All tests were run on 2.2 GHz processors, and a time limit of 1 hour was set. The branch-and-price algorithm found optimal solutions for 19 out of 30 instances tested.

Mankowska, Meisel and Bierwirth (2014) approached a problem that considers a subset of the operation synchronization constraints discussed in Rasmussen et al. (2012). Among the other differences between the two works, the later study modeled the visits as a hard constraints, and proposed soft time-windows for the patient visits to circumvent feasibility issues. Additionally, the authors considered caregivers qualification–or skills–explicitly. With regard to the visits, a patient can request either one or two visits demanding distinct service types. To the case of multiple demands, the visits must obey the either a precedence constraints or simultaneous attendance constraints, as proposed by Rasmussen et al. (2012). The authors solved the problem using both the CPLEX solver, as well as an adaptive variable neighborhood search (AVNS) based meta-heuristic. On the adaptive phase, the meta-heuristic tests all local-search operators the authors proposed and sort them according their effectiveness on improving the solution quality. Furthermore, the objective function of the problem considers three components: the first minimizes the caregivers travel times; the second minimizes the total tardiness w.r.t. violations of the soft time-windows; and the third component minimizes the largest tardiness among all the visits. According their computational results, CPLEX ended up solving only the small instances of the problem, with up to 10 patients and three caregivers. To most of the larger test cases, the solver was able to provide lower bounds solely, even with the generous time limit of ten hours of processing per instance. The AVNS found solutions to instances with 25 patients and five caregivers, up to 300 patients, and 40 caregivers within up to 2 hours of processing. The authors also published their benchmark dataset on the internet, enabling other researchers to compare their solutions against a common dataset.

Decerle et al. (2018) considered an HHCRSP with only simultaneous attendances constraints for patients requesting multiple visits. Unlike the previous works, the authors approached these synchronization requirements as soft constraints and minimized how much they violated these requirements. Like Mankowska, Meisel and Bierwirth (2014), all patients must be serviced, and soft patients time-window were employed to overcome feasibility issues. Besides that, the authors also limited how much a patient time window

can be violated. Additionally, the problem considered the assignment of caregivers to depots, which combines the operational HHCRSP with the staff allocation that is usually taken during the tactical planning of HHC. The authors proposed a MIP model and a memetic algorithm to solve the instances introduced by Bredström and Rönnqvist (2008), ranging from test cases with 20 patients and four caregivers up to 80 patients and 16 caregivers. In addition to the traditional one- and two-point genetic algorithm crossover operators, the authors also proposed tailored ones to the synchronization requirements on the multiple visits. The algorithm runtime was set to fixed values, consuming up to two minutes to solve the smaller instances and 10 minutes for the larger ones. Furthermore, GUROBI solver was employed to benchmark the proposed model, allowing the solver to run up to one hour. According to their computational results, the solver found optimal solutions only to 13 instances of the Bredström and Rönnqvist (2008) dataset. The authors also proposed a new benchmark dataset comprising of larger test cases up to 103 patients and six caregivers. This new dataset also features the problem with multiple operation centers–or depots. As one may expect, the MIP solver only produced an optimal solution to the smallest test cases of the new benchmark dataset. However, surprisingly, it could produce feasible solutions even on the larger test cases–an unexpected achievement when considering the previous results from the literature on similar problems.

Lasfargeas, Gagné and Sioud (2019) also proposed a VNS meta-heuristic for the so-called medium-term HHCP, drawing solution method for the problem within up to a week of planning. In the studied problem, all patient demands are known in advance and must be fulfilled during the planned period. This problem also considers the compatibility between caregivers and patients. Thus some assignments are not allowed, similarly to the previous works approaching the problem with skilled caregivers. The problem allowed multiple visits by distinct caregivers to cope with inter-dependencies between the services, such as precedence constraints and simultaneous visits. The authors also considered a few additional working regulations arriving on such medium-term HHCP, such as the additional constraint that imposes a break in the caregiver schedule to force a resting period after a certain number of worked hours. The study conducted by the authors was two-fold. The first study focused on the different solutions methods consisting of simple variations of a constructive heuristic. The second study assessed the performance of a VNS based meta-heuristic introduced by the authors, which comprises new stochastic neighborhood operators to overcome convergence issues when facing strong local optima attractors. Due to the lack of a benchmark dataset for the medium-term HHCP, they tested their algorithms

over the benchmark dataset proposed by Mankowska, Meisel and Bierwirth (2014). The VNS found improved solutions over the previous best-known solutions from the literature. However, only half of the benchmark dataset was used, and the authors produced these new solutions to instances up to 75 patients and 15 caregivers. On the larger test cases, the authors argued difficulty finding feasible solutions, indicating a failure of their constructive heuristics.

As observed by Fikar and Hirsch (2017), and more recently in Grenouilleau et al. (2019) and Grieco, Utley and Crowe (2020), there is still a missing definition of what objective function and contraints compose a "standard" HHC problems. Furthermore, most authors approach a problem comprising the caregivers' routing and benchmark their solution methods against some dataset from the literature. Due to the features considered in their works, the authors frequently introduce new mathematical models to describe their problem of interest. In few cases, the solution methods are benchmarked against some augmented literature dataset, otherwise a new dataset is introduced. Some authors make these datasets available upon reaching out to the corresponding authors of such works, as we mention in Section 9. In other cases, the datasets are openly available on the internet (MANKOWSKA; MEISEL; BIERWIRTH, 2014; DECERLE et al., 2018; GRENOUILLEAU et al., 2019). Besides the myriad features, most problems consider time windows for visiting the patients, and additional constraints are considered while scheduling patients with multiple visits. For this reason, the following sections of this thesis focus on discussing these features arriving on the HHC context.

4.2.1 Time-window modeling on HHCRSP

Consider the span defined by the interval $[e, l]$. These values e and l indicate the earliest and tardiest arrival times, respectively, in which a caregiver can visit a given patient–in other words, this is the patient time-window (TW). As to the VRPTW, the presence of TW makes finding feasible solutions to HHCRSP potentially tricky (BRÄYSY; GENDREAU, 2005), especially when such requirements are modeled as hard constraints. For this reason, some authors in the literature still approach the single-day HHCRSP with a hard time window, but frequently they allow some patients to remain unscheduled in the optimal solution (RASMUSSEN et al., 2012). In those cases, typically, the objective function has a component that maximizes the number of patients visited (DOHN; KOLIND; CLAUSEN, 2009; RASMUSSEN et al., 2012).

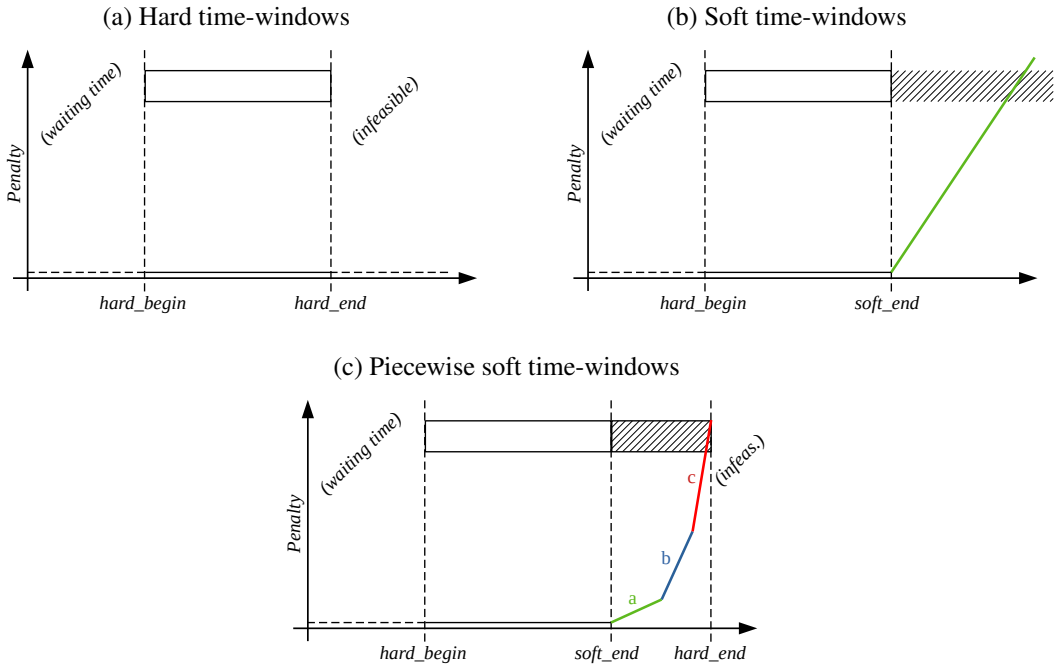
Others guarantee the scheduling of all patients at the expense of incurring additional costs in the solution (BRAEKERS et al., 2016; DECERLE et al., 2018). In most cases, the TW violation is penalized linearly according to the amount of tardiness on the visits (BRAEKERS et al., 2016). To limit how much such violations can degrade the service level, some authors model the penalization differently, e.g., Decerle et al. (2018) defined a piecewise function that increases the solution cost by a little in the presence of relatively small amounts of tardiness. Conversely, large amounts of tardiness are subject to larger penalties. Another interesting characteristic approached by Decerle et al. (2018) is to consider a limit of how much the TWs can be violated, which makes some solutions infeasible but prevents sacrificing the service level of some patients to benefit the others.

Degrading too much the service levels was also of concern to Mankowska, Meisel and Bierwirth (2014). In their work, time windows are modeled as soft constraints so that a solution may include tardiness in the visits. In addition to these soft constraints, the authors also propose a makespan component to the objective function to minimize the largest tardiness over all the patients scheduled. This approach is simpler to model than the piecewise function proposed by Decerle et al. (2018). Their computational results show evidences that both the soft TW constraints and the makespan constraints tend to work well together, leading to high-quality solutions that, in the worst case, have minimal total tardiness and also keeps the tardiest visit under small amounts. Despite that, the authors offer no mechanism to limit how much the time windows can be violated.

Figure 4.1 summarizes the three most common modeling approaches of patient time windows. Note that all them consider a hard beginning of the time windows, indicated by the *hard_begin* on the horizontal axis of the figure. If a caregiver arrives before such time, they need to wait until the patient becomes available. Figure 4.1a depicts a hard TW approach, where the rectangle between *hard_begin* and *hard_end* defines the time frame in which the patient is available. Figure 4.1b illustrates the case of soft time windows. In this case, a visit starting before *soft_end* incurs additional costs, as depicted by the line in green. Figure 4.1c illustrates a third common approach of modeling soft patient TW but includes distinct penalization according to how much the TW is violated. The lines segments a, b, and c depict the behavior of such a piecewise penalization approach. Furthermore, the larger angle of the line segment, the larger is the penalty incurred. Such an approach also considers a maximum tolerated violation through the parameter *hard_end*.

Time windows can also be applied on the operations center to model the time window which the caregivers can work. Such an approach can be used to model a global

Figure 4.1 – Patient's time-window handling approaches found in the literature of the HHCRSP.



Source: the author.

time window for all the caregivers, or one specific TW per caregiver, e.g., to model part-time workers (TRAUTSAMWIESER; HIRSCH, 2011). These later can also model regulation constraints regarding the working times of the staff, so the soft TW approaches can be applied to model overtime (BRAEKERS et al., 2016; LANZARONE; MATTA, 2014).

As time window arrives on several important routing problems, there are important results that should be taken into account when designing an algorithm for such problem. For example, Savelsbergh (1992) propose data structures that allow evaluating the feasibility of changing a solution in $O(1)$ to problems with hard time windows, and this results is very important to develop competitive algorithms. Furthermore, soft time windows are harder to handle, requiring for example custom pricing algorithms to cope with the dual information from the soft time windows in the master problem, in the column generation context (DROR, 1994; RASMUSSEN et al., 2012).

4.2.2 Operations synchronization on multiple-visit patients

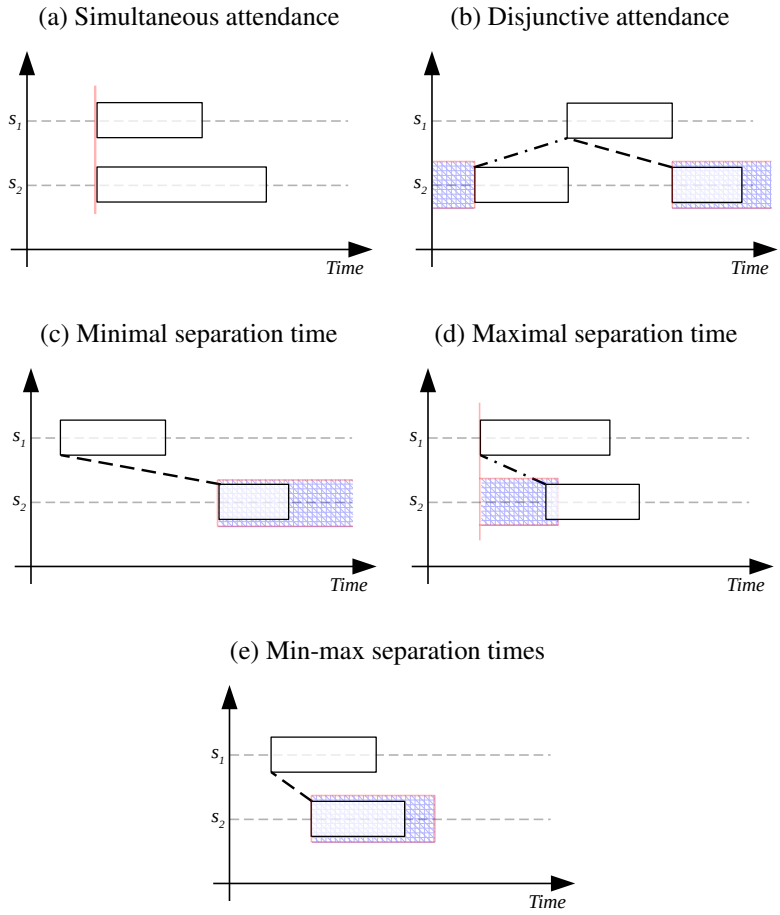
Frequently, the HHC literature discusses a patient's problem requiring multiple visits by caregivers for providing healthcare services. From the practical point of view,

these multiple visits cannot happen freely during the patient's time window. Such an issue becomes even more noticeable when several providers need to coordinate home care services. Often the VRP literature discusses such characteristics through operations synchronization between vehicles (DREXL, 2012). In the HHC context, these characteristics are also known as route inter-dependency constraints (MANKOWSKA; MEISEL; BIERWIRTH, 2014). Like HHC, operations synchronization also arrives in task workforce problems, e.g., where several teams or technicians must be present at the working location before starting the attendance (CASTILLO-SALAZAR; LANDA-SILVA; QU, 2016).

Dohn, Kolind and Clausen (2009) and Rasmussen et al. (2012) studied several generalizations of precedence constraints by manipulating the minimum and maximum separation times between multiple visits into the patient site. Furthermore, some of these route inter-dependency constraints have already been discussed in the HHC literature. For example, Bredstrom and Ronnqvist (2007) were one of the first authors approaching a problem where the patients can request simultaneous multiple visits. The authors motivated such a requirement to model cases that can be dangerous to a single caregiver to handle the patient, e.g., overweight. In their following paper, Bredström and Rönnqvist (2008) also proposed constraints to model multiple visits with minimum and maximum separation times. Both of these types of route inter-dependencies were also considered by Mankowska, Meisel and Bierwirth (2014), as well as in Decerle et al. (2018), Lasfargeas, Gagné and Sioud (2019), and Liu, Tao and Xie (2019).

In total, the literature proposed five types of route inter-dependency constraints applied to HHC services. The simultaneous attendance constraint, as depicted in 4.2a, requires that both service types s_1 and s_2 start their operation simultaneously. As depicted in Figure 4.2b, the opposite case requires the service operations to not overlap over time. In this figure, the blue hatched regions indicate the time spans where s_2 can be operated, given the current scheduling of s_1. Figure 4.2c illustrates the case which requires a minimum separation time between the operation of the two requested services. Analogously, Figure 4.2d illustrates the case where a maximum separation time is set between the services. Note that the span in which s_2 can be performed would extend to $-\infty$. Despite that, in practice, the interval is bounded by the starting time of the other service. Lastly, Figure 4.2e illustrates the case that combines both minimum and maximum separation times. Note that the cases of Figures 4.2a, 4.2c, and 4.2d can be achieved by manipulating the min-max parameters of Figure 4.2e.

Figure 4.2 – Operation synchronization constraints arriving on home health care context.



Source: the author.

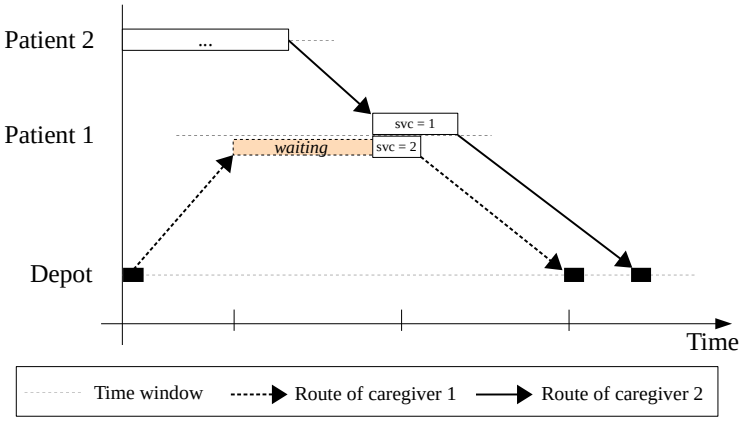
To the best of our knowledge, the router inter-dependencies illustrated in Figures 4.2a and 4.2e are most commonly approached in the HHC literature (CISSÉ et al., 2017; FIKAR; HIRSCH, 2017; GRIECO; UTLEY; CROWE, 2020). Mankowska, Meisel and Bierwirth (2014) proposed an HHCRSP that comprised extensions of these constraints. In addition to some patients requiring simultaneous visits, the authors also introduced the concept of precedence constraint when the patient visits require constraints in the format of min-max separation times. This way, the HHC provider can specify a fix operation order between the multiple services requested by the patient and establish the minimal and maximal separation time among them. These constraints allow modeling common cases arriving on healthcare services, e.g., when the patient requires a shot of some drug before receiving the visit of a physiotherapist, and such a drug requires a minimum time before starting their effect and a maximum time before the effect is gone.

Figure 4.3 depicts the cases of route inter-dependency studied by Mankowska, Meisel and Bierwirth (2014). These figures illustrate such constraints through a time-space

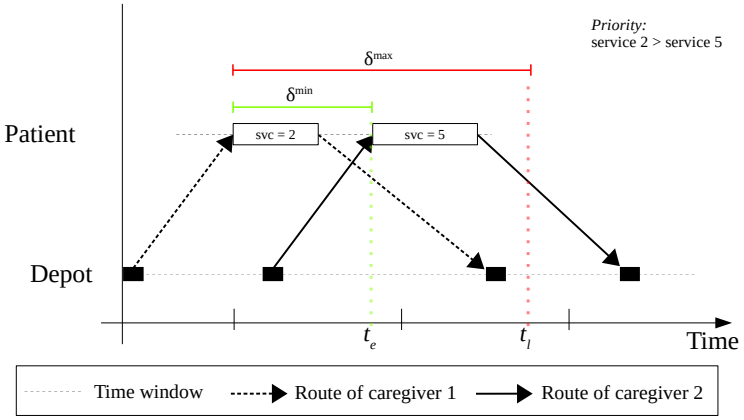
network. In these examples, the locations are disposed of in the y-axis and the time horizon in the x-axis. The horizontal dashed lines indicate the time window of the patients. In 4.3a, Patient 1 requested both the service types 1 and 2 to be operated simultaneously. As the arrows indicate, caregiver 1 arrives at the patient location before caregiver 2, which incurs waiting for the one who arrived later. Figure 4.3b illustrate the min-max separation time approached by Mankowska, Meisel and Bierwirth (2014). This example highlights the presence of a priority order among the services requested by the patient. This example also introduces the convention proposed by the authors to express the minimum ($\delta^{\min}$) and maximum ($\delta^{\max}$) between the multiple visits to the patient.As the reader can note, such a requirement induces the dynamic time window defined by $[t_e, t_l]$ for the visit of caregiver 2. The effects of these dynamic TWs are often discussed, e.g., in dial-a-ride problems (DREXL, 2012; GSCHWIND; IRNICH, 2015).

Figure 4.3 – Visual representation of patient time-line with route inter-dependency constraints.

(a) Simultaneous attendances



(b) Multiple visits with precedences and min-max separation times



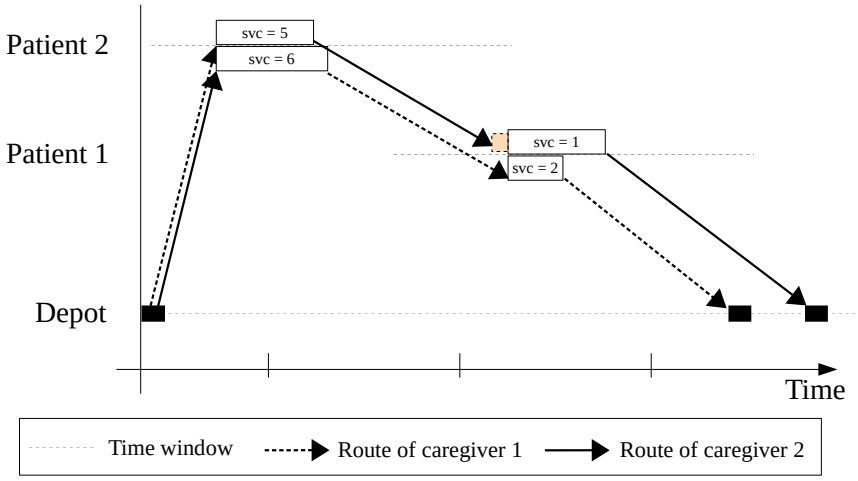
Source: the author.

Despite their simple concept, such route inter-dependency requirements act as a constraint for both heuristic and exact solution methods. Haddadene, Labadie and Prodhon (2016), and more recently by Liu, Tao and Xie (2019), observed that these constraints make devising efficient local search operators very hard, requiring sophisticated data structures to keep tracking of their state in a solution. Even evaluating if a movement is feasible can be quite challenging due to cross-synchronization involving, .e.g., two distinct patients requesting simultaneous multiple visits. Figure 4.4 illustrates this situation. In 4.4a, the solution is feasible, and all constraints are satisfied. A local search operator can lead to the situation illustrated in 4.4b, which no longer admits a feasible solution for simultaneous attendance constraints. It is also worth noting that the more of these constraints present in an instance of the problem, the more computationally intensive it is to check if a change in the solution leads to some improvement, requiring computing large chunks of the solution from scratch. Liu, Tao and Xie (2019) proposed local search operators that employ a "quick inspection" of the solution before evaluating the movement itself. The idea consists of evaluating partially the movement, considering only a subset of the objective function components, e.g., the travel time, while relaxing the other constraints like the soft time windows.

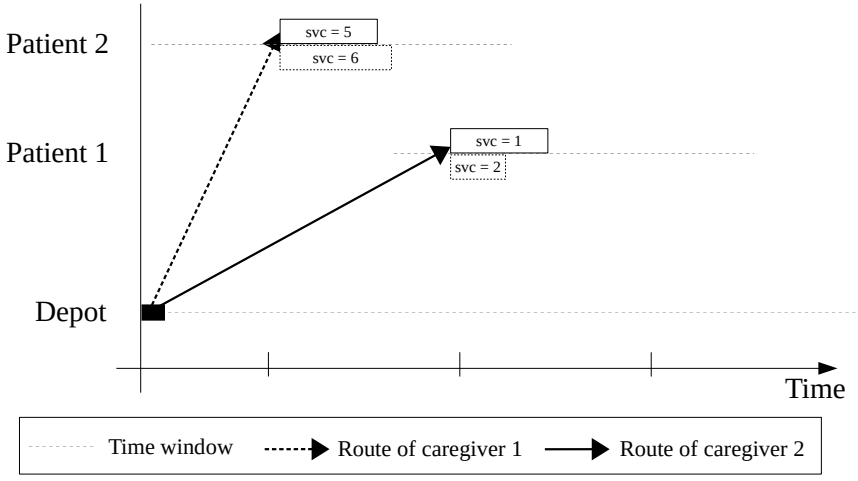
These route inter-dependency constraints also require the development of tailed algorithms when employing mathematical programming-based solution approaches, such as column generation (BREDSTROM; RONNQVIST, 2007; RASMUSSEN et al., 2012). To the best of our knowledge, all the authors proposing BCP-based algorithms handle these constraints during the branching phase. Otherwise, the author would need to adapt the algorithms applied to solve the pricing subproblems to handle the dual information from these constraints from the master problem Dohn, Rasmussen and Larsen (2011).

Figure 4.4 – Visual representation of cross-synchronization issue.

(a) A feasible routing solution



(b) Infeasible partial solution due to cross-synchronization



Source: the author.

5 PROBLEM DEFINITION

After reviewing and identifying the most common features arriving on the literature, we then start seeking a problem that fits our research purposes. This way, our methodology consists of identifying a problem in the literature that is complex enough to produce valuable solutions and be interesting from a scientific perspective. In some sense, we also want to approach a problem that is not too specific regarding some countries' application and working regulations (GRENOUILLEAU et al., 2019). With such a choice for a core problem, we could then devise additional constraints or objective function components to cope somewhat easily as new demands arrive. We approach the problem studied by Mankowska, Meisel and Bierwirth (2014) as our target problem with this mindset.

5.1 A MIP for single-day HHCRSP

As aforementioned, Mankowska, Meisel and Bierwirth (2014) proposed a single-day problem called the home health care routing and scheduling problem (HHCRSP). We interchange the terms HHCRSP and HHCP (home health care problem) as synonyms in the following text. We often refer to the service types by means of the word *skills*. Furthermore, we introduce the problem through a MIP model, but we first introduce some notation before that.

Let $\mathcal{V}$ be the set of the caregivers/vehicles, and $\mathcal{C}$ the set of patients/nodes. Let $\mathcal{C}^0 = \{0\} \cup \mathcal{C}$, where 0 represents the operations center–or depot. The parameter $d_{ij} \geqslant 0$ defines the travel time between all the pairs of locations $(i, j) \in \mathcal{C}^0 \times \mathcal{C}^0$. All caregivers travel at the same speed, i.e., the problem considers a single transportation modality. The patients also have time windows. More precisely, each patient $i \in \mathcal{C}$ has a hard time window starting e_i , meaning that a visit can only happen after the starting of their time window. Conversely, patient i also has a soft time-window ending l_i. Patient visits should be served ideally before the end of the patient time window; otherwise, it incurs in attendance tardiness, which is penalized in the objective function. These definitions are enough to set the routing component inside the problem that aims to assign caregivers to the patients, seeking to minimize the travel time.

In this HHCRSP, a patient can request multiple visits for distinct service types, which an equally skilled caregiver must fulfill. Let S be the set of service types considered in the problem. Caregivers are skilled in the sense that each caregiver has a subset of service types that they can perform. For each $v \in \mathcal{V}$ and service type $s \in \mathcal{S}$, the parameter $a_{vs} = 1$ indicates if the caregiver v can perform the service type s, otherwise 0. Patients have a similar parameter setting. The parameter $r_{is} = 1$ indicates if the patient $i \in \mathcal{C}$ requires attendance for the service type $s \in \mathcal{S}$, otherwise 0. Additionally, the processing time required to complete such attendance is defined according to the parameter $p_{is} > 0$, for the patient $i \in \mathcal{C}$ and service type $s \in \mathcal{S}$. These values are homogeneous regarding all the caregivers. Often this characteristic of multiple service types is presented as a *multi-attribute* VRPTW (CASTILLO-SALAZAR; LANDA-SILVA; QU, 2016; GRENOUILLEAU et al., 2019).

Without loss of generality, a patient can request one service type or two service types. We also refer to these patients as single-service patients and double-service patients, respectively. Double-service patients have either precedence constraints or simultaneous attendance constraints, as we describe in Section 4.2.2. In the first case, the parameters $\delta_i^{\min} \geqslant 0$ and $\delta_i^{\max} \geqslant 0$ indicate the minimal and maximal separation times between the services of patient $i \in \mathcal{C}$. On the latter case, these values are set to $\delta_i^{\min} = \delta_i^{\max} = 0$ so both services must be operated simultaneously. To ease the writing, please consider $\mathcal{C}^{\mathrm{d}} \subseteq \mathcal{C}$ as the set of patients requesting double services. The priority order among services on double-services is common to all the patients, and it follows the lexicographical order for any $s_1 \neq s_2 \in \mathcal{S}$. To illustrate this, consider $\mathcal{S} = \{a, b, c\}$. In this example, the service type a has priority over b and c, and b has priority only over c.

The MIP model introduced by Mankowska, Meisel and Bierwirth (2014) considers of three sets of decision variables. The binary variables $x_{ijvs} \in \{0, 1\}$ indicate whether the caregiver $v \in \mathcal{V}$ departs from the location $i \in \mathcal{C}^0$ to the location $j \in \mathcal{C}^0$ to perform the service type $s \in \mathcal{S}$ on the destination location. The continuous variables $t_{ivs} \geqslant 0$ indicate the start time of service $s \in \mathcal{S}$ for the location $i \in \mathcal{C}^0$ by the caregiver $v \in \mathcal{V}$. As we already mentioned, the model of Mankowska, Meisel and Bierwirth (2014) allows the services to start after the ending of the patient time-window. These are modeled using the continuous variables $z_{is} \geqslant 0$, which compute the tardiness on the start time for the service type $s \in \mathcal{S}$ and patient $i \in \mathcal{S}$. Lastly, the continuous variable $T^{\max}$ holds the value of largest tardiness of the solution. The MIP objective consists of a sum of three weighted components, one for minimizing the travel times, the other for minimizing the

total tardiness, and the third for minimizing the largest tardiness over all the patients. The weights λ_1, λ_2, and λ_3 apply to these components, respectively. Furthermore, the original model from Mankowska, Meisel and Bierwirth (2014) uses some additional variables to implicitly set the objective function components.

$$\text{Minimize} \quad \lambda_1 D + \lambda_2 T + \lambda_3 T^{\max} \tag{5.1}$$

Subject to:

$$D = \sum_{v\in\mathcal{V}} \sum_{i\in\mathcal{C}^0} \sum_{j\in\mathcal{C}^0} \sum_{s\in\mathcal{S}} d_{ij} x_{ijvs} \tag{5.2}$$

$$T = \sum_{i\in\mathcal{C}} \sum_{s\in\mathcal{S}} z_{is} \tag{5.3}$$

$$T^{\max} \geqslant z_{is} \qquad \forall i \in \mathcal{C}, s \in \mathcal{S} \tag{5.4}$$

$$\sum_{i\in\mathcal{C}^0} \sum_{s\in\mathcal{S}} x_{0ivs} = \sum_{i\in\mathcal{C}^0} \sum_{s\in\mathcal{S}} x_{i0vs} = 1 \qquad \forall v \in \mathcal{V} \tag{5.5}$$

$$\sum_{j\in\mathcal{C}^0} \sum_{s\in\mathcal{S}} x_{jivs} = \sum_{j\in\mathcal{C}^0} \sum_{s\in\mathcal{S}} x_{ijvs} \qquad \forall i \in \mathcal{C}, v \in \mathcal{V} \tag{5.6}$$

$$\sum_{v\in\mathcal{V}} \sum_{j\in\mathcal{C}^0} a_{vs} x_{jivs} = r_{is} \qquad \forall i \in \mathcal{C}, s \in \mathcal{S} \tag{5.7}$$

$$t_{ivs_1} + p_{is_1} + d_{ij} \leqslant t_{jvs_2} + M\left(1 - x_{ijvs_2}\right) \qquad \begin{array}{l} \forall i \in \mathcal{C}^0, j \in \mathcal{C}, \\ v \in \mathcal{V}, s_1, s_2 \in \mathcal{S} \end{array} \tag{5.8}$$

$$t_{ivs} \geqslant e_i \qquad \forall i \in \mathcal{C}, v \in \mathcal{V}, s \in \mathcal{S} \tag{5.9}$$

$$t_{ivs} \leqslant l_i + z_{is} \qquad \forall i \in \mathcal{C}, v \in \mathcal{V}, s \in \mathcal{S} \tag{5.10}$$

$$\begin{aligned} t_{iv_2s_2} - t_{iv_1s_1} &\geqslant \delta_i^{\min} \\ &- M\left(2 - \sum_{j\in\mathcal{C}^0} x_{jiv_1s_1} - \sum_{j\in\mathcal{C}^0} x_{jiv_2s_2}\right) \end{aligned} \qquad \begin{array}{l} \forall i \in \mathcal{C}^{\mathrm{d}}, v_1, v_2 \in \mathcal{V}, \\ s_1, s_2 \in \mathcal{S} : s_1 < s_2 \end{array} \tag{5.11}$$

$$\begin{aligned} t_{iv_2s_2} - t_{iv_1s_1} &\leqslant \delta_i^{\max} \\ &+ M\left(2 - \sum_{j\in\mathcal{C}^0} x_{jiv_1s_1} - \sum_{j\in\mathcal{C}^0} x_{jiv_2s_2}\right) \end{aligned} \qquad \begin{array}{l} \forall i \in \mathcal{C}^{\mathrm{d}}, v_1, v_2 \in \mathcal{V}, \\ s_1, s_2 \in \mathcal{S} : s_1 < s_2 \end{array} \tag{5.12}$$

$$x_{ijvs} \in \{0, a_{vs} r_{js}\} \qquad \forall i \in \mathcal{C}^0, v \in \mathcal{V}, s \in \mathcal{S} \tag{5.13}$$

$$t_{ivs} \geqslant 0 \qquad \forall i \in \mathcal{C}^0, v \in \mathcal{V}, s \in \mathcal{S} \qquad (5.14)$$

$$z_{is} \geqslant 0 \qquad \forall i \in \mathcal{C}, s \in \mathcal{S} \qquad (5.15)$$

$$D,\, T,\, T^{\max} \geqslant 0 \qquad (5.16)$$

The weighted objective function is defined by (5.1) while each objective component (distance, tardiness, and largest tardiness) is set by (5.2), (5.3), and (5.4), respectively. The constraints (5.5) and (5.6) model the flow conservation on all locations. The constraints (5.7) model the assignment of the demands for the capable staff members. (5.8) implements the sub-tour elimination, taking into account the travel times and service processing times. The patient time-window are modeled by (5.9) and (5.10). Operations synchronization are set by (5.11) and (5.12). Finally, the constraints (5.13–5.16) set the domain of the decision variables. This MIP model can also be defined in terms of components: (5.1–5.4) defines the objective component; (5.5–5.10) defines the multi-attribute VRPTW; Finally, (5.11,5.12) defines the operation synchronization between the caregivers.

From a scientific perspective, there is a trade-off of minimizing the visit tardiness at the expanse of increasing the caregivers' travel times. A similar trade-off arises for the maximum tardiness because reducing the cost of this component may require increasing the total tardiness of the solution. Due to these inherent properties of the HHCRSP, the problem is considered *multi-objective* (MANKOWSKA; MEISEL; BIERWIRTH, 2014). From a more practical perspective, this typically means an increased HHC provider's effort to understand the weighted objective function (5.1) and fine-tune its coefficients. Depending upon the solution the MIP model produces, it may not suffice to specify the same value for λ_1, λ_2, and λ_3 coefficients as (5.1), requiring specific values for some caregivers or patients. For example, it may be necessary to suppress the weight coefficient for the total tardiness from the objective function, replacing (5.3) to (5.17). This specification for the total tardiness component allows the provider to specify distinct values of λ_2^{is} for some $i \in \mathcal{C}, s \in \mathcal{S}$, otherwise falling back to the default value by setting $\lambda_2^{is} = \lambda_2$ for the cases which a custom weight is not required.

$$T = \sum_{i \in \mathcal{C}} \sum_{s \in \mathcal{S}} \lambda_2^{is} z_{is} \qquad (5.17)$$

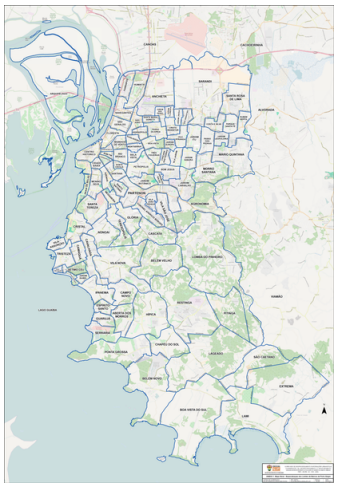
5.2 A foreword on the public HHC services of Porto Alegre

Porto Alegre is both the capital and the most populous city of the Rio Grande do Sul state. Among the most populous Brazilians cities, it is ranked 10th. Compared to the top two cities, Porto Alegre has (roughly) $\frac{1}{11}$ the number of residents of São Paulo and $\frac{1}{6}$ the population size of Rio de Janeiro, respectively (IBGE, 2010). As shown in Figure 5.1a, Porto Alegre is located at the most south Brazilian state. According to the latest data, its metropolitan region covers 94 official boroughs and embeds the territory of smaller cities around it, as depicted in Figure 5.1b.

Figure 5.1 – Porto Alegre location withing the Brazilian territory.

(a) Overview of Rio Grande do Sul state

(b) Borough distribution over the metropolitan region



Source: Wikipedia (2021).

As we mention in Chapter 1, Porto Alegre's Health Department offers since 2011 home health care services through the Brazilian public health services (Ministério da Saúde, 2013). The operations of Porto Alegre's HHC system started back in 2011 as a response to the public authorities' demands for *de-hospitalization*, i.e., to move patients from the hospital to continue their healthcare in their homes, when such is viable and offers a low risk to the patients. Moreover, most users in the system are older people or patients with some chronic health issue or palliative care. More recently, HHC services have also been offered for the recovery period of surgery patients.

Three distinct providers offer healthcare services in the home health care format, and each one of these three providers is installed inside the city's big public hospitals. They all have dedicated teams to provide such services, and all their expenses are paid through public financing. The decision to split the system into three providers is due to the

infrastructures available in each hospital and the demographic distribution of Porto Alegre. Moreover, each provider offers HHC services to only a subset of the city boroughs, and they run independently.

Our knowledge of these systems is limited to the information we got from our interviews with the manager of the largest HHC provider of Porto Alegre, comprising 19 daily working teams. Due to the lack of integration with data sources, the demands are given with a week of advance; usually, a phone call communicates any last-minute change. Moreover, the teams' planning is done daily by an experienced caregiver that selects which patients will be serviced during the day by each team, considering their proximity and the current availability of medical equipment and drug supply. The single objective is to schedule as many patients as possible within the time frame of one day. Each team is composed of at least one vehicle driver, plus the caregiver that attends the patients. Moreover, most routing solutions are made by the vehicle's driver, either using smartphone apps to guide their routing decisions or according to their knowledge regarding road structure and the city's traffic behavior over the day.

This provider offers several medical specialties, so the qualification characteristics of the caregivers are pretty much the same as in the problem studied by Mankowska, Meisel and Bierwirth (2014). As a complicating aspect, the provider has fewer vehicles available than the number of caregivers working daily, and consequently, the number of teams effectively working at any moment is bounded by the number of vehicles available. The current operations resembles the car-sharing system discussed by Fikar and Hirsch (2018). A similar issue arrives w.r.t. the scheduling of medical equipment carried by the teams, since the number of these is also limited. In some occasions, the problem also require the scheduling of medical students that apply for residence in the HHC services. In those cases, the vehicle capacity matters since the equipment can use a lot of space and may harm the safety of the passengers.

Route inter-dependency constraints also arrive in the operations of Porto Alegre's HHC system. A common case is when the provider reserves a few vehicles for what they call *support teams*. Sometimes the operations of these teams are known a priori. In other cases, they can be called up in an ad-hoc fashion, typically for picking samples or delivering medical equipment or supplies upon the caregivers' request. This way, it is necessary to synchronize the visits of the caregivers with these support teams. Another important distinction is that the support teams have very short visit times, and they spent most of their time traveling between the patients' sites and the operations center.

In addition to all these planning requirements and features, the system still requires re-planning capability to handle with last-minute changes. In some cases, subtle changes in the health condition would require to reschedule a critical patient to as early in the routes as possible–sometimes requiring also the team designated to such patient to interrupt their current work and to abandon all the other patients remaining in their route. According to our interview with the manager of this system, these emergencies happen almost in daily basis so the demands for such re-planning must be incorporated in any computational solution from day-0. Other common cause for rescheduling is the uncertainty regarding the visit duration of some patients, typically when a new one arrives at the system, or when e medical student is assisting the caregiver operations. Furthermore, a computational solution system needs to communicate any changes to all the caregivers automatically.

Concerning the rescheduling, we are still not sure how much we can change, say an initial schedule, to better incorporate last-minute changes in the plan. More precisely, we are unsure of how well the teams would accept more subtle changes of their "original" plan since a major modification may also be disruptive to patients and reduce the service levels. Of course, dramatic changes would also burden the teams if the initial planning changes too much or changes too many times during the day. Another important observation is that the increasing stress level of the team after servicing an emergency patient can be so high that the best decision would be to insert the remaining patients of their original schedule in the route of the other caregivers or to return the team to the operation center and then allocate the vehicle to another available team.

With all this said, there is still a budget limitation to develop and implement computational solutions to assist and optimize the planning of the HHC services. Typically the provider cannot afford commercial solvers like CPLEX and GUROBI due to the expensive licensing fees. With this in mind, our approach is to devise solution methods that can employ free and open-source solutions like Coin-OR CBC and GNU Linear Programming Kit (GLPK). Of course, heuristic methods are the most straightforward alternative concerning the performance of these free mathematical programming packages, but the challenge here is how to guarantee the maintenance of such a specialized code over time. From our experience, these maintenance tasks require an experienced developer who understands both the theoretical complexity of the employed solution methods and the abstraction costs involved in the underlying implementation. In other words, the ideal person for maintaining such specialized code is its developer. Furthermore, applying a more standardized approach such as mathematical programming methods would ease these difficulties.

6 OBTAINING STRONGER LOWER BOUNDS FOR THE HHCRSP

When measuring the solution quality of an HHCRSP instance, two values are of great importance. From a practical point of view, an HHC provider is more interested in finding a feasible solution to the problem that holds the smallest cost possible–ideally an optimal solution. Often the literature calls these solution values as *upper bounds* (UB). Still, it is also desirable to measure how good a solution to a particular instance can be from a more scientific perspective. This chapter focuses on methods to provide these so-called *lower bounds* values (LB). The relative gap between the LB and the UB measures what the literature calls as optimality gap of such instance. An optimal solution has this gap at zero when its optimal value has been proved. This way, the efforts in obtaining stronger lower bounds and better upper bounds are both significant. Section 6.1 presents a methodology to benchmark a state-of-art mathematical programming solver, which can be employed to provide both LB and UB values. Section 6.2 further explores the MIP-based solution methods by introducing what we call redundant cuts. Section 6.3 discusses a little widespread technique for calculating stronger LB called *combinatorial lower bounds* (CLB).

6.1 Improved lower bounds from state-of-art MIP solver

According to the computational experiments reported by Mankowska, Meisel and Bierwirth (2014), only tiny instances of up to ten patients could be solved to optimality using for that the MIP model of (5.1)–(5.16) and the solver CPLEX 12.3, and feasible solutions could be found to test cases with 25 patients. All the other experiments comprising larger instances produced no feasible solution at all. Instead, those experiments produced what we call as best lower bounds (LB^+), i.e., a lower bound produced by the branch-and-cut (BC) algorithm employed within the solver. These LB^+ are attractive for two reasons: first, they are stronger than the values produced while solving a linear relaxation of the problem, thanks to the automatic cut generation algorithms that CPLEX employs when solving discrete models. Secondly, these still hold valid lower bounds because they refer to the value of the most promising leaf node of the BC tree when the search stopped.

Considering the progress of CPLEX development since 2014, we re-run the experiments with the MIP model but using the most up-to-date CPLEX version available (version 20.10). Furthermore, we use the constraints (5.13) to skip generating decision variables

bound to zero due to either the lack of caregiver skill or missing service type requests. Besides the solver being capable of eliminating constraints before solving the model, we also apply some manual preprocessing to skip generating empty flow constraints (5.5) and (5.6), which helps to reduce the memory consumption even further. We significantly reduce the MIP memory consumption to run the solver on large instances using a "standard" computer with 8 GB of memory with these simple improvements.

Despite our effort, we still expect the solver to exceed the available memory during the optimization of the models, so we propose a few scenarios in which we change some of the CPLEX parameters attempting to reduce the growth rate of memory consumption as the search progresses. The choice of parameters to consider was taken under the guidelines from CPLEX documentation (IBM, 2020). Furthermore, IBM has this habit of changing the default seed of the pseudo-random number generator that CPLEX embeds in each new solver release, so we change the seed value to 1 in all the scenarios we test.

- **Scenario 1:** We solve solely the linear relaxation of the MIP model;
- **Scenario 2:** We use CPLEX with all of its default parameter settings;
- **Scenario 3:** We use CPLEX with most of its default parameter settings, but we turn on the `memory emphasis` flag;
- **Scenario 4:** We use CPLEX with most of its default parameter settings, but we limit the search to one thread.

After experimenting with these scenarios, we also devise a few others that employ more heavy customization of CPLEX. Scenario 5 is based on the outputs after testing scenarios 3 to 4. In regarding the default parameter settings, CPLEX also seems to stress finding a feasible solution at the start of the search, leading to huge BC trees and high memory consumption. Scenario 6 tries to overcome this by feeding CPLEX with a feasible initial solution through a MIP warm start (IBM, 2020), provided by the genetic algorithm we describe in Chapter 8.

- **Scenario 5:** Combines the parameter settings of scenarios 3 and 4;
- **Scenario 6:** Same as scenario 5, but we also start the search with a feasible solution through CPLEX *warmstart* capability.

6.2 Improving lower bounds with additional information from redundant cuts

Until this date, the existence of an *efficient* algorithm for solving NP-*hard* problems is unknown (GAREY; JOHNSON, 1979). Several authors agree with the unlikeliness of the existence of such an algorithm, and the state-of-art *exact* techniques for solving such problems comprise mainly an enumeration-based procedure with some speedup mechanisms. In the context of mixed-integer programming, a typical optimization package relies on the branch-and-bound (BB) strategy for exploring the problem's solution space, using its linear relaxation for obtaining lower bounds on each node of the tree. This combination of algorithms is often referred to as branch-and-cut, as we mention in Sections 5.1 and 6.1. Solving the LP of the root node can be quite a computation-intensive task, but this approach is somewhat efficient as most of the other node-level LP problems are quickly solvable. Furthermore, the lower bound values can be used to prune the tree, and other exploitation mechanisms can be employed to take advantage of the BC framework (ACHTERBERG; KOCH; MARTIN, 2005).

It is common knowledge that *stronger* LP formulations are essential when solving a combinatorial problem with BC (WOLSEY; NEMHAUSER, 1999). For some problems that a tight formulation is known, the BC can be very effective, requiring the exploration of just a few nodes to prove an optimal solution. Unfortunately, this is not the case of VRP-relate problems like the HHCRSP, to which the lower bounds provided by the LP are generally too fractional to help speed up the search (DESAULNIERS; DESROSIERS; SOLOMON, 2006). Even worse, the BC approach tends to be very inefficient due to the behavior of such models: a branching on a single variable causes others to become more and more fractional, so the expected number of branches required to solve the problem is exponential in the number of variables of the model. This difficulty often motivates the research of new yet stronger formulations from the ground, but the most common approach is to include additional constraints on current models to further "cutaway" fractional solutions from the problems' solution space (WOLSEY; NEMHAUSER, 1999). These types of additional constraints are referred to as valid cuts. According to Bektaş, Erdoğan and Røpke (2011), a third approach is to develop weaker but faster to solve models, maximizing the throughput of nodes explored by the BC algorithm and, hopefully, proving an optimal solution faster than using a stronger–and frequently slower to solve–formulation.

Current MIP solvers embed routines to identify structures in the input models, and such information can be used to trigger the automatic generation of valid cuts (IBM, 2020). Unfortunately, sometimes the optimization software fails to generate some of these valid cuts due to the failure in detecting such structures in its input data. With further development of optimization packages, it is expected that software developers include more robust cut generation routines from time to time. Furthermore, it is not uncommon to see a previously stronger formulation performing worse than a weaker one in more updated solver versions (BEKTAŞ; ERDOĞAN; RØPKE, 2011), thanks to the improved "intelligence" of MIP solvers on recognizing structural properties of the problem in these weaker yet less constrained formulations. Trick (2005) exemplifies this in the traveling tournament problem: generating by hand cuts that the optimization package could automatically lead to larger solver runtimes and a "worse" trajectory to prove an optimal solution. As general advice, the author suggested leaving the generation of valid cuts by the optimization packages rather than manually.

The evolution of optimization packages also opens new opportunities to improve MIP performance through redundant constraints. Aardal (1998) showed that the additional information provided by *redundant cuts* helped the CPLEX solver (version 2.1) identify more structures on the problem space, which triggered the automatic generation of otherwise inaccessible valid cuts. Computational results indicated an improvement of up to 50% on the solver runtime while solving the facility location problem. This approach is also explored by Trick (2005) in the context of a transportation problem. Lastly, it is important to highlight that the performance of a MIP model varies with the optimization packages, so an approach that works well with one software might not be optimal to others.

That said, we propose the following additional cuts for the HHCRSP. Constraints (6.1) are an augmented version of the assignment constraints 5.7, which generates these assignment constraints for every pair of vehicles in the problem. In turn, constraints (6.2) are similar to the flow conservation constraints (5.5, 5.6). They set that for any two nodes j and k, the amount of flow in the arc (j, k) is limited by the amount of incoming flow in node j through the arc (i, j), for every other node $i \neq j \in \mathcal{C}$.

$$\sum_{j \in \mathcal{C}^0} (x_{jiv_1s} + x_{jiv_2s}) \leqslant 1 \qquad \forall v_1 \neq v_2 \in \mathcal{V}, i \in \mathcal{C}, s \in \mathcal{S} \tag{6.1}$$

$$x_{jkvs_k} \leqslant \sum_{\substack{i \in \mathcal{C}: \\ i \neq j}} \sum_{s_j \in \mathcal{S}} x_{ijvs_j} \qquad \forall v \in \mathcal{V}, j \neq k \in \mathcal{C}, s_k \in \mathcal{S} \tag{6.2}$$

By augmenting Mankowska's model with these two constraints, we define another testing scenario with the CPLEX solver. As on its description, scenario 7 combines the parameter setting from scenarios 3 to 4, and we also add two families of redundant cuts (6.1, 6.2). In any way, we still have some concerns because these constraints would contribute to increase the model size significantly, turning the generation of this augmented model impractical in our computational environment.

- **Scenario 7:** Same as scenario 5, but we also enable the generation of a few additional cuts, aiming to improve the LB^+ produced by CPLEX.

6.3 Combinatorial lower bounds for the HHCRSP

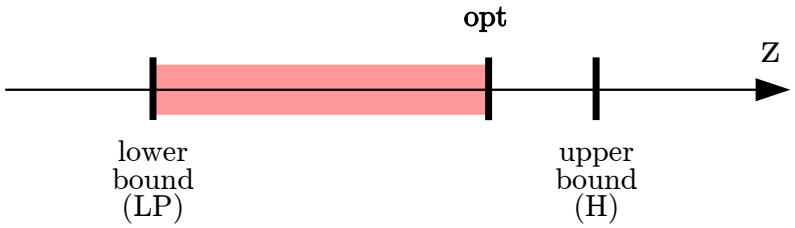
Suppressing constraints of a MIP can improve (or at least keep the same) the optimal solution value of such a model. Additionally, relaxed models tend to be much easier to solve, e.g., in problems in which finding a feasible solution is difficult (DORNELES; ARAÚJO; BURIOL, 2014). This idea behind relaxation is applied to MIP and motivates algorithms such as Lagrangian relaxation (FISHER, 1981). Besides, the more widespread relaxation technique probably consists of relaxing the integrality constraints of a MIP model to achieve the so-called linear relaxation of the problem. This approach is at the core of branch-and-cut algorithms and in all state-of-art mathematical programming packages for providing lower bounds for every node of a BC tree.

For the sake of reading, suppose an optimization problem with a minimizing objective function. A lower bound is some value obtained when solving a relaxation of this problem and limits *how good* the original problem's optimal solution can be, at least. A necessary condition for such value to be a valid lower bound is that the value of the relaxed problem needs to be *smaller than or equal to* the optimal value of the original problem. As this optimal value is generally unknown, a more conservative approach solves the relaxed model *to optimality*. For this reason, this approach is usually only applied

when the relaxed problem is *significantly easier* to solve (to optimality) than the original problem, which is the case of linear relaxation of MIP models to which efficient linear programming algorithms are known.

Figure 6.1 depicts the relation between linear relaxation lower bound (LP), the upper bound produced by some heuristic "H", and a MIP model's optimal solution value. In this example, a naive branch-and-cut algorithm starts exploring the solution space from the lower bound "LP" and goes in the direction of the upper bound value provided by H, exploring the entire solution space highlight in red until achieving the optimal solution "opt." Weaker lower bounds force the exploration of larger solution space, and stronger formulations may require exploring a smaller chunk of the solution space (WOLSEY; NEMHAUSER, 1999). For that reason, state-of-art optimizers like CPLEX trigger during the search embed cut detection routines to improve LP lower bounds. In general, these generated cuts can move the best lower bound far over in the optimal solution direction and effectively reduce the total tree's size during BC.

Figure 6.1 – Relation of lower bounds, upper bound, and optimal solution on a minimizing optimization problem.

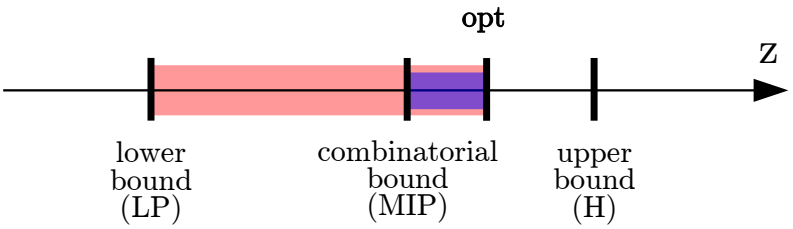


Source: the author.

There are other approaches in OR literature for computing a lower bound of a MIP. In a more general way, one can relax an arbitrary number of constraints of the original MIP to generate other easier-to-solve models. If such a relaxed model still holds the original MIP's integrality constraints, it can provide the so-called *combinatorial lower bound* (CLB) for the original problem (UMA; WEIN, 1998; UMA; WEIN; WILLIAMSON, 2006). Note that it is still required to solve such relaxation to optimality. The idea here is to generate a relaxed model that *sufficiently easier* to solve to optimality than the original MIP for which optimal solution value is stronger–or provably closer to the optimal value–than the standard linear relaxation approach. Figure 6.2 illustrates this situation. In this example, even the naive branch-and-cut algorithm benefits from such a more robust limit, requiring exploring the blue solution space only for achieving the optimal solution.

For the HHCRSP, a possible relaxation consists of dropping the operations synchronization constraints (5.11) and (5.12), which is the same relaxation employed by

Figure 6.2 – Relation of linear programming lower bound and the combinatorial relaxation lower bound on a minimizing optimization problem.



Source: the author.

Mankowska, Meisel and Bierwirth (2014) while solving larger problems with CPLEX. However, we make a small addendum in their definition of the relaxed problem as a multiple traveling salesman problem with time windows (mTSPTW). According to the seminar work of Dantzig and Ramser (1959), the multiple TSP consists of a *single* salesman visiting all the nodes of the problem, with the additional requirement of the periodic visits to the depot after every m visited nodes. Such additional constraint does not apply to this relaxation of the HHCRSP, so it is more precise and correct to refer to the proposed relaxation as a VRPTW with soft time windows.

Mankowska, Meisel and Bierwirth (2014) had to use the relaxed model due to the memory limitation of their benchmark computer, but our motivation is different. Despite relaxing operations synchronization constraints, the resulting VRP is anything but easy to solve, even by industry-standard optimization packages as CPLEX, GUROBI, and XPRESS. As the VRPTW is one of the most studied variations of its field, our approach takes advantage of state-of-art methods to solve the relaxed problem. State-of-art exact methods rely on the so-called column generation procedures (CG), which typically consists of using a set-partition model–usually called master problem–to build a solution from a pool of available routes. CG iteratively solves a linear relaxation of the master problem (MP), and the optimization is guided by the dual variables of the MP, which are used to update the arc costs of the underlying flow problem extracted from the "original" MIP model–which often the literature refers as the compact formulation. This flow problem is often called a *subproblem* or a *pricing* problem. This iterative CG procedure stops when the subproblem can find no column (or route) with a negative reduced cost. Thanks to the duality of linear programming, it is guaranteed that a CG converges to an optimal value if the subproblem is solved to optimality, at least on the last CG iteration. For a detailed description of CG algorithms, we refer to the primer on CG for a vehicle routing problem of Desaulniers, Desrosiers and Solomon (2006).

Column generation was developed initially for linear programming solving but not for binary and integer optimization problems, and in practice, it is very unlikely that the optimal solution of the MP to converge to an integer solution. For these applications, the CG can be embedded into a branch-and-bound to search for integer solutions, devising an algorithm similar to branch-and-cut for MIP. This approach is known as branch-and-cut-and-price (BCP). Implementing BCP algorithms is hard in practice since introducing integer feasibility cuts on the master problem also introduces changes on the subproblem. Consequently, a BCP approach can typically be quite inefficient if the subproblem becomes too difficult to optimize (DESAULNIERS; DESROSIERS; SOLOMON, 2006). Typically, a BC algorithm is more effective in these cases, mainly due to larger research, development effort, and results on BC than on BCP algorithms (PESSOA et al., 2020). For that reason, we use the *VRPSolver* tool from Pessoa et al. (2020) to solve a combinatorial relaxation of the HHCRSP. Pessoa et al. (2020) proposed this tool as a generic solver to VRP-related problems, and it embeds a column generation procedure, an efficient labeling algorithm for solving pricing problems, and several state-of-art components solve the integer problems. VRPSolver is free for academics. Thus it can be used without any costs for scientific purposes.

VRPSolver application programming interface (API) is written in Julia, and its distribution package contains several implementation examples for a variety of optimization problems that can be modeled as a VRP. On its internal, the VRPSolver employs the CPLEX as a linear programming optimizer. The branch-and-cut-and-price is implemented using the *BaPCod* library, designed to help to develop BCP-based algorithms (VANDERBECK; SADYKOV; TAHIRI, 2017). To implement a model with VRPSolver, the user inputs one or more *graphs* to define all the feasible arcs of the problem and their respective travel times or costs. The time windows are modeled as *resources*, which are consumed as the vehicle travels the arcs. The node's time windows can be set as accumulated resource consumption per node. Once the graphs are set, the user defines the set of decision variables of the problem and the objective function coefficients. Note that VRPSolver uses the concept of variable mapping to establish the relation of the routing variables and the routes generated by the column generation algorithm, which allows finding integer solutions without branching directly in route variables and allows to keep subproblem unchanged during the branching phase (PESSOA et al., 2020).

Despite all the flexibility offered by the API, we find no alternatives to access the visit start times of the generated routes. Thus we could not model the soft time

window characteristic of the HHCRSP. Furthermore, this issue is somewhat expected since the VRPSolver does not allow changing how it updates the arc costs in the subproblem, and how the reduced costs are computed. Similarly, we could not model the operations synchronization constraints. This way, we use VRPSolver to optimize the following model, which consists of a relaxed version of the model proposed by Mankowska, Meisel and Bierwirth (2014) without the route inter-dependency constraints, and without the soft time window ending. This way, the objective function of such relaxed model attempts to optimize only the travel times of the caregivers.

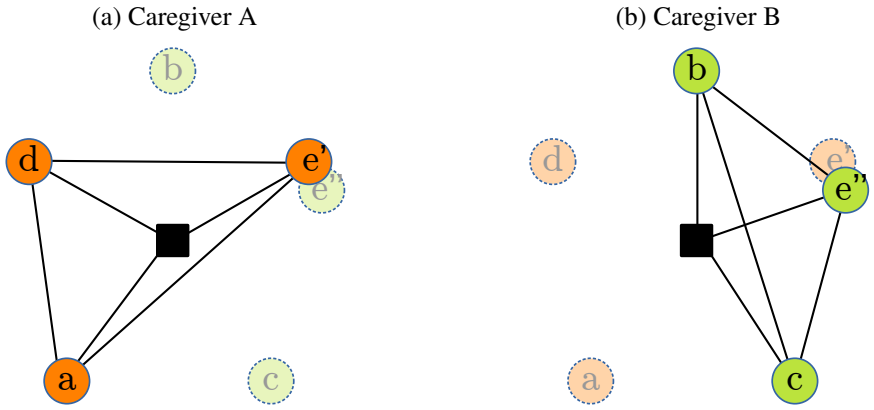
$$\text{Minimize} \quad \lambda_1 D \tag{6.3}$$

Subject to:

$$(5.2)$$
$$(5.5)\text{–}(5.9)$$
$$(5.13, 5.14)$$
$$D \geqslant 0 \tag{5.16}$$

As the reader can perceive, this simpler model involves solving a VRP, which still belongs to the $\mathcal{NP}$-hard class of problems. Furthermore, time windows are only considered partially through constraints (5.9), limiting the earliest visit time. In the source code, we also had to duplicate nodes to model patients requesting multiple visits, similar to the approach that Bredström and Rönnqvist (2008) used to model simultaneous visits patients. Such transformation is very straightforward and does not change any characteristic of the problem. It is also worth mentioning that we still model the service types implicitly by excluding infeasible arcs in the input graphs of VRPSolver. This way, we set distinct flow graphs for each caregiver separately, as Figure 6.3 illustrates. In this example, the problem comprises five patients and two caregivers. Patients in orange require a service type that only caregiver A can perform. Similarly, patients in green require another service type, and they can only be assigned to caregiver B. Note that node e is duplicated, and each replica represents a distinct service type demanded by the patient. This way, Figures 6.3a and 6.3b illustrate the input graph of caregiver A and caregiver B, respectively.

Figure 6.3 – Example of input graphs for the VRPSolver.



Source: the author.

Finally, the VRPSolver can solve a combinatorial lower bound of an HHCRSP instance by following three steps.

1. We apply an algorithm to duplicate double service nodes and generate the input graphs for VRPSolver.
2. We insert these graphs into the solver, specifying the travel times of the arcs and the earliest visit time of each node.
3. We solve the problem using the suggested parameters for a VRPTW problem from the documentation of VRPSolver.

After the optimization, the solver always stops producing no solution–in case the root note did not converge due to tailing effect (DESAULNIERS; DESROSIERS; SOLOMON, 2006)–or with a solution that can be used as combinatorial lower if the branch-and-price phase is completed successfully, as we further discuss in Section 10.3. These bounds can then be used to analyze how far the solutions produced by heuristics of Chapters 7 and 8 are from the theoretical optimal values.

7 THE FIX-AND-OPTIMIZE MATHEURISTIC

After discussing methods for providing the best LB^+ possible for the HHCRSP, we introduce our first solution approach to solve the whole problem using the MIP model introduced by Mankowska, Meisel and Bierwirth (2014). As we already mentioned, the model (5.1–5.16) holds a very poor convergence behavior, and state-of-art solvers like CPLEX fails in finding any feasible solution to test cases larger than 50 patients. This behavior is somewhat provoking since even a naive constructive heuristic can be successful in its intention. Furthermore, providing a "warm start" to the MIP solvers is not enough to kick off the optimization process since there is still a large gap between the lower bound from MIP linear relaxation to the constructive initial solution.

One approach to cope with this convergence behavior is to *fix* some of the decision variables of the problem, allowing the solver to optimize a smaller problem. This variable fixing gimmick makes it possible to lift (artificially) the lower bound provided by the MIP relaxation, allowing the solver to prove–hopefully in short processing times–the optimal solution value when optimizing the routes of the "free caregivers." Helber and Sahling (2010), inspired by the previous work of Pochet and Wolsey (2006), explored this characteristic to devise a model-based solution approach known as fix-and-optimize (F&O). Such a *matheuristic* was firstly introduced to solve lot-sizing problems (HELBER; SAHLING, 2010; CHEN, 2015) and was further applied to solve the high school timetabling problem (DORNELES; ARAÚJO; BURIOL, 2014).

There are some significant motivations to develop F&O-based algorithms. First and foremost, typically is very easy to change the characteristics of the problem embedded in a fix-and-optimize solution framework: As far as no change is introduced in the decision variables of the model, modifying constraints or the objective function may only require a reworking of the underlying model. This resilience is also desirable in more dynamic environments, e.g., when the manager introduces ad-hoc constraints of one-time-use or when the researcher is still in the problem definition phase and the solver is not powerful enough to provide solutions. Moreover, this matheuristic can be more easily introduced in teams where collaborators prefer to sifting their favorite MIP solver rather than devising heuristic solution approaches.

After introducing the historical data and some motivations for its use, we can define the FO matheuristic as the iterative procedure depicted in the following steps.

Step 1. MIP generation. Create the MIP relative to the problem to be solved. Such a model contains all decision variables, constraints, and objective functions from standard formulation to the problem. In the context of the HHCRSP, this model comprises (5.1–5.16). After generating the model, go to Step 2;

Step 2. Initial variable fixing. Generate a solution by employing a heuristic procedure. Then, fix the upper and lower bounds of the decision variables according to the heuristic solution provided. Ideally, the initial solution should be provided quickly. Then, go to Step 3;

Step 3. Subproblem generation. Select some of the decision variables and restore their lower and upper bound to their original values. The choice of which variables to select defines a decomposition scheme and should follow the different elements of the problem. For example, Dorneles, Araújo and Buriol (2014) proposed decomposition by day, teachers, and classes. Ideally, the decomposition scheme should be designed with efficiency in mind to enable finding better solutions while keeping the generated subproblem small enough to be solved efficiently by the solver. Go to Step 4;

Step 4. Optimization of the subproblem. Optimize the modified MIP from Step 3 up to some time limit or optimality. An improved solution can be found during this step. In the worst case, the solver keeps the best solution found so far. If the stopping criterion was reached, then go to Step 6. Otherwise, go to Step 5;

Step 5. Fixing the current solution. Fix all decision variable bounds according to the current solution to the MIP. Jump back to Step 3;

Step 6. Finish. The matheuristic returns the current MIP solution as the best one found during the search.

It is important to highlight that the performance of the F&O matheuristic depends heavily upon the difficulty of generated subproblems and the performance of the employed MIP solver. This way, generating large subproblems can compromise the matheuristic efficiency, leading to poor convergence behavior. Conversely, poor decomposition strategies can lead to an ineffective algorithm with limited capacity to improve an initial solution. According to the literature, the standard approach seems to be the systematic decomposition schemes, e.g., to generated all possible subproblems for pairs of distinct teachers (DORNELES; ARAÚJO; BURIOL, 2014). Moreover, the quality of the initial solution provided in Step 2 is also relevant, which is somewhat intuitive since it would be easier for the matheuristic to find improvement if the initial solution is of bad quality.

With the groundings of the matheuristic set, we can then introduce the problem-specific components necessary for solving the HHCRSP, namely how to provide initial solutions to the matheuristic and the decomposition schemes applied to generate the subproblems.

7.1 Providing initial solutions to the matheuristic

Our proposal of the fix-and-optimize was devised back in 2019 when we started our studies in the HHCRSP. On occasion, most of the information we have about the problem came from Mankowska, Meisel and Bierwirth (2014) and a few other works listed in the survey of Fikar and Hirsch (2017). With such limited knowledge about the problem, we adopted the constructive heuristic of Mankowska, Meisel and Bierwirth (2014) instead of developing another from scratch. Despite the simplicity, their heuristic is very fast–and greed–because it only considers the travel times between locations and patient time windows when constructing the routes. Synchronization constraints are not considered at all. Thus, we expected the solution it produces to incur arbitrarily large waiting times in double service patients. In any case, we thought that it may be a good scenario to check how effective the matheuristic is in improving solutions from such a naive constructive approach.

The entire constructive heuristic is depicted in Algorithm 1. At a glance, the constructive heuristic starts with all caregivers placed at the depot node. Then, it assigns the patients following the nearest neighborhood strategy by selecting the vehicle that arrives early in the patient site. The heuristic starts putting all patients in the list of pending nodes, sorted according to the value of their time-window ending (line 1). The solution is then initialized empty, with all the caregivers placed at the depot node (line 2). The heuristic proceeds by inserting them in the solution following the sorted list to (heuristically) minimize the visit tardiness. Iteratively, it selects the next patient from the list (line 3) and then selects a qualified caregiver to service the patient, following the nearest neighborhood strategy we already mentioned. This step is greedy and only takes into account the costs relative to the travel time of caregivers. More specifically, lines 5 and 6 apply for single service patients, and lines 9–12 apply for double services cases. The process repeats until all demands have been assigned to some caregiver (lines 7 and 13 for single and double service patients, respectively). Moreover, we highlight that we made a minor change in the original heuristic of Mankowska, Meisel and Bierwirth (2014).

For double service patients (line 12), we prevent v_2 from being the same as v_1, which can generate distinct–but correct–solutions than the values reported by the authors.

Algorithm 1: Initial routing heuristic.

```
1  sort ← sort the elements from C according by their increasing l_i values
2  init the routes for all caregivers with the depot node
3  for k ∈ sort do
4      if k ∈ C^s then
5          s ← the service type requested by k
6          select v ∈ V : a_vs = 1, with the earliest arrival time at k
7          append (k, s) in the route of v
8      else
9          s_1 ← the service type requested by k with the highest priority
10         s_2 ← the service type requested by k with the lowest priority
11         select v_1 ∈ V : a_{v_1 s_1} = 1, with the earliest arrival time at k
12         select v_2 ∈ V \ {v_1} : a_{v_2 s_2} = 1, with the earliest arrival time at k
13         append (k, s_1) in the route of v_1, and (k, s_2) in the route of v_2
14     end
15 end
```

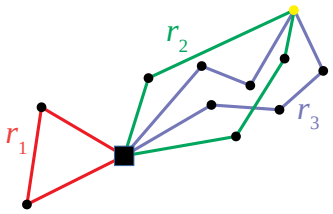
7.2 Decomposition types applied to the HHCRSP

The structural richness of the HHCRSP allows several types of decomposition schemes. One approach may perform the single-route optimization by relaxing the decision variables in a single caregiver's route. Another more interesting approach is to decompose n routes simultaneously, allowing inter-route optimization between n caregivers. Besides being more powerful than the single-route decomposition approach, the latter generates an $\mathcal{NP}$-hard subproblem–or very large neighborhood if one prefers local search terminology. Thus, only a few routes can be relaxed at once when using this route-based decomposition strategy to keep computational times short. One could also propose a decomposition based on the travel times between the locations, clustering nearby patients in groups and then generating the subproblems. An F&O matheuristic with such a decomposition scheme is similar to the *POPMUSIC* heuristic proposed by Taillard and Helsgaun (2019) for the traveling salesman problem. Unfortunately, these instance data-based decomposition may be ineffective when applied to solve the HHCRSP due to the presence of time windows, which tend to "perturb" the solution by including long arcs but fulfill the time windows constraints, and due to the dynamic time windows induced by the route inter-dependency constraints (see 4.2.2).

Due to the difficulty of solving larger subproblems, we propose a route-based decomposition scheme that generates subproblems for $n = 2$ distinct caregivers. By design, our approach only modifies the decision variables that define the routes themselves (the set x), and all the other variables employed in the model are kept unchanged by the matheuristic (sets t and z). Our motivation is threefold. First, fixing only the x variables allows the solver to optimize the routes of the two selected caregivers and adjust the visit times of patients of all the other routes. Without this, the solver becomes limited only to find solutions that match precisely the current visit times of the solution, which is very unlikely. Second, this approach allows the MIP solver to handle double service patients better, which inherently requires adjusting the visit times to keep the solution feasibility w.r.t. the route synchronization constraints in case the solution changes. Third, this approach enables the solver to reduce–or even eliminate–the visit tardiness of the solution. As a side effect, implementing this decomposition scheme requires keeping the complete model in memory, which can be prohibitive when solving large instances of the problem.

During the design phase of the algorithm, we verified that the systematic decomposition of all the pairs of distinct caregivers is often highly inefficient, either because most pairs reduce the solution cost very little or because decomposing the route of a specific caregiver leads to subproblems hard to solve to optimality. Since the systematic decomposition requires solving such a hard subproblem several times for each other vehicle in the problem, this efficiency issue becomes a critical concern. As an alternative to the systematic approach, we can devise an $n = 2$ decomposition procedure that selects these two caregivers, e.g., at random or using some heuristic criteria. The first approach has the strength of allowing escape from local optima, while the latter can exploit the solution's structure regarding both spatial and temporal characteristics of the routes. To illustrate these capabilities, please consider the solution in Figure 7.1. This example illustrates a problem with three routes (r_1, r_2, and r_3) and 11 patients. This example also contains a double service patient, represented by the node in yellow. By using the random selection of caregivers, any of the pairs $\{(r_1, r_2), (r_1, r_3), (r_2, r_3)\}$ can be selected for generating a subproblem. Supposing a guided decomposition that exploits the solution's temporal and spatial characteristics, a heuristic may select the two caregivers with most visits close in space or time. For example, such a guided decomposition would select the pair of routes (r_2, r_3).

Figure 7.1 – A routing solution to HHCRSP problem.



Source: the author.

Both random and guided decomposition approaches are formally defined in Algorithm 2. First, the algorithm makes all the random choices according to a uniform distribution, so there is no bias when choosing one decomposition or the other. This algorithm takes on its input the MIP decision variables x and t and the set of vehicles in the instance (line 1). Next, it sorts the decomposition scheme to be applied (line 2). If it selects the random decomposition approach (line 3), the algorithm then samples two distinct caregivers v_1 and v_2 (line 4). Otherwise, it has selected the guided approach, which requires a few more "pre-processing" steps. First, the algorithm sorts the decision variables t from the MIP in non-increasing order (line 6). Then it selects the two distinct caregivers v_1 and v_2, from the first half of this sorted list at random (line 7). This way, the guided approach tends to select vehicles with larger accumulated travel times. Despite myopic, this implementation matches our goal of devising a heuristic fast to compute. Finally, at line 7, the algorithm sets the original bounds of all decision variables regarding the two selected vehicles, which enables the solver to optimize a problem with $2 \cdot |\mathcal{C}^0| \cdot |\mathcal{C}^0| \cdot |\mathcal{S}|$ free decision variables x.

Algorithm 2: Decomposition schemes to the HHCRSP.

```
1 Algorithm decompose(x, t, V)
2     decomp ← choose randomly from ['random', 'guided']
3     if decomp = 'random' then
4         select v1 ≠ v2 from V using a uniform distribution
5     else
6         cand ← sort t variables in non-decreasing order of values, considering
            the routes of the current solution
7         split cand in half, and select v1 ≠ v2 in the index of t variables from
            the first half of list
8     end
9     restore the original bounds of the x variables for the caregivers v1 and v2
```

Algorithm 3 introduces the fix-and-optimize pseudocode for the HHCRSP, following the steps described in Section 7. The algorithm starts by computing an initial solution to the problem (lines 1 and 2), using the constructive heuristic introduced in Section 7.1.

Then, it enters a loop (lines 3–11) that comprises the following steps. Line 4 fixes the current solution by setting the bounds of decision variables x (line 4). If this is the first iteration, it uses the solution from the constructive heuristic (see line 1) or the current solution from the MIP solver (see line 9). A subproblem is then generated by applying the decompose procedure (line 5), and the resulting subproblem is optimized either to optimality or up to STL ("step time-limit") seconds (line 6). Line 7 extracts the value of the best solution found by the MIP solver, and a test checks if the new, improved solution was found (line 8). If that is the case, this new solution is then stored (line 9), and the value of the best solution found so far is updated accordingly (line 10). When the stopping criterion is met (line 12), the algorithm's main loop stops, and the matheuristic returns the best solution found (line 13).

Algorithm 3: Fix-and-optimize matheuristic for the HHCRSP.

```
1  s* ← create a initial solution using the Algorithm 1
2  z* ← cost of solution s*
3  repeat
4      add fixing constraints for the decision variables x according their values in
         s*
5      decompose(x, t, V)
6      s̄ ← solve the MIP model up to STL seconds
7      z̄ ← cost of solution s̄
8      if z̄ ⩽ z* then
9          s* ← s̄
10         z* ← z̄
11     end
12 until stop criteria met
13 return s*, z*
```

Despite the matheuristic capability of exploring random and guided decomposition schemes, we have a few concerns that too many route inter-dependencies may hurt the algorithm efficacy, requiring the subproblems to optimize (potentially) $n \gg 2$ vehicles to be of any effect. This issue can arise in test cases with many double service patients, e.g., when the instance size grows.

8 A BIASED RANDOM KEY GENETIC ALGORITHM FOR THE HHCRSP

MIP-based solution approaches can benefit from standard methods for modeling and solving the HHCRSP, but current techniques have some irremediable limitations. As we already mentioned, there is an issue regarding the weak linear relaxation of the model, which could be circumvented–on some level–by adding new valid inequalities to strengthen the formulation. Besides that, another major issue arises when solving larger instances sizes: the model grows quickly, to the point that it can only fit in server-class computers when solving instances with 400 patients or more. From a more practical point of view, it is unlikely that all major HHC providers will develop such computational infrastructure due to acquisition and operational costs, especially when the provider is a non-profit organization that operates with a tight budget (GOMES; RAMOS, 2019).

Alternatively, heuristics can be quite efficient w.r.t. holding the problem data in memory and allowing the developer to have much finer control of the algorithms rather when applying some mathematical optimization package. Unfortunately, some other issues arise when applying heuristics. According to results from the literature, methods based on modifying solutions such as local search can also be ineffective due to strong local optima attractors in the optimization landscape (GENDREAU; POTVIN et al., 2010). The route inter-dependency constraints of the HHCRSP tend to intensify such issues. Furthermore, even optimizing the route of a single caregiver route can be quite challenging due to the multiple-visits patients in the route (HADDADENE; LABADIE; PRODHON, 2016). More successful approaches, e.g. Liu, Tao and Xie (2019), were tested only in small instances with less than 100 patients, and they require keeping a more sophisticated solution representation that can be cumbersome when solving large instances of the problem. After all, the HHCRSP belongs to the $\mathcal{NP}$-hard complexity class, and the solution methods tend to be more computationally intensive as one tries to develop more and more efficient solutions.

Considering these observations, we think that methods based on the solution space exploration may be more effective than exploiting the problem structures, e.g., local search. This way, we propose solving the HHCRSP by using a genetic algorithm (GA) known as the Biased Random Key Genetic Algorithm (for short, BRKGA) (GONÇALVES; RESENDE, 2011). BRKGA has already been applied to a variety of combinatorial optimization problems that hold a permutational structure in their solutions. Some application examples are the container loading problem, the parallel machines scheduling problem, and project

scheduling problems (GONÇALVES; RESENDE; MENDES, 2011; GONÇALVES; RESENDE, 2012; CHAVES et al., 2016). To the best of our knowledge, we are the first attempting to solve an HHC problem with route inter-dependency constraints using a BRKGA based algorithm (DREXL, 2012; CISSÉ et al., 2017; FIKAR; HIRSCH, 2017; GRIECO; UTLEY; CROWE, 2020).

8.1 Representation of individuals with vectors of random keys

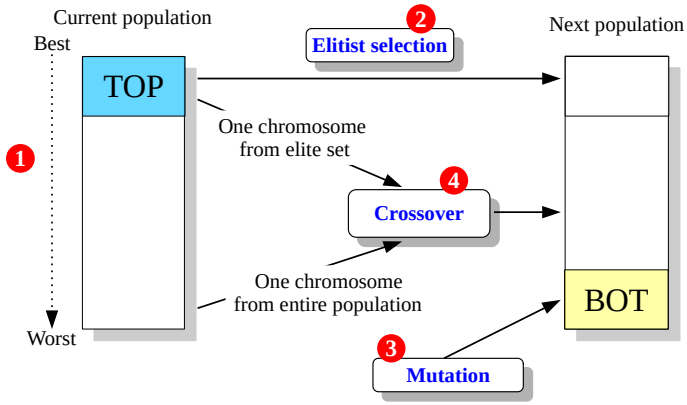
The BRKGA inherits the representation scheme introduced by Bean (1994) for the Random Key Genetic Algorithm (RKGA). In this approach, the chromosomes indirectly represent solutions encoded in terms of randomly generated real numbers from the interval $[0, 1)$. This way, it is possible to architecture the evolutionary framework in terms of this general representation, devising a completely generic algorithm while leaving the task of mapping the chromosome to a solution of the underlying problem to a problem-specific crossover. Typically, BRKGA-based algorithms do not use any initial solution. Thus the first generation generates starts by sampling the initial population, which comprises randomly generated individuals. In terms of the taxonomy of GAs, the indirect representation of a solution through random-keys is called *genotype*, while the explicit solution derived from the random-keys is called *phenotype* (GONÇALVES; RESENDE; MENDES, 2011).

As we already mentioned, it is easy to represent permutational problems like the TSP using the indirect solution representation by random keys. In this example, the keys are used to sort the cities of the problem, defining the visiting order of the salesman (GONÇALVES; RESENDE; MENDES, 2011). In other problems, it is possible to represent binary variables through the random keys, so the discrete value can be deduced by comparing the key with some threshold value, e.g., the variable represented by a key is set to true when the key's value is $\geqslant 0.5$, otherwise false (GONÇALVES; WÄSCHER, 2020). Due to the feasibility issues of this threshold method, it is also possible to use the random keys to guide some constructive heuristic instead of extracting the solution "directly" (GONÇALVES; RESENDE, 2012).

8.2 Mutation, elitism, and crossover in the BRKGA

In BRKGA, the core of the evolutionary framework comprises a tailored mating and mutation procedures, as Figure 8.1 illustrates. Let p be the population size of the genetic algorithm. The GA starts by sorting the individuals of the current population according to their fitness (1). Then the elite individuals from the current population are copied to the next generation (2), and the remaining population is generated by following two different strategies. The first strategy aims to keep the diversity of the new population by inserting p_m new random mutants in the next generation, as indicated by the BOT partition of Figure 8.1 (3). The second strategy consists of mating one member from the elite set with another member from the non-elite set to produce offspring with the so-called parameterized uniform crossover operator (4). First, the crossover algorithm samples the parents at random. Next, each allele from the offspring is inherited elite parent, with a probability of ρ_e, or from the non-elite parent, with a probability of $1 - \rho_e$. This way, the parameter ρ_e allows adjusting the bias to inherit alleles from the elite parent. The mating repeats until generating $p - p_e - p_m$ new individuals into the next population. Finally, the next population's individuals take the current population's place, and the evolutionary process repeats until achieving some stop criteria (GONÇALVES; RESENDE; MENDES, 2011).

Figure 8.1 – Evolutionary framework of the BRKGA.



Source: the author.

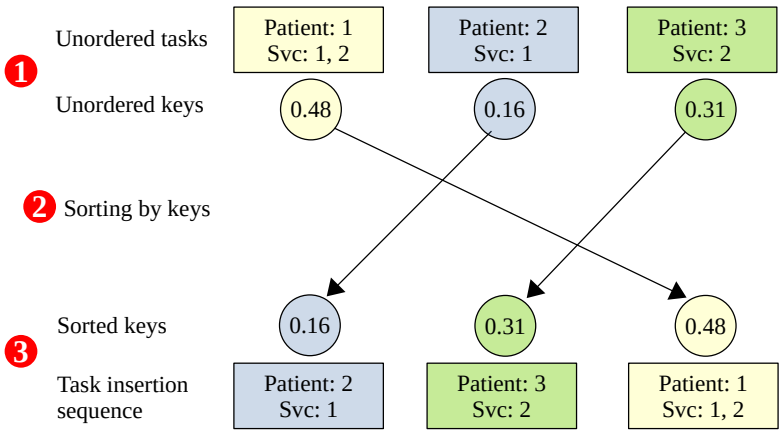
The BRKGA has some strengths that help to manage some complicating aspects of the HHCRSP. It uses a parameterized uniform crossover, dismissing the necessity of repair operators by delegating feasibility issues to the objective function (GONÇALVES; RESENDE, 2012). This way, a well-designed decoder is required to prevent the evolutionary process from getting stuck due to infeasibility. The choice of chromosome representations

with random keys and the constant generation of mutants enables the so-called exploration of the solution space, while the crossover operator and the decoder enable the exploitation of the structural properties of the solutions (EIBEN; SMITH et al., 2003). Such characteristics also relieve the necessity to employ local search, which can be very time-consuming and causes a minor impact on the solution due to myopic behavior in front of the routes inter-dependency induced by double service patients (DREXL, 2012; MANKOWSKA; MEISEL; BIERWIRTH, 2014).

8.3 A greedy constructive decoder to the HHCRSP

The proposed BRKGA evolves the sequence in which the patients are inserted into the solution. For this reason, the individuals have a chromosome with $|\mathcal{C}|$ random keys. For each individual, a sorting algorithm arranges its keys in non-decreasing values, and this sorted key sequence defines the order in which the rest of the decoding process will process the patients. The example of Figure 8.2 illustrates the first part of the decoding of an instance with three patients. The algorithm starts pairing the list of tasks (patient plus the requested service types) with the chromosome (1), then it sorts the task list according to the non-decreasing key values (2), reaching the task sequence depicted in the task insertion sequence (3). The second part of the decoding uses a best-insertion greedy heuristic to assign the patients from the task insertion sequence the caregivers of the problem.

Figure 8.2 – Decoding example for a instance with three patients.



Source: the author.

Algorithm 4 details the implementation of the proposed decoder. From lines 2 and 3, the decoder performs the same steps depicted in the example of Figure 8.2. The decoder takes each task from the sorted task sequence (line 4), and test all the possible caregiver

assignments to such task (lines 7–25), taking into consideration one caregiver to single service patients (lines 6–13) and two caregivers to double service patients (lines 6–25). The algorithm updates the current solution by inserting the task node into the routes of the selected vehicles (lines 26–29). The loop step finishes by updating the solution cost (line 30), considering the same objective components from the MIP model (5.1). Let $n = |\mathcal{V}|$ and $m = |\mathcal{C}|$. In the worst case, the decoder algorithm has a complexity of $O(mn^2)$.

Algorithm 4: Greedy constructive heuristic embedded in the decoder.

```
   Input: A chromosome ch.
   Output: Cost of the computed constructive solution.
 1 s ← solution with all vehicles at depot and ready to leave at time 0
 2 tasks ← list of patients and their associated service types requests
 3 tasks ← sort tasks by using the keys from the chromosome
 4 for i ← 1 to |tasks| do
 5     c* ← ∞
       // Gets the service type of higher priority
 6     s1 ← high_priority_skill(tasks[i])
 7     for v1 ← 1 to |V| with a_{v1 s1} = 1 do
 8         if node(tasks[i]) ∈ C^s then
 9             c̄ ← find_insertion_cost(s, v1, null, tasks[i])
10             if c̄ < c* then
11                 v1* ← v1
12                 c* ← c̄
13             end
14         else
               // Gets the service type of lower priority
15             s2 ← low_priority_skills(tasks[i])
16             for v2 ← 1 to |V| \ {v1} with a_{v2 s2} = 1 do
17                 c̄ ← find_insertion_cost(s, v1, v2, tasks[i])
18                 if c̄ < c* then
19                     v1* ← v1
20                     v2* ← v2
21                     c* ← c̄
22                 end
23             end
24         end
25     end
26     update_route(s, v1*, node(tasks[i]), s1)
27     if node(tasks[i]) ∉ C^s then
28         update_route(s, v2*, node(tasks[i]), skills(tasks[i], 2))
29     end
30     cost(s) ← update_cost(s, c*)
31 end
32 return cost(s)
```

Due to the double services and soft time windows, the cost calculation in find_insertion_cost is not straightforward. Algorithm 5 details how such a function can be implemented, considering all the constraints of the problem. The function calculates the arrival time of each caregiver into the patient, taking the greatest value between the arrival time and the starting of the time window (lines 3–4 for single service patients, and

lines 9–10 for double service patients, respectively.) The function leave_time gives access to when a vehicle departs from the node of its route. In simultaneous double services, the start times are adjusted to the caregiver who arrives later in the patient (line 13). On double services with precedence, the service start time of the second caregiver is adjusted according to the minimum separation time (line 17). If the maximum separation time is violated, then a delay is inserted into the vehicle's start time responsible to the service type with higher priority (line 18). Violations of the time windows are computed in lines 5, 14, and 19–20, and the maximum tardiness is computed in lines 5, 15, and 22. Finally, the function takes the updated solution quality indicators and returns the new cost (lines 22–25), considering the weights λ_1, λ_2, and λ_3 of (5.1). Assuming that all operations inside find_insertion_cost run at constant time, the function has then theoretical complexity of $O(1)$. In practice, the costs involved in querying the distances matrix in lines 3 and 9 can be significant due to the (probable) cache miss triggered in current computer architectures.

Algorithm 5: Implementation of find_insertion_cost.

Input: s, v_1, v_2, $task$.
Output: The new solution cost after inserting the task into the solution.

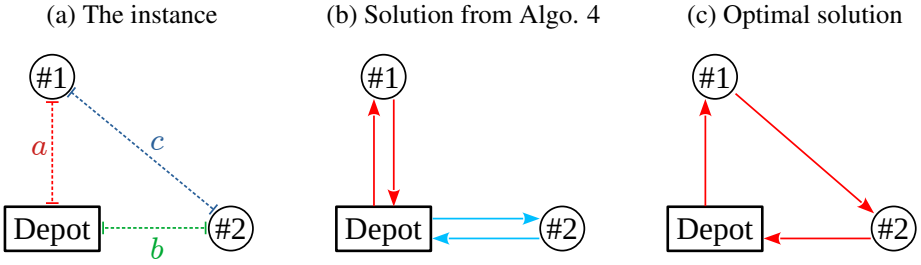
1 $i \leftarrow$ node($task$)
2 $s_1 \leftarrow$ high_priority_skill($task$)
3 $dist \leftarrow$ distance from the last node of v_1 route to i
4 $arrival_1 \leftarrow \max\{e_i, \text{leave\_time}(s, v_1) + dist\}$
5 $tard \leftarrow \max\{0, arrival_1 - l_i\}$
6 $curr\_tmax \leftarrow tard$
7 **if** $i \notin \mathcal{C}^{\text{s}}$ **then**
8 $\quad s_2 \leftarrow$ low_priority_skills($task, 2$)
9 $\quad dist_2 \leftarrow$ distance from the last node of v_2 route to i
10 $\quad arrival_2 \leftarrow \max\{e_i, \text{leave\_time}(s, v_2) + dist_2\}$
11 $\quad dist \leftarrow dist + dist_2$
12 $\quad$ **if** $i \in \mathcal{C}^{\text{sim}}$ **then**
13 $\quad\quad svc\_start \leftarrow \max\{arrival_1, arrival_2\}$
14 $\quad\quad tard \leftarrow 2 \cdot \max\{0, svc\_start - l_i\}$
15 $\quad\quad curr\_tmax \leftarrow tard$
16 $\quad$ **else**
17 $\quad\quad svc\_start_{s_2} \leftarrow \max\{arrival_1 + \delta_i^{\min}, arrival_2\}$
18 $\quad\quad svc\_start_{s_1} \leftarrow arrival_1 + \max\{0, arrival_2 - arrival_1 - \delta_i^{\max}\}$
19 $\quad\quad tard_1 \leftarrow \max\{0, arrival_1 - l_i\}$
20 $\quad\quad tard_2 \leftarrow \max\{0, arrival_2 - l_i\}$
21 $\quad\quad tard \leftarrow tard_1 + tard_2$
22 $\quad\quad curr\_tmax \leftarrow \max\{tard_1, tard_2\}$
23 $\quad$ **end**
24 **end**
25 $dist \leftarrow dist +$ dist(s)
26 $tard \leftarrow tard +$ tard(s)
27 $curr\_tmax \leftarrow \max\{curr\_tmax, \text{tmax}(s)\}$
28 **return** $\lambda_1 \cdot dist + \lambda_2 \cdot tard + \lambda_3 \cdot curr\_tmax$

It is noteworthy that an optimal solution of the problem is not always reachable by the Algorithm 4. Proposition 8.3.1 introduces an example of such limitation, based on the triangular inequality for a toy instance in the metric space.

Proposition 8.3.1. *There are solutions to some HHCRSP instances that cannot be reached with Algorithm 4.*

Proof. Suppose an arbitrary instance of the HHC problem, consisting of one depot, two patients and two caregivers. Both vehicles can serve either of the patients, the time windows are sufficient large, and the processing time is a constant. The placement of the depot and the two patients are according the Figure 8.3a. To this instance, the greedy best-insertion heuristic assigns patient 1 to one of the caregivers, holding a travel distance of a units. One option is to serve patient 2 by using the same caregiver, increasing the solution cost by c units of distance. The other option uses the other caregiver and increases the solution cost by b units of distance. Supposing $b < c$, the heuristic chooses the second option, and the complete solution of Figure 8.3b holds $2a + 2b$ units of distance. However, a solution exists with the cost of $a + b + c \leqslant 2a + 2b$ due to the triangular inequality $c \leqslant a + b$ by using just one of the caregivers. Such solution can not be achieved with the greedy best-insertion heuristic. In this example, this is also the optimal solution of this toy instance. ■

Figure 8.3 – A generic instance to the HHC problem on the metric space.



Source: the author.

8.4 Extensions to the BRKGA

With regard to the original algorithm introduced by Gonçalves and Resende (2011), most of the exploration capability of a BRKGA is based on the frequent generation of mutant individuals, which allows the genetic algorithm to keep the diversity as the population evolves. As a counterpart, the parameterized uniform crossover operator acts as the exploitation mechanism, trying to combine promising characteristics of the

parents in the mating process. Since the elite individuals have been curated through several generations and survived so far, the crossover also tends to benefit the "genetic information" from the elite parent, increasing the offspring's opportunity to inherit some of the properties of a high-quality solution.

Despite the simplicity, the crossover operator can be so aggressive that the resulting offspring is just a scrambled combination of alleles from both parents. For example, consider an application of the BRKGA for solving the TSP. In this example, the presence of a particular subroute may be the property that makes a high-quality solution. Since the crossover algorithm does not know about this, such an "high-quality substructure" of the elite parent will be (probably) lost when generating the offspring. One may think this issue might be circumvented through a stronger bias to inherit alleles from the elite parent, but it is not hard to imagine a scenario in which this approach does not work, e.g., when such a substructure appears in the non-elite parent.

With these limitations in mind, Andrade et al. (2021) proposed some extensions to improve BRKGA's exploitation capability, aiming to improve the algorithm results and convergence. Similar to the `brkgaAPI` open-source library developed by Toso and Resende (2015), Andrade et al. (2021) also published an open-source library that implements these new components. Both libraries share several similarities, but the first is written in C++ while the latter is supported by several other languages, including C++ (`brkga_mp_ipr_cpp`), Julia, and Python. From our experience, the same decoder of Algorithm 4 can be used in both libraries, of course requiring some adaptation. Another important feature is that both libraries support multi-core parallel processing via OpenMP directives, allowing almost linear speedup in most applications.

8.4.1 Parallel independent populations and the island model

The island model is the first additional component for the BRKGA, introduced by Toso and Resende (2015) and natively supported by `brkgaAPI` and `brkga_mp_ipr_cpp`. Sometimes also called a multi-population model, the island model consists of evolving multiple independent islands simultaneously. Each island has its independent population and naturally can achieve a local minimum–ideally a distinct one. Periodically after a few generations, it triggers an immigration mechanism that allows exploiting the problem's solution space by moving some elite individuals from one island to the other. More precisely, the immigration moves m elite individuals from island i to island $\min\{i+1, 1\}$, and this

process repeats for $i = 1, 2, \ldots, k$ in the presence of k islands. Simple criteria can trigger this mechanism, e.g., periodically after a fixed number of generations (TOSO; RESENDE, 2015).

The island model amplifies the BRKGA exploration capability by evolving several distinct populations, and the immigration mechanism intensifies the search. Furthermore, the island model has two major caveats. First, it significantly increases the amount of processing done during the search. Typically the amount of works scales linearly as the number of islands increase. Secondly, the developer has to be careful not to trigger immigration too frequently. Otherwise, all populations will converge early to the same local optima.

8.4.2 Multi-parent mating

As mentioned in Section 8, the standard crossover operator for BRKGA-based algorithms performs the so-called parameterized uniform crossover involving two parents from the population. Further researchers identified simple modifications that could improve the offspring's fitness by tweaking the mating process a little. Computational experiments presented in Lucena et al. (2014) reported that multi-parent mating presented consistently better results than the standard two-parent approach.

In multi-parent BRKGA, the meta-heuristic selects a total of $\pi_t \geqslant 2$ parents each mating, with π_e of these coming from the elite set and the others $\pi_t - \pi_e$ coming from the non-elite set. At this point, another major difference to the "standard" crossover algorithm arises. Instead of specifying the *explicit bias* for selecting alleles from the elite parents, the multi-parent BRKGA requires calculating the bias for each parent as follows.

Step 1. Definition rank. Put all sampled parents into a list, and sort them according to their *fitness*. For a minimization problem, this means that individuals are sorted in non-increasing order of fitness. After sorting, we say that the parent in the first index of the list has the rank 1, the second has rank 2, and so forth. This way, the list indices span from 1 to π_t.

Step 2. Calculation of the weights. Apply some non-decreasing bias function to calculate the weight of each parent explicitly, according to their rank. We say that the parent with rank r has its weight calculated as $\Phi(r)$, where Φ is the bias function employed.

Step 3. Normalization of the weights. Calculate the sum $T = \sum_{r=1}^{\pi_t} \Phi(r)$ and normalize all of individuals's weights as $\frac{\Phi(r)}{T}$, for $r = 1, 2, \ldots, \pi_t$.

In Step 2, Andrade et al. (2021) suggested using one of the non-decreasing bias functions of Bresina (1996) to compute such weights. The `brkga_mp_ipr_cpp` library already supports several of them, including other simple variations as the quadratic $\Phi(r) = r^{-2}$ and cubic $\Phi(r) = r^{-3}$ functions. With all parents' weights defined, the multi-parent crossover continues like the original algorithm of Gonçalves and Resende (2011). According to their normalized weights, the offspring inherits each of its alleles from any of the selected parents. In the end, typically, the offspring inherit more alleles from parents with lower ranks.

8.4.3 Implicit path-relinking in random-keys space

Path-relinking is an intensification heuristic that explores the intermediate solutions between solutions, usually referred to as *base* and a *guide*. This heuristic works by incrementally introducing changes in the base solution according to guide one, and then it evaluates if such an intermediate solution improves the incumbent value. The search finishes when the last intermediate solution generated is equal to the guide one (GLOVER, 1997).

The direction of this exploration matters, and it was the subject of study by Resende and Ribeiro (2016). The authors study several variations of such a heuristic. The *forward path-relinking* explores the path from the base to the guide solution. As the name suggests, the *backward path-relinking* then explores the solution space from the guide to the "direction" of the base solution. Resende and Ribeiro (2016) also propose the *hybrid path-relinking*, which combines both exploration strategies. According to their experiments, this hybrid intensification approach often finds a better solution than the others.

An important aspect is that path-relinking tends to be ineffective when the base and guide solutions are too similar. With that in mind, the authors also propose applying some distance function to measure how similar the base solution is to the guide one. Ideally, such distance measuring should be faster than heuristic, saving processing time running the path-relinking. Unfortunately, such a distance function is problem-specific, as so the implementation of path-relinking is. It is also difficult to generalize an implementation of the heuristic even to a family of similar problems (ANDRADE et al., 2021).

Nevertheless, developing some general-purpose "path-relinking library" is also very unlikely due to the abstraction costs involved. With all these difficulties in mind, Andrade

et al. (2021) then perceived that such a generic path-relinking could indirectly explore the solution space of a generic problem using the same strategy of the BRKGA meta-heuristic. The same idea applies to devise generic functions to evaluate the distance between two chromosomes. This idea originated what the authors call implicit path-relinking (IPR).

The IPR operates differently according to how the solutions are encoded. In some cases, the keys encode binary variables, as we already mentioned in Section 8.1. In this case, Andrade et al. (2021) propose the *direct* IPR, which replaces the alleles from one individual to the other directly, as depicted in Figure 8.4a. A naive approach of direct IPR changes a single key at the time when generating an intermediate individual. Thus for vector with n keys, the heuristic calls the decoder n^2 times when applying a hybrid IPR, which can be prohibitive depending upon the complexity of the decoder algorithm. Alternatively, the authors suggest replacing *blocks of keys* in each step of the heuristic, effectively reducing the number of calls to the decoder. This approach can also benefit problems that the evolutionary process induces some substructure in the random-keys vectors, which allows preserving what we call "high-quality substructure" in Section 8.4. Ultimately, one can tweak the time spent in diversification and intensification in the IPR by changing the block size.

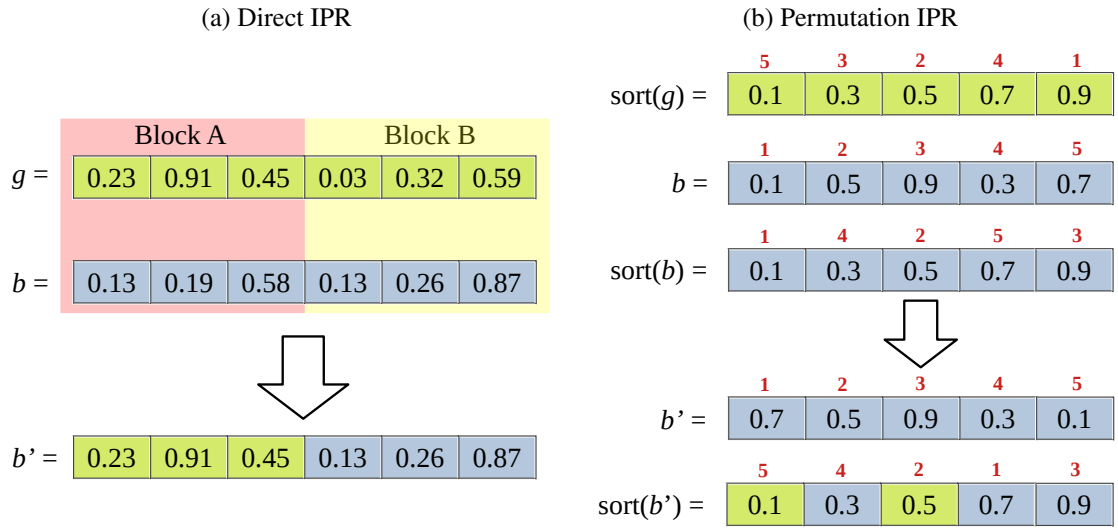
In routing and sequencing problems–like the HHCRSP, the random keys establish a permutation of the solution elements. For a vehicle routing problem, e.g., each client is associated with a chromosome position; thus, the visit order is obtained by sorting the chromosome (RUIZ et al., 2019). For this type of solution encoding, the IPR may not be enough to change the solution encoded in the intermediate chromosome effectively. For such applications, Andrade et al. (2021) proposed the permutation IPR, which uses the guide chromosome to induce permutations in the base chromosome, similar to the example of Figure 8.4b. As in the direct IPR, the block of keys to speed up the search also applies to the permutation IPR. Furthermore, we also focus our computational experiments on the permutation IPR because this is the type of IPR that better fits when solving the HHCRSP.

Despite being illustrative, we think that Figure 8.4 still requires some explanation. Figure 8.4a illustrates the direct IPR between the guide individual g and the base individual b. This example considers a block of keys with size three. As the chromosomes have six keys, the algorithm identifies two blocks, A and B. In one step of the heuristic, block A of the base individual b is replaced by the keys from block A of guide individual g, generating the intermediate individual b'. This individual is submitted to the decoder algorithm, and it is inserted in the current population if it improves the incumbent. In `brkga_mp_ipr`,

the process repeats but inverting the roles of base and guide individuals. The direct IPR finishes when this process is applied to all blocks of keys.

As Figure 8.4b illustrates, the permutation IPR is a little more complex. Let the guide individual $g = \{0.9, 0.5, 0.3, 0.7, 0.1\}$. After sorting this individual, we get the array identified as $\text{sort}(g)$. Note the presence of the "original" key indices highlighted in red above the sorted vector. We repeat the process for the base individual b, highlighting the sorted sequence of keys as $\text{sort}(b)$. Here the permutation IPR starts. As the reader can note, the "original" index of the sorted vectors does not match: for $\text{sort}(g)$, the first position has the index 5, and $\text{sort}(b)$ has the index 1. To force $\text{sort}(b)$ to have the same "original" index of $\text{sort}(g)$ in the first position of the sorted sequence, the permutation IPR swaps the keys with indices 5 and 1 of b, generating the intermediate individual b'. After sorting it, we can see in $\text{sort}(b')$ that the IPR induced the first index of the intermediate individual to have the same "original" index of the sorted vector of g. After this step, the heuristic proceeds as for the direct IPR. This example does not take into account the blocks of keys. On its presence, the algorithm then swaps all the keys from the same block at once. The rest of the algorithm stills unchanged.

Figure 8.4 – Examples of direct and permutation IPR.



Source: the author.

The `brkga_mp_ipr_cpp` already supports implicit path-relinking procedures. If the developer uses a single island model, the IPR takes in place by selecting individuals from the elite set of the population. If multiple islands are available, the IPR is then triggered between individuals from distinct islands, taking the base individual from one island and the guide individual from another one, similar to how the immigration mechanism

works. This choice can be between the top individuals from each island or a random individual from the island's elite sets. As a whole, the intensification capability of the IPR can be greatly enhanced when the heuristic operates integrated with the multiple islands model.

9 INSTANCE DATASETS

This chapter presents some of the available instance datasets for the single-day HHCRSP. Section 9.1 introduces whose we have access, highlighting in which paper they have been introduced and the features considered by the authors. Section 9.2 introduces the primary dataset we approach in our study. Section 9.3 introduces a new realistic dataset that better represents the problem arriving in Porto Alegre, considering the feature set proposed by Mankowska, Meisel and Bierwirth (2014), but adapting the data generated according to the real problem.

9.1 Overview of available datasets

Despite scant, a few synthetic benchmark datasets are available from the literature on the single-day home health care problem. However, more important than the scarcity is that their distinct features make them not cross-compatible, requiring many adaptions when applying a method designed for one dataset to solve another. Table 9.1 lists all the instance datasets for the single-period HHCP we have access to, indicating the publications that introduced them, the data source, the access policy, and the characteristics present in the instance files.

These were generated randomly, with a few inspired in real instances provided by a home health care provider (RASMUSSEN et al., 2012; HIERMANN et al., 2015). Regarding the availability, some datasets are publicly available on the research group's webpages (MANKOWSKA; MEISEL; BIERWIRTH, 2014; HIERMANN et al., 2015; BRAEKERS et al., 2016). Some others are available upon email requests (BREDSTRÖM; RÖNNQVIST, 2008; RASMUSSEN et al., 2012). Fikar and Hirsch (2017) already listed some of these datasets, but only a few of them are still accessible through the hyperlinks indicated in their survey (MANKOWSKA; MEISEL; BIERWIRTH, 2014; HIERMANN et al., 2015; BRAEKERS et al., 2016).

Most datasets consider staff members with unequal qualification levels, frequently discussed as heterogeneous staff. Similarly, most datasets consider euclidean travel times, except for the dataset that used realistic distances. For example, Rasmussen et al. (2012) keep the patients and distances of real instances from the HHC provider while replacing the other data with random information. On the other hand, Braekers et al. (2016) used random points from a geographical area and computed the travel times using the OpenStreetMaps

Table 9.1 – Instance datasets available and their features.

Publication	Source	Access	Features
Bredström and Rönnqvist (2008)	Random	Upon request	Homogeneous staff; Euclidean dist. (rounded); Service duration; Patient time-window; Caregiver time-window; Time-window tightness; Simultaneous att. constr; Soft patient-caregiver pref;
Rasmussen et al. (2012)	Random + HHC Provider	Upon request	Heterogeneous staff; Realistic dist; Service duration; Patient time-window; Caregiver time-window; Simult. att. constr; Precedence constr; Soft patient-caregiver pref;
Mankowska, Meisel and Bierwirth (2014)	Random	Public	Heterogeneous staff; Euclidean dist; Service duration; Patient time-window; Simultaneous att. constr; Precedence constr;
Hiermann et al. (2015)	Random + HHC Provider	Public (Only the random)	Heterogeneous staff; Euclidean dist; Multiple depots; Multi-modal vehicles; Service duration; Soft-limited patient TW; Caregiver time-window; Hard patient-caregiver pref; Pre-allocated tasks;
Braekers et al. (2016)	Random	Public	Heterogeneous staff; Realistic dist; Multi-depot, distinct source and sink nodes; Max. staff working time; Caregiver time-window; Multi-modal vehicles; Service duration; Soft-limited patient TW; Soft patient-caregiver pref;

data. All datasets consider service duration and some patient time window. Mankowska, Meisel and Bierwirth (2014) allowed unbounded tardiness, while Hiermann et al. (2015) and Braekers et al. (2016) used the soft-limited TW approach (c.f. Subsection 4.2.1). Bredström and Rönnqvist (2008) devised their dataset specifically to study the impact of the time-window tightness on solution quality and the difficulty of finding the optimal solution. Most authors approached a single-depot problem, with the notable exceptions of Hiermann et al. (2015) and Braekers et al. (2016) that approached a multi-depot problem.

Almost all the datasets consider time windows for the operations of the caregivers (BREDSTRÖM; RÖNNQVIST, 2008; RASMUSSEN et al., 2012; HIERMANN et al., 2015; BRAEKERS et al., 2016). More than half of the datasets consider either a soft preference or a hard constraint for the caregiver's service assignment (MANKOWSKA; MEISEL; BIERWIRTH, 2014; HIERMANN et al., 2015; BRAEKERS et al., 2016). Hiermann et al. (2015) and Braekers et al. (2016) datasets consider multi-modal transportation to analyze the tradeoffs between travel costs and travel times.

Route inter-dependency constraints are considered in three datasets, either as simultaneous attendance or in the form of operations precedence or non-overlapping constraints. Rasmussen et al. (2012) considered four variations of operations precedence constraints. Bredström and Rönnqvist (2008) and Mankowska, Meisel and Bierwirth (2014) approached only cases with simultaneous visits or visits with precedence constraints. Hiermann et al. (2015) dataset also considered some pre-allocated patients to the staff, aiming to values the bonding between caregiver and patient often discussed as continuity of care (FIKAR; HIRSCH, 2017; CISSÉ et al., 2017; GRIECO; UTLEY; CROWE, 2020).

To the problem of our interest, we identify three datasets that are compatible in different "degrees". Without surprise, we take the Mankowska, Meisel and Bierwirth (2014) instances as our standard dataset. We could also approach the Bredström and Rönnqvist (2008) dataset by augmenting the solution methods to deal with the additional requirements– patient-caregiver preferences, caregiver TW. Finally, the Rasmussen et al. (2012) dataset also requires such amendments, but we need to handle also the missing patient visits and the other three types of route inter-dependency constraints studied by the authors. Experimenting with these other datasets also requires testing of the adapted algorithms and several months to finish all the computational experiments. Furthermore, we think it would be best to limit the scope of our research by testing only the Mankowska, Meisel and Bierwirth (2014) dataset to better keep the comprehensibility of the results.

9.2 Mankowska, Meisel and Bierwirth (2014) dataset

Mankowska, Meisel and Bierwirth (2014) dataset comprises seven smaller subsets, each of which comprises ten synthetic instances.. All instances of the same subset have the same number of caregivers and patients, as depicted in Table 9.2. The columns identify the instance family, the total number of patients ($|\mathcal{C}|$), the number of single ($|\mathcal{C}^{\mathrm{s}}|$) and double service patients ($|\mathcal{C}^{\mathrm{d}}|$), respectively, and the number of caregivers ($|\mathcal{V}|$), respectively. The set of service types is fixed $\mathcal{S} = \{1, 2, 3, 4, 5, 6\}$ for the entire dataset. For writing sake, consider $\mathcal{S}_a = \{1, 2, 3\}$ and $\mathcal{S}_b = \{4, 5, 6\}$ the subsets generated when we split $\mathcal{S}$ in the half.

$|\mathcal{C}^{\mathrm{d}}|$ is calculated to be 30% of the total number of patients ($|\mathcal{C}|$), so 15% of $|\mathcal{C}|$ requires simultaneous visits, and the other 15% requires precedence constraints. The rest of the patients (70% of $|\mathcal{C}|$) require single visits. Remark that single service patients require just one service type, and double service patients require exactly two. This way, the service requested by a single service patient is generated uniformly by sampling $\mathcal{S}$. The two service types requested by a double service patient are sampled from the subsets $\mathcal{S}_a$ and $\mathcal{S}_b$, respectively. Mankowska, Meisel and Bierwirth (2014) original description does not include this sampling from distinct service types subsets, but it seems to be the case after we analyzed their dataset.

The caregiver skills are also set at random. Half of the caregivers can perform up to 3 service types from $\mathcal{S}_a$, and the other half can perform up to 3 service types from $\mathcal{S}_b$. Mankowska, Meisel and Bierwirth (2014) guarantee at least one caregiver capable of performing each service type. Furthermore, all the points (the depot and patient homes) were generated within a 100×100 region, and the travel distances are Euclidean. The authors also consider a fixed travel speed of one distance unit per time unit. Each patient has a two-hour time window drawn from a time horizon of 10h.

The values of the visit duration p_{is} are generated uniformly from $[10, 20]$, for patient $i \in \mathcal{C}$ and service type $s \in \mathcal{S}$. Finally, for double service patients with precedence constraints, the minimal separation time $\delta_i^{\min}$ is generated uniformly from $[0, 60]$, and $\delta_i^{\max}$ is also generated uniformly from $[0, 60] + \delta_i^{\min}$.

Table 9.2 – Characteristics of the instance subsets, indicating the subset name, the total number of patients, the number of single service patients, the number of double service patients, and the number of caregivers.

Instance subset	$\lvert\mathcal{C}\rvert$	$\lvert\mathcal{C}^{\mathrm{s}}\rvert$	$\lvert\mathcal{C}^{\mathrm{d}}\rvert$	$\lvert\mathcal{V}\rvert$
A	10	7	3	3
B	25	17	8	5
C	50	35	15	10
D	75	52	23	15
E	100	70	30	20
F	200	140	60	30
G	300	200	100	40
Instances available = 70				

Source: the author.

9.3 Introducing a new realistic instance dataset

Our primary criticism with Mankowska, Meisel and Bierwirth (2014) is the lack of a "structure" in their dataset. For example, they consider a simple Euclidean model when sampling points and calculating the travel distances, which unlikely represents the real problem. Furthermore, the sampling strategy for caregiver's qualifications and patient service requests allows very unbalanced cases in which only a few caregivers qualify to perform a heavily requested service type. In practice, these extreme cases are unlikely due to how an HHC system is structured, i.e., the eligibility of a new patient to be included in such systems is subject to the availability of qualified caregivers (HULSHOF et al., 2012). For these reasons, we introduce a new realistic dataset that tries to represent a single-day HHCRSP better.

Our dataset comprises random instances similar to the literature, but we employ additional tools to model better the real HHC problem arriving in Porto Alegre. For that purpose, we developed our instance generator on the top of the `ovig` open-source library, introduced by Sartori and Buriol (2020), to generate realistic instances from VRP-related problems. The `ovig` library combines data and state-of-art algorithms to generate realistic instances w.r.t node locations and travel times. As a data source, `ovig` uses the OpenAddresses database (OPENADDRESSES, 2021) to sample realistic addresses. This open database includes factual address information for several cities of the entire globe, including Porto Alegre. Specifically to Brazil, the database contains all the domiciles covered by the latest census of IBGE (2010). Using such data has e few additional benefits. First, it prevents the generation of invalid points, e.g., within lakes or larger

private properties, which is critical when generating data for Porto Alegre. Secondly, OpenAddress (OA) data has additional attributes that we can further explore during the generation, e.g., to select addresses with residences rather than large condominiums.

From an algorithmic point of view, the `ovig` library embeds the Open Source Routing Machine (OSRM) library for calculating realistic travel times (in minutes) between the sampled locations (LUXEN; VETTER, 2011), using the data from the OpenStreetMaps. This usage scenario applies an all-pair shortest path algorithm, considering the characteristics of the urban environment, like speed limits and the number of stops and turns. Thus, the travel times computed by the tool tend to be realistic. Naturally, we could replace the point sampling with data input in the future so that `ovig` can compute the travel times of real problems.

9.3.1 Placing the depot and patient nodes

`Ovig` requires a few parameters when generating geographical locations, mostly to set how the generated nodes should be spread (UCHOA et al., 2017). As illustrated in Figure 9.1, the patient locations can be generated in some ways, as we list below.

1. Node-random. It uses a simple random strategy to place all the patients randomly in any of the addresses from the OA database. It tends to generate instances hard to solve to optimality, but this distribution does not represent a real HHC problem very well;

2. Node-clustered. This strategy generates clusters of nodes and operates in two steps. First, it samples n initial patients as "seeds" for the clusters. Secondly, it places the other patients within these clusters, using the haversine distance as an approximation of the real distances between the candidate point to the cluster seed. This distribution seems to represent better HHC problems involving distinct providers, in which each provider is responsible for service patients from some specific geographical location;

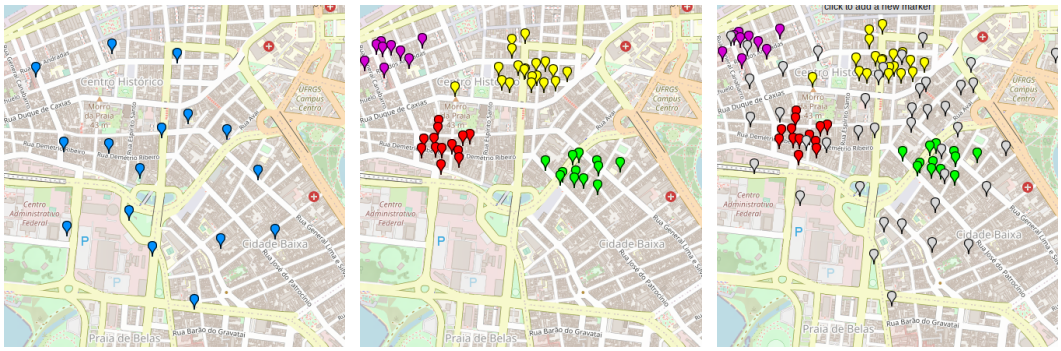
3. Node-random-clustered. It combines the random and clustered approaches. In this approach, ovig generates 40% to 60% of the patients following the clustered strategy. The remaining patients are generated randomly.

For strategies 2 and 3, the library requires two additional inputs. We need to specify the number of clusters to generate–in our experiments we always set five clusters–and the density value of how large the clusters can be. `Ovig` considers this density as the "radius" of the generated clusters. It uses the haversine distance in the computations due to the overhead of calculating and storing the distances matrix for all the points from the

OA database, which comprises 33.6k geographical points for the case of Porto Alegre (SARTORI; BURIOL, 2020).

Figure 9.1 – Node distribution strategies supported by `ovig`.

(a) Node-random (b) Node-clustered (c) Node-random-clustered



Source: the author.

The last step that `ovig` helps with the instance generation is to set the placement of the depot node. Here, the library allows putting the depot in a random location or the closest position to the center of the bounding rectangle set by the most extreme patient nodes. After that, we still need to generate the domain-related data for the HHCRSP, but there is not much that `ovig` can do to help us with this task. For this reason, we had further extend the library for generating such domain-specific data, as we discuss in the following sections.

9.3.2 Generating demands proportionally to the workforce available

As we mentioned in the previous section, one of our criticisms regarding completely random data generation is the possibility of generating instances in which the workforce for some service types is unbalanced regarding the patients' demands. For example, unbalanced scenarios appear in instances B6, C1, and D10 of Mankowska, Meisel and Bierwirth (2014) dataset. To further illustrate this problem, we highlight the case of D10, in which service type 1 has only one qualified caregiver but demands from 15 distinct patients. This issue becomes even more critical in the presence of time windows, turning the visits' tardiness unavoidable, such as for all these three instances.

We employed a revised version of Mankowska, Meisel and Bierwirth (2014) procedure to overcome unbalanced workforce issues when generating the caregivers' qualifications and patients' demands. Let $\mathcal{S} = \{1, 2, \ldots, 6\}$ be the set of service types

considered by the instance generator. For generating the caregivers, we split the set $\mathcal{S}$ into half (subsets $\mathcal{S}_a = \{1, 2, 3\}$ and $\mathcal{S}_b = \{4, 5, 6\}$). Half of the caregivers can perform up to three service types from $\mathcal{S}_a$, and the other half from subset $\mathcal{S}_b$. We also guarantee to generate at least one qualified caregiver for each service type of $\mathcal{S}$.

We use a similar strategy for generating the patients' demands. However, we did not sample these uniformly. Instead, our sampling is *biased* according to the number of qualified caregivers for each service type. We also perturb these bias a little because the real problem is also not perfectly balanced. This way, the skill $s \in \mathcal{S}$ has a weight of $\sum_{v \in \mathcal{V}} a_{vs} + u$ of being sampled, where u is a uniform integer sampled from $[0, 2]$. After calculating these weights, we set 70% of patients to request a single service type. The other 30% are split into simultaneous double services (15%) and double services with precedence constraints (15%). We generate the patient demand for single services considering the entire set $\mathcal{S}$ and the weights we discussed earlier. For the double services patients, we sample the first service type from $\mathcal{S}_a$, and the second service type from $\mathcal{S}_b$, also observing the biased distribution. Finally, the visit duration is generated uniformly from $[15, 60]$ minutes, which is the typical visit duration range of Porto Alegre system.

After generating data for qualification levels and service demands, the last step consists of setting few other characteristics regarding the operations at the patient site.

9.3.3 Generating patient time windows and separation times

All patients have a time windows of two hours placed between 08:00 and 19:00. Simultaneous double service patients have minimal and maximal visit separation time set to zero ($\delta^{\min} = \delta^{\max} = 0$). Double service patients with precedence constraints have the minimum and maximum separation time drawn uniformly from $[15, 60]$. If we generate $\delta^{\min} > \delta^{\max}$, then we swap these parameter values.

9.3.4 Methodology for generating the new dataset

As the reader can note, our instance generation algorithm is highly parameterized. Most parameters are domain-specific, and we also use Mankowksa's generator as a guideline to devise our generator. Our reasoning is to stick to a well-established dataset but changing the range of values to better represent the real problem in Porto Alegre. Table 9.3

highlight the differences between Mankowska, Meisel and Bierwirth (2014) original ranges and the ones we use while generating the new instances.

Table 9.3 – Differences in domain-specific parameters while generating instances.

Parameter	Literature dataset	Our dataset
Node generation	Random in the 100×100 square	Real data sampled from OpenAddresses
Travel times	Euclidean, according to the square region	Realistic, using the outputs of the OSRM
Single service patients	70%	70%
Simultaneous visit double service patients	15%	15%
Double service patients with precedences	15%	15%
Minimum separation time ($\delta^{\min}$)	Uniform in $[0, 60]$	Uniform in $[10, 30]$
Maximum separation time ($\delta^{\max}$)	Uniform in $[0, 60] + \delta^{\min}$	Uniform in $[10, 30]$
Visit duration	Uniform in $[10, 20]$	Uniform in $[15, 60]$
Planning horizon	10 hours	11 hours
Time window	2 hours	2 hours

Source: the author.

For some parameters, we used similar ranges to Mankowska, Meisel and Bierwirth (2014). We set a larger range for visit durations to better match the visit lengths of Porto Alegre's system, and we made a similar adjustment to the ranges of minimum and maximum separation time for double service patients with precedence constraints. The last change considers a slightly longer planning horizon than Mankowska, Meisel and Bierwirth (2014).

In addition to these domain-specific parameters, we also take into consideration the other parameters that `ovig` requires. In this aspect, we preferred to not bias the instance generator at expanse of a increased effort for generating instances for all the possible combination of parameters from Table 9.4. The literature often discuss this systematic methodology as the *one-factor-at-a-time* (UCHOA et al., 2017). As the reader can note, we have in total 30 distinct combinations of these parameters, but we can skip a few ones because the "cluster density" has no effect when the node generation strategy "node-random" is set. Thus we have effectively 22 distinct combinations of parameter values.

Table 9.4 – Node placement parameters supported by `ovig`.

Parameter	Values
Node generation strategy	node-random, node-clustered, node-random-clustered
Depot placement strategy	random, central
Cluster density	0.4, 0.8, 1.0, 1.2, 1.6

Source: the author.

Algorithm 6 summarizes the process of generating a single instance using our proposed methodology. The generator takes as input the seed to be used with then pseudo-random number generator and the instance size in the number of patients. It computes the number of caregivers to generate (line 1). For a single input, this algorithm produces several instances for each combination of parameters of Table 9.4 (line 2). For each combination, it applies the generator seed (line 3), then it calls *ovig* to generate patients and depot locations, as well as the matrix of travel times (line 4). Line 5 generates the data regarding domain-specific parameters of Table 9.3, disregarding its two initial rows. Line 6 stores the new instance, e.g., by writing a file with the generated data.

Algorithm 6: Complete algorithm for instance generation.

Input: seed s, number of patients np
1 $nc \leftarrow \max\{3, np \div 5\}$
2 **for** each combination of parameters c of Table 9.4 **do**
3 seed the generator using s
4 calls `ovig`, configured with c
5 generate domain-specific values using the ranges of column "Our dataset" of Table 9.3
6 store the generated data as a instance for the HHCRSP
7 **end**

Using Algorithm 6, we generate instances with 10, 25, 50, 75, 100, 200, 300, and 400 patients. Like Mankowska, Meisel and Bierwirth (2014), we generate several test cases for each instance size. More precisely, we call the generation algorithm 100 times, generating $22 \cdot 100 = 2200$ unique test cases for each instance size, using the numbers from 1 to 100 as seed for the algorithm. Our idea is to produce a so-called *curated dataset*, to which we provide some evidence of how hard it is to solve such instances to optimality. Appendix F details which of the generated instances we selected to compose this new dataset. Furthermore, we made the instance generator source code available through this

repository: <https://github.com/afkummer/ovig>. The dataset is available in this other repository: <https://github.com/afkummer/hhcrsp-dataset-2021>.

9.3.5 Measuring new instances' hardness

Despite our effort in generating an interesting dataset, there is still an issue of how we could effectively generate a rich mix of easy and hard test cases. To the best of our knowledge, the HHC literature seems to disregard the instance hardness, and the typical approach consists of generating random test cases without any significant consideration. Despite that, some authors from other research areas proposed convoluted methodologies that include the evaluation of the instance hardness while generating such datasets, e.g., for nurse scheduling problems (VANHOUCKE; MAENHOUT, 2009).

According to Vallada, Ruiz and Framinan (2015), one way for measuring the instance hardness is to look at the values of the so-called optimality gap, which is calculated after obtaining lower and upper bounds for these new instances (c.f. Chapter 6). In our experiment, the idea is to generate many test cases for some instance sizes and to compute their LB and UB. After tabulating these data, we can compute their relative optimality gap as $GAP_{\%} = \frac{UB-LB}{UB} \cdot 100$, sorting these new instances in non-increasing order of gap values. Supposing that the LBs are strong and that the UBs are of high quality, we can use such gap values to indicate the instances' hardness. This way, one can take, say, the top-10 instances to compose the final dataset.

Vallada, Ruiz and Framinan (2015) recognized that such an approach might sound naive because it depends on the efficacy of LB and UB-providing methods. In any way, this methodology seems to be the best option when we still do not know what makes a hard instance for some problem, say the HHCRSP, in a non-obvious manner. Furthermore, the authors suggest employing distinct methods for calculating LB and UB and taking the best values for each new instance.

Specifically to our new benchmark dataset for the HHCRSP, we already mentioned generating 100 distinct test cases for each instance size. We then submit these new instances to the hardness evaluation, in which the LBs are provided by the linear relaxation of the model of Subsection 5.1, and the genetic algorithm of Section 8.4 provides the UBs. Furthermore, our choice of LB approach was due to the limited time we had available to run the experiments. In total, we spent four days generating the entire dataset, considering the methodology of Subsection 9.3.4 and instances from 10 up to 400

patients. Additionally, we spent 49 days evaluating all the new instances using the methods mentioned above, distributing the work over three identical computers, and using parallel multicore processing in each one. We describe our computational environment in detail in Section 10.1.

After all this computation, we extracted the top-10 hardest test cases to compose our final dataset, but we also took ten additional instances at random. Thus, for some instance size, we have in total 20 test cases. The reason to include some instances at random is twofold. First, we want to include some *diversity* in our final dataset. Secondly, our measure of instance hardness is based on a single solution method for providing UBs. Thus, these instances can be either, *in fact*, hard or can be too hard *just for* our genetic algorithm.

10 EXPERIMENTAL RESULTS

This chapter presents our computational results regarding the methods we propose in Chapters 6, 7, and 8, for the benchmark dataset of Mankowska, Meisel and Bierwirth (2014).

10.1 Computational environment

Before discussing our computational results, we first define the computational environment used to run all our experiments. We used in total three identical machines, equipped with the same processor, memory, and motherboard. The only notable exception is the storage devices, but this should not significantly impact our experiments. To speed up our computations, we employed GNU Parallel to distribute the processing across the three machines (TANGE et al., 2011). Furthermore, all the processing is done at the machine level, and the generated logfiles are then collected in a single machine to run the analysis. Just for the case, we mention in the following text, our machines are identified as `shiva`, `ifrit`, and `malboro`.

All machines have a quad-core Intel i7-930 processor running at 2.80 GHz and with *turbo boost* technology disabled to prevent performance fluctuation. This equipment support what Intel calls *hyperthreading*, which artificially duplicates the number of cores detected in these chips. In our experiments, we never used these virtual cores. All machines have 12 GB of DDR3 memory, running at 1066 MHz. Regarding the software, we use Ubuntu 18.04 running the Linux kernel 4.15.0. Almost all our code is implemented in C++ and compiled with the GNU GCC 7.3.0. Regarding the proprietary software, we use CPLEX 20.1.0.0 as our default MIP solver. In the combinatorial lower bound experiments, we use VRPSolver 0.4.1, which embeds the slightly older CPLEX 12.10.0.0.

The ideal scenario for comparing solution methods is to test them in the same computational environment. Despite that, it would be tough and impractical to replicate all literature experiments in our computational system. From our experience, it is typically very hard to replicate results from a paper, either because the authors leave out some implementation details or because their environment has some particular characteristic. Instead, we prefer a simpler comparison approach in which we directly use the values reported in their publications.

In such an approach, one may think that this kind of comparison is unfair due to technological differences, especially when the period between the publication comprises more than a couple of years. For this reason, we use two strategies when comparing the runtime of algorithms. The first strategy is to solely use the values reported by the authors, which sounds more or less reasonable if the computational environments share some similarity, e.g., the same Intel x86 architecture. As a second strategy, we compute adjusted runtimes, employing some publicly available performance factors between the machines. In our case, we use the values from the PassMark (2021) benchmark database. This way, we can estimate the runtime of literature methods in our system, using these performance factors as a scale. Another important aspect is that we run our experiments in machines that are *older than* the ones employed in the problem's literature.

Table 10.1 introduces the computational characterization of the literature we compare. As a complicating aspect, neither Mankowska, Meisel and Bierwirth (2014) nor Lasfargeas, Gagné and Sioud (2019) specify which machine model they use in their experiments. Considering the date of the publications and the documentation available on the Intel website, we guessed that Mankowska, Meisel and Bierwirth (2014) used an Intel i3-4130 and that Lasfargeas, Gagné and Sioud (2019) used an Intel i7-6700K. Furthermore, we choose the slowest machines available at their publication time that match the description provided in their respective papers. Thus we do not artificially increase the speed factors, which would give us an unfair advantage in the comparisons. A detailed comparison between these machines is available at <https://www.cpubenchmark.net/compare/Intel-i7-930-vs-Intel-i3-4130-vs-Intel-i7-6700K/835vs2015vs2565>.

Table 10.1 – Computational environments and speed factors applied to adjust runtimes reported in the literature.

Publication	Machine description	PassMark score	Factor
Mankowska, Meisel and Bierwirth (2014)	Intel Core at 3.40 GHz	1880	1.4699
Lasfargeas, Gagné and Sioud (2019)	Intel i7 at 4.00 GHz	2528	1.9765
This thesis	Intel i7-930 at 2.80 GHz	1279	1.0000

10.2 Improved lower-bounds for literature's dataset

We present the results regarding the Mankowska, Meisel and Bierwirth (2014) dataset with the several scenarios proposed in Chapter 6, using the MIP model of Section 5.1. Table 10.2 introduces the average results per instance subset (column *Family*). Column

Lit. presents the average results reported in the literature for the LB^+ computed by CPLEX in ten hours of processing. Column *Sce. 1 (LP)* depicts the average solution values when we solve the linear relaxation of the problem. The following six columns present the average LB^+ for scenarios 2 to 7, respectively. These values were obtained by running CPLEX with a time limit of two hours. The last column indicates how much we could improve, on average, the bounds reported by Mankowska, Meisel and Bierwirth (2014), considering the best average value per instance subset. Appendix B shows the extensive results per instance. We also provide additional results in this attachment.

Table 10.2 – Average results for LB^+ experiments with CPLEX.

Family	Lit.	Sce. 1 (LP)	LB^+						Literature improve. (%)
			Sce. 2	Sce. 3	Sce. 4	Sce. 5	Sce. 6	Sce. 7	
A	**225.18**	110.57	**225.19**	**225.19**	**225.19**	**225.19**	**225.19**	**225.19**	0.01
B	343.11	165.36	387.14	388.95	346.80	345.90	**391.29**	352.63	14.04
C	342.62	207.74	384.24	384.47	381.34	382.00	**393.75**	385.45	14.92
D	377.36	249.14	399.73	410.54	409.52	408.71	**412.71**	412.02	9.37
E	404.53	270.02	402.10	402.16	428.53	428.47	425.55	**434.64**	7.44
F	435.26	367.57	492.88	496.43	525.30	530.97	**536.43**	528.32	23.24
G	462.12	451.47	*OOM	*449.91	603.98	**620.76**	604.59	OOM	34.33

Source: the author.

As the reader can notice, we employ a significantly smaller time limit of 2 hours in our experiments instead of the 10 hours of the literature. We set this more tight time limit for two reasons. First, all run for instances from subset B onwards run in out-of-memory issues after five hours of processing, disregarding the scenario employed. Secondly, we do not think it is worth spending long computational time to obtain such lower bounds due to the poor solution convergence. We also observe that CPLEX only produces optimal solutions for subset A. Subset B onwards, the solver always reached the time limit or stopped earlier due to out-of-memory.

We have found out-of-memory (OOM) issues in almost all test cases for the subset G in scenarios two, three, and seven. The only instance CPLEX can solve in scenario two is G10, so we prefer to mark this result with an asterisk and put an OOM indicator there. The same happened for three instances in scenario three and to all test cases of scenario seven. In the latter, the increased model size by additional cuts leads to the issue, even with the same memory-saving flags of scenario five. In scenarios two and three, we observe the OOM during the preprocessing of the model by CPLEX. In scenario seven, the OOM happens while the model is built. Thus, the average value for scenario three considers 7 out of 10 instances from subset G, with three instances resulting in OOM.

Observing the average results makes the reader think that the best choice is either scenarios 5 and 6, but the right choice varies according to the instance subset. Overall, we find that scenario five produced the best LB^+ for 20 instances of the dataset, while scenario six produced the best LB^+ for 36 instances. Moreover, scenario five produces the best results for 6 out of 10 instances of subset G, while scenario six produces worse bounds for this subset. Conversely, the situation is the opposite for the instance family F, to which scenario six produces the best LB^+ to 5 out of 10 instances, while scenario five is the best choice only for two instances. Despite that, we find the largest improvement of LB+ for subsets F and G, mainly because we consider the route inter-dependency constraints during our experiments, while the literature relaxed such constraints in theirs.

Table 10.3 accounts best performing scenario per instance family. In small test cases with 25 patients or less, default CPLEX suffices in producing the best lower bounds. For instances with 50 patients (subset C), the best approach is the scenario six. Results for subset D are mixed. Subset E seems to be the only that benefit from additional cuts. For subset G, corresponding to instance with 300 patients, the best strategy is to save memory using the parameters of scenario five. In these cases, providing a warmstart solution also seems to be detrimental when seeking for strong LB^+. For detailed results per instance, please check further Table B.1.

Table 10.3 – Accounting of best scenario by instance subset. The numbers indicate how many instances each scenario produces the best LB^+ per instance subset. Remark that each subset comprises 10 instances.

Subset	Sce. 2	Sce. 3	Sce. 4	Sce. 5	Sce. 6	Sce. 7
A	**10**	**10**	**10**	**10**	**10**	**10**
B	**8**	**8**	2	0	**8**	2
C	0	0	0	0	**10**	0
D	1	**3**	0	0	**3**	**3**
	19	21	12	10	**31**	15
E	1	0	0	2	0	**7**
F	0	0	1	2	**5**	2
G	0	0	4	**6**	0	0
	1	0	5	**10**	5	9
Total	20	21	17	20	36	24

Source: the author.

10.3 Combinatorial lower bounds with VRPSolver

In our experiments to obtain combinatorial lower bounds for the HHCRSP, we followed the official documentation and ran the VRPSolver through a Docker container. As far as we know, Docker implements *lightweight containers*, so we expect no significant overhead due to this extra layer of complexity. Just for the record, we employed a rootless installation of Docker version 20.10.6.

To produce the results from Table 10.4, we solve each instance of Mankowska, Meisel and Bierwirth (2014) dataset with VRPSolver, using the combinatorial relaxation model we described in Section 6.3. To hold a fair comparison with the results produced by CPLEX, we also stipulate a time limit of two hours when running VRPSolver. Column *Instance or family* indicates if the results in the row represent either a specific instance or an entire instance subset. Column *Patients (expanded)* indicates the total number of nodes after duplicating the double service patients. Column *Caregivers* indicate the number of vehicles in the instances. Columns *Runtime total (sec)* and *Runtime root (sec)* indicate the total time spent in building and solving the model, and the time spent solving just the root node, respectively. Column *Cost (root)* refers to the column generation's optimal value for the restricted master problem. Column *BCP nodes* indicates how many nodes the VRPSolver explored during the search. Finally, the last column, *Cost (CLB)*, indicates the combinatorial lower bound value produced by VRPSolver, i.e., the optimal solution for the integer problem. This column also indicates if the time limit was reached during the search, but the root node has been solved (TL), or if the column generation did not converge when the solver stopped (TL-R). A value of OOM indicates out-of-memory when building the model.

Table 10.4 – Results per instance and subset for the combinatorial lower bound experiment with VRPSolver.

Instance or family	Patients (expanded)	Caregivers	Runtime total (sec)	Runtime root (sec)	Cost (root)	BCP nodes	Cost (CLB)
A1	13	3	4.05	2.75	170.44	1	170.44
A2	13	3	19.95	18.66	156.70	1	156.70
A3	13	3	2.40	1.12	157.24	1	157.24
A4	13	3	23.01	21.72	128.86	1	128.86
A5	13	3	17.57	16.27	134.56	1	134.56
A6	13	3	2.53	1.24	122.62	1	122.62
A7	13	3	29.95	28.66	147.23	1	147.23
A8	13	3	2.31	1.01	149.33	1	149.33
A9	13	3	2.97	1.68	150.30	1	150.30
A10	13	3	2.23	0.94	220.31	1	220.31
B4	33	5	1510.71	1509.21	172.03	1	TL
B7	33	5	4928.30	393.91	196.42	3	TL
B8	33	5	7214.58	6163.15	211.31	1	TL
C	65	10	–	–	–	–	TL-R
D	98	15	–	–	–	–	TL-R
E	130	20	–	–	–	–	TL-R
F	260	30	–	–	–	–	OOM
G	400	40	–	–	–	–	OOM

Source: the author.

Contrary to what we expected, our experiments with VRPSolver led to very poor results. First, only instances from subset A could be solved to optimality, but for these cases, CPLEX can find their optimal solution in seconds. Three instances of subset B have been solved at the root node level, so in theory, we could use the root cost as a valid lower bound because VRPSolver employs a primal-dual solution method. In any case, such results are useless because the values produced are far below CPLEX results. The column generation did not converge within the time limit of two hours for all other test cases of subset B and all instances of subsets C, D, and E. For subsets F and G, all experiments failed due to out-of-memory while building the models. This attachment contains detailed results regarding all our VRPSolver runs.

Despite speculative, we have a few ideas why VRPSolver performed so poorly in our tests. According to the available documentation, its performance depends heavily upon tight upper bounds. Otherwise, we can expect the algorithm to spend much time proving the optimal solution value, similar to what happens with branch-and-cut methods. However, the main issue we identify is the absence of hard time windows in our problem. Because of that, the problems' solution space becomes too big, which also makes the dominance rules of the labeling algorithm within VRPSolver almost useless to solve subproblems quickly. Consequently, this issue slows down the subproblems so much that converging the column

generation becomes impossible within the time limit we set, even to instances small as the ones from subset B.

In a future experiment, we could try to solve a combinatorial relaxation problem comprising hard time windows by employing the model (5.1–5.10, 5.13–5.16), but setting the weight λ_2, which penalizes TW violations to ∞. In such an experiment, we conjecture that some of Mankowska, Meisel and Bierwirth (2014) instances might prove infeasible for the problem with hard time windows.

10.4 Results to the fix-and-optimize matheuristic

The proposed fix-and-optimize matheuristic has very few parameters. We set a time limit for solving each subproblem (STL, or "step-time-limit") to 25 seconds for all the instances we test (c.f. Algorithm 3). We implemented the matheuristic in C++, and we used the language's standard library for generating random numbers using the Mersenne-Twister (MATSUMOTO; NISHIMURA, 1998). We use the number of iterations without improvement as a stop criterion of the algorithm. Thus, each run stops after $\frac{|\mathcal{C}|}{2}$ iterations without improvement. We also solve each instance from Mankowska, Meisel and Bierwirth (2014) dataset 20 times, by seeding the first run with seed 1, the second run with seed 2, until the 20th run with seed 20. We run the experiments in the system described in Section 10.1.

In the previous revision of this text, we presented only a summary of results per instance subset, but we think a better approach is to present the extensive results directly, as in Table 10.5. The first column indicates to which instance the row presents data. Under the results of Mankowska, Meisel and Bierwirth (2014), the columns indicate which tested methods resulted in the best solution, the runtime as in the paper, the adjusted runtime using factors from Table 10.1, and the solution value. The following columns present Lasfargeas, Gagné and Sioud (2019) results regarding the runtime from their paper, the adjusted runtime, the value of the best solution produced, and the average and standard deviation solution values. For the matheuristic, we present the average runtime, the average number of iterations, the value of the initial solutions, the value of the best solution produced, and the average and standard deviation solution values. The last three columns present a few comparisons. Column *To init* indicates the relative difference between the initial solution and the average solution value produced by the matheuristic. Column *To MK* indicates the relative difference between Mankowska, Meisel and Bierwirth (2014) and the best

solution from the F&O. Similarly, column *To LF* indicates the relative difference between Mankowska, Meisel and Bierwirth (2014) and the best solution from the matheuristic. Bold values under MK, LF, and F&O indicate the best individual solution among all the methods.

Regarding Mankowska, Meisel and Bierwirth (2014) results, they applied 10 hours of CPLEX solver. For the LS and MN heuristics, the neighborhood is explored until achieving a local optimum. The VNS follows the same principle, but the authors also set two additional stop criteria. In AVNS-TL, the runtime of the MN heuristic defines a time limit for the VNS. In AVNS, they set a fixed runtime of 2h.

Conversely, Lasfargeas, Gagné and Sioud (2019) employed randomized local search operators, and they solved each instance 40 times. Due to the randomization, they set a time limit of five minutes for each run, and they accounted at which time the best solution was produced, reporting the minimum and maximum runtime for the 40 runs in each instance. As a drawback, we have no information regarding the average "effective runtime" of their heuristics or the time spent finding their best solutions. We then took a conservative approach, using only the maximum runtime they reported per instance. Nonetheless, the authors only tested instances from subset A to D, reporting that solving larger instances would require relaxing some problem constraints.

Table 10.5 – Extensive results for the fix-and-optimize matheuristic.

Inst.	MK: Mankowska, Meisel and Bierwirth (2014)				LF: Lasfargeas, Gagné and Sioud (2019)					F&O: Fix-and-optimize						Avg. difference (%)		
	Method	Time (sec)	A.Time(sec)	Cost	Time (sec)	A.Time (sec)	Best	Avg	SD	Avg time (sec)	Iters (avg)	Init	Best	Avg	SD	To init	To MK	to LF
A1	CPLEX	2.00	2.94	**218.20**	<1	1.98	**218.20**	224.30	14.60	0.17	8.85	271.21	**218.20**	219.19	4.44	19.18	-0.45	**2.28**
A2	CPLEX	5.00	7.35	**246.60**	<1	1.98	**246.60**	258.10	31.60	0.18	7.35	248.13	**246.63**	246.63	0.00	0.60	-0.01	**4.45**
A3	CPLEX	7.00	10.29	**305.90**	<1	1.98	**305.90**	358.80	38.50	0.47	7.15	339.07	**305.86**	305.86	0.00	9.79	0.01	**14.75**
A4	CPLEX	8.00	11.76	**186.90**	<1	1.98	**186.90**	196.40	11.10	1.30	7.70	210.42	**186.90**	189.25	7.24	10.06	-1.26	**3.64**
A5	CPLEX	2.00	2.94	**189.50**	<1	1.98	**189.50**	216.50	36.60	0.23	9.25	271.43	**189.54**	194.00	13.73	28.53	-2.38	**10.39**
A6	CPLEX	2.00	2.94	**200.10**	<1	1.98	**200.10**	200.10	0.00	0.18	7.95	200.16	**200.10**	200.11	0.03	0.02	-0.01	-0.01
A7	CPLEX	1.00	1.47	**225.40**	<1	1.98	**225.40**	232.40	24.20	0.12	7.35	383.44	**225.37**	225.37	0.00	41.22	0.01	**3.03**
A8	CPLEX	4.00	5.88	**232.00**	<1	1.98	**232.00**	281.40	47.60	0.19	7.45	295.98	**232.05**	232.05	0.00	21.60	-0.02	**17.54**
A9	CPLEX	20.00	29.40	**222.30**	<1	1.98	**222.30**	225.40	4.00	2.50	8.70	275.97	**222.30**	222.30	0.00	19.45	**0.00**	**1.38**
A10	CPLEX	1.00	1.47	**225.00**	<1	1.98	**225.00**	225.00	0.00	0.11	7.70	261.75	**225.01**	226.55	6.91	13.45	-0.69	-0.69
		5.20	7.64	**225.19**	<1	1.98	**225.19**	241.84	20.82	0.54	7.95	275.75	**225.19**	226.13	3.23	16.39	-0.48	**5.68**
B1	AVNS-TL	<1	0.00	458.90	53.10	104.95	434.10	552.80	93.40	42.95	25.65	787.91	**428.10**	434.79	8.06	44.82	**5.25**	**21.35**
B2	CPLEX	36,000.00	52,916.40	476.20	27.70	54.75	**476.00**	561.30	61.40	12.88	25.75	650.85	**476.05**	483.03	17.71	25.78	-1.43	**13.94**
B3	CPLEX	36,000.00	52,916.40	399.20	63.50	125.51	**399.10**	527.60	72.50	210.97	35.45	1328.21	**399.09**	419.51	12.07	68.42	-5.09	**20.49**
B4	CPLEX	36,000.00	52,916.40	576.00	66.80	132.03	414.00	509.70	74.50	20.76	30.00	1034.85	**411.30**	439.23	17.23	57.56	**23.74**	**13.83**
B5	AVNS-TL	<1	0.00	391.10	13.70	27.08	385.60	496.90	98.10	76.08	26.70	684.46	**366.34**	390.17	20.30	43.00	**0.24**	**21.48**
B6	MN	<1	0.00	534.70	43.70	86.37	447.80	611.80	129.90	261.43	36.20	1835.53	**441.70**	516.99	84.59	71.83	**3.31**	**15.50**
B7	MN	<1	0.00	355.50	61.50	121.55	**328.70**	398.80	64.80	42.72	31.40	496.67	**328.67**	334.33	9.72	32.69	**5.96**	**16.17**
B8	CPLEX	36,000.00	52,916.40	357.80	79.30	156.74	359.70	488.70	116.20	36.46	35.75	717.20	**357.68**	373.95	16.85	47.86	-4.51	**23.48**
B9	CPLEX	36,000.00	52,916.40	403.80	62.10	122.74	404.10	483.40	60.30	220.29	33.20	840.20	**402.67**	410.59	18.52	51.13	-1.68	**15.06**
B10	MN	<1	0.00	500.40	8.70	17.20	**462.70**	616.80	147.70	218.70	33.85	987.87	**462.75**	479.48	17.31	51.46	**4.18**	**22.26**
		36000.00	26458.20	445.36	48.01	94.89	411.18	524.78	91.88	114.32	31.40	936.38	**407.43**	428.21	22.24	49.45	**3.00**	**18.36**

Table 10.5 – (Continued.) Extensive results for the fix-and-optimize matheuristic.

Inst.	MK: Mankowska, Meisel and Bierwirth (2014)				LF: Lasfargeas, Gagné and Sioud (2019)					F&O: Fix-and-optimize						Avg. difference (%)		
	Method	Time (sec)	A.Time(sec)	Cost	Time (sec)	A.Time (sec)	Best	Avg	SD	Avg time (sec)	Iters (avg)	Init	Best	Avg	SD	To init	To MK	to LF
C1	AVNS-TL	<1	0.00	1123.60	96.20	190.14	974.20	1350.40	365.30	1256.49	132.15	4084.06	**957.05**	1001.48	60.67	75.48	**10.87**	**25.84**
C2	MN	<1	0.00	673.80	106.40	210.30	605.10	685.50	55.60	392.44	101.65	1014.01	**582.99**	610.95	43.52	39.75	**9.33**	**10.87**
C3	AVNS-TL	<1	0.00	642.40	109.80	217.02	562.90	698.20	82.70	538.64	133.65	1421.51	**558.75**	644.42	91.42	54.67	-0.31	**7.70**
C4	AVNS-TL	<1	0.00	580.40	112.40	222.16	521.90	630.40	101.80	214.54	126.30	816.24	**507.38**	527.13	25.72	35.42	**9.18**	**16.38**
C5	AVNS-TL	<1	0.00	754.60	114.90	227.10	683.10	822.60	119.30	376.06	97.00	1339.31	**667.53**	687.64	20.16	48.66	**8.87**	**16.41**
C6	AVNS-TL	<1	0.00	951.60	115.90	229.08	854.60	1010.60	146.40	1174.61	136.35	4217.62	**822.85**	900.74	82.23	78.64	**5.35**	**10.87**
C7	AVNS-TL	<1	0.00	577.40	109.40	216.23	529.20	572.50	29.70	417.45	131.15	893.74	**521.89**	540.31	10.63	39.54	**6.42**	**5.62**
C8	AVNS-TL	<1	0.00	540.60	110.80	219.00	**471.00**	522.80	29.80	137.04	106.00	783.89	476.66	489.45	10.30	37.56	**9.46**	**6.38**
C9	AVNS-TL	<1	0.00	608.70	115.40	228.09	551.10	642.70	77.60	411.82	126.85	1093.39	**535.87**	572.36	29.68	47.65	**5.97**	**10.94**
C10	AVNS-TL	<1	0.00	679.30	99.00	195.67	608.90	653.00	35.60	405.08	126.40	1394.82	**590.26**	617.35	26.91	55.74	**9.12**	**5.46**
		<1	0.00	713.24	109.02	215.48	636.20	758.87	104.38	532.42	121.75	1705.86	**622.12**	659.18	40.12	51.31	**7.43**	**11.65**
D1	AVNS-TL	5.00	7.35	1321.80	143.00	282.64	1278.20	1498.80	199.00	2670.26	291.30	3055.58	**1149.70**	1202.98	66.99	60.63	**8.99**	**19.74**
D2	AVNS-TL	4.00	5.88	892.70	168.70	333.44	746.90	914.30	97.90	1208.17	252.55	1411.10	**690.21**	761.25	50.31	46.05	**14.73**	**16.74**
D3	AVNS-TL	4.00	5.88	819.40	155.40	307.15	678.60	817.80	80.90	1302.75	270.45	1464.83	**643.23**	680.90	36.63	53.52	**16.90**	**16.74**
D4	AVNS-TL	4.00	5.88	877.40	148.50	293.51	809.70	1073.10	197.50	1578.76	224.70	2282.79	**798.99**	828.36	23.85	63.71	**5.59**	**22.81**
D5	AVNS-TL	5.00	7.35	872.10	150.30	297.07	777.00	924.90	120.80	1462.66	273.50	1412.03	**688.53**	741.15	35.31	47.51	**15.02**	**19.87**
D6	AVNS-TL	5.00	7.35	835.20	154.60	305.57	768.60	886.60	97.40	1201.58	318.85	1217.79	**712.15**	764.51	31.68	37.22	**8.46**	**13.77**
D7	AVNS-TL	6.00	8.82	706.30	168.10	332.25	600.10	680.40	31.60	1331.29	272.30	1104.35	**595.22**	619.66	20.47	43.89	**12.27**	**8.93**
D8	AVNS-TL	4.00	5.88	811.40	149.80	296.08	715.50	775.80	31.10	1158.73	280.55	1268.49	**666.09**	712.81	31.97	43.81	**12.15**	**8.12**
D9	MN	6.00	8.82	842.70	156.00	308.33	741.00	818.20	46.50	785.45	258.40	1192.45	**671.23**	726.78	28.43	39.05	**13.76**	**11.17**
D10	AVNS-TL	3.00	4.41	1306.60	173.10	342.13	1424.60	1867.70	258.60	2058.81	223.65	4508.65	**1239.79**	1329.02	88.15	70.52	-1.72	**28.84**
		4.60	6.76	928.56	156.75	309.82	854.02	1025.76	116.13	1475.85	266.63	1891.81	**785.51**	836.74	41.38	50.59	**10.61**	**16.67**

Table 10.5 – (Continued.) Extensive results for the fix-and-optimize matheuristic.

Inst.	MK: Mankowska, Meisel and Bierwirth (2014)				LF: Lasfargeas, Gagné and Sioud (2019)					F&O: Fix-and-optimize						Avg. difference (%)		
	Method	Time (sec)	A.Time(sec)	Cost	Time (sec)	A.Time (sec)	Best	Avg	SD	Avg time (sec)	Iters (avg)	Init	Best	Avg	SD	To init	To MK	to LF
E1	AVNS-TL	17.00	24.99	1604.90	–	–	–	–	–	4989.21	381.20	3786.22	**1337.87**	1466.10	121.41	61.28	**8.65**	–
E2	LS	10.00	14.70	1101.90	–	–	–	–	–	4584.23	490.75	1730.60	**858.38**	924.91	51.62	46.56	**16.06**	–
E3	AVNS-TL	14.00	20.58	986.40	–	–	–	–	–	3854.88	436.15	1715.90	**795.72**	883.85	43.01	48.49	**10.40**	–
E4	AVNS-TL	19.00	27.93	871.00	–	–	–	–	–	4406.35	493.60	1442.73	**732.28**	791.98	40.14	45.11	**9.07**	–
E5	AVNS-TL	19.00	27.93	1018.00	–	–	–	–	–	5514.02	516.20	1706.20	**791.18**	857.18	34.80	49.76	**15.80**	–
E6	AVNS-TL	19.00	27.93	1003.00	–	–	–	–	–	5007.03	577.35	1614.55	**831.76**	881.96	39.75	45.37	**12.07**	–
E7	AVNS-TL	20.00	29.40	921.10	–	–	–	–	–	4233.73	432.35	1289.07	**744.15**	813.42	42.63	36.90	**11.69**	–
E8	AVNS-TL	19.00	27.93	884.60	–	–	–	–	–	4619.17	516.45	1483.75	**745.18**	808.44	30.05	45.51	**8.61**	–
E9	AVNS-TL	18.00	26.46	1131.70	–	–	–	–	–	4245.42	509.25	1683.32	**926.38**	1011.13	49.52	39.93	**10.65**	–
E10	LS	11.00	16.17	1053.60	–	–	–	–	–	4984.28	502.80	2114.08	**874.51**	931.44	35.06	55.94	**11.59**	–
		16.60	24.40	1057.62	–	–	–	–	–	4643.83	485.61	1856.64	**863.74**	937.04	48.80	47.49	**11.46**	–
F1	AVNS	889.00	1306.74	**1721.40**	–	–	–	–	–	8033.76	300.65	2744.70	2557.06	2681.11	63.85	2.32	-55.75	–
F2	AVNS	909.00	1336.14	**1763.80**	–	–	–	–	–	5212.50	196.35	2648.03	2544.32	2628.75	30.43	0.73	-49.04	–
F3	AVNS	868.00	1275.87	**1549.60**	–	–	–	–	–	6323.53	237.15	2499.23	2433.87	2473.77	16.10	1.02	-59.64	–
F4	AVNS	1321.00	1941.74	**1420.40**	–	–	–	–	–	3147.07	115.95	2127.20	2116.80	2125.39	3.77	0.08	-49.63	–
F5	AVNS	1145.00	1683.04	**1701.90**	–	–	–	–	–	2971.40	106.95	2698.74	2669.68	2697.29	6.50	0.05	-58.49	–
F6	AVNS	836.00	1228.84	**1639.70**	–	–	–	–	–	11475.76	439.25	2609.83	2455.95	2504.51	30.34	4.04	-52.74	–
F7	AVNS	1294.00	1902.05	**1384.30**	–	–	–	–	–	2775.26	101.00	2253.09	2253.09	2253.09	0.00	0.00	-62.76	–
F8	AVNS	924.00	1358.19	**1544.60**	–	–	–	–	–	6890.17	258.55	2412.15	2321.96	2374.64	26.50	1.56	-53.74	–
F9	AVNS	1642.00	2413.58	**1572.90**	–	–	–	–	–	5102.92	191.05	2539.70	2510.01	2529.09	8.81	0.42	-60.79	–
F10	AVNS	1326.00	1949.09	**1581.00**	–	–	–	–	–	3079.60	110.85	2712.22	2686.98	2709.23	7.75	0.11	-71.36	–
		1115.40	1639.53	**1587.96**	–	–	**–**	–	–	5501.20	205.78	2524.49	2454.97	2497.69	19.40	1.03	-57.39	–

Table 10.5 – (Continued.) Extensive results for the fix-and-optimize matheuristic.

Inst.	MK: Mankowska, Meisel and Bierwirth (2014)				LF: Lasfargeas, Gagné and Sioud (2019)					F&O: Fix-and-optimize						Avg. difference (%)		
	Method	Time (sec)	A.Time(sec)	Cost	Time (sec)	A.Time (sec)	Best	Avg	SD	Avg time (sec)	Iters (avg)	Init	Best	Avg	SD	To init	To MK	to LF
G1	AVNS	7200.00	10,583.28	**2248.00**	–	–	–	–	–	4239.05	151.00	5089.92	5089.92	5089.92	0.00	0.00	-126.42	–
G2	AVNS	7200.00	10,583.28	**2316.10**	–	–	–	–	–	4662.85	151.00	10299.50	10299.50	10299.50	0.00	0.00	-344.69	–
G3	AVNS	7147.00	10,505.38	**1885.30**	–	–	–	–	–	4130.04	151.00	3152.55	3152.55	3152.55	0.00	0.00	-67.22	–
G4	AVNS	7200.00	10,583.28	**2023.20**	–	–	–	–	–	4581.85	151.00	3277.34	3277.34	3277.34	0.00	0.00	-61.99	–
G5	AVNS	7200.00	10,583.28	**2247.60**	–	–	–	–	–	4158.78	151.00	3584.11	3584.11	3584.11	0.00	0.00	-59.46	–
G6	AVNS	7200.00	10,583.28	**2144.40**	–	–	–	–	–	4678.47	151.00	3498.75	3498.75	3498.75	0.00	0.00	-63.16	–
G7	AVNS	6934.00	10,192.29	**1971.50**	–	–	–	–	–	4243.56	151.00	3214.17	3214.17	3214.17	0.00	0.00	-63.03	–
G8	AVNS	7200.00	10,583.28	**1987.40**	–	–	–	–	–	4108.06	151.00	3241.68	3241.68	3241.68	0.00	0.00	-63.11	–
G9	AVNS	7023.00	10,323.11	**2415.50**	–	–	–	–	–	4135.18	151.00	3673.47	3673.47	3673.47	0.00	0.00	-52.08	–
G10	AVNS	7003.00	10,293.71	**2373.40**	–	–	–	–	–	4080.58	151.00	3751.71	3751.71	3751.71	0.00	0.00	-58.07	–
		7130.70	10,481.42	**2161.24**	–	–	–	–	–	4301.84	151.00	4278.32	4278.32	4278.32	0.00	0.00	-95.92	–

Source: the author.

The methods are equivalent when solving instances from subset A, to which Mankowska, Meisel and Bierwirth (2014) proved their optimal values in less than 10 seconds of CPLEX. For the instance family B onwards, the matheuristic outperforms all the previous results, with three exceptions from subset B and one from subset C, which Lasfargeas, Gagné and Sioud (2019) produced the best results. Unfortunately, the matheuristic requires much more time, consuming 245 times the runtime reported by Mankowska, Meisel and Bierwirth (2014) and four times the runtime reported by Lasfargeas, Gagné, and Sioud (2019) for subset D, for example. In any way, the almost 20 minutes required by the matheuristic is reasonable when solving medium-sized instances such as subset D, with 75 patients and 15 caregivers.

For subsets A to E, the matheuristic consistently improves the initial solutions by almost 50%, but it makes almost no improvement for subsets F and G. We tried tweaking the matheuristic a little by decomposing more than two routes per iteration, but the only change was making the subproblems harder to solve. After some experiments, we observed that the matheuristic only affects when we relax the route inter-dependency constraints (5.11, 5.12). This result indicates that such constraints tie each route in many others, making our proposed exploitation strategy ineffective. As a result, Mankowska, Meisel and Bierwirth (2014) hold the best solutions for these two subsets.

We perform a succinct analysis of the matheuristic components to identify how much the two decomposition strategies collaborate to improve the solutions during the search. Each time we run the matheuristic, we identify which iterations improved the solution, recording which decomposition strategy was applied and how much it reduces the solution cost. We also compute the total number of applications of each decomposition strategy. Table 10.6 shows a summary of these results per instance subset. The first column indicates the instance subset, and in the second column, the decomposition strategy. The next column indicates how many times a decomposition strategy was applied while solving the instances from such a subset. Column *Improved* indicates how many of these attempts effectively improved the solution. Column % presents this same data in percentage of the total number of iterations. The last two columns present the average and standard deviation of "cost reduction" per decomposition strategy, respectively

As we can observe, the number of successful applications of both decomposition strategies increases as the instance size grows. However, putting these numbers in perspective, we observe that the matheuristic progressively loses its efficacy, as indicated by the decreasing values of improving iterations relative to the total number of iterations

Table 10.6 – Effectiveness analysis of decomposition strategies for the F&O.

Subset	Strategy	Number of applications			Improved objective	
		Total	Improved	%	Average	SD
A	Guided	724	95	8.69	**39.05**	43.36
	Random	865	183	14.43	33.96	37.75
B	Guided	3,060	735	2.99	63.04	142.08
	Random	3,219	757	2.90	**73.05**	156.30
C	Guided	12,145	2442	0.74	**47.28**	235.99
	Random	12,205	2679	0.81	35.04	188.74
D	Guided	26,569	4798	0.36	**23.94**	140.88
	Random	26,756	4771	0.36	20.15	122.09
E	Guided	48,304	8319	0.20	**11.85**	63.88
	Random	48,818	7749	0.19	11.01	48.55

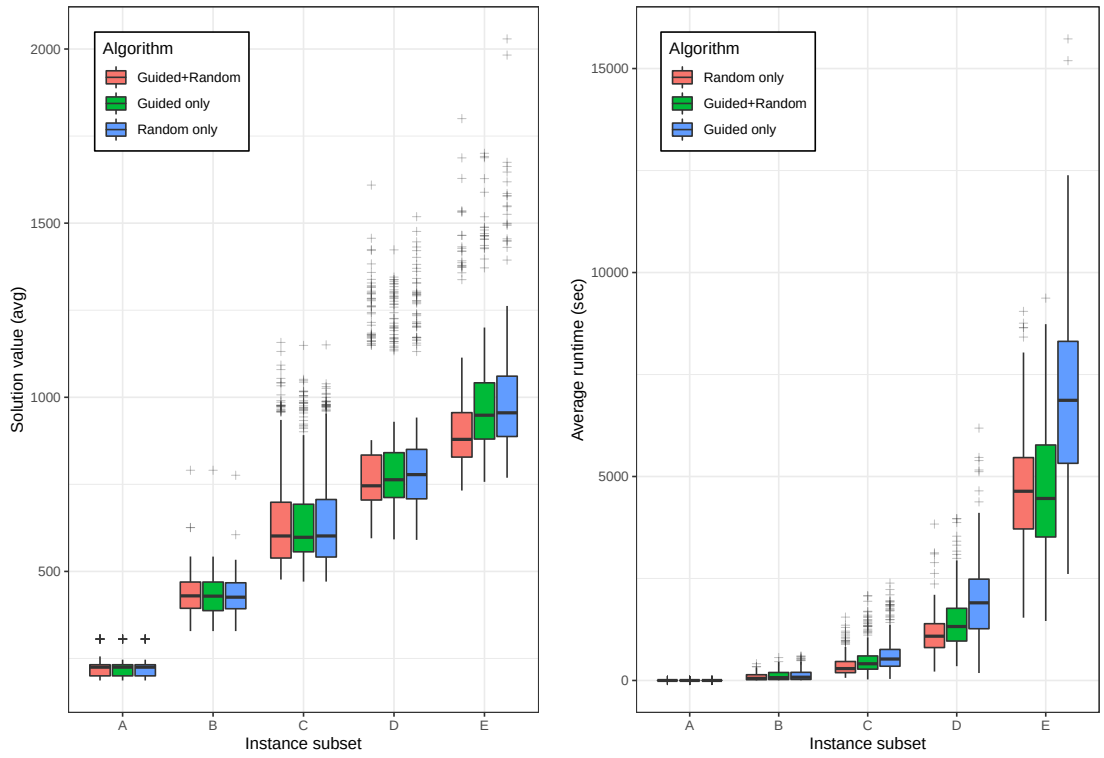
Source: the author.

(columns *Improved* and *Total*, respectively.) The guided strategy holds the best average cost reduction for subsets C to E, but the difference between the guided and random strategies decreases in larger test cases. Both have similar performance for subset D, and they are almost equivalent for subset E in the number of improved applications and average cost reduction.

To better understand the capabilities of each decomposition scheme, we repeated all 20 runs of the matheuristic in the Mankowska, Meisel and Bierwirth (2014) dataset but considering a single decomposition strategy. With that, we had run the experiments three times: once with both decompositions enabled, then with random decomposition disabled, and with guided decomposition disabled. Figure 10.1 presents these results per instance subset, indicating the behavior of solution values in Figure 10.1a and the matheuristic runtime in Figure 10.1b.

Observing the boxplot of Figure 10.1a, we notice that all the three variations of the matheuristic behave the same in the instance subsets A, B, and C. Furthermore, we can already notice a slightly larger variation of solution values for subset C, considering the "Guided+Random" and "Random-only" algorithms. Subset D already indicates a slight advantage of combining decomposition strategies, which holds the smallest median of solution costs among the three algorithms. This advantage becomes even more evident in the results of subset E, although we have a significant number of outliers in these larger subsets. With a significance level of 5%, a paired two-sided Wilcoxon signed-rank test indicates that the “Random+Guided” strategy is statistically different than “Random only” ($p =$ 2.2e-16) and “Guided only” (also $p =$ 2.2e-16). Moreover, the same test indicates that the individual strategies “Random only” and “Guided only” are not statistically distinct

Figure 10.1 – Matheuristic behavior regarding the decomposition schemes.
(a) Solution cost (b) Running times



Source: the author.

($p = 0.2391$). This result was more or less expected because we already noticed that both strategies seem to have similar efficacy as the instance sizes grow (c.f Table 10.6.)

The boxplot of Figure 10.1b indicates something very unexpected. The matheuristic exhibits much longer running times in the presence of the guided decomposition, which reflects in large values for "Guided only" and "Random+Guided" algorithms. To be more precise, our surprise is caused by the scale of such values, but not by this behavior itself, because we knew that subproblems generated by the guided decomposition tend to be harder to solve to optimality than those generated by the random strategy. The reason is somewhat apparent: the guided strategy tends to select longer routes, thus leading to bigger subproblems. Consequently, these subproblems frequently run for the entire "step-time-limit" STL within the matheuristic main loop, the opposite of several subproblems created by the random strategy. Nevertheless, according to data of subset E, the median runtime of "Guided+Random" is close to the fast "Random only," so we think that the best approach is to run the matheuristic with both decomposition schemes enabled, even because the "Random+Guided" seems to be the best algorithm according to data of Figure 10.1a. Additional results are available in this attachment, as well as in Appendix C.

10.5 Results from the BRKGA

In this first implementation of our BRKGA-based heuristic, we used the library `brkgaAPI` proposed by Toso and Resende (2015), which eases the implementation by providing a framework that encapsulates all the "low-level" operations of the BRKGA. The library has most parts of the genetic algorithm implemented out-of-the-box, requiring only an implementation of the problem-specific decoder. We used Algorithm 4 as a decoder for this meta-heuristic.

The code was implemented in C++, and we enabled its native support for decoding individuals in parallel through `OpenMP` directives. In our computational system, we observe almost a linear speedup in the heuristic runtime thanks to this parallel processing, mainly because the decoder we use is somewhat complex from the point of view of the asymptotic analysis. Considering the characteristics of our benchmark computers (c.f. Section 10.1,) we employed parallel decoding with four cores.

As a minor detail, we set the compiler to generate optimized code through the flag `-O3`. We also enabled the compiler flags to generate optimized code specifically to our processor model through the flags `-march=native` and `-mtune=native`. After analyzing some results, we noted that these two extra flags have almost no impact on the metaheuristic's performance because such optimizations benefit float-point intensive computations primarily (GNU Project, 2018).

10.5.1 Automatic algorithm configuration

The behavior of the BRKGA is set according to several parameters, ranging from the population size, the number of generations, the population percentage of elite and mutants, and the uniform crossover inheritance bias (c.f. Section 8.2). Configuring these parameters is not simple, so some researchers considered this task more art than a science (GONÇALVES; RESENDE, 2012). Instead of manually testing the meta-heuristic, we entrust to the `irace` software this automatic algorithm configuration (AAC) task of finding good values considering a set of parameters and their respective ranges (LÓPEZ-IBÁÑEZ et al., 2016).

The AAC has several inherent advantages over the manual configuration procedure. Mainly, it reduces the human effort, allowing the practical evaluation of thousands of configurations automatically. It also helps prevent experimental errors or biases from

human inclination regarding the algorithm behavior (EGGENSPERGER; LINDAUER; HUTTER, 2019). Tools like `irace` (LÓPEZ-IBÁÑEZ et al., 2016) require a very minimal effort to set up an AAC experiment. First, we need to specify the *range of values* for each parameter we want the `irace` to configure. We also need to provide *training instances* and a *budget* of maximum runs of the meta-heuristic we want to test. `Irace` then evaluates lots of configurations automatically by iteratively sampling new configurations, evaluating them over the training dataset, and then refining the following configurations' sampling step towards promising configurations generated so far. Moreover, this approach of sampling configurations and keeping the statistically best ones are often called a racing algorithm (BIRATTARI et al., 2002). Just for the record, we `irace` version 3.4.1, with R version 3.4.4, to configure our meta-heuristic.

In our experiment, we use Mankowska, Meisel and Bierwirth (2014) subset C as the training dataset for the AAC experiment, for configuring the parameters of Table 10.7. The first column presents the parameter description. Column *Range* indicates the valid range of values for sampling each parameter, and column *Best value* indicates the values of the best-ranked configuration produced by `irace`. As a stop criterion of the experiment, we establish a budget of 5,000 runs of the meta-heuristic. We spent almost six days completing this AAC experiment, using just one of our benchmark computers.

Table 10.7 – Parameter calibration setting tested with `irace`. Ranges in square and round brackets indicate integer and real sequences, respectively.

Parameter	Range	Best value
Population size (p)	$[50, 1000]$	885
Number of generations	$[100, 3000]$	1823
Percentual of elite set (p_e)	$(0.01, 0.50)$	0.20665
Percentual of mutant set (p_m)	$(0.01, 0.70)$	0.05408
Bias to inherit from elite parent (ρ_e)	$(0.01, 1.0)$	0.32728

Source: the author.

10.5.2 Extensive results for the literature dataset

Using the parameters from column *Best value* of Table 10.7, we solve each instance of Mankowska, Meisel and Bierwirth (2014) 20 times by using distinct random seeds from 1 to 20. Table 10.8 presents the genetic algorithm's results, comparing with the previous best-known solution (BKS). The first column identifies the instance, and the second column presents the best known lower bound for such instance (c.f Table B.1).

Under Previous BKS, we identify the source of such value (*Source*), the original algorithm runtime as presented in their paper (*column Time (sec)*), the adjusted algorithm runtime using the speed factors of Table 10.1 (column *T.Adj (sec)*), and best, average, and standard deviation of solution values, respectively. The source MK indicates Mankowska, Meisel and Bierwirth (2014) results, while the source LF indicates Lasfargeas, Gagné and Sioud (2019). F&O refers to results from the matheuristic from Chapter 7. The asterisk in LF* indicates that the F&O produced the same best solution value as Lasfargeas, Gagné and Sioud (2019). MK* indicates a similar situation regarding results from Mankowska, Meisel and Bierwirth (2014).

Under *BRKGA* of Table 10.8, the column *Avg time(sec)* indicates the average runtime of the genetic algorithm. The following three columns present the best, average, and standard deviation of the solution produced by BRKGA, respectively. Column *Rel.Diff (%) previous BKS* indicates the improvement of the best solution produced by the genetic algorithm relative to the best solution value from BKS, calculated as $\frac{\text{Best BRKGA - Best BKS}}{\text{Best BKS}} \cdot 100$. The last two columns indicates the optimality gap of the best and average solution values produced by BRKGA relative to LB$^+$, calulated as $\frac{\text{X - LB}^+}{\text{X}} \cdot 100$, where X represents either the best or the average solution value from the genetic algorithm.

Table 10.8 – Extensive computational results for the BRKGA heuristic.

Instance	LB$^+$	Previous BKS						BRKGA				Rel.Diff (%) previous BKS	Opt. gap (%)	
		Source	Time (sec)	T.Adj (sec)	Best	Avg	SD	Avg time (sec)	Best	Avg	SD		Best	Avg
A1	218.20	MK*	2.00	2.94	**218.20**	–	–	0.72	226.98	226.98	0.00	4.02	3.87	3.87
A2	246.63	MK*	5.00	7.35	**246.60**	–	–	0.72	**246.63**	246.63	0.00	0.01	0.00	0.00
A3	305.86	MK*	7.00	10.29	**305.90**	–	–	0.73	**305.86**	305.86	0.00	-0.01	0.00	0.00
A4	186.90	MK*	8.00	11.76	**186.90**	–	–	0.72	**186.90**	186.90	0.00	0.00	0.00	0.00
A5	189.54	MK*	2.00	2.94	**189.50**	–	–	0.72	191.97	191.97	0.00	1.30	1.27	1.27
A6	200.10	MK*	2.00	2.94	**200.10**	–	–	0.73	**200.13**	200.13	0.00	0.01	0.01	0.01
A7	225.37	MK*	1.00	1.47	**225.40**	–	–	0.69	**225.37**	225.37	0.00	-0.01	0.00	0.00
A8	232.05	MK*	4.00	5.88	**232.00**	–	–	0.70	**232.05**	232.05	0.00	0.02	0.00	0.00
A9	222.30	MK*	20.00	29.40	**222.30**	–	–	0.76	234.21	234.21	0.00	5.36	5.09	5.09
A10	225.01	MK*	1.00	1.47	**225.00**	–	–	0.70	**225.01**	225.01	0.00	0.00	0.00	0.00
	225.19		5.20	7.64	**225.19**			0.72	227.51	227.51	0.00	1.03	1.02	1.02
B1	428.10	F&O	42.95	–	**428.10**	434.79	8.06	1.86	**428.10**	428.32	0.25	0.00	0.00	0.05
B2	476.05	LF*	27.70	54.75	**476.00**	561.30	61.40	1.86	483.63	485.31	1.54	1.60	1.57	1.91
B3	399.09	LF*	63.50	125.51	**399.10**	527.60	72.50	1.92	402.80	402.80	0.00	0.93	0.92	0.92
B4	411.30	F&O	20.76	–	**411.30**	439.23	17.23	1.85	420.29	432.55	3.37	2.19	2.14	4.91
B5	366.34	F&O	76.08	–	**366.34**	390.17	20.30	1.77	372.16	374.65	3.09	1.59	1.56	2.22
B6	405.58	F&O	261.43	–	**441.70**	516.99	84.59	1.93	471.00	471.95	1.43	6.63	13.89	14.06
B7	328.67	LF*	61.50	121.55	**328.70**	398.80	64.80	1.98	**328.67**	328.67	0.00	-0.01	0.00	0.00
B8	357.68	F&O	36.46	–	**357.68**	373.95	16.85	1.93	359.70	359.70	0.00	0.56	0.56	0.56
B9	330.41	F&O	220.29	–	**402.67**	410.59	18.52	2.09	**402.67**	404.25	1.06	0.00	17.95	18.27
B10	420.99	LF*	8.70	17.20	**462.70**	616.80	147.70	1.96	469.58	469.58	0.00	1.49	10.35	10.35
	392.42		81.94		**407.43**	467.02	51.19	1.92	413.86	415.78	1.07	1.58	5.18	5.62

Table 10.8 – (Continued.) Extensive computational results for the BRKGA heuristic.

Instance	LB$^+$	Previous BKS						BRKGA				Rel.Diff (%) previous BKS	Opt. gap (%)	
		Source	Time (sec)	T.Adj (sec)	Best	Avg	SD	Avg time (sec)	Best	Avg	SD		Best	Avg
C1	459.25	F&O	1256.49	–	**957.05**	1001.48	60.67	5.89	965.15	977.56	14.39	0.85	52.42	53.02
C2	373.94	F&O	392.44	–	**582.99**	610.95	43.52	6.06	583.39	590.45	8.53	0.07	35.90	36.67
C3	390.48	F&O	538.64	–	558.75	644.42	91.42	5.84	**548.79**	559.53	6.44	**-1.78**	28.85	30.21
C4	371.99	F&O	214.54	–	**507.38**	527.13	25.72	5.68	519.91	531.91	7.38	2.47	28.45	30.07
C5	464.97	F&O	376.06	–	**667.53**	687.64	20.16	5.01	678.49	698.14	19.54	1.64	31.47	33.40
C6	360.73	F&O	1174.61	–	**822.85**	900.74	82.23	5.85	840.69	845.30	4.30	2.17	57.09	57.33
C7	354.15	F&O	417.45	–	**521.89**	540.31	10.63	6.58	534.01	540.42	5.33	2.32	33.68	34.47
C8	375.52	LF	110.80	219.00	**471.00**	522.80	29.80	5.69	474.55	479.75	3.60	0.75	20.87	21.73
C9	355.29	F&O	411.82	–	535.87	572.36	29.68	6.62	**534.30**	551.72	9.25	**-0.29**	33.50	35.60
C10	431.18	F&O	405.08	–	**590.26**	617.35	26.91	5.51	611.25	618.83	4.36	3.56	29.46	30.32
	393.75		529.79		**621.56**	662.52	42.07	5.87	629.05	639.36	8.31	1.21	37.41	38.42
D1	492.09	F&O	2670.26	–	**1149.70**	1202.98	66.99	13.81	1186.20	1208.19	13.98	3.17	58.52	59.27
D2	385.83	F&O	1208.17	–	**690.21**	761.25	50.31	11.58	693.28	719.52	15.00	0.44	44.35	46.38
D3	385.34	F&O	1302.75	–	643.23	680.90	36.63	13.95	**635.67**	650.29	7.63	**-1.17**	39.38	40.74
D4	419.21	F&O	1578.76	–	**798.99**	828.36	23.85	11.64	809.45	840.61	13.72	1.31	48.21	50.13
D5	418.44	F&O	1462.66	–	**688.53**	741.15	35.31	13.48	691.50	703.23	12.10	0.43	39.49	40.50
D6	393.92	F&O	1201.58	–	**712.15**	764.51	31.68	14.43	733.67	744.99	7.32	3.02	46.31	47.12
D7	372.49	F&O	1331.29	–	595.22	619.66	20.47	15.03	**590.64**	603.65	6.42	**-0.77**	36.93	38.29
D8	409.35	F&O	1158.73	–	666.09	712.81	31.97	13.84	**661.78**	681.10	12.47	**-0.65**	38.14	39.90
D9	394.65	F&O	785.45	–	**671.23**	726.78	28.43	13.18	704.63	722.09	10.35	4.98	43.99	45.35
D10	485.63	F&O	2058.81	–	1239.79	1329.02	88.15	11.00	**1208.71**	1294.66	66.45	**-2.51**	59.82	62.49
	415.69		1475.85		**785.51**	836.74	41.38	13.20	791.55	816.83	16.54	0.77	47.48	49.11

Table 10.8 – (Continued.) Extensive computational results for the BRKGA heuristic.

Instance	LB$^+$	Previous BKS						BRKGA				Rel.Diff (%) previous BKS	Opt. gap (%)	
		Source	Time (sec)	T.Adj (sec)	Best	Avg	SD	Avg time (sec)	Best	Avg	SD		Best	Avg
E1	437.82	F&O	4989.21	–	1337.87	1466.10	121.41	24.46	**1331.49**	1352.33	10.75	**-0.48**	67.12	67.62
E2	444.88	F&O	4584.23	–	858.38	924.91	51.62	23.57	**848.08**	869.45	15.00	**-1.20**	47.54	48.83
E3	465.31	F&O	3854.88	–	795.72	883.85	43.01	23.10	**788.03**	815.31	15.16	**-0.97**	40.95	42.93
E4	412.08	F&O	4406.35	–	732.28	791.98	40.14	24.69	**711.19**	731.24	11.75	**-2.88**	42.06	43.65
E5	416.62	F&O	5514.02	–	791.18	857.18	34.80	23.10	**781.50**	806.81	13.73	**-1.22**	46.69	48.36
E6	418.12	F&O	5007.03	–	831.76	881.96	39.75	24.44	**788.08**	803.28	7.86	**-5.25**	46.94	47.95
E7	402.04	F&O	4233.73	–	744.15	813.42	42.63	26.69	**711.11**	731.83	9.38	**-4.44**	43.46	45.06
E8	444.35	F&O	4619.17	–	**745.18**	808.44	30.05	22.91	748.48	761.06	7.40	0.44	40.63	41.61
E9	453.86	F&O	4245.42	–	926.38	1011.13	49.52	22.44	**921.78**	950.89	13.90	**-0.50**	50.76	52.27
E10	458.81	F&O	4984.28	–	874.51	931.44	35.06	23.99	**825.24**	847.64	14.48	**-5.63**	44.40	45.87
	435.39		4643.83		863.74	937.04	48.80	23.94	**845.50**	866.98	11.94	**-2.11**	48.51	49.78
F1	548.88	MK	1721.40	1306.74	1721.40	–	–	89.15	**1372.69**	1421.76	17.34	**-20.26**	60.01	61.39
F2	543.32	MK	1763.80	1336.14	1763.80	–	–	95.02	**1336.33**	1383.55	23.01	**-24.24**	59.34	60.73
F3	547.64	MK	1549.60	1275.87	1549.60	–	–	91.24	**1263.39**	1288.66	14.08	**-18.47**	56.65	57.50
F4	532.31	MK	1420.40	1941.74	1420.40	–	–	110.09	**1124.24**	1145.53	10.23	**-20.85**	52.65	53.53
F5	538.14	MK	1701.90	1683.04	1701.90	–	–	101.85	**1316.54**	1361.64	20.75	**-22.64**	59.12	60.48
F6	525.95	MK	1639.70	1228.84	1639.70	–	–	85.45	**1322.89**	1369.38	25.66	**-19.32**	60.24	61.59
F7	512.98	MK	1384.30	1902.05	1384.30	–	–	107.73	**1131.27**	1163.97	17.53	**-18.28**	54.65	55.93
F8	540.05	MK	1544.60	1358.19	1544.60	–	–	96.05	**1132.77**	1165.67	16.72	**-26.66**	52.32	53.67
F9	543.16	MK	1572.90	2413.58	1572.90	–	–	101.87	**1293.78**	1347.78	22.91	**-17.75**	58.02	59.70
F10	546.84	MK	1581.00	1949.09	1581.00	–	–	98.96	**1418.53**	1446.56	15.68	**-10.28**	61.45	62.20
	537.93		1587.96	1639.53	1587.96			97.74	**1271.24**	1309.45	18.39	**-19.94**	57.69	58.92

Table 10.8 – (Continued.) Extensive computational results for the BRKGA heuristic.

Instance	LB^{+}	Previous BKS						BRKGA				Rel.Diff (%) previous BKS	Opt. gap (%)	
		Source	Time (sec)	T.Adj (sec)	Best	Avg	SD	Avg time (sec)	Best	Avg	SD		Best	Avg
G1	612.37	MK	7200.00	10,583.28	2248.00	–	–	221.55	**1778.54**	1851.25	32.13	**-20.88**	65.57	66.92
G2	614.82	MK	7200.00	10,583.28	2316.10	–	–	243.95	**1824.74**	1899.33	34.71	**-21.21**	66.31	67.63
G3	614.20	MK	7147.00	10,505.38	1885.30	–	–	216.58	**1514.23**	1546.44	13.92	**-19.68**	59.44	60.28
G4	604.30	MK	7200.00	10,583.28	2023.20	–	–	253.52	**1564.42**	1599.35	23.92	**-22.68**	61.37	62.22
G5	633.66	MK	7200.00	10,583.28	2247.60	–	–	211.75	**1694.50**	1750.03	28.07	**-24.61**	62.60	63.79
G6	621.46	MK	7200.00	10,583.28	2144.40	–	–	264.39	**1714.38**	1779.29	21.68	**-20.05**	63.75	65.07
G7	603.48	MK	6934.00	10,192.29	1971.50	–	–	224.19	**1640.07**	1677.66	22.99	**-16.81**	63.20	64.03
G8	618.74	MK	7200.00	10,583.28	1987.40	–	–	221.25	**1547.63**	1582.71	20.05	**-22.13**	60.02	60.91
G9	668.16	MK	7023.00	10,323.11	2415.50	–	–	229.19	**1942.21**	1974.16	19.69	**-19.59**	65.60	66.15
G10	638.92	MK	7003.00	10,293.71	2373.40	–	–	205.69	**1872.08**	1931.99	25.31	**-21.12**	65.87	66.93
	623.01		7130.70	10,481.42	2161.24			229.20	**1709.28**	1759.22	24.25	**-20.91**	63.55	64.59

Source: the author.

The BRKGA can find the optimal solution to most instances of subset A, except for instances A1, A5, and A9. We verified that the decoder could not reach a optimal solution in these cases due to a characteristic of the algorithm, as observed in Proposition 8.3.1. Considering the entire dataset, the BRKGA improves the previous best-known solution for C9 by 0.29%, and the previous BKS for F8 the impressive 26.66%. Compared to results from Mankowska, Meisel and Bierwirth (2014), the genetic algorithm finds improved solutions for 67 out of 70 instances; Compared to Lasfargeas, Gagné and Sioud (2019), it outperforms their VNS-based meta-heuristic in 28 out of 40 instances; It also outperforms our matheuristic in 49 out of 70 instances. Furthermore, the BRKGA has the best average results on the larger instance subsets, improving in 19.94% the average solution value for subset F and 20.91% for subset G, compared to the previous average BKS.

Despite these promising results, we think that one of the notable strengths of the genetic algorithm is its speed. For subset G, with the largest test cases, the BRKGA finishes in about 3.8 minutes. Compared to the average runtime of two hours (2.91 hours using the speed factors), it is 31 times faster than Mankowska, Meisel and Bierwirth (2014) AVNS (or 45.6 times according to the speed factors). For subset F, the average runtime is close to 1.6 minutes, and from subset E to below, the algorithm finishes in less than 30 seconds. These short running times justify our choice of employing the BRKGA in providing MIP warm start solutions for Scenario 6 we described in Chapter 6.

In general, the genetic algorithm is the best option for solving the HHCRSP in time-constrained situations. It also seems to be the best option when solving larger instances considering both solution quality and running times. Otherwise, the best option seems to be the fix-and-optimize matheuristic, especially on moderate-sized instances up to 75 patients as long as the matheuristic runtime is not of concern. Furthermore, the speed of BRKGA would allow some local-search operators within the decoder, devising what the literature calls a memetic algorithm (MOSCATO, 1999). Appendix D has additional data for BRKGA experiments. The data of Table 10.8 is also available in this attachment.

10.6 Results from the BRKGA-MP-IPR

We use the open-source `brkga_mp_ipr_cpp` to implement our extended BRKGA algorithm (PESSOA et al., 2020). This library holds many similarities with the previous `brkgaAPI` of Toso and Resende (2015) but is more modern, and it is designed to take advantage of new language resources of C++ standard from 2017. That said,

this may turn the library unusable in old systems lacking support of these more recent C++ features. Fortunately, the `brkga_mp_ipr_cpp` library already implements the multi-parent mating algorithm, implicit path relinking, and island model, which eases a lot in designing and implementing meta-heuristics for specific problems. For the sake of writing and helping the reader, we will further refer to the standard genetic algorithm as BRKGA and the algorithm with intensification components as BRKGA-MP-IPR.

We implemented the heuristic in C++, and we compiled the application using the same compiler and flags described in Section 10.5. Furthermore, we also use the same computational environment we described in Section 10.1, also using four cores for parallel decoding of individuals through OpenMP directives.

10.6.1 Automatic algorithm configuration

As we discuss in the previously, the presence of intensification components makes our proposed BRKGA-MP-IPR even more parameterized than the standard BRKGA. As we do not have a clear idea of how the intensification components would work with the evolutionary process, we prefer to re-run the AAC experiment with `irace`, considering the additional parameters that control the intensification components of BRKGA-MP-IPR. For this task, we use the same versions of `irace` and R described in Subsection 10.5.1.

In this AAC experiment, we consider in total the 17 parameters listed in Table 10.9. We described such parameters in Section 8.4. The first column presents the parameter name, and the second column presents the values we allow testing with `irace`. The reader can note that we use a discrete set of values this time instead of specifying ranges as in Subsection 10.5.1. Finally, the last column presents the parameter value set by `irace` as the best configuration found after a budget of 1500 runs of the meta-heuristic. In this experiment, we also changed the training instances from subset G. Furthermore, we fixed the values of some parameters, e.g., IPR to be `permutation` and distance function to be `Kendall-Tau`. We omitted these parameters from the AAC experiment and in Table 10.9.

Table 10.9 – Parameters and range of values configured by *irace*.

Parameter name	Values allowed	Best value
Population size	300, 350, 400, 450, 500, 550, 600, 650, 700, 750, 800, 850, 900, 950, 1000, 1100	1000
Generations	200, 400, 500, 600, 700, 800, 850, 900, 950, 1000, 1100, 1200, 300, 1400, 1500 ,1600, 1700, 1800, 1900, 2000	1900
Elite (%)	0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3	0.1
Mutant (%)	0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3	0.01
# of parents	2, 3, 4, 5	5
# of elite parents	1, 2, 3, 4	1
Bias function	`constant`, `cubic`, `exponential`, `linear`, `loginverse`, `quadatic`	`constant`
Independent populations	1, 2, 3, 4	3
Immigrants	1, 5, 10, 15, 20, 25, 30, 50, 60, 70, 80, 100, 120, 150	50
Exchange elite frequency	10, 50, 80, 120, 150, 180, 210, 260, 300, 350, 400	80
IPR pairs	1, 10, 50, 75, 100, 125, 150, 180, 220	75
IPR selection criteria	`best`, `random`	`random`
IPR path (%) to explore	0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.6, 0.8, 1.0	0.01
IPR block size	0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.6, 0.8	–
IPR minimum distance	0.001, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.98	0.001
IPR frequency	5, 10, 50, 80, 120, 150, 180, 210, 260, 300, 350, 400	180
Reset frequency	350, 400, 450, 500, 550, 600, 650, 700, 800, 900, 1000, 1100	700

Source: the author.

10.6.2 Extensive results for the literature dataset

We run a very similar experiment with BRKGA-MP-IPR than we did for BRKGA. We solve each instance of Mankowska, Meisel and Bierwirth (2014) 20 times by using distinct random seeds from 1 to 20. Table 10.10 presents BRKGA-MP-IPR results, comparing with the previous best-known solution (BKS). The first column identifies the instance, and the second column presents the best known lower bound for such instance (c.f Table B.1). Under Previous BKS, we identify the source of such value (*Source*), the original algorithm runtime as presented in their paper (*column Time (sec)*), the adjusted algorithm runtime using the speed factors of Table 10.1 (column *T.Adj (sec)*), and best, average, and standard deviation of solution values, respectively. The source MK indicates Mankowska, Meisel and Bierwirth (2014) results, while the source LF indicates Lasfargeas, Gagné and Sioud (2019). F&O refers to results from the matheuristic from Chapter 7. Finally, BRKGA indicates the previous results from Section 10.5. The asterisk in LF* indicates that the F&O produced the same best solution value as Lasfargeas, Gagné and Sioud (2019). MK* indicates a similar situation regarding results from Mankowska, Meisel and Bierwirth (2014).

Under *BRKGA-MP-IPR*, column *Avg time (sec)* indicates the average runtime of the genetic algorithm. The following three columns present the best, average, and standard

deviation of the solution produced by BRKGA, respectively. Column *Rel.Diff (%) previous BKS* indicates the improvement of the best solution produced by the genetic algorithm relative to the best solution value from BKS, calculated as $\frac{\text{Best BRKGA - Best BKS}}{\text{Best BKS}} \cdot 100$. The last two columns indicates the optimality gap of the best and average solution values produced by BRKGA relative to LB^{+}, calulated as $\frac{\text{X - LB}^{+}}{\text{X}} \cdot 100$, where X represents either the best and the average solution value from the genetic algorithm, respectively.

Table 10.10 – Extensive computational results for the BRKGA-MP-IPR heuristic.

Instance	LB$^+$	Previous BKS						BRKGA-MP-IPR				Rel.Diff (%) previous BKS	Opt. gap (%)	
		Source	Time (sec)	T.Adj (sec)	Best	Avg	SD	Avg time (sec)	Best	Avg	SD		Best	Avg
A1	218.20	MK	2.00	2.94	**218.20**	–	–	97.22	226.98	226.98	0.00	4.02	3.87	3.87
A2	246.63	MK	5.00	7.35	**246.60**	–	–	97.40	**246.63**	246.63	0.00	0.01	0.00	0.00
A3	305.86	MK	7.00	10.29	**305.90**	–	–	98.34	**305.86**	305.86	0.00	-0.01	0.00	0.00
A4	186.90	MK	8.00	11.76	**186.90**	–	–	97.54	**186.90**	186.90	0.00	0.00	0.00	0.00
A5	189.54	MK	2.00	2.94	**189.50**	–	–	96.72	191.97	191.97	0.01	1.30	1.26	1.27
A6	200.10	MK	2.00	2.94	**200.10**	–	–	97.74	**200.13**	200.14	0.01	0.01	0.02	0.02
A7	225.37	MK	1.00	1.47	**225.40**	–	–	96.88	**225.37**	225.37	0.00	-0.01	0.00	0.00
A8	232.05	MK	4.00	5.88	**232.00**	–	–	98.25	**232.05**	232.05	0.00	0.02	0.00	0.00
A9	222.30	MK	20.00	29.40	**222.30**	–	–	96.02	234.21	234.21	0.00	5.36	5.09	5.09
A10	225.01	MK	1.00	1.47	**225.00**	–	–	95.77	**225.01**	225.01	0.00	0.00	0.00	0.00
	225.19		5.20	7.64	**225.19**	–	–	97.18	227.51	227.51	0.00	1.03	1.02	1.02
B1	428.10	F&O	42.95	–	**428.10**	434.79	8.06	104.28	**428.10**	428.10	0.00	0.00	0.00	0.00
B2	476.05	*LF	27.70	54.75	**476.00**	561.30	61.40	104.52	483.63	483.63	0.00	1.60	1.57	1.57
B3	399.09	*LF	63.50	125.51	**399.10**	527.60	72.50	103.83	402.80	402.80	0.00	0.93	0.92	0.92
B4	411.30	F&O	20.76	–	**411.30**	439.23	17.23	105.64	420.29	429.87	4.71	2.19	2.14	4.32
B5	366.34	F&O	76.08	–	**366.34**	390.17	20.30	102.08	372.16	372.86	1.92	1.59	1.56	1.75
B6	405.58	F&O	261.43	–	**441.70**	516.99	84.59	106.34	471.00	471.00	0.00	6.63	13.89	13.89
B7	328.67	*LF	61.50	121.55	**328.70**	398.80	64.80	103.87	**328.67**	328.67	0.00	-0.01	0.00	0.00
B8	357.68	F&O	36.46	–	**357.68**	373.95	16.85	105.07	359.70	359.70	0.00	0.56	0.56	0.56
B9	330.41	F&O	220.29	–	**402.67**	410.59	18.52	106.52	**402.67**	403.03	0.64	0.00	17.95	18.02
B10	420.99	*LF	8.70	17.20	**462.70**	616.80	147.70	105.67	469.58	469.58	0.00	1.49	10.35	10.35
	392.42		81.94		**407.43**	467.02	51.19	104.78	413.86	414.92	0.73	1.58	5.18	5.42

Table 10.10 – (Continued.) Extensive computational results for the BRKGA-MP-IPR heuristic.

Instance	LB$^+$	Previous BKS						BRKGA-MP-IPR				Rel.Diff (%) previous BKS	Opt. gap (%)	
		Source	Time (sec)	T.Adj (sec)	Best	Avg	SD	Avg time (sec)	Best	Avg	SD		Best	Avg
C1	459.25	F&O	1256.49	–	**957.05**	1001.48	60.67	126.60	965.15	966.64	1.64	0.85	52.42	52.49
C2	373.94	F&O	392.44	–	582.99	610.95	43.52	132.42	**582.22**	582.80	1.15	-0.13	35.77	35.84
C3	390.48	BRKGA	5.84	–	**548.79**	559.53	6.44	128.82	**548.79**	551.41	1.54	0.00	28.85	29.19
C4	371.99	F&O	214.54	–	**507.38**	527.13	25.72	127.71	519.84	522.95	2.53	2.46	28.44	28.87
C5	464.97	F&O	376.06	–	**667.53**	687.64	20.16	127.77	670.61	675.06	4.51	0.46	30.66	31.12
C6	360.73	F&O	1174.61	–	**822.85**	900.74	82.23	127.22	830.30	841.77	2.97	0.91	56.55	57.15
C7	354.15	F&O	417.45	–	**521.89**	540.31	10.63	132.54	528.53	532.65	3.11	1.27	32.99	33.51
C8	375.52	LF	110.80	219.00	**471.00**	522.80	29.80	131.10	471.33	474.67	1.69	0.07	20.33	20.89
C9	355.29	BRKGA	6.62	–	534.30	551.72	9.25	131.01	**531.15**	537.90	4.38	-0.59	33.11	33.95
C10	431.18	F&O	405.08	–	**590.26**	617.35	26.91	128.51	610.86	612.06	1.81	3.49	29.41	29.55
	393.75		435.99		**620.40**	651.96	31.53	129.37	625.88	629.79	2.53	0.88	37.09	37.48
D1	492.09	F&O	2670.26	–	**1149.70**	1202.98	66.99	166.18	1179.85	1196.97	6.57	2.62	58.29	58.89
D2	385.83	F&O	1208.17	–	690.21	761.25	50.31	155.33	**684.28**	693.70	6.95	-0.86	43.61	44.38
D3	385.34	BRKGA	13.95	–	635.67	650.29	7.63	168.48	**633.43**	640.72	3.55	-0.35	39.17	39.86
D4	419.21	F&O	1578.76	–	**798.99**	828.36	23.85	158.69	800.17	827.96	14.03	0.15	47.61	49.37
D5	418.44	F&O	1462.66	–	688.53	741.15	35.31	163.74	**682.77**	691.52	4.19	-0.84	38.71	39.49
D6	393.92	F&O	1201.58	–	**712.15**	764.51	31.68	169.51	**712.11**	727.49	6.28	-0.01	44.68	45.85
D7	372.49	BRKGA	15.03	–	590.64	603.65	6.42	175.32	**585.58**	592.97	5.10	-0.86	36.39	37.18
D8	409.35	BRKGA	13.84	–	661.78	681.10	12.47	168.60	**657.81**	663.39	3.85	-0.60	37.77	38.29
D9	394.65	F&O	785.45	–	**671.23**	726.78	28.43	162.42	692.93	711.92	11.58	3.23	43.05	44.56
D10	485.63	BRKGA	11.00	–	1208.71	1294.66	66.45	155.17	**1201.68**	1225.40	19.83	-0.58	59.59	60.37
	415.69		896.07		**780.76**	825.47	32.95	164.34	783.06	797.20	8.19	0.29	46.91	47.86

Table 10.10 – (Continued.) Extensive computational results for the BRKGA-MP-IPR heuristic.

Instance	LB$^+$	Previous BKS						BRKGA-MP-IPR				Rel.Diff (%) previous BKS	Opt. gap (%)	
		Source	Time (sec)	T.Adj (sec)	Best	Avg	SD	Avg time (sec)	Best	Avg	SD		Best	Avg
E1	437.82	BRKGA	24.46	–	1331.49	1352.33	10.75	217.90	**1318.89**	1334.23	10.04	-0.95	66.80	67.19
E2	444.88	BRKGA	23.57	–	848.08	869.45	15.00	216.35	**824.18**	846.40	15.30	-2.82	46.02	47.44
E3	465.31	BRKGA	23.10	–	788.03	815.31	15.16	214.39	**785.10**	795.12	6.71	-0.37	40.73	41.48
E4	412.08	BRKGA	24.69	–	711.19	731.24	11.75	220.07	**703.11**	719.59	8.71	-1.14	41.39	42.73
E5	416.62	BRKGA	23.10	–	781.50	806.81	13.73	215.20	**760.49**	777.38	10.69	-2.69	45.22	46.41
E6	418.12	BRKGA	24.44	–	788.08	803.28	7.86	220.59	**772.09**	789.17	8.39	-2.03	45.85	47.02
E7	402.04	BRKGA	26.69	–	711.11	731.83	9.38	229.40	**705.85**	718.66	6.45	-0.74	43.04	44.06
E8	444.35	F&O	4619.17	–	745.18	808.44	30.05	215.46	**725.62**	744.82	9.33	-2.62	38.76	40.34
E9	453.86	BRKGA	22.44	–	921.78	950.89	13.90	211.47	**877.08**	917.28	16.77	-4.85	48.25	50.52
E10	458.81	BRKGA	23.99	–	825.24	847.64	14.48	217.22	**814.97**	829.15	9.64	-1.24	43.70	44.66
	435.39		483.57		845.17	871.72	14.21	217.80	**828.74**	847.18	10.20	-1.94	47.46	48.61
F1	548.88	BRKGA	89.15	–	1372.69	1421.76	17.34	528.29	**1342.08**	1374.11	17.30	-2.23	59.10	60.06
F2	543.32	BRKGA	95.02	–	1336.33	1383.55	23.01	555.84	**1296.03**	1324.05	15.99	-3.02	58.08	58.97
F3	547.64	BRKGA	91.24	–	1263.39	1288.66	14.08	541.07	**1201.14**	1228.36	13.57	-4.93	54.41	55.42
F4	532.31	BRKGA	110.09	–	1124.24	1145.53	10.23	631.01	**1093.90**	1108.59	13.77	-2.70	51.34	51.98
F5	538.14	BRKGA	101.85	–	1316.54	1361.64	20.75	584.98	**1286.79**	1316.58	14.78	-2.26	58.18	59.13
F6	525.95	BRKGA	85.45	–	1322.89	1369.38	25.66	517.20	**1295.58**	1316.40	11.30	-2.06	59.40	60.05
F7	512.98	BRKGA	107.73	–	1131.27	1163.97	17.53	615.18	**1085.26**	1117.70	18.93	-4.07	52.73	54.10
F8	540.05	BRKGA	96.05	–	1132.77	1165.67	16.72	557.76	**1098.74**	1132.50	12.09	-3.00	50.85	52.31
F9	543.16	BRKGA	101.87	–	1293.78	1347.78	22.91	586.57	**1257.97**	1292.89	19.93	-2.77	56.82	57.99
F10	546.84	BRKGA	98.96	–	1418.53	1446.56	15.68	574.09	**1357.96**	1394.40	15.24	-4.27	59.73	60.78
	537.93		97.74		1271.24	1309.45	18.39	569.20	**1231.55**	1260.56	15.29	-3.12	56.32	57.33

Table 10.10 – (Continued.) Extensive computational results for the BRKGA-MP-IPR heuristic.

Instance	LB$^+$	Previous BKS						BRKGA-MP-IPR				Rel.Diff (%) previous BKS	Opt. gap (%)	
		Source	Time (sec)	T.Adj (sec)	Best	Avg	SD	Avg time (sec)	Best	Avg	SD		Best	Avg
G1	612.37	BRKGA	221.55	–	1778.54	1851.25	32.13	1182.79	**1726.53**	1750.11	19.03	-2.92	64.53	65.01
G2	614.82	BRKGA	243.95	–	1824.74	1899.33	34.71	1277.04	**1714.04**	1770.65	21.63	-6.07	64.13	65.28
G3	614.20	BRKGA	216.58	–	1514.23	1546.44	13.92	1160.42	**1441.85**	1475.60	15.29	-4.78	57.40	58.38
G4	604.30	BRKGA	253.52	–	1564.42	1599.35	23.92	1320.43	**1490.08**	1523.43	19.27	-4.75	59.45	60.33
G5	633.66	BRKGA	211.75	–	1694.50	1750.03	28.07	1120.27	**1615.69**	1650.55	22.18	-4.65	60.78	61.61
G6	621.46	BRKGA	264.39	–	1714.38	1779.29	21.68	1384.55	**1656.79**	1683.23	19.84	-3.36	62.49	63.08
G7	603.48	BRKGA	224.19	–	1640.07	1677.66	22.99	1191.06	**1567.74**	1593.97	19.76	-4.41	61.51	62.14
G8	618.74	BRKGA	221.25	–	1547.63	1582.71	20.05	1168.12	**1477.58**	1505.58	15.99	-4.53	58.13	58.90
G9	668.16	BRKGA	229.19	–	1942.21	1974.16	19.69	1195.74	**1827.74**	1875.77	34.55	-5.89	63.44	64.38
G10	638.92	BRKGA	205.69	–	1872.08	1931.99	25.31	1095.57	**1781.16**	1828.04	28.94	-4.86	64.13	65.05
	623.01		229.20		1709.28	1759.22	24.25	1209.60	**1629.92**	1665.69	21.65	-4.64	61.78	62.60

Source: the author.

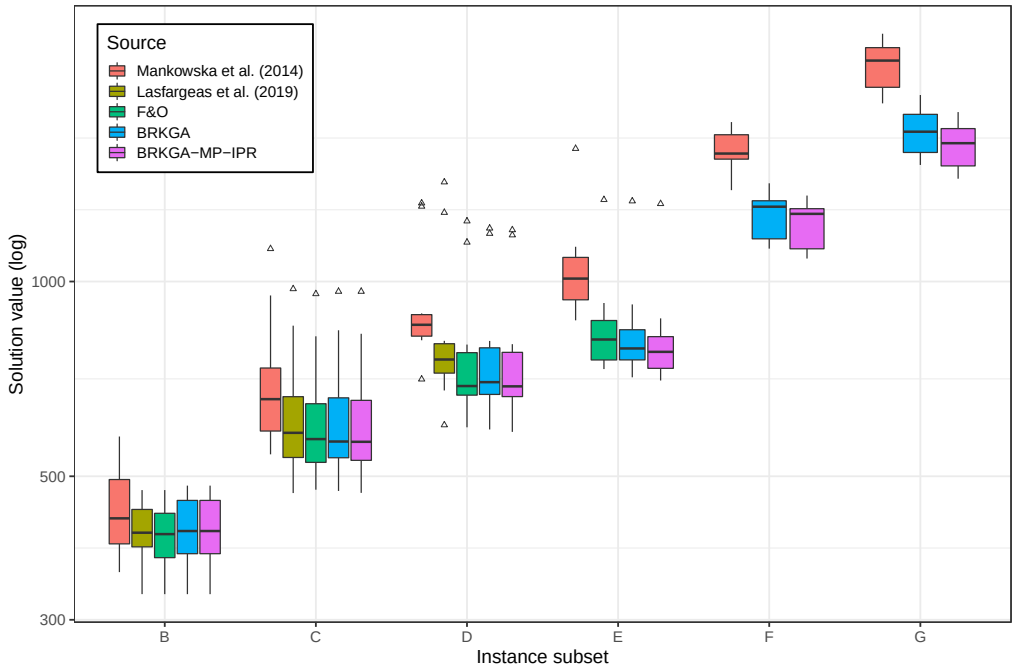
Regarding small and moderate-sized instances, the fix-and-optimize matheuristic stills the best solution method. Between the test cases of up 50 patients, the BRKGA-MP-IPR produces worse results in 1.58% than the previous BKS for subset C. The only notable exception is for C9, in which BRKGA-MP-IPR improved the best solution by 0.59% from the previous result from BRKGA. The results for subset D are mixed, but the genetic algorithm managed to find new solutions for seven out of 10 instances. Despite that, we verify a very diminished reduction of costs between 0.01% (instance D6) of up 0.86% (instances D2 and D7).

Subset E onwards seems to benefit from the more sophisticated genetic algorithm. We observed an average reduction of 1.94% of the previous BKS. Regarding the individual test cases, the BRKGA-MP-IPR improved the solution of E3 by only 0.37% and the solution for E9 by up 4.85%. The improvement is even more impressive for subsets F and G, to which we found an average cost reduction of 3.12% and 4.64%, respectively. Moreover, the algorithm showed the most promising result in the test case G2, in which the new solution improved the previous BKS by 6.07%. Figure 10.2 illustrates the behavior of the best solutions produced by all the methods we propose, including results from the literature. We suppressed results for subset A because these instances can be easily solved with CPLEX. Once again, the evidence indicates that BRKGA-MP-IPR is the best method when solving instances with 100 patients or more.

One may concern regarding the statistical difference between BRKGA and BRKGA-MP-IPR. With a significance level of 5%, a paired two-sided Wilcoxon signed-rank test indicates that the best solution of BRKGA and BRKGA-MP-IPR are statistically different ($p = 1.684\text{e-}09$). The statistical difference becomes more evident when applying the test to average solution values ($p = 5.572\text{e-}11$), although both tests indicate that the algorithms produce statistically distinct solutions.

A natural caveat of using a more sophisticated version of the genetic algorithm is the heuristic's general increase in computational times. Taking subset G as an example, we observe that BRKGA-MP-IPR consumed 5.28 times more time to complete than the simpler BRKGA. After some meticulous analysis of heuristic parameters, we verified that the meta-heuristic runtime scales are almost linear in the number of independent populations. Furthermore, a single run of IPR is not exactly time-consuming, but running this intensification component often also impacts overall performance. According to the parameters from Table 10.9, the IPR is triggered ten times during the entire execution of the genetic algorithm, which becomes significant when solving large test cases.

Figure 10.2 – Boxplot of the average best solutions for each instance subset. We purposely suppressed data for subset A because these instances are easily solved to optimality, and their results distort the graph significantly.



Source: the authors.

We also run a small component analysis for BRKGA-MP-IPR. For that, we run two new experiments with the meta-heuristics, considering the following variations of the meta-heuristic.

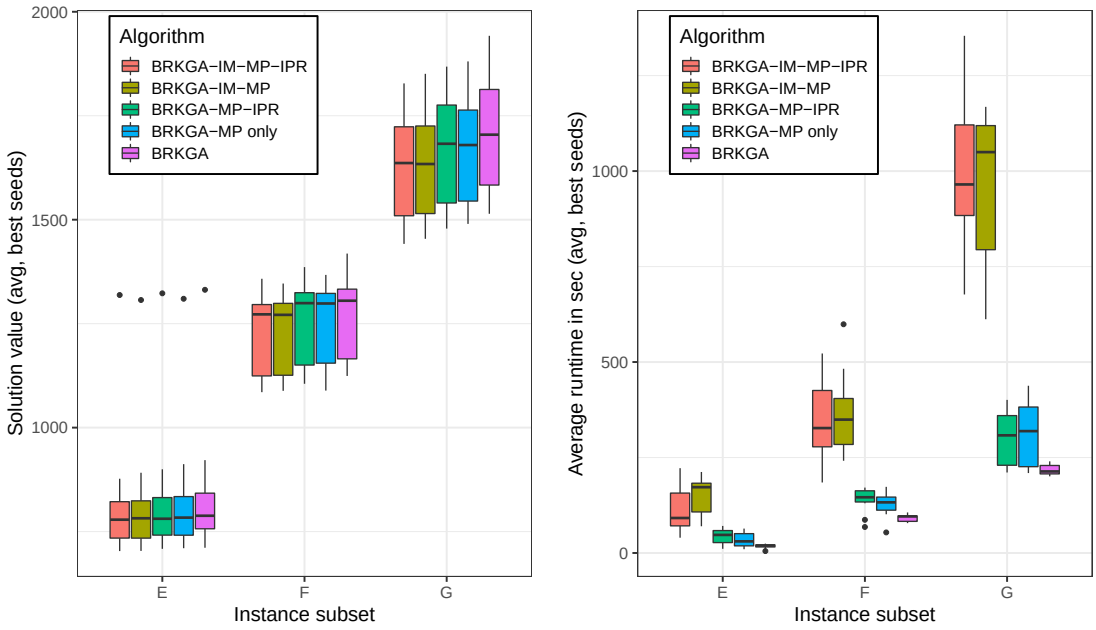
- **BRKGA.** The meta-heuristic based on the "standard" algorithm of Gonçalves and Resende (2011);
- **BRKGA-IM-MP-IPR**. The meta-heuristic with all the extensions proposed by Andrade et al. (2021), as we described in Section 8.4;
- **BRKGA-MP.** The meta-heuristic augmented only by the multi-parent mating component;
- **BRKGA-IM-MP.** The meta-heuristic augmented only by the multi-parent mating, and the multiple islands components;
- **BRKGA-MP-IPR.** The meta-heuristic augmented only by the multi-parent mating, and the implicit path relinking components;

Figure 10.3 illustrates the behavior of all these variations of the genetic algorithm, both in the average best solutions produced per instance subset and in the algorithms' runtimes. Regarding Figure 10.3a, we can draw three conclusions. First, the best-performing algorithm between the options is *BRKGA-IM-MP-IPR*, as evidenced by results for subset G.

Secondly, we observed that the *BRKGA-IM-MP-IPR* behaves almost the same with the IPR disabled. This observation is also supported by a paired two-sided Wilcoxon signed-rank test, which indicates no statistical evidence that two algorithms produce different results ($p = 0.3624$), at a significance level of 5%. Third, the multiple islands component seems to be, in large part, the most responsible for the superior results of *BRKGA-IM-MP-IPR*.

Additionally, it is not like the IPR heuristic has no effect at all. By comparing the *BRKGA-MP-IPR* against *BRKGA-MP*, the former algorithm takes advantage of the IPR component and produces better results. Regarding the algorithm runtimes, we already know that the multiple islands are responsible for the increase in the runtimes. Furthermore, the multi-parent mating and the IPR heuristic impact the algorithm runtime a little. Thus, usage scenarios that can not afford the increased runtimes of *BRKGA-IM-MP-IPR* still benefit from the *BRKGA-MP-IPR* variant.

Figure 10.3 – Genetic algorithm behavior regarding their components.
(a) Effect in solution cost (b) Effect in algorithm runtime



Source: the author.

Additional information regarding the solutions produced by the meta-heuristic is available in Appendix E. If one prefer, the tabulated data is also available in this attachment.

10.6.3 Extensive results for the new dataset

As we mentioned in Subsection 9.3.4, our methodology to generate a new interesting benchmark dataset consists of generating many instances for each case we want to represent, then solve these "candidade" instances to obtain lower bounds and upper bounds and to finally select the instances with the largest relative gap between their LB and UB. For providing the lower bounds, our algorithm of choice was to solve the linear relaxation of the MIP model from Subsection 5.1. Despite our knowledge regarding stronger lower bounds (c.f. Chapter 6), we could not employ such a methodology due to the time required in all the computations.

We have eight instance subsets, and we consider all the 22 combinations of parameters supported by the library underneath the generator for each subset. Then, for each subset and combination, we generate 100 instances. In total, we generated 17,600 instances, and we solved each one using both the model and the "standard" BRKGA. Taking an average runtime of five minutes for the heuristic and ten minutes for building and solving the model, the estimated number of days to evaluate all the instances is about 153 days if we disregard the time spent generating the instances themselves. Fortunately, we could cut our time estimates by a third by employing the distributed processing of our three computers.

Recall that we selected 20 test cases for each instance subset by selecting the 10 "hardest" instances among the 100, plus the other ten instances at random. This way, the final dataset comprises in total of 160 instances. This is twice the number of instances tested by Bredström and Rönnqvist (2008) and for Mankowska, Meisel and Bierwirth (2014) The list of instances that compose our dataset is available in Appendix F.

We employed our best-performing heuristic BRKGA-MP-IPR to solve each of these new instances 20 times. Table 10.11 presents the results for this set of instances. The first column identifies the instance, followed by the three columns with data regarding the instance generation. Column *LB* indicates the linear programming lower bound of the instance, followed by the upper bound obtained by BRKGA, and the reason the instance was selected to compose the final dataset. The columns below *BRKGA-MP-IPR* present the result for the genetic algorithm regarding the average meta-heuristic runtime (column *Avg time (sec)*), the best solution produced (column *Best*), the average solution value (column *Avg*), and standard deviation among solution values (column *SD*). The following columns present the optimality gap for the best and average solution values, respectively, regarding

the value of column *LB*. The last three columns present the average value for the three components of the HHCRSP objective function: the total travel time (D, in minutes), the total tardiness among all patients (T, also in minutes), and the tardiest visit ($T^{\max}$, in minutes).

Regarding the instance naming, we follow the templace "HHCRSP_<number of patients>_<number of caregivers>_<seed used in the instance generator>_<cluster density>_<depot placement>_<patient placement>".

The first significant result regarding these instances is that the values of the objective function are meaningful. Thanks to how the instances were generated, both information of travel times and visit tardiness represent values in minutes. We made this for easing the understanding of the solution cost, e.g., by the HHC practitioners of Porto Alegre. Regarding computational results, we first observe that instances selected by "hardness" have in general quicker runtimes than the instances sampled at random, within the same subset. Compared to Mankowska, Meisel and Bierwirth (2014), the new instances consume almost the same computational times.

Despite the instance sizes, we frequently observe that "hard" instances have an average travel time greater than the "sampled" counterpart. This difference seems to be more evident in "hard" instances when the depot is placed at random, which indicates that in practice, the best option would be to keep the operation center close to the region covered by the provider, corroborating with results from the literature. The negative effect of the random placement of the operation center is more or less reduced when we also place the patient nodes at random or with low-density clusters. Furthermore, the hardest instances, in general, features random placement of the depot, with the patients spread within high-density clusters (densities greater than 1.0). As a result, this causes the caregivers to have long arcs when departing and arriving at the depot, which artificially increases the difficulty of these instances. For this reason, we made the entire 17,600 instances available, so one could apply more convoluted selection strategies to compose their own benchmark set.

The data of Table 10.11 is also available in this attachment.

Table 10.11 – Extensive results for the new dataset with BRKGA-MP-IPR.

Instance	Gen. characteristics			BRKGA-MP-IPR				Opt. gap (%)		Avg. sol. ind.		
	LB	UB	Sel	Avg time (sec)	Best	Avg	SD	Best	Avg	D	T	$T^{\max}$
HHCRSP_10_3_88_1.6_R_C	16.69	150.70	Hardness	43.72	150.67	150.67	0.00	88.92	88.92	335.00	79.00	38.00
HHCRSP_10_3_40_0.8_R_C	18.13	129.70	Hardness	43.91	129.67	129.67	0.00	86.02	86.02	377.00	6.00	6.00
HHCRSP_10_3_40_1.0_R_C	18.13	129.70	Hardness	43.96	129.67	129.67	0.00	86.02	86.02	377.00	6.00	6.00
HHCRSP_10_3_35_1.0_R_R	55.70	385.00	Hardness	43.90	385.00	385.00	0.00	85.53	85.53	355.00	545.00	255.00
HHCRSP_10_3_88_1.0_R_C	17.68	113.70	Hardness	43.78	113.67	113.67	0.00	84.44	84.44	341.00	0.00	0.00
HHCRSP_10_3_88_1.2_R_C	17.68	113.70	Hardness	44.06	113.67	113.67	0.00	84.44	84.44	341.00	0.00	0.00
HHCRSP_10_3_81_1.2_R_C	38.06	241.70	Hardness	43.76	241.67	241.67	0.00	84.25	84.25	309.00	341.00	75.00
HHCRSP_10_3_86_1.2_C_C	31.77	198.70	Hardness	43.83	198.67	198.67	0.00	84.01	84.01	346.00	175.00	75.00
HHCRSP_10_3_88_1.6_C_C	14.35	89.30	Hardness	43.82	89.33	89.33	0.00	83.94	83.94	99.00	115.00	54.00
HHCRSP_10_3_16_1.6_C_C	31.03	183.70	Hardness	44.30	183.67	183.67	0.00	83.10	83.10	273.00	216.00	62.00
	25.92	173.59		43.90	173.57	173.57	0.00	85.07	85.07	315.30	148.30	57.10
HHCRSP_10_3_96_1.0_R_C	36.75	126.70	Sampling	44.19	126.67	126.67	0.00	70.99	70.99	380.00	0.00	0.00
HHCRSP_10_3_46_1.6_R_C	33.35	86.00	Sampling	43.93	86.00	86.00	0.00	61.23	61.23	234.00	12.00	12.00
HHCRSP_10_3_80_1.0_C_R	42.36	102.30	Sampling	60.96	102.33	102.33	0.00	58.60	58.60	269.00	27.00	11.00
HHCRSP_10_3_56_1.2_R_RC	42.36	92.30	Sampling	51.60	92.33	92.33	0.00	54.12	54.12	277.00	0.00	0.00
HHCRSP_10_3_22_1.6_R_C	42.02	79.00	Sampling	44.09	79.00	79.00	0.00	46.81	46.81	237.00	0.00	0.00
HHCRSP_10_3_11_0.4_R_C	61.02	110.30	Sampling	44.01	110.33	110.33	0.00	44.70	44.70	331.00	0.00	0.00
HHCRSP_10_3_51_1.0_R_RC	54.68	97.00	Sampling	44.14	97.00	97.00	0.00	43.63	43.63	289.00	1.00	1.00
HHCRSP_10_3_90_1.2_C_C	42.34	64.30	Sampling	43.76	64.33	64.33	0.00	34.18	34.18	193.00	0.00	0.00
HHCRSP_10_3_90_1.2_C_RC	49.68	75.00	Sampling	43.89	75.00	75.00	0.00	33.77	33.77	225.00	0.00	0.00
HHCRSP_10_3_60_1.0_R_RC	57.01	83.00	Sampling	44.04	83.00	83.00	0.00	31.31	31.31	249.00	0.00	0.00
	46.15	91.59		46.46	91.60	91.60	0.00	49.61	49.61	268.40	4.00	2.40

Table 10.11 – (Continued.) Extensive results for the new dataset with BRKGA-MP-IPR.

Instance	Gen. characteristics			BRKGA-MP-IPR				Opt. gap (%)		Avg. sol. ind.		
	LB	UB	Sel	Avg time (sec)	Best	Avg	SD	Best	Avg	D	T	$T^{\max}$
HHCRSP_25_5_22_1.0_C_C	55.02	899.30	Hardness	49.07	820.33	831.47	10.93	93.29	93.38	621.45	1552.60	320.35
HHCRSP_25_5_69_1.6_R_RC	72.73	1171.00	Hardness	49.01	1149.00	1149.00	0.00	93.67	93.67	494.00	2440.00	513.00
HHCRSP_25_5_47_0.8_R_C	53.70	848.70	Hardness	49.15	848.00	848.00	0.00	93.67	93.67	555.30	1505.75	482.95
HHCRSP_25_5_37_0.8_R_RC	93.38	1441.30	Hardness	49.23	1434.67	1434.67	0.00	93.49	93.49	620.00	3144.00	540.00
HHCRSP_25_5_37_1.0_R_RC	93.38	1441.30	Hardness	49.15	1434.67	1434.67	0.00	93.49	93.49	620.00	3144.00	540.00
HHCRSP_25_5_26_0.8_C_C	62.35	956.30	Hardness	49.20	956.33	956.33	0.00	93.48	93.48	580.00	1804.00	485.00
HHCRSP_25_5_6_0.8_R_C	48.35	706.00	Hardness	49.55	681.00	687.60	9.94	92.90	92.97	575.15	1198.00	289.65
HHCRSP_25_5_69_1.6_R_C	52.67	715.30	Hardness	49.06	695.00	696.06	0.55	92.42	92.43	586.20	1252.00	250.00
HHCRSP_25_5_42_0.8_R_RC	93.08	1196.70	Hardness	48.98	1161.67	1168.45	5.11	91.99	92.03	702.70	2329.75	472.90
HHCRSP_25_5_96_0.8_R_C	57.38	730.30	Hardness	49.49	707.67	707.67	0.00	91.89	91.89	634.00	1216.00	273.00
	68.20	1010.62		49.19	988.83	991.39	2.65	93.10	93.12	598.88	1958.61	416.69
HHCRSP_25_5_80_0.8_C_RC	116.06	437.30	Sampling	49.07	436.67	436.67	0.00	73.42	73.42	732.00	417.00	161.00
HHCRSP_25_5_48_1.0_C_C	78.38	280.30	Sampling	49.53	275.00	277.18	2.24	71.50	71.72	517.55	238.60	75.40
HHCRSP_25_5_42_0.8_C_RC	101.02	287.70	Sampling	49.41	287.67	287.67	0.00	64.88	64.88	523.85	264.15	75.00
HHCRSP_25_5_56_0.8_R_RC	63.34	179.70	Sampling	50.09	178.33	178.80	0.66	64.48	64.58	522.40	7.00	7.00
HHCRSP_25_5_76_1.2_C_C	71.70	184.00	Sampling	49.65	183.67	183.67	0.00	60.96	60.96	533.00	12.00	6.00
HHCRSP_25_5_70_1.0_R_C	65.02	146.00	Sampling	50.02	146.00	146.00	0.00	55.47	55.47	438.00	0.00	0.00
HHCRSP_25_5_21_1.6_R_RC	93.67	205.00	Sampling	50.04	204.67	204.67	0.00	54.24	54.24	607.00	4.00	3.00
HHCRSP_25_5_73_0.4_R_RC	73.69	155.30	Sampling	49.59	151.00	152.85	0.82	51.20	51.79	443.55	7.50	7.50
HHCRSP_25_5_97_1.0_R_R	123.35	258.70	Sampling	49.41	258.67	258.67	0.00	52.31	52.31	706.00	46.00	24.00
HHCRSP_25_5_49_0.4_C_C	86.34	141.70	Sampling	49.86	138.67	138.69	0.07	37.74	37.75	416.05	0.00	0.00
	87.26	227.57		49.66	226.04	226.49	0.38	61.40	61.47	543.94	99.63	35.89

Table 10.11 – (Continued.) Extensive results for the new dataset with BRKGA-MP-IPR.

Instance	Gen. characteristics			BRKGA-MP-IPR				Opt. gap (%)		Avg. sol. ind.		
	LB	UB	Sel	Avg time (sec)	Best	Avg	SD	Best	Avg	D	T	$T^{\max}$
HHCRSP_50_10_80_1.0_C_C	102.19	855.30	Hardness	58.27	826.67	829.05	1.47	87.64	87.67	1025.60	1120.00	341.55
HHCRSP_50_10_81_1.0_R_RC	133.37	1053.70	Hardness	58.12	999.67	1010.04	12.06	86.66	86.80	1374.85	1316.40	338.85
HHCRSP_50_10_80_1.0_C_RC	152.02	1081.30	Hardness	58.28	1057.33	1063.25	1.76	85.62	85.70	1015.20	1825.95	348.60
HHCRSP_50_10_97_0.8_C_RC	129.70	784.00	Hardness	59.21	759.67	770.72	11.18	82.93	83.17	1070.70	1019.35	222.10
HHCRSP_50_10_32_1.2_R_C	85.68	473.70	Hardness	60.76	442.33	463.30	14.24	80.63	81.51	919.55	384.10	86.25
HHCRSP_50_10_3_0.8_R_C	112.01	617.30	Hardness	60.90	589.00	596.18	5.42	80.98	81.21	972.80	589.75	226.00
HHCRSP_50_10_80_0.4_C_C	144.69	788.70	Hardness	58.77	762.67	762.84	0.51	81.03	81.03	1059.95	1008.55	220.00
HHCRSP_50_10_26_1.0_C_RC	121.01	629.30	Hardness	59.83	610.33	611.10	0.68	80.17	80.20	709.30	862.00	262.00
HHCRSP_50_10_26_1.2_R_C	67.33	349.30	Hardness	60.91	312.00	317.02	4.23	78.42	78.76	950.80	0.15	0.10
HHCRSP_50_10_44_1.6_R_C	60.00	305.30	Hardness	62.56	279.33	284.55	4.17	78.52	78.91	852.85	0.40	0.40
	110.80	693.79		59.76	663.90	670.80	5.57	83.31	83.48	995.16	812.67	204.59
HHCRSP_50_10_3_1.6_C_C	66.67	272.70	Sampling	61.58	262.00	266.25	2.71	74.55	74.96	798.30	0.30	0.15
HHCRSP_50_10_53_1.2_R_C	82.67	261.30	Sampling	60.59	250.67	254.92	2.33	67.02	67.57	764.45	0.15	0.15
HHCRSP_50_10_62_1.2_R_C	74.00	216.00	Sampling	62.81	211.33	218.25	3.55	64.98	66.09	654.55	0.10	0.10
HHCRSP_50_10_74_1.0_R_RC	113.67	319.00	Sampling	60.31	290.33	291.93	1.71	60.85	61.06	866.40	6.25	3.15
HHCRSP_50_10_61_1.0_R_RC	107.00	288.30	Sampling	62.71	276.33	279.62	2.20	61.28	61.73	838.35	0.25	0.25
HHCRSP_50_10_88_1.6_C_RC	130.00	342.70	Sampling	61.92	317.67	320.93	1.45	59.08	59.49	962.50	0.15	0.15
HHCRSP_50_10_60_0.4_C_RC	107.01	281.30	Sampling	62.30	266.33	267.38	1.30	59.82	59.98	801.10	0.65	0.40
HHCRSP_50_10_2_1.0_C_R	155.67	370.70	Sampling	60.77	345.00	348.70	1.39	54.88	55.36	1046.10	0.00	0.00
HHCRSP_50_10_100_1.6_C_C	75.00	162.30	Sampling	61.92	159.00	159.00	0.00	52.83	52.83	477.00	0.00	0.00
HHCRSP_50_10_33_0.4_R_RC	140.33	287.00	Sampling	62.12	279.67	280.64	0.97	49.82	49.99	841.50	0.30	0.10
	105.20	280.13		61.70	265.83	268.76	1.76	60.43	60.86	805.03	0.82	0.45

Table 10.11 – (Continued.) Extensive results for the new dataset with BRKGA-MP-IPR.

Instance	Gen. characteristics			BRKGA-MP-IPR				Opt. gap (%)		Avg. sol. ind.		
	LB	UB	Sel	Avg time (sec)	Best	Avg	SD	Best	Avg	D	T	$T^{\max}$
HHCRSP_75_15_97_1.0_R_RC	144.35	924.30	Hardness	72.50	798.00	814.52	8.76	81.91	82.28	1701.00	531.65	210.90
HHCRSP_75_15_35_0.8_R_C	130.34	774.30	Hardness	72.90	719.00	730.60	6.63	81.87	82.16	1281.90	665.90	244.00
HHCRSP_75_15_35_1.0_C_RC	195.68	1145.30	Hardness	73.27	1051.33	1058.83	3.84	81.39	81.52	1474.35	1402.20	299.95
HHCRSP_75_15_97_1.0_C_C	116.67	662.30	Hardness	74.86	612.33	635.02	6.41	80.95	81.63	1030.75	674.40	199.80
HHCRSP_75_15_79_1.2_R_C	89.33	495.00	Hardness	79.09	404.33	421.37	11.19	77.91	78.80	1261.55	1.35	1.20
HHCRSP_75_15_10_1.6_R_C	78.67	426.00	Hardness	74.60	376.00	390.67	6.78	79.08	79.86	1165.90	4.25	1.85
HHCRSP_75_15_88_1.6_R_C	98.67	522.70	Hardness	79.34	461.33	480.05	9.22	78.61	79.45	1438.90	0.75	0.50
HHCRSP_75_15_98_1.6_R_C	81.51	423.30	Hardness	76.71	375.67	393.70	9.25	78.30	79.30	1180.00	0.65	0.45
HHCRSP_75_15_21_1.6_R_C	80.67	411.30	Hardness	81.12	384.67	389.02	2.21	79.03	79.26	1166.70	0.20	0.15
HHCRSP_75_15_87_1.6_C_C	78.67	400.30	Hardness	76.80	386.33	388.57	2.48	79.64	79.75	1164.65	0.70	0.35
	109.46	618.48		76.12	556.90	570.23	6.68	80.35	80.81	1286.57	328.21	95.92
HHCRSP_75_15_97_0.4_C_C	162.03	541.30	Sampling	74.83	480.00	483.37	1.99	66.24	66.48	1336.05	60.05	54.00
HHCRSP_75_15_63_1.0_R_RC	147.33	453.30	Sampling	78.22	408.00	413.10	3.54	63.89	64.33	1237.80	0.85	0.65
HHCRSP_75_15_5_0.8_C_RC	183.33	503.30	Sampling	78.09	439.67	446.17	4.25	58.30	58.91	1336.50	1.35	0.65
HHCRSP_75_15_19_0.8_R_C	141.01	377.00	Sampling	77.86	368.67	377.88	4.75	61.75	62.68	1132.80	0.50	0.35
HHCRSP_75_15_84_0.8_R_RC	162.01	431.30	Sampling	76.96	392.33	401.47	4.14	58.70	59.64	1202.45	1.15	0.80
HHCRSP_75_15_30_1.0_R_C	128.00	328.70	Sampling	80.97	303.67	309.13	3.17	57.85	58.59	923.55	2.00	1.85
HHCRSP_75_15_32_1.2_R_RC	151.67	380.00	Sampling	76.31	353.67	357.68	2.90	57.12	57.60	1065.90	4.80	2.35
HHCRSP_75_15_40_0.8_R_RC	184.33	452.70	Sampling	82.88	423.67	429.10	2.72	56.49	57.04	1283.45	2.10	1.75
HHCRSP_75_15_63_0.4_R_C	142.33	340.30	Sampling	78.69	317.33	322.50	3.40	55.15	55.87	966.55	0.55	0.40
HHCRSP_75_15_65_1.0_C_C	118.67	280.00	Sampling	78.81	257.00	259.53	2.01	53.83	54.28	778.40	0.10	0.10
	152.07	408.79		78.36	374.40	379.99	3.29	59.38	59.98	1126.35	7.35	6.29

Table 10.11 – (Continued.) Extensive results for the new dataset with BRKGA-MP-IPR.

Instance	Gen. characteristics			BRKGA-MP-IPR				Opt. gap (%)		Avg. sol. ind.		
	LB	UB	Sel	Avg time (sec)	Best	Avg	SD	Best	Avg	D	T	$T^{\max}$
HHCRSP_100_20_8_1.2_R_C	117.00	729.70	Hardness	105.73	607.33	635.73	13.90	80.74	81.60	1901.70	3.55	1.95
HHCRSP_100_20_63_1.2_R_C	112.00	655.70	Hardness	107.07	548.33	570.03	7.65	79.57	80.35	1709.65	0.30	0.15
HHCRSP_100_20_39_1.6_R_C	112.34	648.00	Hardness	99.28	554.33	567.50	7.30	79.73	80.20	1697.75	3.15	1.60
HHCRSP_100_20_76_1.6_R_C	100.67	568.00	Hardness	98.36	500.00	510.58	6.28	79.87	80.28	1530.90	0.55	0.30
HHCRSP_100_20_57_1.6_R_C	105.00	574.70	Hardness	105.03	505.67	523.40	9.46	79.24	79.94	1565.70	2.65	1.85
HHCRSP_100_20_25_0.8_R_C	138.33	750.70	Hardness	97.75	654.67	670.80	12.76	78.87	79.38	2010.30	1.35	0.75
HHCRSP_100_20_77_1.0_R_C	134.67	730.00	Hardness	105.58	613.67	640.15	14.39	78.06	78.96	1913.45	4.60	2.40
HHCRSP_100_20_11_1.2_R_C	118.00	634.00	Hardness	95.21	535.33	551.07	6.21	77.96	78.59	1651.30	1.10	0.80
HHCRSP_100_20_30_1.0_R_C	133.33	707.00	Hardness	101.69	621.67	635.85	9.63	78.55	79.03	1903.50	2.75	1.30
HHCRSP_100_20_23_1.2_R_C	102.00	528.70	Hardness	107.64	454.00	467.57	6.83	77.53	78.18	1402.35	0.20	0.15
	117.33	652.65		102.33	559.50	577.27	9.44	79.03	79.67	1728.66	2.02	1.13
HHCRSP_100_20_68_1.6_R_C	87.00	351.30	Sampling	115.38	304.67	313.48	3.88	71.44	72.25	940.35	0.05	0.05
HHCRSP_100_20_49_0.4_C_C	211.33	586.30	Sampling	103.04	525.67	536.42	5.54	59.80	60.60	1605.50	2.45	1.30
HHCRSP_100_20_61_0.4_R_RC	187.00	510.30	Sampling	110.32	471.33	477.83	4.04	60.33	60.87	1430.35	1.70	1.45
HHCRSP_100_20_88_1.2_C_C	111.33	301.70	Sampling	99.41	269.67	273.62	2.97	58.72	59.31	820.85	0.00	0.00
HHCRSP_100_20_9_1.0_R_R	237.67	625.30	Sampling	110.90	554.67	564.48	5.95	57.15	57.90	1690.75	1.95	0.75
HHCRSP_100_20_35_1.6_C_RC	185.33	481.30	Sampling	103.82	442.67	453.77	6.41	58.13	59.16	1361.00	0.15	0.15
HHCRSP_100_20_10_1.2_R_C	132.67	340.30	Sampling	103.62	300.00	308.20	3.09	55.78	56.95	924.60	0.00	0.00
HHCRSP_100_20_65_0.4_R_RC	212.00	533.70	Sampling	104.77	495.00	503.87	6.51	57.17	57.93	1507.75	2.25	1.60
HHCRSP_100_20_99_0.4_C_RC	216.33	543.70	Sampling	107.93	487.00	492.63	4.16	55.58	56.09	1476.10	1.15	0.65
HHCRSP_100_20_54_1.2_R_RC	176.00	435.30	Sampling	101.42	398.67	410.53	5.49	55.85	57.13	1230.30	0.70	0.60
	175.67	470.92		106.06	424.94	433.48	4.80	58.66	59.48	1298.76	1.04	0.66

Table 10.11 – (Continued.) Extensive results for the new dataset with BRKGA-MP-IPR.

Instance	Gen. characteristics			BRKGA-MP-IPR				Opt. gap (%)		Avg. sol. ind.		
	LB	UB	Sel	Avg time (sec)	Best	Avg	SD	Best	Avg	D	T	$T^{\max}$
HHCRSP_200_40_52_1.6_R_C	171.67	1109.70	Hardness	276.99	932.00	972.72	26.07	81.58	82.35	2914.25	2.55	1.35
HHCRSP_200_40_7_1.2_R_C	187.00	1207.70	Hardness	263.19	1081.33	1113.22	20.26	82.71	83.20	3329.70	6.15	3.80
HHCRSP_200_40_30_1.6_R_C	180.33	1143.00	Hardness	280.84	923.67	953.12	18.50	80.48	81.08	2856.05	1.95	1.35
HHCRSP_200_40_76_1.2_R_C	218.00	1352.00	Hardness	301.42	1133.33	1194.73	42.47	80.76	81.75	3578.65	3.65	1.75
HHCRSP_200_40_7_1.6_R_C	149.67	923.00	Hardness	263.28	847.67	876.15	18.13	82.34	82.92	2626.25	1.25	0.95
HHCRSP_200_40_51_1.0_R_C	214.00	1317.30	Hardness	292.73	1136.67	1206.20	24.01	81.17	82.26	3615.95	1.65	1.00
HHCRSP_200_40_17_1.6_R_C	170.67	1005.00	Hardness	340.20	835.00	858.85	17.73	79.56	80.13	2574.10	1.50	0.95
HHCRSP_200_40_40_1.0_R_C	200.67	1128.30	Hardness	297.18	913.00	942.42	20.24	78.02	78.71	2825.80	0.95	0.50
HHCRSP_200_40_71_1.6_R_C	182.00	1018.30	Hardness	274.77	857.67	883.70	21.26	78.78	79.40	2646.10	3.25	1.75
HHCRSP_200_40_98_1.2_R_C	202.00	1114.30	Hardness	314.45	936.00	973.03	25.78	78.42	79.24	2914.50	3.00	1.60
	187.60	1131.86		290.50	959.63	997.41	23.45	80.45	81.19	2988.14	2.59	1.50
HHCRSP_200_40_17_1.6_C_C	158.00	650.00	Sampling	343.10	569.67	577.58	5.00	72.26	72.64	1732.15	0.35	0.25
HHCRSP_200_40_75_1.6_R_C	200.33	791.70	Sampling	296.38	679.00	693.62	7.97	70.50	71.12	2079.50	0.75	0.60
HHCRSP_200_40_80_1.0_C_C	214.67	756.70	Sampling	372.74	635.33	646.52	7.72	66.21	66.80	1937.35	1.35	0.85
HHCRSP_200_40_82_1.0_C_R	329.00	1123.00	Sampling	296.24	944.00	977.73	25.33	65.15	66.35	2929.05	2.75	1.40
HHCRSP_200_40_69_1.0_C_R	342.67	1136.00	Sampling	341.85	991.67	1015.40	25.32	65.45	66.25	3042.05	2.90	1.25
HHCRSP_200_40_60_0.4_C_RC	306.33	1005.00	Sampling	268.88	870.67	900.08	12.94	64.82	65.97	2695.60	3.20	1.45
HHCRSP_200_40_10_1.6_C_RC	309.00	1002.30	Sampling	323.51	843.33	859.15	9.71	63.36	64.03	2576.45	0.55	0.45
HHCRSP_200_40_98_1.6_C_RC	294.00	944.30	Sampling	311.00	830.33	844.53	12.36	64.59	65.19	2531.75	1.10	0.75
HHCRSP_200_40_100_0.4_C_C	268.33	824.30	Sampling	342.13	711.67	726.58	7.71	62.30	63.07	2177.70	1.30	0.75
HHCRSP_200_40_71_1.6_C_C	182.33	511.70	Sampling	298.01	458.67	468.78	7.10	60.25	61.11	1406.05	0.15	0.15
	260.47	874.50		319.38	753.43	771.00	12.12	65.43	66.22	2310.77	1.44	0.79

Table 10.11 – (Continued.) Extensive results for the new dataset with BRKGA-MP-IPR.

Instance	Gen. characteristics			BRKGA-MP-IPR				Opt. gap (%)		Avg. sol. ind.		
	LB	UB	Sel	Avg time (sec)	Best	Avg	SD	Best	Avg	D	T	$T^{\max}$
HHCRSP_300_60_4_1.6_R_C	264.00	1976.00	Hardness	881.04	1568.67	1629.28	41.78	83.17	83.80	4879.85	5.20	2.75
HHCRSP_300_60_89_1.6_R_C	244.00	1805.00	Hardness	672.63	1431.33	1507.85	49.72	82.95	83.82	4518.65	3.20	1.70
HHCRSP_300_60_82_1.2_R_C	256.67	1852.70	Hardness	729.01	1499.67	1586.38	39.05	82.89	83.82	4748.15	7.35	3.65
HHCRSP_300_60_95_1.2_R_C	276.00	1946.00	Hardness	845.22	1596.67	1696.10	58.33	82.71	83.73	5078.65	6.60	3.05
HHCRSP_300_60_22_1.2_R_C	271.33	1852.70	Hardness	823.69	1458.67	1522.58	53.99	81.40	82.18	4562.20	4.15	1.40
HHCRSP_300_60_85_1.2_R_C	271.33	1767.30	Hardness	715.23	1412.33	1479.60	31.16	80.79	81.66	4422.65	12.40	3.75
HHCRSP_300_60_81_1.6_R_C	250.00	1627.00	Hardness	785.84	1230.67	1313.85	40.99	79.69	80.97	3933.90	5.25	2.40
HHCRSP_300_60_23_1.2_R_C	241.67	1567.00	Hardness	823.18	1271.33	1322.22	31.81	80.99	81.72	3961.05	3.50	2.10
HHCRSP_300_60_34_1.2_R_C	288.67	1867.70	Hardness	801.04	1635.00	1709.00	33.08	82.34	83.11	5105.95	15.55	5.50
HHCRSP_300_60_69_1.0_R_C	298.00	1920.00	Hardness	789.10	1530.00	1579.32	36.26	80.52	81.13	4723.30	11.35	3.30
	266.17	1818.14		786.60	1463.43	1534.62	41.62	81.81	82.66	4593.44	7.46	2.96
HHCRSP_300_60_7_1.2_R_RC	382.00	1696.70	Sampling	627.71	1434.33	1486.10	36.19	73.37	74.30	4445.15	9.80	3.35
HHCRSP_300_60_75_1.2_C_RC	342.33	1327.70	Sampling	762.75	1088.33	1134.57	36.60	68.55	69.83	3400.05	2.55	1.10
HHCRSP_300_60_26_0.4_C_RC	399.33	1528.00	Sampling	689.58	1284.67	1343.53	32.52	68.92	70.28	4023.75	5.00	1.85
HHCRSP_300_60_68_0.8_C_RC	391.33	1479.30	Sampling	847.66	1182.67	1230.22	34.69	66.91	68.19	3684.85	3.95	1.85
HHCRSP_300_60_4_1.0_R_C	329.67	1200.30	Sampling	890.83	1016.00	1078.32	32.34	67.55	69.43	3230.40	3.20	1.35
HHCRSP_300_60_73_0.8_R_RC	377.67	1369.00	Sampling	740.73	1175.67	1234.60	22.04	67.88	69.41	3697.55	4.45	1.80
HHCRSP_300_60_45_1.6_C_C	248.33	854.70	Sampling	834.72	726.00	747.50	17.19	65.79	66.78	2239.90	1.65	0.95
HHCRSP_300_60_46_0.4_R_C	387.00	1269.00	Sampling	801.14	1068.00	1113.27	30.07	63.76	65.24	3336.80	1.95	1.00
HHCRSP_300_60_81_0.8_C_C	302.00	933.70	Sampling	806.91	814.00	835.68	16.41	62.90	63.86	2505.20	1.15	0.70
HHCRSP_300_60_48_0.8_C_C	327.33	980.70	Sampling	811.38	826.00	850.58	15.99	60.37	61.52	2550.90	0.45	0.40
	348.70	1263.91		781.34	1061.57	1105.44	27.40	67.15	68.46	3311.46	3.42	1.44

Table 10.11 – (Continued.) Extensive results for the new dataset with BRKGA-MP-IPR.

Instance	Gen. characteristics			BRKGA-MP-IPR				Opt. gap (%)		Avg. sol. ind.		
	LB	UB	Sel	Avg time (sec)	Best	Avg	SD	Best	Avg	D	T	$T^{\max}$
HHCRSP_400_80_59_1.0_R_C	330.67	2891.00	Hardness	1602.10	2400.67	2464.35	42.39	86.23	86.58	7355.70	28.85	8.50
HHCRSP_400_80_16_1.2_R_C	309.00	2669.00	Hardness	1631.95	2201.33	2311.37	57.52	85.96	86.63	6907.15	20.50	6.45
HHCRSP_400_80_51_1.0_R_C	342.00	2811.70	Hardness	1522.17	2237.33	2425.50	74.50	84.71	85.90	7236.15	31.60	8.75
HHCRSP_400_80_97_1.6_R_C	275.67	2165.30	Hardness	1733.84	1759.67	1820.27	37.83	84.33	84.86	5435.25	18.50	7.05
HHCRSP_400_80_30_1.6_R_C	287.00	2173.70	Hardness	1474.27	1834.00	1908.13	51.92	84.35	84.96	5704.70	14.65	5.05
HHCRSP_400_80_34_1.0_R_C	307.00	2302.30	Hardness	1605.12	2064.67	2145.38	48.64	85.13	85.69	6406.10	22.95	7.10
HHCRSP_400_80_92_1.6_R_C	291.33	2180.70	Hardness	1495.99	1791.33	1851.02	37.13	83.74	84.26	5535.05	13.25	4.75
HHCRSP_400_80_91_1.2_R_C	335.00	2476.00	Hardness	1608.22	1909.33	2025.52	63.13	82.45	83.46	6058.05	14.00	4.50
HHCRSP_400_80_28_1.2_R_C	365.67	2692.30	Hardness	1553.66	2079.00	2172.27	63.25	82.41	83.17	6484.65	25.80	6.35
HHCRSP_400_80_81_1.6_R_C	294.00	2131.00	Hardness	1710.40	1743.00	1815.32	43.09	83.13	83.80	5428.35	13.30	4.30
	313.73	2449.30		1593.77	2002.03	2093.91	51.94	84.33	85.02	6255.12	20.34	6.28
HHCRSP_400_80_54_1.2_C_RC	467.00	1945.70	Sampling	1757.88	1562.33	1675.90	43.19	70.11	72.13	5016.05	8.60	3.05
HHCRSP_400_80_20_0.4_C_RC	502.00	2088.30	Sampling	1430.20	1636.33	1731.70	52.77	69.32	71.01	5183.30	8.70	3.10
HHCRSP_400_80_61_0.4_C_RC	469.33	1948.30	Sampling	1627.19	1625.00	1666.07	31.91	71.12	71.83	4990.00	5.90	2.30
HHCRSP_400_80_33_1.0_C_RC	457.33	1876.30	Sampling	1718.09	1527.33	1620.13	47.74	70.06	71.77	4849.60	8.25	2.55
HHCRSP_400_80_95_1.0_C_R	484.33	1934.70	Sampling	1660.34	1547.33	1600.53	46.48	68.70	69.74	4792.75	6.25	2.60
HHCRSP_400_80_65_1.0_C_R	523.67	1879.70	Sampling	1650.58	1618.00	1689.10	42.77	67.63	69.00	5054.75	9.50	3.05
HHCRSP_400_80_45_0.4_C_C	437.67	1509.00	Sampling	1526.54	1265.67	1324.67	31.99	65.42	66.96	3968.00	4.35	1.65
HHCRSP_400_80_22_1.6_C_C	324.67	1106.30	Sampling	1725.22	948.00	1001.80	24.67	65.75	67.59	3000.60	3.40	1.40
HHCRSP_400_80_77_1.2_R_RC	471.00	1576.70	Sampling	1904.06	1447.33	1504.32	32.14	67.46	68.69	4505.20	5.60	2.15
HHCRSP_400_80_60_0.8_C_C	406.67	1286.30	Sampling	1413.34	1065.67	1128.53	24.54	61.84	63.97	3380.85	3.10	1.65
	454.37	1715.13		1641.34	1424.30	1494.28	37.82	68.10	69.59	4474.11	6.37	2.35

Source: the author.

11 CONCLUSIONS AND FUTURE WORKS

We propose in this thesis a few algorithms for solving the home health care routing and scheduling problem, which consists of a routing problem to minimize costs and maximize the service levels and satisfaction of both caregivers and patients. In the problem we approached in this thesis, such service levels are measured according to visit tardiness, given the patients' preferred time window. In case of violation of such a time window, then additional costs are incurred. Travel times represent the operational costs of the problem. Some additional constraints are set for patients requiring multiple visits by caregivers with distinct qualifications. In such cases, these multiple visits must start simultaneously or operate within a predefined amount of minimum and maximum separation times between them. This problem is hard to solve at optimality, and these additional constraints define what the literature calls route inter-dependency constraints (DREXL, 2012).

We propose several scenarios employing a MIP solver for obtaining stronger lower bounds for the problem. Despite promising, these scenarios rely on solver-specific features such as *memory emphasis flags* to trigger reduced memory consumption. In any way, the strategies of saving memory pay it out. After a comprehensive experiment, we verified that CPLEX could benefit from providing redundant cuts and a feasible solution through the MIP warm start. Conversely, as the instance size grows, the best strategy is only to toggle flags to reduce memory consumption.

We propose a fix-and-optimize matheuristic as our first method studied to solve the problem. This matheuristic attempts to improve pairs of routes of caregivers, selecting these routes either randomly or following a simple guiding heuristic. After evaluating algorithm variations with a single decomposition scheme by time, we concluded that the best approach is to combine both the guided and random decomposition schemes within the matheuristic. Unfortunately, this solution method could not scale well on larger test cases, and we have some evidence that traces this issue back to the route inter-dependency constraints in the problem.

Due to the inability of the F&O to solve larger test cases, we moved further and developed some variations of meta-heuristic algorithms. Thanks to our literature review, we have been aware of the difficulties arriving when solving problems with such complicated constraints as soft time windows and multiple visits. In observation with Drexl (2012) comment of indirectly exploring the solution space of such tough problems, we devise a biased random-key genetic algorithm for the HHCRSP. Despite the simplicity, the heuristic

produced high-quality solutions, especially for medium-sized to larger test cases. In great part, the effectiveness of such an algorithm is due to the decoding strategy we employed, which embeds a greedy best-insertion heuristic on it. Furthermore, there is still room for improving such decoder, mainly because its greediness makes it blind to important characteristics of the problem, such as the triangle inequality.

We devised another genetic algorithm that embeds the new components proposed by Andrade et al. (2021) to evaluate if our genetic algorithm would benefit from an additional intensification mechanism. These components act as intensification mechanisms in several aspects of the genetic algorithm. First, the multi-parent mating allows combining promising characteristics from several parents, instead of the 2-parent approach from the "standard" algorithm of Gonçalves and Resende (2011). Secondly, we employed an implicit path-relinking algorithm that further intensifies the search by exploring intermediate solutions indirectly through the random-keys space. The third component consists of an island model that allows evolving of several independent populations in parallel. This component allows the meta-heuristic to immigrate individuals between populations from time to time, which helps to combine promising characteristics individually reached by distinct populations. We also run a component analysis for such a meta-heuristic, and we verify that these new components work well together to improve the algorithm's outputs.

Lastly, we propose a methodology for generating new realistic instances for home health care problems. This methodology uses state-of-art algorithms and databases when generating new random instances. Thanks to how the generator was structured, one can extract a library for obtaining realistic travel times for real problems, using data from the OpenStreetMaps and algorithms from Open Source Routing Machine. Handling this material becomes easy through the open-source library `ovig`. Using this library, we generate in total 17,600 new random instances for Porto Alegre. Inspired by Vallada, Ruiz and Framinan (2015) work, we devise a methodology to filter these instances and provide a so-called curated dataset. For that, we need to employ algorithms to compute lower and upper bounds for all the +17k instances, which consumed almost 50 days of distributed processing across three computers. After all this processing is done, we select 160 of these instances to compose the "final dataset" of this thesis.

Furthermore, we select 80 of these instances according to their hardness and the others 80 at random. The reasoning behind that is that our hardness measure is biased to the algorithms we employed when calculating the instances' LBs and UBs. Thus, we expect to circumvent this issue to some degree by this sampling strategy. Finally, we

provide extensive results for this “final dataset” from our best-performing meta-heuristic.

As future work, we want to run a more comprehensive experiment with our new dataset to understand better which characteristics make a hard instance of the problem. Regarding the meta-heuristic, we still need to develop a smarter decoding algorithm capable of circumventing the difficulties arousing due to its greedy nature. On the matheuristic, we still need to verify the reasons for the algorithm's bad performance when solving larger instances. We also think that we could further improve the matheuristic by devising more sophisticated decomposition schemes, maybe spending a little more time to select which caregiver routes to optimize in future iterations. Similarly, we think that the fix-and-optimize would benefit some memory mechanism to keep track of what pairs of routes have been tried so far, mainly because the guided decomposition selects the same caregiver routes from time to time. This issue also happens when the random decomposition is selected but is unlikely in test cases with ten caregivers or more.

We have some initial work on route recombination-based algorithms for set partitioning problems. This approach seems promising, and some initial experiments indicate that mixing the genetic algorithm as a route generator for a set partition MIP model can lead to better solutions than the ones from the genetic algorithm solely. The arriving question here is how we could better handle the route synchronization constraints because using MIP makes adjusting the visit times very hard. For that reason, we think that the heuristic method is the way to go. Otherwise, we could try some state-space reduction strategy to eliminate unpromising arcs from the HHCRP MIP model (GUEDES et al., 2016).

REFERENCES

AARDAL, K. Reformulation of capacitated facility location problems: How redundant information can help. **Annals of Operations Research**, Springer, v. 82, p. 289–308, 1998.

ACHTERBERG, T.; KOCH, T.; MARTIN, A. Branching rules revisited. **Operations Research Letters**, Elsevier, v. 33, n. 1, p. 42–54, 2005.

AIANE, D.; EL-AMRAOUI, A.; MESGHOUNI, K. A new optimization approach for a home health care problem. **Proceedings of 2015 International Conference on Industrial Engineering and Systems Management, IEEE IESM 2015**, n. October, p. 285–290, 2016.

AKJIRATIKARL, C.; YENRADEE, P.; DRAKE, P. R. PSO-based algorithm for home care worker scheduling in the UK. **Computers & Industrial Engineering**, Elsevier, v. 53, n. 4, p. 559–583, 2007.

ANDRADE, C. E. et al. The multi-parent biased random-key genetic algorithm with implicit path-relinking and its real-world applications. **European Journal of Operational Research**, v. 289, n. 1, p. 17–30, 2021. ISSN 0377-2217. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S0377221719309488>.

BALLOU, R. H.; SRIVASTAVA, S. K. **Business logistics/supply chain management: planning, organizing, and controlling the supply chain**. [S.l.]: Pearson Education India, 2007.

BASHIR, B.; CHABROL, M.; CAUX, C. Literature review in home care. In: **9th International Conference on Modeling, Optimization & Simulation**. [S.l.: s.n.], 2012.

BEAN, J. C. Genetic algorithms and random keys for sequencing and optimization. **ORSA journal on computing**, INFORMS, v. 6, n. 2, p. 154–160, 1994.

BEKTAŞ, T.; ERDOĞAN, G.; RØPKE, S. Formulations and branch-and-cut algorithms for the generalized vehicle routing problem. **Transportation Science**, Informs, v. 45, n. 3, p. 299–316, 2011.

BENZARTI, E.; SAHIN, E.; DALLERY, Y. Operations management applied to home care services: Analysis of the districting problem. **Decision Support Systems**, v. 55, n. 2, p. 587 – 598, 2013. ISSN 0167-9236.

BERTELS, S.; FAHLE, T. A hybrid setup for a hybrid scenario: combining heuristics for the home health care problem. **Computers & Operations Research**, Elsevier, v. 33, n. 10, p. 2866–2890, 2006.

BIRATTARI, M. et al. A racing algorithm for configuring metaheuristics. In: **Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2002. (GECCO'02), p. 11–18. ISBN 1558608788.

BLAIS, M.; LAPIERRE, S. D.; LAPORTE, G. Solving a home–care districting problem in an urban setting. **Journal of the Operational Research Society**, Taylor & Francis, v. 54, n. 11, p. 1141–1147, 2003.

BORSANI, V. et al. A home care scheduling model for human resources. In: IEEE. **2006 International Conference on Service Systems and Service Management**. [S.l.], 2006. v. 1, p. 449–454.

BRAEKERS, K. et al. A bi-objective home care scheduling problem: Analyzing the trade-off between costs and client inconvenience. **European Journal of Operational Research**, Elsevier, v. 248, n. 2, p. 428–443, 2016.

BRÄYSY, O.; GENDREAU, M. Vehicle routing problem with time windows, part i: Route construction and local search algorithms. **Transportation Science**, INFORMS, v. 39, n. 1, p. 104–118, 2005.

BREDSTROM, D.; RONNQVIST, M. A branch and price algorithm for the combined vehicle routing and scheduling problem with synchronization constraints. **SSRN Electronic Journal**, n. 2, 2007. ISSN 1556-5068.

BREDSTRÖM, D.; RÖNNQVIST, M. Combined vehicle routing and scheduling with temporal precedence and synchronization constraints. **European Journal of Operational Research**, Elsevier, v. 191, n. 1, p. 19–31, 2008.

BRESINA, J. L. Heuristic-biased stochastic sampling. In: **Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 1**. [S.l.]: AAAI Press, 1996. (AAAI'96), p. 271–278. ISBN 026251091X.

CARELLO, G.; LANZARONE, E. A cardinality-constrained robust model for the assignment problem in home care services. **European Journal of Operational Research**, Elsevier B.V., v. 236, n. 2, p. 748–762, 2014. ISSN 03772217.

CASTILLO-SALAZAR, J. A.; LANDA-SILVA, D.; QU, R. Workforce scheduling and routing problems: Literature survey and computational study. **Annals of Operations Research**, Springer, v. 239, n. 1, p. 39–67, 2016.

CHAHED, S. et al. Exploring new operational research opportunities within the home care context: The chemotherapy at home. **Health Care Management Science**, Springer, v. 12, n. 2, p. 179–191, 2009.

CHAVES, A. A. et al. Hybrid method with CS and BRKGA applied to the minimization of tool switches problem. **Computers & Operations Research**, Elsevier, v. 67, p. 174–183, 2016.

CHEN, H. Fix-and-optimize and variable neighborhood search approaches for multi-level capacitated lot sizing problems. **Omega**, Elsevier, v. 56, p. 25–36, 2015.

CHENG, E.; RICH, J. L. **A home health care routing and scheduling problem**. [S.l.], 1998. Available in <https://hdl.handle.net/1911/101899>.

CISSÉ, M. et al. Or problems related to home health care: A review of relevant routing and scheduling problems. **Operations Research for Health Care**, Elsevier, v. 13, p. 1–22, 2017.

DANTZIG, G. B.; RAMSER, J. H. The truck dispatching problem. **Management Science**, Informs, v. 6, n. 1, p. 80–91, 1959.

DASKIN, M. S.; DEAN, L. K. Location of health care facilities. In: **Operations Research and Health Care**. [S.l.]: Springer, 2005. p. 43–76.

DE ANGELIS, V. Planning home assistance for AIDS patients in the city of Rome, Italy. **Interfaces**, INFORMS, v. 28, n. 3, p. 75–83, 1998.

DECERLE, J. et al. A memetic algorithm for a home health care routing and scheduling problem. **Operations Research for Health Care**, Elsevier Ltd, v. 16, p. 59–71, 2018. ISSN 22116923.

DESAULNIERS, G.; DESROSIERS, J.; SOLOMON, M. M. **Column generation**. [S.l.]: Springer Science & Business Media, 2006.

DOHN, A.; KOLIND, E.; CLAUSEN, J. The manpower allocation problem with time windows and job-teaming constraints: A branch-and-price approach. **Computers & Operations Research**, v. 36, n. 4, p. 1145–1157, 2009.

DOHN, A.; RASMUSSEN, M. S.; LARSEN, J. The vehicle routing problem with time windows and temporal dependencies. **Networks**, Wiley Online Library, v. 58, n. 4, p. 273–289, 2011.

DORNELES, Á. P.; ARAÚJO, O. C. de; BURIOL, L. S. A fix-and-optimize heuristic for the high school timetabling problem. **Computers & Operations Research**, Elsevier, v. 52, p. 29–38, 2014.

DREXL, M. Synchronization in vehicle routing – a survey of VRPs with multiple synchronization constraints. **Transportation Science**, INFORMS, v. 46, n. 3, p. 297–316, 2012.

DROR, M. Note on the complexity of the shortest path models for column generation in vrptw. **Operations Research**, v. 42, n. 5, p. 977–978, 1994.

EBERHART, R.; KENNEDY, J. A new optimizer using particle swarm theory. In: IEEE. **MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science**. [S.l.], 1995. p. 39–43.

EGGENSPERGER, K.; LINDAUER, M.; HUTTER, F. Pitfalls and best practices in algorithm configuration. **arXiv.**, 2019. Available in <https://arxiv.org/abs/1705.06058v3>.

EIBEN, A. E.; SMITH, J. E. et al. **Introduction to evolutionary computing**. [S.l.]: Springer, 2003.

EMILIANO, W.; TELHADA, J.; CARVALHO, M. do S. Home health care logistics planning: a review and framework. **Procedia Manufacturing**, Elsevier, v. 13, p. 948–955, 2017.

ERNST, A. T. et al. Staff scheduling and rostering: A review of applications, methods and models. **European Journal of Operational Research**, Elsevier, v. 153, n. 1, p. 3–27, 2004.

EVEBORN, P.; FLISBERG, P.; RÖNNQVIST, M. Laps care–an operational system for staff planning of home care. **European Journal of Operational Research**, Elsevier, v. 171, n. 3, p. 962–976, 2006.

FERNANDEZ, A. et al. A model for community nursing in a rural county. **Journal of the Operational Research Society**, Taylor & Francis, v. 25, n. 2, p. 231–239, 1974.

FIKAR, C.; HIRSCH, P. A matheuristic for routing real-world home service transport systems facilitating walking. **Journal of Cleaner Production**, Elsevier, v. 105, p. 300–310, 2015.

FIKAR, C.; HIRSCH, P. Home health care routing and scheduling: A review. **Computers & Operations Research**, Elsevier, v. 77, p. 86–95, 2017.

FIKAR, C.; HIRSCH, P. Evaluation of trip and car sharing concepts for home health care services. **Flexible Services and Manufacturing Journal**, Springer, v. 30, n. 1-2, p. 78–97, 2018.

FISHER, M. L. The lagrangian relaxation method for solving integer programming problems. **Management Science**, INFORMS, v. 27, n. 1, p. 1–18, 1981.

GAREY, M.; JOHNSON, D. **Computers and Intractability: A Guide to the Theory of NP-completeness**. [S.l.]: W. H. Freeman, 1979. (Mathematical Sciences Series). ISBN 9780716710448.

GENDREAU, M.; POTVIN, J.-Y. et al. **Handbook of metaheuristics**. [S.l.]: Springer, 2010.

GENET, N. et al. **Home care across Europe: current structure and future challenges**. [S.l.]: World Health Organization. Regional Office for Europe, 2012.

GERSHON, R. R. et al. Home health care patients and safety hazards in the home: preliminary findings. **Advances in patient safety: New directions and alternative approaches (Vol. 1: Assessment)**, Agency for Healthcare Research and Quality, 2008.

GLOVER, F. Tabu search and adaptive memory programming — advances, applications and challenges. In: ____. **Interfaces in Computer Science and Operations Research: Advances in Metaheuristics, Optimization, and Stochastic Modeling Technologies**. Boston, MA: Springer US, 1997. chp. 1, p. 1–75. ISBN 978-1-4615-4102-8.

GNU Project. **Optimize options for GNU GCC 7.3.0**. 2018. Available in <https://gcc.gnu.org/onlinedocs/gcc-7.3.0/gcc/Optimize-Options.html>. Last visited in May 2020.

GOMES, M. I.; RAMOS, T. R. P. Modelling and (re-) planning periodic home social care services with loyalty and non-loyalty features. **European Journal of Operational Research**, Elsevier, v. 277, n. 1, p. 284–299, 2019.

GONÇALVES, J. F.; RESENDE, M. G. Biased random-key genetic algorithms for combinatorial optimization. **Journal of Heuristics**, Springer, v. 17, n. 5, p. 487–525, 2011.

GONÇALVES, J. F.; RESENDE, M. G. A parallel multi-population biased random-key genetic algorithm for a container loading problem. **Computers & Operations Research**, Elsevier, v. 39, n. 2, p. 179–190, 2012.

GONÇALVES, J. F.; RESENDE, M. G.; MENDES, J. J. A biased random-key genetic algorithm with forward-backward improvement for the resource constrained project scheduling problem. **Journal of Heuristics**, Springer, v. 17, n. 5, p. 467–486, 2011.

GONÇALVES, J. F.; WÄSCHER, G. A MIP model and a biased random-key genetic algorithm based approach for a two-dimensional cutting problem with defects. **European Journal of Operational Research**, v. 286, n. 3, p. 867–882, nov 2020. ISSN 03772217.

GRENOUILLEAU, F. et al. A set partitioning heuristic for the home health care routing and scheduling problem. **European Journal of Operational Research**, Elsevier, v. 275, n. 1, p. 295–303, 2019.

GRIECO, L.; UTLEY, M.; CROWE, S. Operational research applied to decisions in home health care: A systematic literature review. **Journal of the Operational Research Society**, Taylor & Francis, p. 1–32, 2020.

GSCHWIND, T.; IRNICH, S. Effective handling of dynamic time windows and its application to solving the dial-a-ride problem. **Transportation Science**, INFORMS, v. 49, n. 2, p. 335–354, 2015.

GUEDES, P. C. et al. Simple and efficient heuristic approach for the multiple-depot vehicle scheduling problem. **Optimization Letters**, Springer, v. 10, n. 7, p. 1449–1461, 2016.

GUTIÉRREZ, E. V.; VIDAL, C. J. Home health care logistics management problems: A critical review of models and methods. **Revista Facultad de Ingeniería Universidad de Antioquia**, Universidad de Antioquia, p. 160–175, 2013.

GUTIÉRREZ, V.; VIDAL, C. J. Inventory management models in supply chains: A literature review. **Revista Facultad de Ingeniería Universidad de Antioquia**, Universidad de Antioquia, n. 43, p. 134–149, 2008.

HADDADENE, S. R. A.; LABADIE, N.; PRODHON, C. A GRASP×ILS for the vehicle routing problem with time windows, synchronization and precedence constraints. **Expert Systems with Applications**, v. 66, p. 274–294, 2016. ISSN 0957–4174.

HAKLAY, M.; WEBER, P. Openstreetmap: User-generated street maps. **IEEE Pervasive computing**, Ieee, v. 7, n. 4, p. 12–18, 2008.

HELBER, S.; SAHLING, F. A fix-and-optimize approach for the multi-level capacitated lot sizing problem. **International Journal of Production Economics**, Elsevier, v. 123, n. 2, p. 247–256, 2010.

HIERMANN, G. et al. Metaheuristics for solving a multimodal home-healthcare scheduling problem. **Central European Journal of Operations Research**, Springer, v. 23, n. 1, p. 89–113, 2015.

HULSHOF, P. J. et al. Taxonomic classification of planning decisions in health care: A structured review of the state of the art in OR/MS. **Health systems**, Taylor & Francis, v. 1, n. 2, p. 129–175, 2012.

IBGE. **Brazilian census of 2010**. 2010. Available at <https://censo2010.ibge.gov.br/resultados.html>. Last visited in April 2021.

IBGE. **Resident population per Brazilian city**. 2010. Available in <https://cidades.ibge.gov.br/brasil/rs/porto-alegre/pesquisa/23/25207?tipo=ranking>. Last visited in August 2021.

IBGE. **Brazilian and Federal Unit Estimations of population**. 2019. Available in <https://www.ibge.gov.br/apps/populacao/projecao/>. Last visited in February 2019.

IBM. **IBM ILOG CPLEX Optimization Studio 20.1**. 2020. <https://community.ibm.com/community/user/datascience/blogs/xavier-nodet1/2020/12/11/cos-201-is-available>. Last visited in March 2021.

KARP, R. M. Reducibility among combinatorial problems. In: **Complexity of computer computations**. [S.l.]: Springer, 1972. p. 85–103.

KERGOSIEN, Y.; LENTÉ, C.; BILLAUT, J.-C. Home health care problem: An extended multiple traveling salesman problem. In: **Proceedings of the 4th multidisciplinary international scheduling conference: theory and applications (MISTA 2009)**. [S.l.: s.n.], 2009. p. 85–92.

KUMMER, A. F.; BURIOL, L. S.; ARAÚJO, O. C. de. A matheuristic algorithm applied to the home health care problem. In: **Eletronic Proceedings of the LI Brazilian Operational Research Symposium**. [S.l.: s.n.], 2019. v. 2. Available in https://proceedings.science/sbpo-2019/papers/a-matheuristic-algorithm-applied-to-the-home-health-care-problem?lang=en.

KUMMER, A. F.; BURIOL, L. S.; ARAÚJO, O. C. de. A biased random key genetic algorithm applied to the vrptw with skill requirements and synchronization constraints. In: **Proceedings of the 2020 Genetic and Evolutionary Computation Conference**. [S.l.: s.n.], 2020. p. 717–724.

LAHRICHI, N. et al. Analysis of a territorial approach to the delivery of nursing home care services based on historical data. **Journal of Medical Systems**, Springer, v. 30, n. 4, p. 283–291, 2006.

LANDERS, S. et al. The future of home health care: a strategic framework for optimizing value. **Home Health Care Management & Practice**, SAGE Publications Sage CA: Los Angeles, CA, v. 28, n. 4, p. 262–278, 2016.

LANZARONE, E.; MATTA, A. Robust nurse-to-patient assignment in home care services to minimize overtimes under continuity of care. **Operations Research for Health Care**, v. 3, n. 2, p. 48–58, 2014. ISSN 2211-6923. Special Issue of the 2012 conference of the EURO working group Operational Research Applied To Health Services (ORAHS).

LANZARONE, E.; MATTA, A.; SAHIN, E. Operations management applied to home care services: the problem of assigning human resources to patients. **IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans**, IEEE, v. 42, n. 6, p. 1346–1363, 2012.

LASFARGEAS, S.; GAGNÉ, C.; SIOUD, A. Solving the home health care problem with temporal precedence and synchronization. In: **Bioinspired Heuristics for Optimization**. [S.l.]: Springer, 2019. p. 251–267.

LIU, R.; TAO, Y.; XIE, X. An adaptive large neighborhood search heuristic for the vehicle routing problem with time windows and synchronized visits. **Computers & Operations Research**, Elsevier, v. 101, p. 250–262, 2019.

LIU, R.; YUAN, B.; JIANG, Z. Mathematical model and exact algorithm for the home care worker scheduling and routing problem with lunch break requirements. **International Journal of Production Research**, Taylor & Francis, v. 55, n. 2, p. 558–575, 2017.

LÓPEZ-IBÁÑEZ, M. et al. The irace package: Iterated racing for automatic algorithm configuration. **Operations Research Perspectives**, Elsevier, v. 3, p. 43–58, 2016.

LUCENA, M. L. et al. Some extensions of biased random-key genetic algorithms. In: **Proceedings of the 46th Brazilian Symposium of Operational Research**. [S.l.: s.n.], 2014. p. 1–12.

LUXEN, D.; VETTER, C. Real-time routing with OpenStreetMap data. In: **Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems**. New York, NY, USA: ACM, 2011. (GIS '11), p. 513–516. ISBN 978-1-4503-1031-4.

MANKOWSKA, D. S.; MEISEL, F.; BIERWIRTH, C. The home health care routing and scheduling problem with interdependent services. **Health care management science**, Springer, v. 17, n. 1, p. 15–30, 2014.

MARCHILDON, G.; ALLIN, S. **Health systems in transition: Canada**. [S.l.]: University of Toronto Press, 2021.

MATSUMOTO, M.; NISHIMURA, T. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. **ACM Transactions on Modeling and Computer Simulation (TOMACS)**, ACM New York, NY, USA, v. 8, n. 1, p. 3–30, 1998.

MCKEE, M. Reducing hospital beds: what are the lessons to be learned? World Health Organization. Regional Office for Europe, 2004.

MELO, M.; NICKEL, S.; SALDANHA-DA-GAMA, F. Facility location and supply chain management — a review. **European Journal of Operational Research**, v. 196, n. 2, p. 401 – 412, 2009. ISSN 0377-2217.

MESQUITA, M.; PAIAS, A. Set partitioning/covering-based approaches for the integrated vehicle and crew scheduling problem. **Computers & Operations Research**, v. 35, n. 5, p. 1562–1575, 2008. ISSN 0305-0548. Part Special Issue: Algorithms and Computational Methods in Feasibility and Infeasibility.

Ministério da Saúde. **Serviço de Atenção Domiciliar "Melhor em Casa"**. 2013. Available in <http://portalms.saude.gov.br/acoes-e-programas/melhor-em-casa-servico-de-atencao-domiciliar/melhor-em-casa>. Last visited in January 2019.

MIRANDA, G. M. D.; MENDES, A. d. C. G.; SILVA, A. L. A. d. O envelhecimento populacional brasileiro: desafios e consequências sociais atuais e futuras. **Revista Brasileira de Geriatria e Gerontologia**, SciELO Brasil, v. 19, p. 507–519, 2016.

MLADENOVIĆ, N.; HANSEN, P. Variable neighborhood search. **Computers & Operations Research**, Elsevier, v. 24, n. 11, p. 1097–1100, 1997.

MOSCATO, P. Memetic algorithms: A short introduction. In: ____. **New Ideas in Optimization**. GBR: McGraw-Hill Ltd., UK, 1999. p. 219–234. ISBN 0077095065.

NASRI, F. O envelhecimento populacional no brasil. **Einstein**, v. 6, n. Supl 1, p. S4–S6, 2008.

NICKEL, S.; SCHRÖDER, M.; STEEG, J. Mid-term and short-term planning support for home health care services. **European Journal of Operational Research**, Elsevier, v. 219, n. 3, p. 574–587, 2012.

OPENADDRESSES. **The free and open global address collection**. 2021. Available online: <https://openaddresses.io/>. Last accessed May 2021.

PAPADAKOS, N. Integrated airline scheduling. **Computers & Operations Research**, Elsevier, v. 36, n. 1, p. 176–195, 2009.

PassMark. **PassMark CPU benchmark dataset**. 2021. Available in <https://www.cpubenchmark.net/>. Last visited in July 2021.

PESSOA, A. et al. A generic exact solver for vehicle routing and related problems. **Mathematical Programming**, Springer, v. 183, n. 1, p. 483–523, 2020.

PEZZELLA, F.; BONANNO, R.; NICOLETTI, B. A system approach to the optimal health-care districting. **European Journal of Operational Research**, Elsevier, v. 8, n. 2, p. 139–146, 1981.

POCHET, Y.; WOLSEY, L. A. **Production planning by mixed integer programming**. [S.l.]: Springer Science & Business Media, 2006.

POLNIK, M.; RICCARDI, A.; AKARTUNAL, K. A multistage optimisation algorithm for the large vehicle routing problem with time windows and synchronised visits. **Journal of the Operational Research Society**, Taylor & Francis, v. 0, n. 0, p. 1–16, aug 2020. ISSN 0160-5682.

RASMUSSEN, M. S. et al. The home care crew scheduling problem: Preference-based visit clustering and temporal dependencies. **European Journal of Operational Research**, Elsevier, v. 219, n. 3, p. 598–610, 2012.

RECHEL, B. et al. Ageing in the European Union. **The Lancet**, v. 381, n. 9874, p. 1312–1322, 2013. ISSN 0140-6736. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S014067361262087X>.

REGIS-HERNÁNDEZ, F.; CARELLO, G.; LANZARONE, E. **An optimization tool to dimension innovative home health care services with devices and disposable materials**. [S.l.]: Springer US, 2020. 561–598 p. ISSN 19366590. ISBN 0123456789.

RESENDE, M. G. C.; RIBEIRO, C. C. Path-relinking. In: ____. **Optimization by GRASP: Greedy Randomized Adaptive Search Procedures**. New York, NY: Springer New York, 2016. chp. 8, p. 167–188. ISBN 978-1-4939-6530-4. Available from Internet: <https://doi.org/10.1007/978-1-4939-6530-4_8>.

ROSENFELD, P.; RUSSELL, D. A review of factors influencing utilization of home and community-based long-term care: Trends and implications to the nursing workforce. **Policy, Politics, & Nursing Practice**, Sage Publications Sage CA: Los Angeles, CA, v. 13, n. 2, p. 72–80, 2012.

RUIZ, E. et al. Solving the open vehicle routing problem with capacity and distance constraints with a biased random key genetic algorithm. **Computers & Industrial Engineering**, Elsevier, v. 133, p. 207–219, 2019.

SARTORI, C. S.; BURIOL, L. S. A study on the pickup and delivery problem with time windows: Matheuristics and new instances. **Computers & Operations Research**, Elsevier, v. 124, p. 105065, 2020.

SAVELSBERGH, M. W. The vehicle routing problem with time windows: Minimizing route duration. **ORSA Journal on Computing**, INFORMS, v. 4, n. 2, p. 146–154, 1992.

SHI, Y.; BOUDOUH, T.; GRUNDER, O. A robust optimization for a home health care routing and scheduling problem with consideration of uncertain travel and service times. **Transportation Research Part E: Logistics and Transportation Review**, Elsevier, v. 128, n. June, p. 52–95, 2019. ISSN 13665545. Available from Internet: <https://doi.org/10.1016/j.tre.2019.05.015>.

STEEG, J.; SCHRÖDER, M. A hybrid approach to solve the periodic home health care problem. In: **Operations Research Proceedings 2007**. [S.l.]: Springer, 2008. p. 297–302.

TAILLARD, É. et al. A tabu search heuristic for the vehicle routing problem with soft time windows. **Transportation Science**, INFORMS, v. 31, n. 2, p. 170–186, 1997.

TAILLARD, É. D.; HELSGAUN, K. POPMUSIC for the travelling salesman problem. **European Journal of Operational Research**, v. 272, n. 2, p. 420–429, 2019. ISSN 0377-2217.

TANGE, O. et al. GNU Parallel—the command-line power tool. **The USENIX Magazine**, <usenix.org>, v. 36, n. 1, p. 42–47, 2011.

TOSO, R. F.; RESENDE, M. G. A C++ application programming interface for biased random-key genetic algorithms. **Optimization Methods and Software**, Taylor & Francis, v. 30, n. 1, p. 81–93, 2015.

TRAUTSAMWIESER, A.; GRONALT, M.; HIRSCH, P. Securing home health care in times of natural disasters. **OR Spectrum**, Springer, v. 33, n. 3, p. 787–813, 2011.

TRAUTSAMWIESER, A.; HIRSCH, P. Optimization of daily scheduling for home health care services. **Journal of Applied Operational Research**, v. 3, n. 3, p. 124–136, 2011.

TRICK, M. Formulations and reformulations in integer programming. In: SPRINGER. **International conference on integration of artificial intelligence (AI) and operations research (OR) techniques in constraint programming**. [S.l.], 2005. p. 366–379.

UCHOA, E. et al. New benchmark instances for the capacitated vehicle routing problem. **European Journal of Operational Research**, Elsevier, v. 257, n. 3, p. 845–858, 2017.

UMA, R.; WEIN, J. On the relationship between combinatorial and lp-based approaches to np-hard scheduling problems. In: SPRINGER. **International Conference on Integer Programming and Combinatorial Optimization**. [S.l.], 1998. p. 394–408.

UMA, R.; WEIN, J.; WILLIAMSON, D. P. On the relationship between combinatorial and lp-based lower bounds for np-hard scheduling problems. **Theoretical Computer Science**, Elsevier, v. 361, n. 2-3, p. 241–256, 2006.

VALLADA, E.; RUIZ, R.; FRAMINAN, J. M. New hard benchmark for flowshop scheduling problems minimising makespan. **European Journal of Operational Research**, Elsevier, v. 240, n. 3, p. 666–677, 2015.

VANDERBECK, F.; SADYKOV, R.; TAHIRI, I. **BaPCod – a generic Branch-And-Price Code**. 2017. Available online: <https://realopt.bordeaux.inria.fr/?page_id=2>. Last accessed June 2021.

VANHOUCKE, M.; MAENHOUT, B. On the characterization and generation of nurse scheduling problem instances. **European Journal of Operational Research**, v. 196, n. 2, p. 457–467, 2009. ISSN 0377-2217. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S0377221708003457>.

WIKIPEDIA. **Geografia de Porto Alegre**. 2021. Available in <https://pt.wikipedia.org/wiki/Geografia_de_Porto_Alegre>. Last visited in July 2021.

WOLSEY, L.; NEMHAUSER, G. **Integer and Combinatorial Optimization**. [S.l.]: Wiley, 1999. (Wiley Series in Discrete Mathematics and Optimization). ISBN 9780471359432.

APPENDIX A — RESUMO EXPANDIDO

Um estudo sobre o problema de roteamento e agendamento para assistência de saúde domiciliar

Esta tese estuda uma variação do problema de atendimento de saúde domiciliar, com ênfase nos subproblemas operacionais de roteamento de equipes de saúde e agendamento de visita dos pacientes (em inglês *Home Health Care Routing and Scheduling Problem*, ou HHCRSP). Esse tipo de problema vem sendo estudado desde 1974, e vem sendo formatado como uma generalização do problema de roteamento de veículos com janela de tempo desde 1998. Esse tipo de problema tem recebido mais atenção nos últimos anos por conta da tendência global de envelhecimento da população (GUTIÉRREZ; VIDAL, 2008; HULSHOF et al., 2012; GRIECO; UTLEY; CROWE, 2020). Nesse contexto, os serviços de atendimento de saúde domiciliar são utilizadas em conjunto ao sistema tradicional de saúde com hospitais, e tem como objetivo a redução dos custos de internação, e o aumento da qualidade de vida dos pacientes (REGIS-HERNÁNDEZ; CARELLO; LANZARONE, 2020). Esse tipo de serviço é chave para implementação das políticas públicas brasileiras de *desospitalização* através do programa nacional *Melhor em Casa* (Ministério da Saúde, 2013).

Dentre as diversas variações da literatura, escolheu-se um problema que compartilha as características mais importantes com o caso real de Porto Alegre (Rio Grande do Sul), na expectativa que os métodos de solução propostos possam ser usados em um futuro próximo. Além disso, a escolha pela variante proposta por Mankowska, Meisel and Bierwirth (2014) levou em conta a relevância científica e computacional do problema. Tal variante considera um conjunto de agentes de saúde para o atendimento de um conjunto de pacientes com demandas por especialidades médicas específicas. Os agentes de saúde são heterogêneos, no sentido que cada agente é qualificado para realizar as visitas de apenas algumas especialidades médicas. Os pacientes, por sua vez, possuem uma janela de tempo ideal para recepção dos agentes de saúde, que pode ser violada caso necessário. Além disso, alguns pacientes exigem múltiplas visitas por equipes distintas, nas quais se exige o cumprimento de restrições adicionais que impactam significativamente na resolução do problema.

A primeira contribuição desta tese são os novos limites inferiores, obtidos para um dataset da literatura, utilizando-se como base os resolvedores CPLEX (IBM, 2020) e VRPSolver (PESSOA et al., 2020). Com CPLEX, estudou-se os efeitos de diversas

parametrizações do resolvedor, assim como os efeitos do fornecimento de uma solução inicial factível, e da geração de cortes redundantes para o problema. Também propôs-se uma etapa de pré-processamento para redução do tamanho dos modelos gerados com CPLEX, que foi utilizada junto a sete cenários de parametrização do resolvedor. Já o VRPSolver foi empregado para resolução de uma relaxação inteira do problema, que supostamente forneceria limites inferiores combinatórios mais fortes que a abordagem com CPLEX. Em todos os casos de teste, fixou-se a execução dos algoritmos em duas horas.

Os resultados computacionais indicaram que CPLEX é o melhor método para obtenção de tais limites. Considerando-se que o VRPSolver foi especificamente desenvolvido para resolução de problemas de roteamento e afins, tal resultado é surpreendente. Utilizando-se o resolvedor CPLEX, encontrou-se novos limites inferiores para 60 das 70 instâncias da literatura. Comparativamente, tais resultados foram obtidos dentro de duas horas de processamento, enquanto a literatura reporta valores após dez horas de execução do resolvedor. Para o subconjunto de instâncias da literatura com 25 pacientes, os novos limites são entre 2,02% a 47,92% melhores que os reportados anteriormente. Já para o subconjunto de instâncias com 100 pacientes, obteve-se novos limites inferiores entre 6,17% e 9,47% apenas. Para os subconjuntos de instâncias com 200 e 300 pacientes (as maiores instâncias do dataset), a melhora média dos limites é de 23,24% e 34,33%, respectivamente. Uma análise geral indica a vantagem de um dos cenários de teste com CPLEX sobre os demais testados, o que serve de guia para experimentos futuros ou experimentos com um conjunto de instâncias diferentes. Além disso, estes novos valores são importantes para avaliar o quanto ainda é possível melhorar as soluções para o dataset de Mankowska, Meisel and Bierwirth (2014).

A segunda contribuição desta tese são os novos limites superiores para o dataset de Mankowska, Meisel and Bierwirth (2014), que foram obtidos por uma *matheurística* "fixa-e-otimiza". Similar aos resultados da literatura, a *matheurística* obteve os valores de solução ótimos para instâncias com 10 pacientes. Já para instâncias com 75 pacientes, o algoritmo proposto obteve soluções, em média, 10,61% melhores que os resultados de Mankowska, Meisel and Bierwirth (2014), e 16,67% melhores que Lasfargeas, Gagné and Sioud (2019). Para instâncias com 100 pacientes, a *matheurística* obteve resultados, em média, 11,46% melhores que Mankowska, Meisel and Bierwirth (2014). Em instâncias maiores, com 200 e 300 pacientes, a *matheurística* se mostrou ineficaz em decorrência do grande número de restrições de interdependência entre rotas. Por fim, uma análise de componentes mostrou que a há vantagem de usar uma combinação das as técnicas de

seleção de rotas propostas, tanto em termos de qualidade de solução quanto em tempo de execução do algoritmo.

Por conta da incapacidade da *matheurística* em otimizar instâncias maiores, propôs-se a resolução do problema por meio de um algoritmo genético de chaves aleatórias viciadas (GONÇALVES; RESENDE, 2011). Nesta abordagem, o algoritmo genético evolui a ordem de execução das visitas dos pacientes, que é então utilizada por uma heurística gulosa para o cálculo das rotas de cada agente de saúde. Tal heurística, portanto, encontra-se embutida dentro do decodificador proposto, e configura a terceira maior contribuição desta tese. Até onde se sabe, esta foi a primeira tentativa de uso de um algoritmo genético de chaves aleatórias viciadas na resolução do problema de assistência de saúde domiciliar.

Em linhas gerais, o melhor método para resolução de instâncias com até 75 pacientes é a *matheurística* fixa-e-otimiza. Nessas instâncias, os resultados computacionais obtidos com o algoritmo genético são mistos, com soluções entre 3,17% piores até 2,51% melhores que as provenientes da *matheurística*. Para o subconjunto de instâncias de 100 pacientes, o algoritmo genético produziu novas soluções para nove dos dez casos de teste, com melhoras entre 0,48% e 5,63% dos resultados obtidos pela *matheurística*. Para os subconjuntos com 200 e 300 pacientes, o algoritmo genético apresentou uma melhora média de 19,94% e 20,91%, respectivamente, em relação aos resultados da literatura.

Algoritmos genéticos de chaves aleatórias viciadas possuem grande capacidade de exploração do espaço de soluções dos problemas, mas uma capacidade de intensificação de busca limitada pelo operador de cruzamento uniforme parametrizável. Tendo em vista esta limitação, Andrade et al. (2021) propuseram uma série de melhorias para o framework do algoritmo genético com chaves aleatórias viciadas com o objetivo de melhorar sua capacidade de intensificação, que foram testadas em conjunto ao decodificador proposto para o algoritmo genético "padrão" de Gonçalves and Resende (2011). Por meio da biblioteca open source `brkga_mp_ipr_cpp`, desenvolveu-se uma nova versão do algoritmo genético que conta com os novos componentes propostos por Andrade et al. (2021): cruzamento com múltiplos progenitores, modelo de ilhas independentes, e heurística de religamento de caminhos implícita no espaço de chaves aleatórias.

Resultados computacionais indicam que, para instâncias até 50 pacientes, não há diferença estatística nos resultados obtidos com e sem os componentes adicionais de intensificação. Em instâncias maiores, os componentes adicionais de intensificação passam a surtir efeito. Para o subconjunto de instâncias com 100 pacientes, as novas soluções são entre 0,74% e 4,85% melhores que os resultados obtidos pelo algoritmo genético

padrão. Para instâncias com 200 e 300 pacientes, a melhora média em relação ao algoritmo genético padrão é de 3,12% e 4,64%, respectivamente.

Um aspecto importante acerca do uso dos componentes de intensificação é seu impacto no tempo de execução do algoritmo. Quando comparado com o algoritmo padrão, o uso dos novos componentes de intensificação aumentam o tempo de execução do algoritmo em duas e seis vezes, nas instâncias com 100 e 300 pacientes, respectivamente. Uma análise de componentes mostrou que a presença de múltiplas ilhas é a maior responsável por esse acréscimo dos tempos de execução. Apesar disso, testes computacionais indicam que o algoritmo genético padrão não alcança resultados comparáveis quando se é permitido uma execução por longos períodos. Este é um forte indício que os responsáveis por tais melhorias nas soluções são, de fato, os componentes adicionais de intensificação.

Por fim, a quarta contribuição desta tese é um gerador de instâncias realísticas para o problema de atendimento de saúde domiciliar, e um conjunto de novas instâncias curadas para o problema. O gerador em questão utiliza de dados geográficos de endereços reais da biblioteca *Open Addresses*, e utiliza dados malha de tráfego da ferramenta *OpenStreetMaps* para cálculo dos tempos de deslocamento realísticos, através por meio da biblioteca *Open Source Routing Machine* (OPENADDRESSES, 2021; HAKLAY; WEBER, 2008; LUXEN; VETTER, 2011). Utilizando-se de dados públicos de Porto Alegre, gerou-se ao todo mais de 17 mil novas instâncias, para as quais se obteve limites inferiores e superiores usando os métodos propostos nesta tese. As novas instâncias foram geradas para diversos tamanhos de problema, desde 10 pacientes e três médicos até 400 pacientes e 80 médicos. Para cada tamanho de problema, gerou-se 2.200 novas instâncias, das quais 20 foram selecionadas para compor o novo dataset da seguinte maneira: selecionou-se as 10 instâncias mais difíceis dentre as 2.200, e 10 outras instâncias ao acaso. Como métrica de dificuldade, utilizou-se o desvio relativo entre os limites inferiores e superiores de cada instância gerada. Após a composição do novo dataset, reportou-se um experimento computacional detalhado, utilizando-se o algoritmo genético com chaves aleatórias viciadas e componentes de intensificação adicionais.

APPENDIX B — LOWER BOUNDS FOR MANKOWSKA ET AL. (2014) DATASET

This appendix presents extensive results for LB^+ experiments with Mankowska, Meisel and Bierwirth (2014) dataset, using the solver parameter described by scenarios in Chapter 6. Column *Instance* presents the instance being solved. Column *Literature* presents the results reported by Mankowska, Meisel and Bierwirth (2014) for the LB^+ computed by CPLEX in ten hours of processing. Column *Scenario 1 (LP)* depicts the solution values when we solve the linear relaxation of the problem. The following six columns present the LB^+ for scenarios 2 to 7, respectively. These values were obtained by running CPLEX with a time limit of two hours. The last column indicates how much we could improve, on average, the best bounds reported by Mankowska, Meisel and Bierwirth (2014), taking into consideration the best result for each instance. Relative differences between -0.02 and 0.02 are spurious due to the rounded values presented in the literature.

Table B.1 – Results for MIP experiments for obtaining stronger LB^+.

Instance	Literature	Scenario 1 (LP)	LB^+						Literature improvement (%)
			Scenario 2	Scenario 3	Scenario 4	Scenario 5	Scenario 6	Scenario 7	
A1	**218.20**	149.08	**218.20**	**218.20**	**218.20**	**218.20**	**218.20**	**218.20**	0.00
A2	**246.60**	144.69	**246.63**	**246.63**	**246.63**	**246.63**	**246.63**	**246.63**	0.01
A3	305.80	127.31	**305.86**	**305.86**	**305.86**	**305.86**	**305.86**	**305.86**	0.02
A4	**186.90**	62.45	**186.90**	**186.90**	**186.90**	**186.90**	**186.90**	**186.90**	0.00
A5	**189.50**	80.73	**189.54**	**189.54**	**189.54**	**189.54**	**189.54**	**189.54**	0.02
A6	**200.10**	78.89	**200.10**	**200.10**	**200.10**	**200.10**	**200.10**	**200.10**	0.00
A7	**225.40**	103.23	**225.37**	**225.37**	**225.37**	**225.37**	**225.37**	**225.37**	-0.01
A8	**232.00**	130.83	**232.05**	**232.05**	**232.05**	**232.05**	**232.05**	**232.05**	0.02
A9	**222.30**	91.01	**222.30**	**222.30**	**222.30**	**222.30**	**222.30**	**222.30**	0.00
A10	**225.00**	137.43	**225.01**	**225.01**	**225.01**	**225.01**	**225.01**	**225.01**	0.00
	225.18	110.57	**225.19**	**225.19**	**225.19**	**225.19**	**225.19**	**225.19**	0.01

Table B.1 – (Continued.) Results for MIP experiments for obtaining stronger LB^+.

Instance	Literature	Scenario 1 (LP)	LB^+						Literature improvement (%)
			Scenario 2	Scenario 3	Scenario 4	Scenario 5	Scenario 6	Scenario 7	
B1	378.70	163.06	**428.10**	**428.10**	398.38	417.73	**428.10**	408.87	13.04
B2	427.90	221.74	**476.05**	**476.05**	450.76	452.55	**476.05**	**476.05**	11.25
B3	391.20	159.19	**399.09**	**399.09**	316.76	306.15	**399.09**	339.51	2.02
B4	330.40	180.53	**411.30**	**411.30**	341.23	336.63	**411.30**	342.68	24.48
B5	311.00	177.04	**366.34**	**366.34**	335.61	333.12	**366.34**	337.41	17.79
B6	274.20	119.77	**405.58**	379.88	271.39	289.81	394.37	296.90	47.92
B7	310.60	158.56	**328.67**	**328.67**	**328.67**	302.39	**328.67**	315.99	5.82
B8	332.40	163.96	**357.68**	**357.68**	**357.68**	346.07	**357.68**	**357.68**	7.61
B9	293.70	120.58	307.37	**330.41**	284.49	276.33	330.30	273.35	12.50
B10	381.00	189.11	391.17	411.99	383.07	398.26	**420.99**	377.91	10.50
	343.11	165.36	387.14	388.95	346.80	345.90	**391.29**	352.63	15.29
C1	401.10	237.78	455.64	451.07	448.79	458.09	**459.25**	452.16	14.50
C2	314.90	195.73	360.08	365.15	354.31	358.67	**373.94**	365.01	18.75
C3	323.50	197.32	379.24	376.08	379.09	379.37	**390.48**	382.56	20.70
C4	329.40	224.54	359.41	356.54	357.39	357.79	**371.99**	362.37	12.93
C5	404.20	215.64	456.59	461.99	448.52	443.01	**464.97**	449.86	15.04
C6	308.20	196.55	347.19	346.80	351.09	350.04	**360.73**	348.67	17.04
C7	315.70	190.21	350.95	350.48	345.89	348.89	**354.15**	347.36	12.18
C8	336.30	194.91	367.95	371.44	362.61	364.50	**375.52**	370.11	11.66
C9	306.50	187.84	346.95	346.31	344.88	345.52	**355.29**	351.55	15.92
C10	386.40	236.90	418.38	418.88	420.84	414.08	**431.18**	424.86	11.59
	342.62	207.74	384.24	384.47	381.34	382.00	**393.75**	385.45	15.03

Table B.1 – (Continued.) Results for MIP experiments for obtaining stronger LB^+.

Instance	Literature	Scenario 1 (LP)	LB^+						Literature improvement (%)
			Scenario 2	Scenario 3	Scenario 4	Scenario 5	Scenario 6	Scenario 7	
D1	456.70	280.90	488.15	489.55	487.57	483.16	**492.09**	487.72	7.75
D2	336.80	230.45	379.63	**385.83**	383.95	383.50	384.68	385.56	14.56
D3	355.70	231.31	380.05	374.39	372.92	375.00	378.99	**385.34**	8.33
D4	379.90	224.35	405.30	**419.21**	412.76	413.01	418.94	411.01	10.35
D5	372.20	263.27	413.21	**418.44**	413.37	415.81	411.16	400.14	12.42
D6	368.80	231.61	365.53	390.45	393.65	392.08	391.87	**393.92**	6.81
D7	333.40	225.92	333.49	363.15	364.14	361.67	**372.49**	371.95	11.72
D8	373.30	262.90	**409.35**	408.78	406.62	404.81	406.40	405.92	9.66
D9	362.40	238.12	385.89	383.59	384.69	381.17	384.91	**394.65**	8.90
D10	434.40	302.61	436.74	472.02	475.56	476.86	**485.63**	484.02	11.79
	377.36	249.14	399.73	410.54	409.52	408.71	**412.71**	412.02	10.23
E1	406.00	281.04	398.13	400.53	430.51	427.74	430.36	**437.82**	7.84
E2	411.90	293.75	**444.88**	437.57	431.76	433.43	431.65	439.54	8.01
E3	437.20	292.80	420.12	424.89	455.69	454.27	451.50	**465.31**	6.43
E4	384.30	256.19	378.97	379.76	410.31	**412.08**	404.07	411.42	7.23
E5	392.40	252.90	386.18	386.26	414.64	**416.62**	409.20	415.13	6.17
E6	390.20	254.18	409.57	411.28	412.90	416.60	416.00	**418.12**	7.16
E7	374.40	266.33	361.16	362.73	392.96	388.71	389.57	**402.04**	7.38
E8	407.70	279.91	393.28	392.85	435.62	433.75	433.89	**444.35**	8.99
E9	422.10	270.75	411.88	411.63	445.60	446.49	439.25	**453.86**	7.53
E10	419.10	252.33	416.80	414.07	455.31	455.07	450.00	**458.81**	9.47
	404.53	270.02	402.10	402.16	428.53	428.47	425.55	**434.64**	7.62

Table B.1 – (Continued.) Results for MIP experiments for obtaining stronger LB^+.

Instance	Literature	Scenario 1 (LP)	LB^+						Literature improvement (%)
			Scenario 2	Scenario 3	Scenario 4	Scenario 5	Scenario 6	Scenario 7	
F1	445.10	377.81	519.58	518.90	546.64	547.88	**548.88**	548.29	23.32
F2	457.90	374.62	515.52	517.52	539.14	541.82	**543.32**	536.17	18.65
F3	481.80	366.38	513.64	520.24	519.04	**547.64**	545.73	532.57	13.67
F4	417.40	379.11	379.11	379.11	**532.31**	527.06	531.84	511.76	27.53
F5	452.30	375.07	509.91	512.99	510.70	514.02	**538.14**	514.28	18.98
F6	367.10	331.05	487.59	486.58	517.49	514.90	518.47	**525.95**	43.27
F7	408.10	353.76	466.82	480.80	508.82	**512.98**	511.80	496.48	25.70
F8	454.30	366.85	502.89	507.02	534.46	534.52	536.15	**540.05**	18.88
F9	426.80	374.74	512.33	514.46	515.27	542.34	**543.16**	541.24	27.26
F10	441.80	376.29	521.36	526.71	529.16	526.53	**546.84**	536.44	23.78
	435.26	367.57	492.88	496.43	525.30	530.97	**536.43**	528.32	24.10
G1	455.00	444.14	OOM	444.14	604.49	**612.37**	603.08	OOM	34.59
G2	463.10	452.83	OOM	OOM	**614.82**	605.84	604.81	OOM	32.76
G3	464.30	454.39	OOM	454.39	606.25	**614.20**	586.04	OOM	32.28
G4	461.50	451.33	OOM	OOM	451.50	**604.30**	594.49	OOM	30.94
G5	449.40	439.11	OOM	439.11	627.81	**633.66**	608.34	OOM	41.00
G6	471.50	461.15	OOM	OOM	607.63	**621.46**	587.73	OOM	31.80
G7	459.10	449.11	OOM	449.11	**603.48**	600.54	602.42	OOM	31.45
G8	472.40	461.90	OOM	461.90	616.68	**618.74**	592.07	OOM	30.98
G9	473.30	459.30	OOM	459.30	**668.16**	662.70	644.86	OOM	41.17
G10	451.60	441.41	441.41	441.41	**638.92**	633.76	622.04	OOM	41.48
	462.12	451.47	441.41	449.91	603.98	**620.76**	604.59	–	34.85

Source: the author.

APPENDIX C — EXTENSIVE RESULTS FOR THE FIX-AND-OPTIMIZE *MATHEURISTIC*

This appendix contains detailed information regarding the runs of the fix-and-optimize matheuristic proposed in Chapter 7. Table C.1 introduces the results for the matheuristic with both guided, and random decomposition schemes turned on. Tables C.2 and C.3 present detailed results when we disable either the guided decomposition or the random decomposition schemes, respectively. These tables present detailed results for each instance from the Mankowska, Meisel and Bierwirth (2014) dataset (column *Instance*), indicating metrics regarding how many iterations the matheuristic performs (under *Iterations*, the average and standard deviation of the number of iterations, as well as the minimum and maximum total iterations in the shortest and longest run of the matheuristic, respectively.) Under *Time (sec)*, we present the average and standard deviation of algorithm runtime. Under *Solution cost*, we present the cost of the initial solution, the average and standard deviation of solution cost, and the best and the worst values produced. We also present the relative improvement of average and best solutions found regarding the value of the initial solution. The last columns present the average and the standard deviation regarding the objective function components.

Table C.1 – Extensive results for fix-and-optimize mathreuristic, combining random and guided decomposition schemes.

Instance	Iterations				Time (sec.)		Solution cost					Improvement (%)		Travel time		Total tardiness		Maximum tard.	
	Mean	SD	Min	Max	Mean	SD	Initial	Mean	SD	Min	Max	Mean	Min	Mean	SD	Mean	SD	Mean	SD
A1	8.85	1.69	7	12	0.17	0.03	271.21	219.19	4.44	218.20	238.04	19.18	19.55	657.57	13.31	0.00	0.00	0.00	0.00
A2	7.35	0.67	7	9	0.18	0.02	248.13	246.63	0.00	246.63	246.63	0.60	0.60	687.29	0.00	26.29	0.00	26.29	0.00
A3	7.15	0.37	7	8	0.47	0.12	339.07	305.86	0.00	305.86	305.86	9.79	9.79	741.14	0.00	99.31	0.00	77.13	0.00
A4	7.70	1.22	6	11	1.30	0.52	210.42	189.25	7.24	186.90	210.42	10.06	11.18	467.43	37.43	61.54	10.48	38.77	5.24
A5	9.25	1.92	7	16	0.23	0.06	271.43	194.00	13.73	189.54	234.14	28.53	30.17	582.01	41.18	0.00	0.00	0.00	0.00
A6	7.95	1.76	6	12	0.18	0.05	200.16	200.11	0.03	200.10	200.16	0.02	0.03	600.34	0.08	0.00	0.00	0.00	0.00
A7	7.35	0.59	7	9	0.12	0.01	383.44	225.37	0.00	225.37	225.37	41.22	41.22	676.11	0.00	0.00	0.00	0.00	0.00
A8	7.45	0.51	7	8	0.19	0.02	295.98	232.05	0.00	232.05	232.05	21.60	21.60	653.27	0.00	26.51	0.00	16.37	0.00
A9	8.70	2.34	7	14	2.50	1.53	275.97	222.30	0.00	222.30	222.30	19.45	19.45	666.89	0.00	0.00	0.00	0.00	0.00
A10	7.70	0.66	7	9	0.11	0.02	261.75	226.55	6.91	225.01	255.93	13.45	14.04	679.66	20.74	0.00	0.00	0.00	0.00
	7.95	1.17	6.80	10.80	0.54	0.24	275.75	226.13	3.23	225.19	237.09	16.39	16.76	641.17	11.27	21.37	1.05	15.86	0.52

Table C.1 – (Continued.) Extensive results for fix-and-optimize mathreuristic, combining random and guided decomposition schemes.

Instance	Iterations				Time (sec.)		Solution cost					Improvement (%)		Travel time		Total tardiness		Maximum tard.	
	Mean	SD	Min	Max	Mean	SD	Initial	Mean	SD	Min	Max	Mean	Min	Mean	SD	Mean	SD	Mean	SD
B1	25.65	7.51	17	44	42.95	24.28	787.91	434.79	8.06	428.10	454.79	44.82	45.67	1281.97	12.10	12.38	9.62	10.01	6.93
B2	25.75	6.10	16	38	12.88	6.28	650.85	483.03	17.71	476.05	533.08	25.78	26.86	1330.29	58.94	65.42	14.76	53.38	0.00
B3	35.45	13.27	17	68	210.97	91.93	1328.21	419.51	12.07	399.09	456.24	68.42	69.95	972.24	36.13	205.38	3.12	80.90	0.00
B4	30.00	9.99	18	49	20.76	10.70	1034.85	439.23	17.23	411.30	480.18	57.56	60.26	1229.64	46.83	58.57	10.60	29.48	0.00
B5	26.70	7.50	17	45	76.08	48.33	684.46	390.17	20.30	366.34	433.35	43.00	46.48	1125.96	71.10	27.14	16.93	17.42	10.76
B6	36.20	11.56	15	59	261.43	129.02	1835.53	516.99	84.59	441.70	790.42	71.83	75.94	1074.30	132.15	351.40	133.10	125.28	34.34
B7	31.40	6.48	21	45	42.72	33.77	496.67	334.33	9.72	328.67	357.22	32.69	33.82	1002.98	29.16	0.00	0.00	0.00	0.00
B8	35.75	6.58	27	50	36.46	20.17	717.20	373.95	16.85	357.68	404.95	47.86	50.13	1101.56	46.66	10.86	15.02	9.41	13.44
B9	33.20	8.56	21	51	220.29	79.95	840.20	410.59	18.52	402.67	488.39	51.13	52.07	1186.33	42.09	29.99	32.52	15.45	7.30
B10	33.85	11.16	20	59	218.70	93.54	987.87	479.48	17.31	462.75	532.96	51.46	53.16	1364.47	27.08	46.64	28.69	27.34	5.13
	31.40	8.87	18.90	50.80	114.32	53.80	936.38	428.21	22.24	407.43	493.16	49.45	51.43	1166.98	50.22	80.78	26.44	36.86	7.79
C1	132.15	37.89	66	223	1256.49	409.40	4084.06	1001.48	60.67	957.05	1157.44	75.48	76.57	1732.24	39.69	1078.32	134.46	193.88	32.46
C2	101.65	28.08	34	154	392.44	138.02	1014.01	610.95	43.52	582.99	785.19	39.75	42.51	1781.77	141.66	33.81	26.16	17.28	12.33
C3	133.65	33.33	91	225	538.64	228.27	1421.51	644.42	91.42	558.75	935.01	54.67	60.69	1780.36	77.83	124.13	188.43	28.77	28.10
C4	126.30	35.49	48	228	214.54	84.31	816.24	527.13	25.72	507.38	629.43	35.42	37.84	1555.97	74.26	16.74	9.89	8.68	4.07
C5	97.00	30.63	53	158	376.06	154.94	1339.31	687.64	20.16	667.53	728.05	48.66	50.16	1764.44	59.07	231.52	14.83	66.94	13.87
C6	136.35	38.92	60	220	1174.61	367.84	4217.62	900.74	82.23	822.85	1079.58	78.64	80.49	1615.34	68.55	897.81	215.45	189.06	19.33
C7	131.15	45.68	66	256	417.45	141.15	893.74	540.31	10.63	521.89	557.13	39.54	41.61	1557.25	66.79	44.36	40.09	19.32	15.08
C8	106.00	18.24	74	151	137.04	63.57	783.89	489.45	10.30	476.66	513.27	37.56	39.19	1435.79	41.09	24.37	12.65	8.19	3.75
C9	126.85	27.86	81	179	411.82	114.82	1093.39	572.36	29.68	535.87	658.37	47.65	50.99	1684.37	85.44	22.28	19.22	10.43	7.23
C10	126.40	34.81	58	200	405.08	150.94	1394.82	617.35	26.91	590.26	706.11	55.74	57.68	1738.90	67.24	88.45	36.37	24.70	12.09
	121.75	33.09	63.10	199.40	532.42	185.33	1705.86	659.18	40.12	622.12	774.96	51.31	53.77	1664.64	72.16	256.18	69.76	56.73	14.83

Table C.1 – (Continued.) Extensive results for fix-and-optimize mathreuristic, combining random and guided decomposition schemes.

Instance	Iterations				Time (sec.)		Solution cost					Improvement (%)		Travel time		Total tardiness		Maximum tard.	
	Mean	SD	Min	Max	Mean	SD	Initial	Mean	SD	Min	Max	Mean	Min	Mean	SD	Mean	SD	Mean	SD
D1	291.30	91.62	103	464	2670.26	844.95	3055.58	1202.98	66.99	1149.70	1383.02	60.63	62.37	2477.16	94.82	958.76	181.27	173.01	29.65
D2	252.55	59.71	150	366	1208.17	462.28	1411.10	761.25	50.31	690.21	870.61	46.05	51.09	2218.57	153.97	45.84	35.74	19.35	12.06
D3	270.45	84.43	141	443	1302.75	448.04	1464.83	680.90	36.63	643.23	807.00	53.52	56.09	1965.13	76.89	53.13	38.49	24.44	12.15
D4	224.70	66.54	113	403	1578.76	503.38	2282.79	828.36	23.85	798.99	877.01	63.71	65.00	2001.33	74.96	385.51	18.28	98.22	0.95
D5	273.50	74.85	164	481	1462.66	483.38	1412.03	741.15	35.31	688.53	828.50	47.51	51.24	2191.03	90.89	22.43	26.24	9.98	11.27
D6	318.85	97.60	199	559	1201.58	446.11	1217.79	764.51	31.68	712.15	846.68	37.22	41.52	2271.23	79.88	16.53	27.03	5.77	7.23
D7	272.30	64.61	129	381	1331.29	397.61	1104.35	619.66	20.47	595.22	663.85	43.89	46.10	1839.09	64.56	12.49	8.01	7.41	4.52
D8	280.55	78.67	178	435	1158.73	407.01	1268.49	712.81	31.97	666.09	783.55	43.81	47.49	2126.79	89.16	7.33	10.43	4.32	6.16
D9	258.40	72.52	130	388	785.45	258.01	1192.45	726.78	28.43	671.23	805.77	39.05	43.71	2160.64	90.37	13.82	11.96	5.89	3.89
D10	223.65	68.79	89	364	2058.81	650.70	4508.65	1329.02	88.15	1239.79	1609.29	70.52	72.50	2493.90	107.71	1286.89	177.38	206.25	30.51
	266.63	75.93	139.60	428.40	1475.85	490.15	1891.81	836.74	41.38	785.51	947.53	50.59	53.71	2174.49	92.32	280.27	53.48	55.46	11.84
E1	381.20	151.13	176	720	4989.21	1887.22	3786.22	1466.10	121.41	1337.87	1800.17	61.28	64.66	2950.57	180.23	1209.73	334.28	238.01	29.31
E2	490.75	180.28	230	808	4584.23	1703.08	1730.60	924.91	51.62	858.38	1037.80	46.56	50.40	2758.04	147.58	11.44	13.52	5.24	5.11
E3	436.15	120.49	250	666	3854.88	1227.08	1715.90	883.85	43.01	795.72	980.73	48.49	53.63	2629.20	122.65	14.93	16.07	7.43	8.51
E4	493.60	123.20	285	702	4406.35	1263.29	1442.73	791.98	40.14	732.28	890.93	45.11	49.24	2361.40	121.17	9.31	8.45	5.24	4.69
E5	516.20	110.83	333	770	5514.02	1280.92	1706.20	857.18	34.80	791.18	928.20	49.76	53.63	2542.95	109.73	20.18	21.26	8.40	8.04
E6	577.35	166.45	314	865	5007.03	1506.55	1614.55	881.96	39.75	831.76	965.79	45.37	48.48	2621.03	117.59	16.66	12.93	8.18	3.89
E7	432.35	139.28	201	714	4233.73	1336.09	1289.07	813.42	42.63	744.15	914.92	36.90	42.27	2423.40	120.23	11.09	11.67	5.76	5.51
E8	516.45	179.04	312	1078	4619.17	1462.20	1483.75	808.44	30.05	745.18	850.16	45.51	49.78	2396.18	83.03	20.09	15.65	9.03	5.66
E9	509.25	161.55	180	863	4245.42	1437.06	1683.32	1011.13	49.52	926.38	1113.88	39.93	44.97	3015.98	143.93	11.74	15.58	5.67	6.80
E10	502.80	120.49	296	756	4984.28	1363.53	2114.08	931.44	35.06	874.51	1014.50	55.94	58.63	2753.99	100.24	28.92	26.76	11.41	9.36
	485.61	145.28	257.70	794.20	4643.83	1446.70	1856.64	937.04	48.80	863.74	1049.71	47.49	51.57	2645.27	124.64	135.41	47.62	30.44	8.69

Table C.1 – (Continued.) Extensive results for fix-and-optimize mathreuristic, combining random and guided decomposition schemes.

Instance	Iterations				Time (sec.)		Solution cost					Improvement (%)		Travel time		Total tardiness		Maximum tard.	
	Mean	SD	Min	Max	Mean	SD	Initial	Mean	SD	Min	Max	Mean	Min	Mean	SD	Mean	SD	Mean	SD
F1	300.65	162.35	152	817	8033.76	4341.77	2744.70	2681.11	63.85	2557.06	2737.77	2.32	6.84	7686.55	65.32	298.90	129.01	57.90	15.20
F2	196.35	158.66	101	621	5212.50	4206.87	2648.03	2628.75	30.43	2544.32	2648.03	0.73	3.92	7879.35	93.96	3.64	7.54	3.27	7.41
F3	237.15	117.56	101	623	6323.53	3107.80	2499.23	2473.77	16.10	2433.87	2499.23	1.02	2.62	7295.45	41.60	98.85	19.40	27.01	1.99
F4	115.95	27.31	101	194	3147.07	736.89	2127.20	2125.39	3.77	2116.80	2127.20	0.08	0.49	6375.25	13.57	0.47	1.14	0.47	1.14
F5	106.95	26.61	101	220	2971.40	727.36	2698.74	2697.29	6.50	2669.68	2698.74	0.05	1.08	8087.45	39.20	3.39	15.17	1.02	4.54
F6	439.25	193.73	185	854	11475.76	5021.52	2609.83	2504.51	30.34	2455.95	2555.52	4.04	5.90	7471.49	93.40	30.85	12.22	11.18	8.95
F7	101.00	0.00	101	101	2775.26	27.63	2253.09	2253.09	0.00	2253.09	2253.09	0.00	0.00	6759.26	0.00	0.00	0.00	0.00	0.00
F8	258.55	112.66	101	507	6890.17	3002.74	2412.15	2374.64	26.50	2321.96	2412.15	1.56	3.74	6904.52	88.15	176.49	39.63	42.90	1.57
F9	191.05	88.78	101	369	5102.92	2362.00	2539.70	2529.09	8.81	2510.01	2539.70	0.42	1.17	7586.82	27.51	0.22	1.00	0.22	1.00
F10	110.85	25.56	101	192	3079.60	720.68	2712.22	2709.23	7.75	2686.98	2712.22	0.11	0.93	7494.43	26.02	573.48	10.75	59.79	0.00
	205.78	91.32	114.50	449.80	5501.20	2425.53	2524.49	2497.69	19.40	2454.97	2518.37	1.03	2.67	7354.06	48.88	118.63	23.58	20.37	4.18
G1	151.00	0.00	151	151	4239.05	128.18	5089.92	5089.92	0.00	5089.92	5089.92	0.00	0.00	10,833.10	0.00	4306.19	0.00	130.49	0.00
G2	151.00	0.00	151	151	4662.85	135.85	10,299.50	10299.50	0.00	10299.50	10299.50	0.00	0.00	11,291.80	0.00	19,381.00	0.00	225.63	0.00
G3	151.00	0.00	151	151	4130.04	13.59	3152.55	3152.55	0.00	3152.55	3152.55	0.00	0.00	9457.66	0.00	0.00	0.00	0.00	0.00
G4	151.00	0.00	151	151	4581.85	130.83	3277.34	3277.34	0.00	3277.34	3277.34	0.00	0.00	9810.20	0.00	10.91	0.00	10.91	0.00
G5	151.00	0.00	151	151	4158.78	67.35	3584.11	3584.11	0.00	3584.11	3584.11	0.00	0.00	10,700.50	0.00	35.91	0.00	15.97	0.00
G6	151.00	0.00	151	151	4678.47	133.27	3498.75	3498.75	0.00	3498.75	3498.75	0.00	0.00	10,496.20	0.00	0.00	0.00	0.00	0.00
G7	151.00	0.00	151	151	4243.56	126.82	3214.17	3214.17	0.00	3214.17	3214.17	0.00	0.00	9527.08	0.00	84.75	0.00	30.67	0.00
G8	151.00	0.00	151	151	4108.06	50.95	3241.68	3241.68	0.00	3241.68	3241.68	0.00	0.00	9725.04	0.00	0.00	0.00	0.00	0.00
G9	151.00	0.00	151	151	4135.18	27.24	3673.47	3673.47	0.00	3673.47	3673.47	0.00	0.00	10,807.80	0.00	171.58	0.00	41.02	0.00
G10	151.00	0.00	151	151	4080.58	12.56	3751.71	3751.71	0.00	3751.71	3751.71	0.00	0.00	11,255.10	0.00	0.00	0.00	0.00	0.00
	151.00	0.00	151.00	151.00	4301.84	82.66	4278.32	4278.32	0.00	4278.32	4278.32	0.00	0.00	10,390.45	0.00	2399.03	0.00	45.47	0.00

Source: the author.

Table C.2 – Extensive results for fix-and-optimize matheuristic, with only the guided decomposition scheme.

Instance	Iterations				Time (sec.)		Solution cost					Improvement (%)		Travel time		Total tardiness		Maximum tard.	
	Mean	SD	Min	Max	Mean	SD	Initial	Mean	SD	Min	Max	Mean	Min	Mean	SD	Mean	SD	Mean	SD
A1	8.70	1.69	7	12	0.19	0.05	271.21	221.18	7.27	218.20	238.04	18.45	19.55	663.52	21.81	0.00	0.00	0.00	0.00
A2	7.10	0.45	7	9	0.20	0.02	248.13	246.63	0.00	246.63	246.63	0.60	0.60	687.29	0.00	26.29	0.00	26.29	0.00
A3	7.00	0.00	7	7	0.58	0.14	339.07	305.86	0.00	305.86	305.86	9.79	9.79	741.14	0.00	99.31	0.00	77.13	0.00
A4	7.90	1.37	7	12	1.59	0.45	210.42	186.90	0.00	186.90	186.90	11.18	11.18	455.27	0.00	64.95	0.00	40.47	0.00
A5	9.70	2.20	7	14	0.27	0.08	271.43	198.46	18.30	189.54	234.14	26.88	30.17	595.39	54.91	0.00	0.00	0.00	0.00
A6	8.25	2.00	6	12	0.19	0.05	200.16	200.11	0.03	200.10	200.16	0.02	0.03	600.34	0.08	0.00	0.00	0.00	0.00
A7	7.10	0.31	7	8	0.15	0.01	383.44	225.37	0.00	225.37	225.37	41.22	41.22	676.11	0.00	0.00	0.00	0.00	0.00
A8	7.30	0.47	7	8	0.23	0.02	295.98	232.05	0.00	232.05	232.05	21.60	21.60	653.27	0.00	26.51	0.00	16.37	0.00
A9	8.95	1.76	7	12	2.70	1.34	275.97	222.89	2.67	222.30	234.23	19.23	19.45	667.28	1.79	0.91	4.08	0.48	2.14
A10	7.75	0.79	7	9	0.11	0.03	261.75	225.01	0.00	225.01	225.01	14.04	14.04	675.02	0.00	0.00	0.00	0.00	0.00
	7.98	1.10	6.90	10.30	0.62	0.22	275.75	226.45	2.83	225.19	232.84	16.30	16.76	641.46	7.86	21.80	0.41	16.08	0.21
B1	24.55	3.56	19	32	65.94	37.32	787.91	431.74	3.56	428.10	440.06	45.21	45.67	1276.21	15.77	11.17	6.39	7.83	1.04
B2	23.20	6.37	16	37	13.29	8.86	650.85	485.29	17.38	476.05	533.08	25.44	26.86	1339.69	54.60	62.80	13.66	53.38	0.00
B3	33.65	8.51	19	46	206.56	78.36	1328.21	421.82	12.25	399.09	458.44	68.24	69.95	969.96	29.18	212.44	25.26	83.05	9.58
B4	27.40	10.24	16	60	22.29	15.79	1034.85	447.60	24.91	411.30	501.00	56.75	60.26	1243.79	63.35	68.17	36.74	30.85	6.14
B5	30.00	7.74	19	51	97.71	38.91	684.46	382.85	10.65	366.34	399.96	44.06	46.48	1118.54	35.34	18.16	11.33	11.86	8.35
B6	38.20	11.69	15	59	296.25	179.82	1835.53	510.89	76.99	441.70	790.42	72.17	75.94	1090.01	130.63	330.55	126.02	112.12	12.29
B7	31.25	8.41	19	50	40.35	27.23	496.67	346.85	28.59	328.67	436.37	30.16	33.82	1035.61	74.52	3.48	10.65	1.46	4.21
B8	32.40	9.79	18	48	35.05	20.28	717.20	373.82	23.38	357.68	427.77	47.88	50.13	1099.41	53.44	11.52	16.09	10.55	14.82
B9	33.45	7.44	21	46	261.97	77.97	840.20	414.67	17.54	403.29	450.00	50.65	52.00	1211.68	67.83	20.50	21.47	11.85	5.86
B10	30.80	12.66	18	61	287.35	138.76	987.87	483.23	17.85	462.75	536.56	51.08	53.16	1371.60	34.64	50.83	25.25	27.26	7.03
	30.49	8.64	18.00	49.00	132.68	62.33	936.38	429.88	23.31	407.50	497.37	49.16	51.43	1175.65	55.93	78.96	29.29	35.02	6.93

Table C.2 – (Continued.) Extensive results for fix-and-optimize matheuristic, with only the guided decomposition scheme.

Instance	Iterations				Time (sec.)		Solution cost					Improvement (%)		Travel time		Total tardiness		Maximum tard.	
	Mean	SD	Min	Max	Mean	SD	Initial	Mean	SD	Min	Max	Mean	Min	Mean	SD	Mean	SD	Mean	SD
C1	115.30	31.86	46	163	1457.23	431.43	4084.06	992.92	31.79	943.62	1051.57	75.69	76.90	1759.60	61.39	1030.15	48.69	188.99	15.44
C2	102.35	31.57	42	181	518.92	166.20	1014.01	618.99	41.83	582.58	766.53	38.96	42.55	1817.78	129.51	26.10	24.26	13.07	9.79
C3	131.85	53.17	38	249	593.11	243.69	1421.51	610.81	62.22	557.63	810.02	57.03	60.77	1767.36	101.84	49.61	99.43	15.45	18.43
C4	112.25	24.30	80	174	222.69	94.41	816.24	542.38	21.84	509.77	586.80	33.55	37.55	1607.51	66.84	11.62	6.50	8.00	3.03
C5	100.05	24.80	50	143	519.12	154.90	1339.31	697.80	35.03	667.70	818.75	47.90	50.15	1790.85	94.02	232.93	26.52	69.64	16.43
C6	127.10	36.36	64	202	1488.89	414.39	4217.62	906.99	75.63	829.30	1148.90	78.50	80.34	1613.81	50.21	913.12	220.12	194.02	19.88
C7	127.80	32.08	80	187	481.33	174.90	893.74	549.99	20.90	519.68	597.91	38.46	41.85	1590.42	75.91	41.69	40.47	17.85	15.73
C8	109.50	31.51	72	181	223.04	149.46	783.89	494.67	17.64	470.80	530.79	36.90	39.94	1454.13	59.22	22.30	12.24	7.58	3.96
C9	144.05	35.64	92	228	609.94	201.70	1093.39	569.41	12.81	551.50	607.81	47.92	49.56	1684.47	37.64	15.64	10.36	8.14	4.24
C10	121.05	28.00	63	171	482.19	148.54	1394.82	623.14	22.20	593.18	672.01	55.32	57.47	1773.86	63.08	75.00	19.64	20.56	8.71
	119.13	32.93	62.70	187.90	659.65	217.96	1705.86	660.71	34.19	622.58	759.11	51.02	53.71	1685.98	73.97	241.82	50.82	54.33	11.56
D1	279.45	105.97	128	544	3372.12	1226.89	3055.58	1183.19	35.84	1133.45	1275.81	61.28	62.91	2471.03	78.43	906.94	33.90	171.60	26.58
D2	266.00	63.28	121	366	1844.30	745.49	1411.10	767.48	53.90	691.47	869.06	45.61	51.00	2232.07	165.96	48.43	38.51	21.92	12.69
D3	259.55	69.30	168	388	1765.37	764.62	1464.83	691.07	29.30	641.76	754.27	52.82	56.19	2007.55	91.99	44.36	11.04	21.29	1.55
D4	225.70	55.54	128	342	2291.15	589.92	2282.79	836.44	39.07	784.64	929.72	63.36	65.63	2030.25	114.95	381.01	12.34	98.08	0.67
D5	289.50	86.15	159	477	2217.04	894.49	1412.03	744.63	40.19	659.21	813.59	47.27	53.31	2210.56	110.16	15.79	16.26	7.55	8.77
D6	287.25	82.86	170	494	1352.20	433.57	1217.79	780.79	38.99	727.51	889.99	35.88	40.26	2317.13	113.32	18.07	20.80	7.17	6.86
D7	245.15	73.19	151	370	1830.79	681.90	1104.35	629.51	25.02	592.02	687.48	43.00	46.39	1866.41	78.36	14.74	12.43	7.37	5.64
D8	270.70	83.12	149	451	1615.60	653.61	1268.49	731.58	40.68	659.96	814.99	42.33	47.97	2181.39	118.07	8.41	10.82	4.93	6.11
D9	233.40	62.08	136	328	1066.32	413.74	1192.45	755.35	31.42	713.96	828.49	36.66	40.13	2238.38	106.45	19.85	27.38	7.83	9.94
D10	227.30	79.30	117	386	2854.65	1120.71	4508.65	1307.12	39.45	1242.98	1423.29	71.01	72.43	2566.28	118.34	1167.40	69.16	187.67	12.55
	258.40	76.08	142.70	414.60	2020.95	752.49	1891.81	842.72	37.39	784.70	928.67	49.92	53.62	2212.11	109.60	262.50	25.26	53.54	9.14

Table C.2 – (Continued.) Extensive results for fix-and-optimize matheuristic, with only the guided decomposition scheme.

Instance	Iterations				Time (sec.)		Solution cost					Improvement (%)		Travel time		Total tardiness		Maximum tard.	
	Mean	SD	Min	Max	Mean	SD	Initial	Mean	SD	Min	Max	Mean	Min	Mean	SD	Mean	SD	Mean	SD
E1	403.45	99.83	179	545	7163.01	1880.11	3786.22	1505.07	99.35	1371.61	1700.59	60.25	63.77	3139.32	182.85	1141.52	176.24	234.37	24.22
E2	508.95	156.28	241	687	7317.84	2248.52	1730.60	980.12	55.43	899.62	1097.19	43.37	48.02	2907.83	164.37	22.96	15.75	9.55	6.68
E3	444.20	136.71	171	675	6268.43	2010.13	1715.90	941.05	64.17	864.17	1112.06	45.16	49.64	2794.74	181.17	20.45	22.64	7.97	7.30
E4	511.85	164.91	218	842	7094.99	2272.06	1442.73	840.77	70.43	757.22	1021.63	41.72	47.51	2503.67	202.09	12.81	11.69	5.82	5.22
E5	460.30	142.34	180	680	7381.00	2214.04	1706.20	929.35	66.15	803.12	1092.81	45.53	52.93	2768.76	190.74	13.13	16.25	6.16	6.96
E6	505.70	151.89	287	777	6813.30	2087.13	1614.55	982.39	70.07	863.68	1110.04	39.15	46.51	2916.05	211.36	22.93	24.43	8.18	6.49
E7	448.60	141.64	216	913	6365.28	1995.34	1289.07	837.74	42.01	758.32	926.68	35.01	41.17	2498.48	125.11	9.27	8.20	5.48	5.35
E8	504.15	171.62	247	898	7441.44	2795.75	1483.75	892.19	65.04	801.94	1075.96	39.87	45.95	2638.91	190.66	26.69	17.63	10.96	5.73
E9	502.25	116.92	316	667	5938.89	1375.88	1683.32	1030.87	39.76	957.04	1120.57	38.76	43.15	3071.71	116.65	14.21	14.70	6.69	5.34
E10	514.25	182.54	204	986	7666.06	2805.60	2114.08	1055.62	88.52	913.76	1200.28	50.07	56.78	3118.93	245.25	37.16	31.40	10.77	6.98
	480.37	146.47	225.90	767.00	6945.02	2168.46	1856.64	999.52	66.09	899.05	1145.78	43.89	49.54	2835.84	181.02	132.11	33.89	30.59	8.03

Source: the author.

Table C.3 – Extensive results for fix-and-optimize matheuristic, with only the random decomposition scheme.

Instance	Iterations				Time (sec.)		Solution cost					Improvement (%)		Travel time		Total tardiness		Maximum tard.	
	Mean	SD	Min	Max	Mean	SD	Initial	Mean	SD	Min	Max	Mean	Min	Mean	SD	Mean	SD	Mean	SD
A1	9.45	2.04	7	14	0.17	0.03	271.21	218.20	0.00	218.20	218.20	19.55	19.55	654.60	0.00	0.00	0.00	0.00	0.00
A2	7.30	0.57	7	9	0.20	0.02	248.13	246.63	0.00	246.63	246.63	0.60	0.60	687.29	0.00	26.29	0.00	26.29	0.00
A3	7.30	0.57	7	9	0.52	0.15	339.07	305.86	0.00	305.86	305.86	9.79	9.79	741.14	0.00	99.31	0.00	77.13	0.00
A4	8.50	1.70	6	12	1.13	0.46	210.42	189.25	7.24	186.90	210.42	10.06	11.18	467.43	37.43	61.54	10.48	38.77	5.24
A5	9.40	1.57	7	13	0.22	0.05	271.43	191.77	9.97	189.54	234.14	29.35	30.17	575.32	29.92	0.00	0.00	0.00	0.00
A6	8.50	1.70	6	12	0.21	0.05	200.16	200.11	0.02	200.10	200.16	0.03	0.03	600.32	0.06	0.00	0.00	0.00	0.00
A7	7.50	0.83	7	10	0.13	0.02	383.44	225.37	0.00	225.37	225.37	41.22	41.22	676.11	0.00	0.00	0.00	0.00	0.00
A8	8.00	1.08	7	10	0.19	0.03	295.98	232.05	0.00	232.05	232.05	21.60	21.60	653.27	0.00	26.51	0.00	16.37	0.00
A9	8.05	2.14	7	14	1.84	1.20	275.97	222.30	0.00	222.30	222.30	19.45	19.45	666.89	0.00	0.00	0.00	0.00	0.00
A10	7.95	0.83	7	9	0.12	0.03	261.75	225.01	0.00	225.01	225.01	14.04	14.04	675.02	0.00	0.00	0.00	0.00	0.00
	8.20	1.30	6.80	11.20	0.47	0.21	275.75	225.65	1.72	225.19	232.01	16.57	16.76	639.74	6.74	21.37	1.05	15.86	0.52
B1	26.90	4.82	19	35	40.59	27.41	787.91	433.86	8.46	428.10	454.79	44.94	45.67	1273.88	17.72	16.87	9.56	10.83	6.68
B2	28.25	6.46	20	43	8.71	2.85	650.85	477.62	3.85	476.05	486.94	26.62	26.86	1320.52	12.26	58.98	1.32	53.38	0.00
B3	32.90	7.57	22	49	170.88	49.61	1328.21	418.29	5.84	414.50	438.58	68.51	68.79	967.82	15.69	206.13	6.25	80.90	0.00
B4	30.65	10.52	16	50	21.19	8.31	1034.85	435.79	17.07	411.30	466.87	57.89	60.26	1219.70	53.72	57.55	24.90	30.11	2.83
B5	28.25	8.06	16	47	58.99	32.79	684.46	389.62	17.47	366.34	433.35	43.08	46.48	1129.25	62.84	24.02	16.88	15.58	11.44
B6	35.15	11.47	19	58	202.11	95.99	1835.53	505.97	76.72	441.70	776.07	72.43	75.94	1063.76	129.90	339.53	126.63	114.62	12.25
B7	31.20	5.79	16	40	29.78	24.81	496.67	335.73	21.35	328.67	421.54	32.40	33.82	999.69	34.10	5.38	24.07	2.11	9.45
B8	33.75	8.06	22	50	25.21	16.19	717.20	374.56	16.95	357.68	404.39	47.78	50.13	1104.42	46.60	10.59	16.09	8.66	13.35
B9	35.35	9.65	22	54	173.08	76.85	840.20	407.43	6.13	402.67	430.82	51.51	52.07	1179.61	32.06	28.44	33.46	14.24	5.74
B10	31.80	7.88	19	49	146.92	49.68	987.87	479.61	14.24	462.75	517.48	51.45	53.16	1366.05	36.85	45.93	18.70	26.84	5.33
	31.42	8.03	19.10	47.50	87.74	38.45	936.38	425.85	18.81	408.98	483.08	49.66	51.32	1162.47	44.17	79.34	27.79	35.73	6.71

Table C.3 – (Continued.) Extensive results for fix-and-optimize matheuristic, with only the random decomposition scheme.

Instance	Iterations				Time (sec.)		Solution cost					Improvement (%)		Travel time		Total tardiness		Maximum tard.	
	Mean	SD	Min	Max	Mean	SD	Initial	Mean	SD	Min	Max	Mean	Min	Mean	SD	Mean	SD	Mean	SD
C1	127.30	32.33	74	192	827.25	250.70	4084.06	995.05	43.84	947.72	1150.56	75.64	76.79	1729.72	39.58	1064.71	122.05	190.70	14.95
C2	131.50	49.21	59	253	304.52	122.27	1014.01	608.29	23.14	584.89	664.26	40.01	42.32	1757.64	66.57	45.72	26.76	21.52	13.20
C3	142.20	40.45	76	218	415.70	159.32	1421.51	660.85	109.58	554.26	961.65	53.51	61.01	1785.28	111.92	162.40	200.58	34.86	31.09
C4	115.65	41.08	55	214	155.96	68.51	816.24	529.14	16.95	506.73	561.10	35.17	37.92	1555.96	47.01	21.30	14.55	10.15	6.25
C5	105.20	25.89	69	158	216.93	77.88	1339.31	687.16	17.35	665.83	725.34	48.69	50.29	1761.15	55.78	233.38	13.93	66.94	13.87
C6	149.10	41.86	71	244	835.44	286.92	4217.62	902.54	61.92	815.12	1008.48	78.60	80.67	1603.73	41.01	907.55	166.72	196.34	17.25
C7	120.35	33.99	70	195	300.18	97.25	893.74	540.39	16.52	516.27	583.06	39.54	42.23	1538.78	77.92	58.32	38.43	24.05	13.42
C8	133.35	24.33	99	193	147.61	55.38	783.89	484.20	10.54	470.80	503.58	38.23	39.94	1417.00	36.68	26.31	9.83	9.28	2.61
C9	144.85	39.50	69	217	386.49	124.42	1093.39	570.58	28.79	536.61	631.86	47.82	50.92	1684.30	78.51	17.94	19.28	9.50	9.51
C10	108.70	27.22	56	150	224.43	78.75	1394.82	621.96	25.37	590.26	685.94	55.41	57.68	1751.19	78.85	90.06	47.44	24.61	14.84
	127.82	35.59	69.80	203.40	381.45	132.14	1705.86	660.01	35.40	618.85	747.58	51.26	53.98	1658.48	63.38	262.77	65.96	58.80	13.70
D1	288.50	98.94	170	539	1932.52	688.99	3055.58	1208.31	55.15	1131.43	1370.30	60.46	62.97	2483.85	99.56	949.72	115.03	191.35	42.86
D2	281.40	98.35	160	563	806.30	392.15	1411.10	781.93	45.12	686.95	850.49	44.59	51.32	2273.74	142.47	48.54	33.33	23.52	16.45
D3	281.00	108.73	119	608	1326.69	521.39	1464.83	716.29	76.00	644.21	897.51	51.10	56.02	2021.30	131.26	87.56	75.79	40.01	38.79
D4	246.70	85.12	102	457	1051.08	370.73	2282.79	835.58	38.58	793.34	941.79	63.40	65.25	2019.62	115.12	388.76	21.33	98.38	0.97
D5	282.15	89.93	160	509	999.30	294.48	1412.03	759.47	46.24	672.76	849.18	46.21	52.35	2249.17	128.62	19.34	15.59	9.91	7.71
D6	277.60	88.99	142	517	989.87	359.63	1217.79	788.22	42.67	731.95	865.79	35.27	39.90	2323.47	118.44	29.23	22.17	11.95	9.52
D7	261.05	66.25	171	371	1227.14	275.18	1104.35	644.52	35.66	590.55	725.01	41.64	46.53	1914.12	111.92	11.85	11.11	7.59	5.74
D8	294.75	86.68	168	490	1175.68	361.67	1268.49	711.91	37.56	669.55	818.80	43.88	47.22	2128.43	110.05	4.55	5.69	2.74	2.86
D9	256.15	81.15	160	472	522.92	208.42	1192.45	745.09	31.39	694.27	795.48	37.52	41.78	2204.50	92.39	21.09	21.08	9.69	10.70
D10	240.75	65.54	147	387	1365.07	352.05	4508.65	1353.21	73.05	1268.66	1518.59	69.99	71.86	2483.66	88.70	1354.35	196.24	221.63	55.40
	271.01	86.97	149.90	491.30	1139.66	382.47	1891.81	854.45	48.14	788.37	963.29	49.40	53.52	2210.19	113.85	291.50	51.74	61.68	19.10

Table C.3 – (Continued.) Extensive results for fix-and-optimize matheuristic, with only the random decomposition scheme.

Instance	Iterations				Time (sec.)		Solution cost					Improvement (%)		Travel time		Total tardiness		Maximum tard.	
	Mean	SD	Min	Max	Mean	SD	Initial	Mean	SD	Min	Max	Mean	Min	Mean	SD	Mean	SD	Mean	SD
E1	410.85	118.52	179	599	4891.36	1508.99	3786.22	1582.33	165.13	1393.96	2029.05	58.21	63.18	3074.04	170.35	1416.94	429.48	255.99	38.73
E2	429.90	106.67	258	593	4351.01	1033.44	1730.60	1009.32	55.77	926.40	1143.10	41.68	46.47	2990.53	155.16	26.19	14.26	11.25	7.43
E3	474.75	234.59	133	1070	4492.91	1947.06	1715.90	927.70	68.84	825.14	1128.68	45.94	51.91	2764.46	202.71	12.73	9.91	5.91	4.05
E4	422.25	125.05	214	628	4180.35	1230.37	1442.73	870.21	50.69	795.94	952.04	39.68	44.83	2586.79	155.88	16.62	14.48	7.21	5.11
E5	447.05	127.44	179	619	5162.25	1382.66	1706.20	979.57	70.12	883.89	1108.46	42.59	48.20	2879.80	199.85	41.95	33.95	16.97	12.64
E6	528.45	187.09	266	956	5181.55	1679.29	1614.55	959.11	58.59	869.10	1078.90	40.60	46.17	2845.57	166.90	22.49	20.14	9.28	6.10
E7	392.60	103.97	178	618	4192.73	1124.55	1289.07	852.35	37.90	769.00	937.02	33.88	40.34	2539.30	109.60	11.17	9.44	6.56	5.33
E8	479.70	133.65	182	692	4623.30	1236.94	1483.75	903.67	64.14	778.70	1092.14	39.10	47.52	2679.04	171.42	22.85	22.35	9.11	7.46
E9	534.15	159.39	271	755	4692.33	1194.48	1683.32	1059.39	58.29	954.88	1169.04	37.07	43.27	3146.95	158.80	21.72	19.88	9.50	7.09
E10	492.45	221.00	186	886	4956.17	2022.59	2114.08	1059.10	111.17	882.33	1262.31	49.90	58.26	3131.03	321.82	35.15	28.68	11.13	6.44
	461.22	151.74	204.60	741.60	4672.39	1436.04	1856.64	1020.27	74.06	907.93	1190.07	42.86	49.02	2863.75	181.25	162.78	60.26	34.29	10.04

Source: the author.

Table C.4 – Comparison of the three matheuristic variations.

Instance	Iterations (mean)			Time (sec.)			Cost (mean)			Cost (best)		
	R+G	G	R	R+G	G	R	R+G	G	R	R+G	G	R
A1	8.85	8.70	9.45	0.17	0.19	0.17	219.19	221.18	**218.20**	**218.20**	**218.20**	**218.20**
A2	7.35	7.10	7.30	0.18	0.20	0.20	**246.63**	**246.63**	**246.63**	**246.63**	**246.63**	**246.63**
A3	7.15	7.00	7.30	0.47	0.58	0.52	**305.86**	**305.86**	**305.86**	**305.86**	**305.86**	**305.86**
A4	7.70	7.90	8.50	1.30	1.59	1.13	189.25	**186.90**	189.25	**186.90**	**186.90**	**186.90**
A5	9.25	9.70	9.40	0.23	0.27	0.22	194.00	198.46	**191.77**	**189.54**	**189.54**	**189.54**
A6	7.95	8.25	8.50	0.18	0.19	0.21	200.11	200.11	**200.11**	**200.10**	**200.10**	**200.10**
A7	7.35	7.10	7.50	0.12	0.15	0.13	**225.37**	**225.37**	**225.37**	**225.37**	**225.37**	**225.37**
A8	7.45	7.30	8.00	0.19	0.23	0.19	**232.05**	**232.05**	**232.05**	**232.05**	**232.05**	**232.05**
A9	8.70	8.95	8.05	2.50	2.70	1.84	**222.30**	222.89	**222.30**	**222.30**	**222.30**	**222.30**
A10	7.70	7.75	7.95	0.11	0.11	0.12	226.55	**225.01**	**225.01**	**225.01**	**225.01**	**225.01**
	7.95	7.98	8.20	0.54	0.62	0.47	226.13	226.45	**225.65**	**225.19**	**225.19**	**225.19**
B1	25.65	24.55	26.90	42.95	65.94	40.59	434.79	**431.74**	433.86	**428.10**	**428.10**	**428.10**
B2	25.75	23.20	28.25	12.88	13.29	8.71	483.03	485.29	**477.62**	**476.05**	**476.05**	**476.05**
B3	35.45	33.65	32.90	210.97	206.56	170.88	419.51	421.82	**418.29**	**399.09**	**399.09**	414.50
B4	30.00	27.40	30.65	20.76	22.29	21.19	439.23	447.60	**435.79**	**411.30**	**411.30**	**411.30**
B5	26.70	30.00	28.25	76.08	97.71	58.99	390.17	**382.85**	389.62	**366.34**	**366.34**	**366.34**
B6	36.20	38.20	35.15	261.43	296.25	202.11	516.99	510.89	**505.97**	**441.70**	**441.70**	**441.70**
B7	31.40	31.25	31.20	42.72	40.35	29.78	**334.33**	346.85	335.73	**328.67**	**328.67**	**328.67**
B8	35.75	32.40	33.75	36.46	35.05	25.21	373.95	**373.82**	374.56	**357.68**	**357.68**	**357.68**
B9	33.20	33.45	35.35	220.29	261.97	173.08	410.59	414.67	**407.43**	**402.67**	403.29	**402.67**
B10	33.85	30.80	31.80	218.70	287.35	146.92	**479.48**	483.23	479.61	**462.75**	**462.75**	**462.75**
	31.40	30.49	31.42	114.32	132.68	87.74	428.21	429.88	**425.85**	**407.43**	407.50	408.98

Table C.4 – (Continued.) Comparison of the three matheuristic variations.

Instance	Iterations (mean)			Time (sec.)			Cost (mean)			Cost (best)		
	R+G	G	R	R+G	G	R	R+G	G	R	R+G	G	R
C1	132.15	115.30	127.30	1256.49	1457.23	827.25	1001.48	**992.92**	995.05	957.05	**943.62**	947.72
C2	101.65	102.35	131.50	392.44	518.92	304.52	610.95	618.99	**608.29**	582.99	**582.58**	584.89
C3	133.65	131.85	142.20	538.64	593.11	415.70	644.42	**610.81**	660.85	558.75	557.63	**554.26**
C4	126.30	112.25	115.65	214.54	222.69	155.96	**527.13**	542.38	529.14	507.38	509.77	**506.73**
C5	97.00	100.05	105.20	376.06	519.12	216.93	687.64	697.80	**687.16**	667.53	667.70	**665.83**
C6	136.35	127.10	149.10	1174.61	1488.89	835.44	**900.74**	906.99	902.54	822.85	829.30	**815.12**
C7	131.15	127.80	120.35	417.45	481.33	300.18	**540.31**	549.99	540.39	521.89	519.68	**516.27**
C8	106.00	109.50	133.35	137.04	223.04	147.61	489.45	494.67	**484.20**	476.66	**470.80**	**470.80**
C9	126.85	144.05	144.85	411.82	609.94	386.49	572.36	**569.41**	570.58	**535.87**	551.50	536.61
C10	126.40	121.05	108.70	405.08	482.19	224.43	**617.35**	623.14	621.96	**590.26**	593.18	**590.26**
	121.75	119.13	127.82	532.42	659.65	381.45	**659.18**	660.71	660.01	622.12	622.58	**618.85**
D1	291.30	279.45	288.50	2670.26	3372.12	1932.52	1202.98	**1183.19**	1208.31	1149.70	1133.45	**1131.43**
D2	252.55	266.00	281.40	1208.17	1844.30	806.30	**761.25**	767.48	781.93	690.21	691.47	**686.95**
D3	270.45	259.55	281.00	1302.75	1765.37	1326.69	**680.90**	691.07	716.29	643.23	**641.76**	644.21
D4	224.70	225.70	246.70	1578.76	2291.15	1051.08	**828.36**	836.44	835.58	798.99	**784.64**	793.34
D5	273.50	289.50	282.15	1462.66	2217.04	999.30	**741.15**	744.63	759.47	688.53	**659.21**	672.76
D6	318.85	287.25	277.60	1201.58	1352.20	989.87	**764.51**	780.79	788.22	**712.15**	727.51	731.95
D7	272.30	245.15	261.05	1331.29	1830.79	1227.14	**619.66**	629.51	644.52	595.22	592.02	**590.55**
D8	280.55	270.70	294.75	1158.73	1615.60	1175.68	712.81	731.58	**711.91**	666.09	**659.96**	669.55
D9	258.40	233.40	256.15	785.45	1066.32	522.92	**726.78**	755.35	745.09	**671.23**	713.96	694.27
D10	223.65	227.30	240.75	2058.81	2854.65	1365.07	1329.02	**1307.12**	1353.21	**1239.79**	1242.98	1268.66
	266.63	258.40	271.01	1475.85	2020.95	1139.66	**836.74**	842.72	854.45	785.51	**784.70**	788.37

Table C.4 – (Continued.) Comparison of the three matheuristic variations.

Instance	Iterations (mean)			Time (sec.)			Cost (mean)			Cost (best)		
	R+G	G	R	R+G	G	R	R+G	G	R	R+G	G	R
E1	381.20	403.45	410.85	4989.21	7163.01	4891.36	**1466.10**	1505.07	1582.33	**1337.87**	1371.61	1393.96
E2	490.75	508.95	429.90	4584.23	7317.84	4351.01	**924.91**	980.12	1009.32	**858.38**	899.62	926.40
E3	436.15	444.20	474.75	3854.88	6268.43	4492.91	**883.85**	941.05	927.70	**795.72**	864.17	825.14
E4	493.60	511.85	422.25	4406.35	7094.99	4180.35	**791.98**	840.77	870.21	**732.28**	757.22	795.94
E5	516.20	460.30	447.05	5514.02	7381.00	5162.25	**857.18**	929.35	979.57	**791.18**	803.12	883.89
E6	577.35	505.70	528.45	5007.03	6813.30	5181.55	**881.96**	982.39	959.11	**831.76**	863.68	869.10
E7	432.35	448.60	392.60	4233.73	6365.28	4192.73	**813.42**	837.74	852.35	**744.15**	758.32	769.00
E8	516.45	504.15	479.70	4619.17	7441.44	4623.30	**808.44**	892.19	903.67	**745.18**	801.94	778.70
E9	509.25	502.25	534.15	4245.42	5938.89	4692.33	**1011.13**	1030.87	1059.39	**926.38**	957.04	954.88
E10	502.80	514.25	492.45	4984.28	7666.06	4956.17	**931.44**	1055.62	1059.10	**874.51**	913.76	882.33
	485.61	480.37	461.22	4643.83	6945.02	4672.39	**937.04**	999.52	1020.27	**863.74**	899.05	907.93

Source: the author.

APPENDIX D — EXTENSIVE RESULTS FOR THE BRKGA.

This appendix contains detailed information regarding the runs of the BRKGA meta-heuristic proposed in Chapter 8. Table D.1 presents detailed results for each instance from the Mankowska, Meisel and Bierwirth (2014) dataset (column *Instance*). Under *Time (sec)*, we present the average and standard deviation of algorithm runtime. Under *Cost*, we present the cost of the average and standard deviation of solution cost, and the best and the worst values produced. The last columns present the average and the standard deviation regarding the objective function components.

Table D.1 – Detailed results for the biased random-key genetic algorithm.

Instance	Time (sec.)		Cost				Travel time		Total tardiness		Maximum tardiness	
	Mean	SD	Mean	SD	Min	Max	Mean	SD	Mean	SD	Mean	SD
A1	0.72	0.02	226.98	0.00	226.98	226.98	680.93	0.00	0.00	0.00	0.00	0.00
A2	0.72	0.02	246.63	0.00	246.63	246.63	687.29	0.00	26.29	0.00	26.29	0.00
A3	0.73	0.02	305.86	0.00	305.86	305.86	741.14	0.00	99.31	0.00	77.13	0.00
A4	0.72	0.02	186.90	0.00	186.90	186.90	455.27	0.00	64.95	0.00	40.47	0.00
A5	0.72	0.02	191.97	0.00	191.97	191.97	575.92	0.00	0.00	0.00	0.00	0.00
A6	0.73	0.02	200.13	0.00	200.13	200.13	600.38	0.00	0.00	0.00	0.00	0.00
A7	0.69	0.02	225.37	0.00	225.37	225.37	676.11	0.00	0.00	0.00	0.00	0.00
A8	0.70	0.02	232.05	0.00	232.05	232.05	653.27	0.00	26.51	0.00	16.37	0.00
A9	0.76	0.02	234.21	0.00	234.21	234.21	677.00	0.00	12.82	0.00	12.82	0.00
A10	0.70	0.03	225.01	0.00	225.01	225.01	675.02	0.00	0.00	0.00	0.00	0.00
	0.72	0.02	227.51	0.00	227.51	227.51	642.23	0.00	22.99	0.00	17.31	0.00

Table D.1 – (Continued.) Detailed results for the biased random-key genetic algorithm.

Instance	Time (sec.)		Cost				Travel time		Total tardiness		Maximum tardiness	
	Mean	SD	Mean	SD	Min	Max	Mean	SD	Mean	SD	Mean	SD
B1	1.86	0.04	428.32	0.25	428.10	428.58	1261.23	9.19	15.19	7.28	8.53	1.18
B2	1.86	0.03	485.31	1.54	483.63	486.68	1372.55	31.43	45.34	12.76	38.03	14.06
B3	1.92	0.04	402.80	0.00	402.80	402.80	914.89	0.00	212.59	0.00	80.90	0.00
B4	1.85	0.03	432.55	3.37	420.29	436.91	1213.87	13.98	54.32	6.04	29.48	0.00
B5	1.77	0.04	374.65	3.09	372.16	378.39	1085.41	37.39	24.24	18.17	14.30	9.95
B6	1.93	0.04	471.95	1.43	471.00	476.38	979.84	5.36	293.08	1.31	142.92	1.11
B7	1.98	0.05	328.67	0.00	328.67	328.67	986.01	0.00	0.00	0.00	0.00	0.00
B8	1.93	0.05	359.70	0.00	359.70	359.70	1075.06	0.00	2.01	0.00	2.01	0.00
B9	2.09	0.05	404.25	1.06	402.67	407.95	1127.96	23.09	62.44	16.78	22.35	3.50
B10	1.96	0.05	469.58	0.00	469.58	469.58	1349.95	0.00	31.05	0.00	27.75	0.00
	1.92	0.04	415.78	1.07	413.86	417.56	1136.68	12.04	74.03	6.23	36.63	2.98
C1	5.89	0.13	977.56	14.39	965.15	1032.31	1755.24	22.44	986.40	28.26	191.05	7.05
C2	6.06	0.13	590.45	8.53	583.39	614.36	1711.21	18.09	44.68	12.97	15.45	8.85
C3	5.84	0.14	559.53	6.44	548.79	571.40	1665.60	18.81	8.16	3.29	4.82	1.66
C4	5.68	0.13	531.91	7.38	519.91	545.72	1570.75	18.60	15.02	7.13	9.95	3.56
C5	5.01	0.11	698.14	19.54	678.49	748.30	1812.64	64.44	203.13	17.96	78.65	21.52
C6	5.85	0.14	845.30	4.30	840.69	860.27	1630.49	17.69	724.95	11.36	180.46	6.31
C7	6.58	0.15	540.42	5.33	534.01	552.25	1620.39	15.30	0.53	2.34	0.34	1.49
C8	5.69	0.14	479.75	3.60	474.55	487.85	1403.47	10.76	26.71	0.30	9.06	0.00
C9	6.62	0.14	551.72	9.25	534.30	572.79	1647.07	30.12	4.77	4.56	3.32	3.97
C10	5.51	0.14	618.83	4.36	611.25	625.83	1774.94	8.90	65.01	4.94	16.52	0.28
	5.87	0.13	639.36	8.31	629.05	661.11	1659.18	22.52	207.94	9.31	50.96	5.47

Table D.1 – (Continued.) Detailed results for the biased random-key genetic algorithm.

Instance	Time (sec.)		Cost				Travel time		Total tardiness		Maximum tardiness	
	Mean	SD	Mean	SD	Min	Max	Mean	SD	Mean	SD	Mean	SD
D1	13.81	0.32	1208.19	13.98	1186.20	1235.16	2320.03	39.85	1053.89	5.78	250.65	0.00
D2	11.58	0.27	719.52	15.00	693.28	745.75	2140.75	41.93	11.84	13.07	5.96	6.44
D3	13.95	0.35	650.29	7.63	635.67	669.30	1891.97	27.40	36.72	11.42	22.17	2.32
D4	11.64	0.25	840.61	13.72	809.45	857.65	2039.38	35.84	375.32	13.66	107.14	9.34
D5	13.48	0.27	703.23	12.10	691.50	738.03	2106.21	35.78	2.03	2.59	1.44	1.47
D6	14.43	0.32	744.99	7.32	733.67	757.09	2223.24	24.75	7.90	5.84	3.83	1.48
D7	15.03	0.32	603.65	6.42	590.64	614.55	1798.04	23.93	8.25	12.16	4.64	6.13
D8	13.84	0.29	681.10	12.47	661.78	701.03	2039.46	38.20	2.25	2.94	1.59	1.58
D9	13.18	0.30	722.09	10.35	704.63	741.09	2155.53	37.30	6.20	6.62	4.54	4.44
D10	11.00	0.28	1294.66	66.45	1208.71	1410.00	2547.23	44.48	1135.44	138.49	201.31	63.68
	13.20	0.30	816.83	16.54	791.55	846.96	2126.18	34.95	263.99	21.26	60.33	9.69
E1	24.46	0.49	1352.33	10.75	1331.49	1373.09	2794.45	37.40	1037.83	10.24	224.72	3.67
E2	23.57	0.46	869.45	15.00	848.08	903.72	2597.63	42.38	7.08	10.48	3.63	4.13
E3	23.10	0.47	815.31	15.16	788.03	848.40	2437.06	46.06	5.61	5.84	3.26	3.30
E4	24.69	0.57	731.24	11.75	711.19	757.48	2186.51	37.84	4.43	2.82	2.79	1.55
E5	23.10	0.47	806.81	13.73	781.50	827.63	2413.29	45.51	4.45	6.43	2.68	3.54
E6	24.44	0.42	803.28	7.86	788.08	815.16	2399.33	25.40	6.45	7.69	4.07	4.51
E7	26.69	0.53	731.83	9.38	711.11	746.44	2190.65	29.30	2.85	3.48	2.00	2.11
E8	22.91	0.49	761.06	7.40	748.48	775.06	2272.41	21.30	6.72	3.76	4.06	2.24
E9	22.44	0.54	950.89	13.90	921.78	974.64	2836.77	40.29	10.75	9.70	5.13	3.40
E10	23.99	0.49	847.64	14.48	825.24	876.57	2530.17	44.30	8.33	9.15	4.41	4.69
	23.94	0.49	866.98	11.94	845.50	889.82	2465.83	36.98	109.45	6.96	25.67	3.31

Table D.1 – (Continued.) Detailed results for the biased random-key genetic algorithm.

Instance	Time (sec.)		Cost				Travel time		Total tardiness		Maximum tardiness	
	Mean	SD	Mean	SD	Min	Max	Mean	SD	Mean	SD	Mean	SD
F1	89.15	1.65	1421.76	17.34	1372.69	1449.11	4248.11	52.15	12.43	10.28	4.72	3.79
F2	95.02	2.00	1383.55	23.01	1336.33	1422.51	4128.87	72.84	15.00	12.87	6.79	4.05
F3	91.24	1.66	1288.66	14.08	1263.39	1313.00	3851.55	41.66	10.11	11.30	4.32	4.24
F4	110.09	1.77	1145.53	10.23	1124.24	1165.45	3428.19	26.46	5.43	7.08	2.97	4.14
F5	101.85	1.56	1361.64	20.75	1316.54	1394.58	4060.32	66.47	19.10	12.63	5.49	2.14
F6	85.45	1.19	1369.38	25.66	1322.89	1415.66	4081.17	81.57	20.15	16.16	6.81	4.07
F7	107.73	1.54	1163.97	17.53	1131.27	1202.33	3475.11	57.17	12.36	12.09	4.43	3.44
F8	96.05	1.62	1165.67	16.72	1132.77	1199.45	3484.96	48.51	8.07	10.71	3.99	4.19
F9	101.87	1.79	1347.78	22.91	1293.78	1384.01	4021.57	71.26	15.40	18.28	6.37	5.41
F10	98.96	1.80	1446.56	15.68	1418.53	1474.77	4306.22	47.75	26.08	21.84	7.37	5.91
	97.74	1.66	1309.45	18.39	1271.24	1342.09	3908.61	56.58	14.41	13.32	5.33	4.14
G1	221.55	4.15	1851.25	32.13	1778.54	1928.02	5502.19	94.30	38.88	16.68	12.68	6.97
G2	243.95	4.40	1899.33	34.71	1824.74	1972.62	5657.46	104.96	30.49	19.31	10.05	6.45
G3	216.58	4.16	1546.44	13.92	1514.23	1566.15	4615.27	41.61	18.63	10.02	5.42	2.46
G4	253.52	5.65	1599.35	23.92	1564.42	1639.61	4773.77	69.34	18.23	11.64	6.05	3.05
G5	211.75	3.56	1750.03	28.07	1694.50	1784.78	5217.32	87.56	23.29	16.87	9.48	6.68
G6	264.39	4.04	1779.29	21.68	1714.38	1812.47	5315.27	73.92	16.97	13.80	5.63	4.37
G7	224.19	4.15	1677.66	22.99	1640.07	1718.83	5015.67	69.91	13.26	9.82	4.04	2.92
G8	221.25	3.97	1582.71	20.05	1547.63	1627.78	4734.06	63.57	10.73	8.52	3.33	2.25
G9	229.19	3.74	1974.16	19.69	1942.21	2027.64	5878.43	71.53	34.57	23.43	9.47	6.45
G10	205.69	4.68	1931.99	25.31	1872.08	1977.14	5763.49	75.87	24.16	18.92	8.31	4.60
	229.20	4.25	1759.22	24.25	1709.28	1805.50	5247.29	75.26	22.92	14.90	7.45	4.62

APPENDIX E — EXTENDED RESULTS FOR BRKGA-MP-IPR

This appendix contains detailed information regarding the runs of the BRKGA meta-heuristic proposed in Chapter 8, with the extensions of Section 8.4. Table E.1 presents detailed results for each instance from the Mankowska, Meisel and Bierwirth (2014) dataset (column *Instance*). Under *Time (sec)*, we present the average and standard deviation of algorithm runtime. Under *Cost*, we present the cost of the average and standard deviation of solution cost, and the best and the worst values produced. The last columns present the average and the standard deviation regarding the objective function components.

Table E.1 – Detailed results for the BRKGA-MP-IPR.

Instance	Time (sec.)		Cost				Travel time		Total tardiness		Maximum tardiness		Resets	
	Mean	SD	Mean	SD	Min	Max	Mean	SD	Mean	SD	Mean	SD	Mean	SD
A1	97.22	4.99	226.98	0.00	226.98	226.98	680.93	0.00	0.00	0.00	0.00	0.00	2.00	0.00
A2	97.40	3.75	246.63	0.00	246.63	246.63	687.29	0.00	26.30	0.00	26.30	0.00	2.00	0.00
A3	98.34	4.49	305.86	0.00	305.86	305.86	741.14	0.00	99.30	0.00	77.10	0.00	2.00	0.00
A4	97.54	3.83	186.90	0.00	186.90	186.90	455.27	0.00	64.90	0.00	40.50	0.00	2.00	0.00
A5	96.72	4.51	191.97	0.01	191.97	191.99	575.92	0.01	0.00	0.00	0.00	0.00	2.00	0.00
A6	97.74	4.47	200.14	0.01	200.13	200.16	600.40	0.04	0.00	0.00	0.00	0.00	2.00	0.00
A7	96.88	4.32	225.37	0.00	225.37	225.37	676.11	0.00	0.00	0.00	0.00	0.00	2.00	0.00
A8	98.25	3.74	232.05	0.00	232.05	232.05	653.27	0.00	26.50	0.00	16.40	0.00	2.00	0.00
A9	96.02	4.23	234.21	0.00	234.21	234.21	677.00	0.00	12.80	0.00	12.80	0.00	2.00	0.00
A10	95.77	4.10	225.01	0.00	225.01	225.01	675.02	0.00	0.00	0.00	0.00	0.00	2.00	0.00
	97.18	4.24	227.51	0.00	227.51	227.52	642.23	0.00	22.98	0.00	17.31	0.00	2.00	0.00

Table E.1 – (Continued.) Detailed results for the BRKGA-MP-IPR.

Instance	Time (sec.)		Cost				Travel time		Total tardiness		Maximum tardiness		Resets	
	Mean	SD	Mean	SD	Min	Max	Mean	SD	Mean	SD	Mean	SD	Mean	SD
B1	104.28	4.10	428.10	0.00	428.10	428.10	1253.02	0.00	21.70	0.00	9.60	0.00	1.60	0.50
B2	104.52	4.62	483.63	0.00	483.63	483.63	1338.24	0.00	59.30	0.00	53.40	0.00	2.00	0.00
B3	103.83	4.86	402.80	0.00	402.80	402.80	914.89	0.00	212.60	0.00	80.90	0.00	2.00	0.00
B4	105.64	3.77	429.87	4.71	420.29	432.16	1203.20	13.69	56.88	1.72	29.50	0.00	1.90	0.31
B5	102.08	3.90	372.86	1.92	372.16	378.39	1065.43	25.19	33.78	12.58	19.37	7.16	1.90	0.31
B6	106.34	4.63	471.00	0.00	471.00	471.00	977.48	0.00	292.20	0.00	143.30	0.00	1.95	0.22
B7	103.87	4.74	328.67	0.00	328.67	328.67	986.01	0.00	0.00	0.00	0.00	0.00	2.00	0.00
B8	105.07	4.81	359.70	0.00	359.70	359.70	1075.06	0.00	2.00	0.00	2.00	0.00	2.00	0.00
B9	106.52	4.25	403.03	0.64	402.67	404.11	1117.62	1.92	68.00	0.00	23.50	0.00	1.95	0.22
B10	105.67	4.82	469.58	0.00	469.58	469.58	1349.95	0.00	31.00	0.00	27.80	0.00	2.00	0.00
	104.78	4.45	414.92	0.73	413.86	415.81	1128.09	4.08	77.75	1.43	38.94	0.72	1.93	0.16
C1	126.60	5.45	966.64	1.64	965.15	971.58	1737.22	3.61	971.91	4.54	190.80	0.00	1.60	0.50
C2	132.42	5.93	582.80	1.15	582.22	586.42	1698.14	4.03	37.54	5.69	12.69	0.17	1.85	0.37
C3	128.82	5.73	551.41	1.54	548.79	554.24	1644.27	4.89	5.36	1.16	4.60	0.00	1.60	0.50
C4	127.71	4.39	522.95	2.53	519.84	526.15	1547.21	13.18	14.10	5.54	7.50	0.21	1.70	0.47
C5	127.77	4.92	675.06	4.51	670.61	681.47	1741.50	14.20	216.73	10.23	66.91	13.88	1.65	0.49
C6	127.22	5.78	841.77	2.97	830.30	845.75	1619.16	8.10	726.42	7.54	179.68	4.17	1.75	0.55
C7	132.54	5.54	532.65	3.11	528.53	537.47	1595.30	12.45	1.36	6.06	1.29	5.77	1.30	0.47
C8	131.10	5.10	474.67	1.69	471.33	477.66	1388.41	5.06	26.60	0.00	9.10	0.00	1.45	0.51
C9	131.01	4.31	537.90	4.38	531.15	544.84	1595.13	21.78	11.12	8.37	7.45	4.74	1.65	0.49
C10	128.51	5.67	612.06	1.81	610.86	618.40	1758.98	5.95	60.74	2.70	16.40	0.00	1.90	0.31
	129.37	5.28	629.79	2.53	625.88	634.40	1632.53	9.33	207.19	5.18	49.64	2.89	1.65	0.47

Table E.1 – (Continued.) Detailed results for the BRKGA-MP-IPR.

Instance	Time (sec.)		Cost				Travel time		Total tardiness		Maximum tardiness		Resets	
	Mean	SD	Mean	SD	Min	Max	Mean	SD	Mean	SD	Mean	SD	Mean	SD
D1	166.18	6.97	1196.97	6.57	1179.85	1211.00	2288.80	19.38	1051.47	0.65	250.70	0.00	1.20	0.41
D2	155.33	5.73	693.70	6.95	684.28	709.55	2064.80	22.18	10.44	10.57	5.88	4.42	1.25	0.64
D3	168.48	6.83	640.72	3.55	633.43	647.98	1861.99	18.06	37.67	8.89	22.47	2.00	1.20	0.52
D4	158.69	5.97	827.96	14.03	800.17	848.83	2003.85	37.70	373.05	5.71	106.96	7.36	1.05	0.60
D5	163.74	6.65	691.52	4.19	682.77	701.29	2072.45	12.83	1.23	1.59	0.86	1.05	1.00	0.46
D6	169.51	6.96	727.49	6.28	712.11	734.58	2167.90	19.60	10.76	2.41	3.83	0.40	1.25	0.44
D7	175.32	7.03	592.97	5.10	585.58	600.93	1773.18	14.24	3.58	6.71	2.18	3.75	1.00	0.00
D8	168.60	7.71	663.39	3.85	657.81	673.01	1987.43	13.45	1.60	2.72	1.15	1.79	1.45	0.51
D9	162.42	5.61	711.92	11.58	692.93	736.30	2118.30	43.44	12.25	13.33	5.22	4.67	1.10	0.45
D10	155.17	6.38	1225.40	19.83	1201.68	1287.58	2529.19	32.58	996.88	31.85	150.17	29.46	0.90	0.45
	164.34	6.58	797.20	8.19	783.06	815.11	2086.79	23.35	249.89	8.44	54.94	5.49	1.14	0.45
E1	217.90	6.66	1334.23	10.04	1318.89	1353.43	2739.93	31.94	1038.87	8.57	223.90	0.00	0.70	0.47
E2	216.35	7.17	846.40	15.30	824.18	872.54	2532.85	43.97	4.03	5.15	2.31	2.86	0.80	0.52
E3	214.39	7.60	795.12	6.71	785.10	817.51	2371.28	25.28	8.70	6.53	5.39	4.17	1.00	0.00
E4	220.07	7.75	719.59	8.71	703.11	736.62	2150.39	28.38	5.19	2.20	3.18	1.44	0.90	0.45
E5	215.20	6.53	777.38	10.69	760.49	804.86	2324.25	34.90	4.20	5.15	3.68	4.93	0.90	0.45
E6	220.59	7.90	789.17	8.39	772.09	804.28	2360.09	25.90	4.10	4.15	3.33	3.18	0.90	0.31
E7	229.40	6.61	718.66	6.45	705.85	730.97	2152.90	21.05	1.87	3.55	1.20	1.85	0.95	0.39
E8	215.46	4.41	744.82	9.33	725.62	765.32	2224.86	27.94	6.09	4.46	3.50	2.27	0.95	0.22
E9	211.47	7.71	917.28	16.77	877.08	943.36	2736.06	50.77	10.60	6.33	5.17	2.53	0.80	0.41
E10	217.22	7.65	829.15	9.64	814.97	858.35	2469.03	31.51	12.06	11.09	6.35	4.72	0.70	0.57
	217.80	7.00	847.18	10.20	828.74	868.72	2406.16	32.17	109.57	5.72	25.80	2.79	0.86	0.38

Table E.1 – (Continued.) Detailed results for the BRKGA-MP-IPR.

Instance	Time (sec.)		Cost				Travel time		Total tardiness		Maximum tardiness		Resets	
	Mean	SD	Mean	SD	Min	Max	Mean	SD	Mean	SD	Mean	SD	Mean	SD
F1	528.29	16.27	1374.11	17.30	1342.08	1406.75	4097.98	58.80	16.89	11.15	7.48	4.39	0.45	0.51
F2	555.84	18.03	1324.05	15.99	1296.03	1347.74	3954.40	53.81	12.62	8.45	5.14	3.28	0.26	0.45
F3	541.07	17.87	1228.36	13.57	1201.14	1255.92	3669.87	42.33	10.90	7.65	4.29	2.55	0.75	0.44
F4	631.01	20.61	1108.59	13.77	1093.90	1137.91	3318.15	43.20	5.11	4.51	2.50	2.11	0.45	0.51
F5	584.98	16.18	1316.58	14.78	1286.79	1343.10	3929.20	45.20	15.84	8.87	4.71	2.32	0.60	0.50
F6	517.20	12.42	1316.40	11.30	1295.58	1336.61	3918.37	41.98	23.69	15.76	7.13	3.76	0.40	0.50
F7	615.18	21.38	1117.70	18.93	1085.26	1152.99	3347.73	60.89	3.45	3.52	1.94	2.23	0.60	0.50
F8	557.76	18.35	1132.50	12.09	1098.74	1150.00	3391.93	37.25	3.56	2.99	2.03	1.15	0.65	0.49
F9	586.57	17.19	1292.89	19.93	1257.97	1320.67	3868.07	57.14	7.51	8.94	3.11	2.90	0.60	0.50
F10	574.09	18.16	1394.40	15.24	1357.96	1422.20	4155.46	46.94	21.18	14.39	6.57	3.21	0.40	0.50
	569.20	17.65	1260.56	15.29	1231.55	1287.39	3765.11	48.75	12.07	8.62	4.49	2.79	0.52	0.49
G1	1182.79	41.82	1750.11	19.03	1726.53	1783.79	5212.08	63.10	29.74	20.57	8.50	6.50	0.35	0.49
G2	1277.04	50.63	1770.65	21.63	1714.04	1809.72	5285.96	64.59	20.47	16.26	5.53	4.44	0.10	0.31
G3	1160.42	47.26	1475.60	15.29	1441.85	1502.72	4414.32	49.04	9.61	8.93	2.85	1.78	0.10	0.31
G4	1320.43	57.01	1523.43	19.27	1490.08	1563.33	4546.89	65.58	17.55	12.63	5.86	3.44	0.35	0.49
G5	1120.27	40.78	1650.55	22.18	1615.69	1702.54	4928.76	69.78	16.36	10.41	6.52	3.98	0.25	0.44
G6	1384.55	58.68	1683.23	19.84	1656.79	1725.85	5033.22	57.88	12.12	9.45	4.35	3.20	0.30	0.47
G7	1191.06	44.74	1593.97	19.76	1567.74	1640.50	4764.69	63.31	13.42	6.87	3.80	1.61	0.35	0.49
G8	1168.12	41.87	1505.58	15.99	1477.58	1532.46	4512.23	48.26	3.04	3.19	1.47	1.45	0.10	0.31
G9	1195.74	34.53	1875.77	34.55	1827.74	1980.77	5593.79	102.71	26.12	14.23	7.40	3.19	0.00	0.00
G10	1095.57	36.13	1828.04	28.94	1781.16	1903.30	5469.21	86.83	11.18	10.64	3.75	2.99	0.05	0.22
	1209.60	45.35	1665.69	21.65	1629.92	1714.50	4976.11	67.11	15.96	11.32	5.00	3.26	0.20	0.35

APPENDIX F — DETAILS OF SELECTED INSTANCES FOR THE NEW DATASET

Table F.1 – Selected instances for the new dataset.

Instance	Seed	Patients	Caregivers	Generation config			Hardness measure			Sel. reason
				Cluster density	Depot placement	Node placement	LB	UB	Gap (%)	
HHCRSP_10_3_88_1.6_R_C	88	10	3	1.6	Random	Clustered	16.69	150.70	88.93	Hardness
HHCRSP_10_3_40_0.8_R_C	40	10	3	0.8	Random	Clustered	18.13	129.70	86.02	Hardness
HHCRSP_10_3_40_1.0_R_C	40	10	3	1	Random	Clustered	18.13	129.70	86.02	Hardness
HHCRSP_10_3_35_1.0_R_R	35	10	3	1	Random	Random	55.70	385.00	85.53	Hardness
HHCRSP_10_3_88_1.0_R_C	88	10	3	1	Random	Clustered	17.68	113.70	84.45	Hardness
HHCRSP_10_3_88_1.2_R_C	88	10	3	1.2	Random	Clustered	17.68	113.70	84.45	Hardness
HHCRSP_10_3_81_1.2_R_C	81	10	3	1.2	Random	Clustered	38.06	241.70	84.26	Hardness
HHCRSP_10_3_86_1.2_C_C	86	10	3	1.2	Central	Clustered	31.77	198.70	84.01	Hardness
HHCRSP_10_3_88_1.6_C_C	88	10	3	1.6	Central	Clustered	14.35	89.30	83.93	Hardness
HHCRSP_10_3_16_1.6_C_C	16	10	3	1.6	Central	Clustered	31.03	183.70	83.11	Hardness
							25.92	173.59	569.68	
HHCRSP_10_3_96_1.0_R_C	96	10	3	1	Random	Clustered	36.75	126.70	71.00	Sampling
HHCRSP_10_3_46_1.6_R_C	46	10	3	1.6	Random	Clustered	33.35	86.00	61.23	Sampling
HHCRSP_10_3_80_1.0_C_R	80	10	3	1	Central	Random	42.36	102.30	58.59	Sampling
HHCRSP_10_3_56_1.2_R_RC	56	10	3	1.2	Random	Random-clustered	42.36	92.30	54.11	Sampling
HHCRSP_10_3_22_1.6_R_C	22	10	3	1.6	Random	Clustered	42.02	79.00	46.81	Sampling
HHCRSP_10_3_11_0.4_R_C	11	10	3	0.4	Random	Clustered	61.02	110.30	44.68	Sampling
HHCRSP_10_3_51_1.0_R_RC	51	10	3	1	Random	Random-clustered	54.68	97.00	43.63	Sampling
HHCRSP_10_3_90_1.2_C_C	90	10	3	1.2	Central	Clustered	42.34	64.30	34.15	Sampling
HHCRSP_10_3_90_1.2_C_RC	90	10	3	1.2	Central	Random-clustered	49.68	75.00	33.77	Sampling
HHCRSP_10_3_60_1.0_R_RC	60	10	3	1	Random	Random-clustered	57.01	83.00	31.31	Sampling
							46.15	91.59	98.44	

Table F.1 – (Continued.) Selected instances for the new dataset.

Instance	Seed	Patients	Caregivers	Generation config			Hardness measure			Sel. reason
				Cluster density	Depot placement	Node placement	LB	UB	Gap (%)	
HHCRSP_25_5_22_1.0_C_C	22	25	5	1	Central	Clustered	55.02	899.30	93.88	Hardness
HHCRSP_25_5_69_1.6_R_RC	69	25	5	1.6	Random	Random-clustered	72.73	1171.00	93.79	Hardness
HHCRSP_25_5_47_0.8_R_C	47	25	5	0.8	Random	Clustered	53.70	848.70	93.67	Hardness
HHCRSP_25_5_37_0.8_R_RC	37	25	5	0.8	Random	Random-clustered	93.38	1441.30	93.52	Hardness
HHCRSP_25_5_37_1.0_R_RC	37	25	5	1	Random	Random-clustered	93.38	1441.30	93.52	Hardness
HHCRSP_25_5_26_0.8_C_C	26	25	5	0.8	Central	Clustered	62.35	956.30	93.48	Hardness
HHCRSP_25_5_6_0.8_R_C	6	25	5	0.8	Random	Clustered	48.35	706.00	93.15	Hardness
HHCRSP_25_5_69_1.6_R_C	69	25	5	1.6	Random	Clustered	52.67	715.30	92.64	Hardness
HHCRSP_25_5_42_0.8_R_RC	42	25	5	0.8	Random	Random-clustered	93.08	1196.70	92.22	Hardness
HHCRSP_25_5_96_0.8_R_C	96	25	5	0.8	Random	Clustered	57.38	730.30	92.14	Hardness
							68.20	1010.62	1381.75	
HHCRSP_25_5_80_0.8_C_RC	80	25	5	0.8	Central	Random-clustered	116.06	437.30	73.46	Sampling
HHCRSP_25_5_48_1.0_C_C	48	25	5	1	Central	Clustered	78.38	280.30	72.04	Sampling
HHCRSP_25_5_42_0.8_C_RC	42	25	5	0.8	Central	Random-clustered	101.02	287.70	64.89	Sampling
HHCRSP_25_5_56_0.8_R_RC	56	25	5	0.8	Random	Random-clustered	63.34	179.70	64.75	Sampling
HHCRSP_25_5_76_1.2_C_C	76	25	5	1.2	Central	Clustered	71.70	184.00	61.03	Sampling
HHCRSP_25_5_70_1.0_R_C	70	25	5	1	Random	Clustered	65.02	146.00	55.47	Sampling
HHCRSP_25_5_21_1.6_R_RC	21	25	5	1.6	Random	Random-clustered	93.67	205.00	54.31	Sampling
HHCRSP_25_5_73_0.4_R_RC	73	25	5	0.4	Random	Random-clustered	73.69	155.30	52.55	Sampling
HHCRSP_25_5_97_1.0_R_R	97	25	5	1	Random	Random	123.35	258.70	52.32	Sampling
HHCRSP_25_5_49_0.4_C_C	49	25	5	0.4	Central	Clustered	86.34	141.70	39.07	Sampling
							87.26	227.57	160.81	

Table F.1 – (Continued.) Selected instances for the new dataset.

Instance	Seed	Patients	Caregivers	Generation config			Hardness measure			Sel. reason
				Cluster density	Depot placement	Node placement	LB	UB	Gap (%)	
HHCRSP_50_10_80_1.0_C_C	80	50	10	1	Central	Clustered	102.19	855.30	88.05	Hardness
HHCRSP_50_10_81_1.0_R_RC	81	50	10	1	Random	Random-clustered	133.37	1053.70	87.34	Hardness
HHCRSP_50_10_80_1.0_C_RC	80	50	10	1	Central	Random-clustered	152.02	1081.30	85.94	Hardness
HHCRSP_50_10_97_0.8_C_RC	97	50	10	0.8	Central	Random-clustered	129.70	784.00	83.46	Hardness
HHCRSP_50_10_32_1.2_R_C	32	50	10	1.2	Random	Clustered	85.68	473.70	81.91	Hardness
HHCRSP_50_10_3_0.8_R_C	3	50	10	0.8	Random	Clustered	112.01	617.30	81.85	Hardness
HHCRSP_50_10_80_0.4_C_C	80	50	10	0.4	Central	Clustered	144.69	788.70	81.65	Hardness
HHCRSP_50_10_26_1.0_C_RC	26	50	10	1	Central	Random-clustered	121.01	629.30	80.77	Hardness
HHCRSP_50_10_26_1.2_R_C	26	50	10	1.2	Random	Clustered	67.33	349.30	80.72	Hardness
HHCRSP_50_10_44_1.6_R_C	44	50	10	1.6	Random	Clustered	60.00	305.30	80.35	Hardness
							110.80	693.79	526.16	
HHCRSP_50_10_3_1.6_C_C	3	50	10	1.6	Central	Clustered	66.67	272.70	75.55	Sampling
HHCRSP_50_10_53_1.2_R_C	53	50	10	1.2	Random	Clustered	82.67	261.30	68.36	Sampling
HHCRSP_50_10_62_1.2_R_C	62	50	10	1.2	Random	Clustered	74.00	216.00	65.74	Sampling
HHCRSP_50_10_74_1.0_R_RC	74	50	10	1	Random	Random-clustered	113.67	319.00	64.37	Sampling
HHCRSP_50_10_61_1.0_R_RC	61	50	10	1	Random	Random-clustered	107.00	288.30	62.89	Sampling
HHCRSP_50_10_88_1.6_C_RC	88	50	10	1.6	Central	Random-clustered	130.00	342.70	62.07	Sampling
HHCRSP_50_10_60_0.4_C_RC	60	50	10	0.4	Central	Random-clustered	107.01	281.30	61.96	Sampling
HHCRSP_50_10_2_1.0_C_R	2	50	10	1	Central	Random	155.67	370.70	58.01	Sampling
HHCRSP_50_10_100_1.6_C_C	100	50	10	1.6	Central	Clustered	75.00	162.30	53.79	Sampling
HHCRSP_50_10_33_0.4_R_RC	33	50	10	0.4	Random	Random-clustered	140.33	287.00	51.10	Sampling
							105.20	280.13	166.28	

Table F.1 – (Continued.) Selected instances for the new dataset.

Instance	Seed	Patients	Caregivers	Generation config			Hardness measure			Sel. reason
				Cluster density	Depot placement	Node placement	LB	UB	Gap (%)	
HHCRSP_75_15_97_1.0_R_RC	97	75	15	1	Random	Random-clustered	144.35	924.30	84.38	Hardness
HHCRSP_75_15_35_0.8_R_C	35	75	15	0.8	Random	Clustered	130.34	774.30	83.17	Hardness
HHCRSP_75_15_35_1.0_C_RC	35	75	15	1	Central	Random-clustered	195.68	1145.30	82.91	Hardness
HHCRSP_75_15_97_1.0_C_C	97	75	15	1	Central	Clustered	116.67	662.30	82.38	Hardness
HHCRSP_75_15_79_1.2_R_C	79	75	15	1.2	Random	Clustered	89.33	495.00	81.95	Hardness
HHCRSP_75_15_10_1.6_R_C	10	75	15	1.6	Random	Clustered	78.67	426.00	81.53	Hardness
HHCRSP_75_15_88_1.6_R_C	88	75	15	1.6	Random	Clustered	98.67	522.70	81.12	Hardness
HHCRSP_75_15_98_1.6_R_C	98	75	15	1.6	Random	Clustered	81.51	423.30	80.74	Hardness
HHCRSP_75_15_21_1.6_R_C	21	75	15	1.6	Random	Clustered	80.67	411.30	80.39	Hardness
HHCRSP_75_15_87_1.6_C_C	87	75	15	1.6	Central	Clustered	78.67	400.30	80.35	Hardness
							109.46	618.48	465.05	
HHCRSP_75_15_97_0.4_C_C	97	75	15	0.4	Central	Clustered	162.03	541.30	70.07	Sampling
HHCRSP_75_15_63_1.0_R_RC	63	75	15	1	Random	Random-clustered	147.33	453.30	67.50	Sampling
HHCRSP_75_15_5_0.8_C_RC	5	75	15	0.8	Central	Random-clustered	183.33	503.30	63.57	Sampling
HHCRSP_75_15_19_0.8_R_C	19	75	15	0.8	Random	Clustered	141.01	377.00	62.60	Sampling
HHCRSP_75_15_84_0.8_R_RC	84	75	15	0.8	Random	Random-clustered	162.01	431.30	62.44	Sampling
HHCRSP_75_15_30_1.0_R_C	30	75	15	1	Random	Clustered	128.00	328.70	61.06	Sampling
HHCRSP_75_15_32_1.2_R_RC	32	75	15	1.2	Random	Random-clustered	151.67	380.00	60.09	Sampling
HHCRSP_75_15_40_0.8_R_RC	40	75	15	0.8	Random	Random-clustered	184.33	452.70	59.28	Sampling
HHCRSP_75_15_63_0.4_R_C	63	75	15	0.4	Random	Clustered	142.33	340.30	58.17	Sampling
HHCRSP_75_15_65_1.0_C_C	65	75	15	1	Central	Clustered	118.67	280.00	57.62	Sampling
							152.07	408.79	168.81	

Table F.1 – (Continued.) Selected instances for the new dataset.

Instance	Seed	Patients	Caregivers	Generation config			Hardness measure			Sel. reason
				Cluster density	Depot placement	Node placement	LB	UB	Gap (%)	
HHCRSP_100_20_8_1.2_R_C	8	100	20	1.2	Random	Clustered	117.00	729.70	83.97	Hardness
HHCRSP_100_20_63_1.2_R_C	63	100	20	1.2	Random	Clustered	112.00	655.70	82.92	Hardness
HHCRSP_100_20_39_1.6_R_C	39	100	20	1.6	Random	Clustered	112.34	648.00	82.66	Hardness
HHCRSP_100_20_76_1.6_R_C	76	100	20	1.6	Random	Clustered	100.67	568.00	82.28	Hardness
HHCRSP_100_20_57_1.6_R_C	57	100	20	1.6	Random	Clustered	105.00	574.70	81.73	Hardness
HHCRSP_100_20_25_0.8_R_C	25	100	20	0.8	Random	Clustered	138.33	750.70	81.57	Hardness
HHCRSP_100_20_77_1.0_R_C	77	100	20	1	Random	Clustered	134.67	730.00	81.55	Hardness
HHCRSP_100_20_11_1.2_R_C	11	100	20	1.2	Random	Clustered	118.00	634.00	81.39	Hardness
HHCRSP_100_20_30_1.0_R_C	30	100	20	1	Random	Clustered	133.33	707.00	81.14	Hardness
HHCRSP_100_20_23_1.2_R_C	23	100	20	1.2	Random	Clustered	102.00	528.70	80.71	Hardness
							117.33	652.65	456.23	
HHCRSP_100_20_68_1.6_R_C	68	100	20	1.6	Random	Clustered	87.00	351.30	75.23	Sampling
HHCRSP_100_20_49_0.4_C_C	49	100	20	0.4	Central	Clustered	211.33	586.30	63.95	Sampling
HHCRSP_100_20_61_0.4_R_RC	61	100	20	0.4	Random	Random-clustered	187.00	510.30	63.35	Sampling
HHCRSP_100_20_88_1.2_C_C	88	100	20	1.2	Central	Clustered	111.33	301.70	63.10	Sampling
HHCRSP_100_20_9_1.0_R_R	9	100	20	1	Random	Random	237.67	625.30	61.99	Sampling
HHCRSP_100_20_35_1.6_C_RC	35	100	20	1.6	Central	Random-clustered	185.33	481.30	61.49	Sampling
HHCRSP_100_20_10_1.2_R_C	10	100	20	1.2	Random	Clustered	132.67	340.30	61.01	Sampling
HHCRSP_100_20_65_0.4_R_RC	65	100	20	0.4	Random	Random-clustered	212.00	533.70	60.28	Sampling
HHCRSP_100_20_99_0.4_C_RC	99	100	20	0.4	Central	Random-clustered	216.33	543.70	60.21	Sampling
HHCRSP_100_20_54_1.2_R_RC	54	100	20	1.2	Random	Random-clustered	176.00	435.30	59.57	Sampling
							175.67	470.92	168.08	

Table F.1 – (Continued.) Selected instances for the new dataset.

Instance	Seed	Patients	Caregivers	Generation config			Hardness measure			Sel. reason
				Cluster density	Depot placement	Node placement	LB	UB	Gap (%)	
HHCRSP_200_40_52_1.6_R_C	52	200	40	1.6	Random	Clustered	171.67	1109.70	84.53	Hardness
HHCRSP_200_40_7_1.2_R_C	7	200	40	1.2	Random	Clustered	187.00	1207.70	84.52	Hardness
HHCRSP_200_40_30_1.6_R_C	30	200	40	1.6	Random	Clustered	180.33	1143.00	84.22	Hardness
HHCRSP_200_40_76_1.2_R_C	76	200	40	1.2	Random	Clustered	218.00	1352.00	83.88	Hardness
HHCRSP_200_40_7_1.6_R_C	7	200	40	1.6	Random	Clustered	149.67	923.00	83.78	Hardness
HHCRSP_200_40_51_1.0_R_C	51	200	40	1	Random	Clustered	214.00	1317.30	83.75	Hardness
HHCRSP_200_40_17_1.6_R_C	17	200	40	1.6	Random	Clustered	170.67	1005.00	83.02	Hardness
HHCRSP_200_40_40_1.0_R_C	40	200	40	1	Random	Clustered	200.67	1128.30	82.22	Hardness
HHCRSP_200_40_71_1.6_R_C	71	200	40	1.6	Random	Clustered	182.00	1018.30	82.13	Hardness
HHCRSP_200_40_98_1.2_R_C	98	200	40	1.2	Random	Clustered	202.00	1114.30	81.87	Hardness
							187.60	1131.86	503.34	
HHCRSP_200_40_17_1.6_C_C	17	200	40	1.6	Central	Clustered	158.00	650.00	75.69	Sampling
HHCRSP_200_40_75_1.6_R_C	75	200	40	1.6	Random	Clustered	200.33	791.70	74.70	Sampling
HHCRSP_200_40_80_1.0_C_C	80	200	40	1	Central	Clustered	214.67	756.70	71.63	Sampling
HHCRSP_200_40_82_1.0_C_R	82	200	40	1	Central	Random	329.00	1123.00	70.70	Sampling
HHCRSP_200_40_69_1.0_C_R	69	200	40	1	Central	Random	342.67	1136.00	69.84	Sampling
HHCRSP_200_40_60_0.4_C_RC	60	200	40	0.4	Central	Random-clustered	306.33	1005.00	69.52	Sampling
HHCRSP_200_40_10_1.6_C_RC	10	200	40	1.6	Central	Random-clustered	309.00	1002.30	69.17	Sampling
HHCRSP_200_40_98_1.6_C_RC	98	200	40	1.6	Central	Random-clustered	294.00	944.30	68.87	Sampling
HHCRSP_200_40_100_0.4_C_C	100	200	40	0.4	Central	Clustered	268.33	824.30	67.45	Sampling
HHCRSP_200_40_71_1.6_C_C	71	200	40	1.6	Central	Clustered	182.33	511.70	64.37	Sampling
							260.47	874.50	235.74	

Table F.1 – (Continued.) Selected instances for the new dataset.

Instance	Seed	Patients	Caregivers	Generation config			Hardness measure			Sel. reason
				Cluster density	Depot placement	Node placement	LB	UB	Gap (%)	
HHCRSP_300_60_4_1.6_R_C	4	300	60	1.6	Random	Clustered	264.00	1976.00	86.64	Hardness
HHCRSP_300_60_89_1.6_R_C	89	300	60	1.6	Random	Clustered	244.00	1805.00	86.48	Hardness
HHCRSP_300_60_82_1.2_R_C	82	300	60	1.2	Random	Clustered	256.67	1852.70	86.15	Hardness
HHCRSP_300_60_95_1.2_R_C	95	300	60	1.2	Random	Clustered	276.00	1946.00	85.82	Hardness
HHCRSP_300_60_22_1.2_R_C	22	300	60	1.2	Random	Clustered	271.33	1852.70	85.35	Hardness
HHCRSP_300_60_85_1.2_R_C	85	300	60	1.2	Random	Clustered	271.33	1767.30	84.65	Hardness
HHCRSP_300_60_81_1.6_R_C	81	300	60	1.6	Random	Clustered	250.00	1627.00	84.63	Hardness
HHCRSP_300_60_23_1.2_R_C	23	300	60	1.2	Random	Clustered	241.67	1567.00	84.58	Hardness
HHCRSP_300_60_34_1.2_R_C	34	300	60	1.2	Random	Clustered	288.67	1867.70	84.54	Hardness
HHCRSP_300_60_69_1.0_R_C	69	300	60	1	Random	Clustered	298.00	1920.00	84.48	Hardness
							266.17	1818.14	583.08	
HHCRSP_300_60_7_1.2_R_RC	7	300	60	1.2	Random	Random-clustered	382.00	1696.70	77.49	Sampling
HHCRSP_300_60_75_1.2_C_RC	75	300	60	1.2	Central	Random-clustered	342.33	1327.70	74.22	Sampling
HHCRSP_300_60_26_0.4_C_RC	26	300	60	0.4	Central	Random-clustered	399.33	1528.00	73.87	Sampling
HHCRSP_300_60_68_0.8_C_RC	68	300	60	0.8	Central	Random-clustered	391.33	1479.30	73.55	Sampling
HHCRSP_300_60_4_1.0_R_C	4	300	60	1	Random	Clustered	329.67	1200.30	72.53	Sampling
HHCRSP_300_60_73_0.8_R_RC	73	300	60	0.8	Random	Random-clustered	377.67	1369.00	72.41	Sampling
HHCRSP_300_60_45_1.6_C_C	45	300	60	1.6	Central	Clustered	248.33	854.70	70.95	Sampling
HHCRSP_300_60_46_0.4_R_C	46	300	60	0.4	Random	Clustered	387.00	1269.00	69.50	Sampling
HHCRSP_300_60_81_0.8_C_C	81	300	60	0.8	Central	Clustered	302.00	933.70	67.66	Sampling
HHCRSP_300_60_48_0.8_C_C	48	300	60	0.8	Central	Clustered	327.33	980.70	66.62	Sampling
							348.70	1263.91	262.46	

Table F.1 – (Continued.) Selected instances for the new dataset.

Instance	Seed	Patients	Caregivers	Generation config			Hardness measure			Sel. reason
				Cluster density	Depot placement	Node placement	LB	UB	Gap (%)	
HHCRSP_400_80_59_1.0_R_C	59	400	80	1	Random	Clustered	330.67	2891.00	88.56	Hardness
HHCRSP_400_80_16_1.2_R_C	16	400	80	1.2	Random	Clustered	309.00	2669.00	88.42	Hardness
HHCRSP_400_80_51_1.0_R_C	51	400	80	1	Random	Clustered	342.00	2811.70	87.84	Hardness
HHCRSP_400_80_97_1.6_R_C	97	400	80	1.6	Random	Clustered	275.67	2165.30	87.27	Hardness
HHCRSP_400_80_30_1.6_R_C	30	400	80	1.6	Random	Clustered	287.00	2173.70	86.80	Hardness
HHCRSP_400_80_34_1.0_R_C	34	400	80	1	Random	Clustered	307.00	2302.30	86.67	Hardness
HHCRSP_400_80_92_1.6_R_C	92	400	80	1.6	Random	Clustered	291.33	2180.70	86.64	Hardness
HHCRSP_400_80_91_1.2_R_C	91	400	80	1.2	Random	Clustered	335.00	2476.00	86.47	Hardness
HHCRSP_400_80_28_1.2_R_C	28	400	80	1.2	Random	Clustered	365.67	2692.30	86.42	Hardness
HHCRSP_400_80_81_1.6_R_C	81	400	80	1.6	Random	Clustered	294.00	2131.00	86.20	Hardness
							313.73	2449.30	680.69	
HHCRSP_400_80_54_1.2_C_RC	54	400	80	1.2	Central	Random-clustered	467.00	1945.70	76.00	Sampling
HHCRSP_400_80_20_0.4_C_RC	20	400	80	0.4	Central	Random-clustered	502.00	2088.30	75.96	Sampling
HHCRSP_400_80_61_0.4_C_RC	61	400	80	0.4	Central	Random-clustered	469.33	1948.30	75.91	Sampling
HHCRSP_400_80_33_1.0_C_RC	33	400	80	1	Central	Random-clustered	457.33	1876.30	75.63	Sampling
HHCRSP_400_80_95_1.0_C_R	95	400	80	1	Central	Random	484.33	1934.70	74.97	Sampling
HHCRSP_400_80_65_1.0_C_R	65	400	80	1	Central	Random	523.67	1879.70	72.14	Sampling
HHCRSP_400_80_45_0.4_C_C	45	400	80	0.4	Central	Clustered	437.67	1509.00	71.00	Sampling
HHCRSP_400_80_22_1.6_C_C	22	400	80	1.6	Central	Clustered	324.67	1106.30	70.65	Sampling
HHCRSP_400_80_77_1.2_R_RC	77	400	80	1.2	Random	Random-clustered	471.00	1576.70	70.13	Sampling
HHCRSP_400_80_60_0.8_C_C	60	400	80	0.8	Central	Clustered	406.67	1286.30	68.38	Sampling
							454.37	1715.13	277.48	

Source: the author.