Metaheuristic Optimization for Complex Routing Problems

Von der Fakultät für Wirtschaftswissenschaften der Rheinisch-Westfälischen Technischen Hochschule Aachen zur Erlangung des akademischen Grades einer Doktorin der Wirtschafts- und Sozialwissenschaften genehmigte Dissertation

vorgelegt von
Alina Theiß

Berichter: Univ.-Prof. Dr. rer. pol. Michael Schneider
Berichterin: Univ.-Prof. Dr. rer. pol. Britta Peis

Tag der mündlichen Prüfung: 26. August 2025

Articles included in this thesis

R. Cavagnini, M. Schneider, and A. Theiß (2024a). “A granular iterated local search for the asymmetric single truck and trailer routing problem with satellite depots at DHL Group”. In: *Networks* 83.1, pp. 3–29. DOI: 10.1002/net.22178.

R. Cavagnini, M. Schneider, and A. Theiß (2024b). “A tabu search with geometry-based sparsification methods for angular traveling salesman problems”. In: *Networks* 83.1, pp. 30–52. DOI: 10.1002/net.22180.

A. Theiß, R. Cavagnini, and D. Gellert (2025). *An iterated local search for the capacitated team orienteering problem with time-dependent and piecewise linear score functions.* Working Paper. Chair of Computational Logistics, RWTH Aachen University, Germany.

A. Theiß, R. Cavagnini, and M. Schneider (2025). *The vehicle routing problem with depot operation constraints at DHL Group.* Working Paper. Chair of Computational Logistics, RWTH Aachen University, Germany.

Abstract

Optimizing transportation systems has become essential for addressing today's logistics challenges. As global trade grows consistently and consumer expectations for faster and on-time deliveries rise, companies face increasing pressure to deliver quickly and cost-efficiently.

This thesis addresses four routing problems: First, the single truck and trailer routing problem with satellite depots (STTRPSD), which can be used to model the problem of optimizing routes of mail carriers in the last mile delivery stage of a mail delivery network. Second, the vehicle routing problem with depot operation constraints (VRPDOC), which additionally includes the assignment of households to mail carriers in the route planning and incorporates depot operations. Third, the angular-metric traveling salesman problem (AngleTSP) and the angular-distance-metric traveling salesman problem (AngleDistanceTSP), relevant for minimizing sharp turns in the routing of heavy vehicles. Last, the capacitated team orienteering problem with time-dependent and piecewise-linear score functions (C-TOP-TDPLSF) often used in the context of customer-focused deliveries.

To efficiently solve realistically sized instances of these $\mathcal{NP}$-hard problems, we use metaheuristics like iterated local search or tabu search. Each heuristic is designed to incorporate problem specific features to enhance their performance. Extensive computational experiments show significant improvements compared to state-of-the-art algorithms from the literature and practices implemented in the real world. For the STTRPSD, we use its natural decomposition into subproblems in the design of our heuristic, that reduces the travel times of real-world solutions currently used in practice by our industry partner on average by approximately 2%. Addressing the VRPDOC, our work is one of the first ones to incorporate depot operations into the route planning. In our algorithm, we use problem-specific neighborhood operators and incorporate the instance structure of real-world street networks. On real-world instances, our heuristic is not only reducing total travel times by approximately 6.5% compared to the currently implemented solutions from our industry partner but also provides significantly simpler solutions with regards to the letter handling operations at the depot, highlighting the operational benefits of considering depot operation constraints. For the AngleTSP and the AngleDistanceTSP, we incorporate the geometric features of the problems. Our heuristic provides a good trade-off between runtime and solution quality, and we find new best-known solutions for around 80% of benchmark instances for which an optimal solution was not available. For the C-TOP-TDPLSF, our heuristic is tailored to the specific structure of the score function.

Through in-depth analysis of the obtained solutions, we provide practical recommendations to companies, offering insights into improving operational efficiency and decision-making.

Acknowledgements

First and foremost, I want to thank Michael for giving me the opportunity to work on his team and to write this thesis and for his support over the past five years. Your guidance has been invaluable throughout this journey. Thank you for enduring the corrections of my overly long sentences and never getting tired of my endless comma mistakes.

A heartfelt thanks to Britta for dedicating your time to serve as my second supervisor.

I also thank André, from DHL, for a great collaboration and for providing me with fascinating research topics.

Thanks to my colleagues at the chair, it has been a real pleasure to work with all of you.

Rossana, I could not have imagined a better research partner and mentor on my PhD journey. Together we have experienced all the ups and downs that writing four papers together bring. From countless calls to thousands of messages, your support, feedback, and encouragement mean the world to me.

Stefan, thank you for being my go-to problem solver, no matter how trivial or complex the issue. Thanks for countless procrastination calls, thanks for always listening to me and giving advice. I really hope that we will work together again in the future.

To my family and friends, thank you for your unwavering support, for cheering me on, and for always being there for me. I sincerely apologize to everyone who has been on vacation with me while I brought my laptop: thank you for your understanding and patience. To my parents, thank you for your belief in me, for always having my back and for laying the foundation that made all this possible. To my best friend Vera, words cannot express how much your constant support over the last ten years has meant to me. Without you, I would never have been able to do this.

To everyone who contributed to this journey, whether through advice, encouragement, shared laughter, or simply being there, I am deeply grateful. This thesis would have not been possible without your support.

Contents

Chapter 1

Introduction

In today's rapidly changing global economy, the efficiency of transportation systems is more critical than ever. As industries become increasingly connected, the demand for transporting goods has reached an all-time high. The global trade volume is expected to grow by 2.7% in 2024 and by 3% in 2025 (World Trade Organization, 2024). Solving classical routing problems, such as the traveling salesman problem (TSP) or the vehicle routing problem (VRP) efficiently, has become a core problem for many companies. However, operational constraints, such as time windows or capacity constraints, add complexity to the problems and make finding good solutions a challenging task.

In addition to cost-efficiency, companies must focus on customer satisfaction and employee well-being. Growing customer expectations, such as the demand for timely deliveries, make service quality critical for maintaining competitiveness in the market. At the same time, driver satisfaction plays an important role. High levels of stress, for example caused by driving new routes daily, finding parking in urban areas, or unbalanced workloads, can lead to increased turnover rates. Incorporating constraints such as equal workload distribution into route planning not only improves driver well-being but also enhances long-term operational efficiency.

Another critical aspect is the environmental impact. The transportation sector is responsible for 16% of global greenhouse gas emissions (Ritchie et al., 2020). Typically, in routing problems, the distance is minimized. Therefore, by optimizing routing decisions, companies can significantly reduce fuel consumption and CO2 emissions. This aligns with regulatory frameworks such as the Sustainable Development Goals of the United Nations (United Nations, 2015) and the European Commission's "Fit for 55" package, which aims to reduce EU greenhouse gas emissions by at least 55% by 2030 (European Commission, 2021).

Most routing problems belong to the class of $\mathcal{NP}$-hard problems. This means that exact methods, which guarantee optimal solutions, become computationally expensive as the instance size grows. This is often the case when dealing with real-world instances of routing problems that involve numerous customers and constraints. Consequently, developing efficient solution methods is essential for solving routing problems, especially if they arise at the operational decision level of companies, where new solutions need to be computed every day. This is where heuristics come into play. Unlike exact methods, heuristics explore only parts of the solution space, with the aim of achieving a good trade-off between solution quality and runtime. At their core, many heuristics rely on local search, a technique that iteratively improves a given initial solution by applying incremental changes (moves). The local search terminates as soon as none of the possible moves leads to an improvement, meaning that we have reached a locally optimal solution. To enhance the efficiency of local search, a commonly employed principle is granular search (Toth and Vigo, 2003). Instead of evaluating all possible moves, granular search restricts the search to a smaller subset. This subset is determined using sparsification methods, which seek to identify moves that result in promising solutions (Escobar et al., 2014). Because we only apply small changes to the given initial solution, with local search, we only explore a small area of the solution space close to the initial solution and end up in a local optimum, which can be arbitrarily bad. To overcome this issue, we use metaheuristics such as iterated local search, tabu search, genetic algorithms, and simulated annealing (Gendreau and Potvin, 2019). The metaheuristic most commonly used in this thesis is iterated local search (ILS). In an ILS, we iterate between two phases, the local search phase and the perturbation phase. In the local search phase, we improve a given solution until a local optimum is reached. In the perturbation phase, we modify this local optimum to reach a different area of the solution space. We iterate the two phases until a given stopping criterion is met, which could, for example, be a given number of iterations without improvement or a time limit. Two iterations of the procedure are illustrated in Figure 1.1.

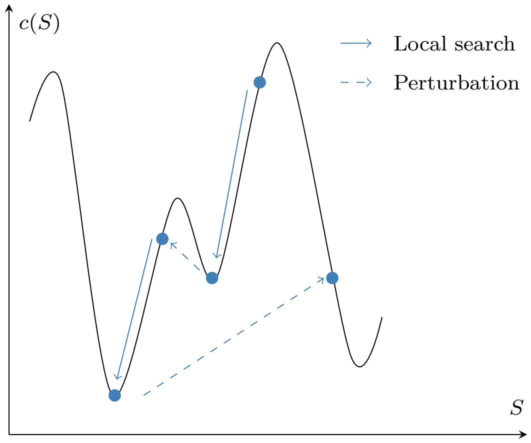


Figure 1.1: Example of two iterations of an iterated local search.

In this thesis, we develop heuristics to efficiently solve four practical routing problems.

1.1 Optimization problems studied in this thesis

In the following section, we present the practical motivation, the problem description, and our contribution for each optimization problem studied in this thesis.

The single truck and trailer routing problem with satellite depots. One example of a transportation system in which route optimization plays a major role is the mail delivery network of DHL Group (DHL). DHL delivers an average of 57 million letters every day, making efficient logistics operations crucial to meeting service quality standards implied by German law (Postgesetz (PostG), 2024) while minimizing costs and environmental impact. The structure of DHL's mail delivery network is illustrated in Figure 1.2. The process begins with the *first mile collection*, in which letters enter the network either through direct collection from customers, retail outlets, or drop-offs at letter boxes and post offices by customers. The letters are then transported to sorting centers, where they undergo an initial sorting and consolidation based on their destinations, a process known as *outbound sorting*. Next, the letters are transported to the destination sorting centers, a stage referred to as *line haul*, for which DHL employs various modes of transport. Upon arrival, the letters are sorted according to the delivery depots, in a process called *inbound sorting* and then brought to the corresponding delivery depot in the *distribution* stage. In the final *last mile delivery* stage, mail carriers deliver the letters directly to the households using either a trolley, bike, or car.

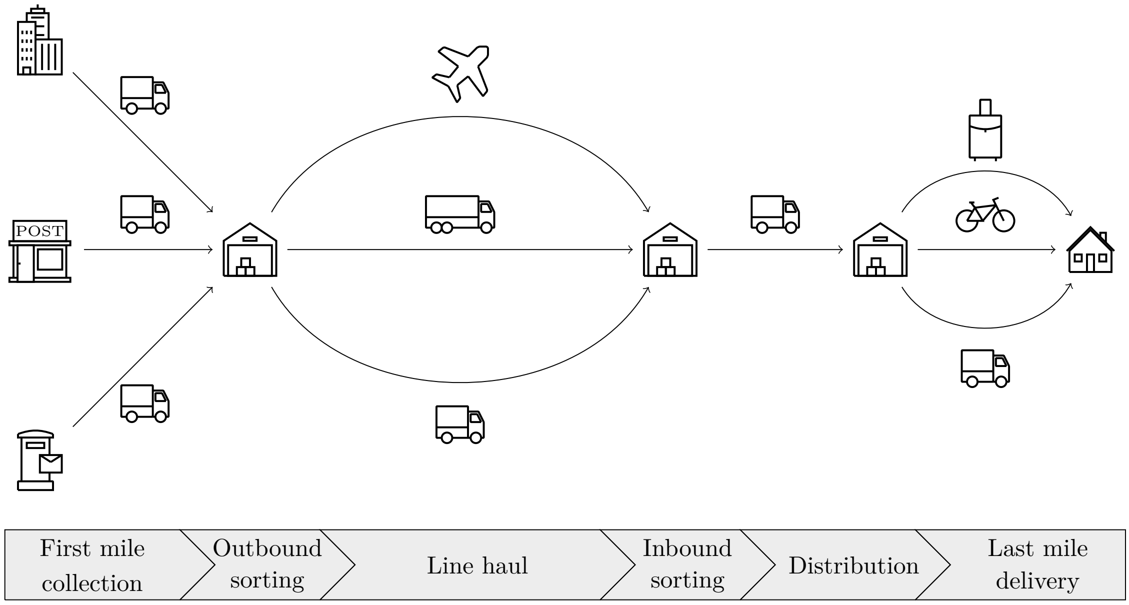


Figure 1.2: Mail delivery network at DHL Group.

In this thesis we focus on the problem of routing mail carriers in the *last mile delivery* stage. The problem of determining a route for one mail carrier serving a given set of households can be modeled as a park and loop problem (PLP, see Bodin and Levy, 2000), in which the mail carrier departs from a delivery depot using a vehicle, that could be a trolley, bike or car, to serve a set of households, that are reachable by foot only. The mail carrier parks the vehicle at designated parking spots, gets off, and loads their bag with letters to serve a group of households. Given the limited capacity of the bag, the mail carrier may need to complete multiple subtours, all originating and ending at the same parking spot. The aim of DHL is (i) to select a set of parking spots and determine the sequence in which they are visited, and (ii) to sequence the household visits from each selected parking spot ensuring that each household is visited exactly once. The objective is to minimize the total travel time across the tours of all mail carriers. This problem corresponds to the single truck and trailer routing problem with satellite depots (STTRPSD). We refer to the tour traveled by the mail carrier using a vehicle as *first-level tour* and to the walking subtours as *second-level tours*.

Existing approaches like Villegas et al. (2010), Accorsi and Vigo (2020) or Arnold and Sörensen (2021) primarily address the symmetric variant of the STTRPSD. However, because DHL operates on a real street network, they solve an asymmetric STTRPSD (ASTTRPSD). Apart from the asymmetry, DHL instances differ in size and parking-spot-to-household ratio from the artificial instances used in the literature. To address these challenges, we propose an ILS, called ILS-ASTTRPSD, designed to solve both the symmetric and asymmetric variant of the STTRPSD.

To build our initial solution, we use an iterative variant of a clustering algorithm inspired by the one proposed by Fischetti et al. (1997) and an improvement phase, both tailored to our problem. We incorporate the natural decomposition of the problem into first-level and second-level tours into the algorithmic design by tackling the first-level tour at the perturbation level, and improving the second-level tours in the local search phase, resulting in decreased computational effort compared to applying the perturbation and the local search phase to the first- and second-level tours simultaneously as done in prior methods. Our neighborhood operators are designed to generate moves on the second-level tours while incorporating first-level tour savings in move evaluations, ensuring the total travel time is minimized. To speed up the search, our ILS employs a vertex-based granular search technique motivated by the structure of instances based on real street networks. In the perturbation phase, we obtain a new parking spot configuration that is possibly infeasible with respect to the second-level tours. To overcome this issue, we propose two transformation heuristics that are applied to the second-level tours.

In our computational experiments, we first compare ILS-ASTTRPSD to the state-of-the-art heuristic by Reed et al. (2024) for a similar problem. ILS-ASTTRPSD outperforms this method and finds new best-known solutions for a large number of instances. Compared to the real-world solutions provided by DHL, our heuristic is able to reduce the total travel time by approximately 2% on average. By analyzing the resulting solutions, we draw insights into efficient mail carrier practices and the influence of parking and loading times in shaping solution structures. Additionally, we derive conditions under which specific optimal solution structures emerge and evaluate the robustness of solutions under parking time fluctuations.

The vehicle routing problem with depot operation constraints. In the STTRPSD we consider only one mail carrier and a given set of households. However, the assignment of households to mail carriers is often not predetermined and must be considered in the route planning. Instead of planning for just one mail carrier, the problem expands to routing multiple mail carriers within a designated delivery area and including the assignment of households to mail carriers into the optimization problem. To simplify the problem, DHL clusters single households into street segments. For each of these street segments a service time is computed, representing the total time required for the mail carrier to serve all households within the street segment, i.e., it includes parking their vehicle, walking to the letter box of the household, and delivering the letters. With this simplification the mail delivery problem can be modeled as a VRP with route duration constraints, in which the mail carriers act as vehicles and the street segments as customers. The objective is to minimize the total travel time.

In addition to delivering the letters, an important task for mail carriers at DHL is to collect the letters of the households assigned to their route from so-called preparation tables located at the depot. To avoid wasting time on finding the correct letters while executing the route, mail carriers already collect the letters in the order they will be delivered. Each preparation table consists of shelves that are divided into sections. Each shelf section is labeled with the names and numbers of the delivery addresses. The delivery addresses are sorted according to their visiting order in their corresponding street segment. An example of such a preparation table is shown in Figure 1.3. The labels cannot be changed on a daily basis, so the order according to which the letters are sorted on the preparation tables is fixed.

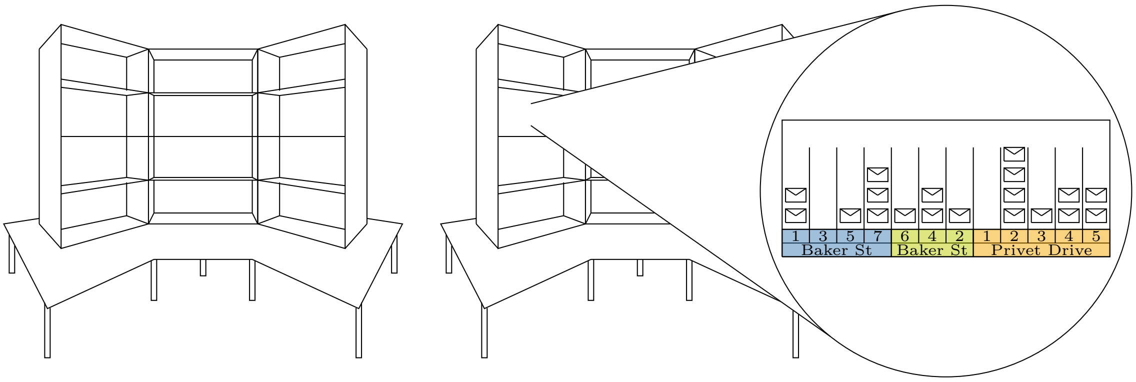


Figure 1.3: Example of a preparation table and detailed view of a shelf.

In practice, a different number of mail carriers might be needed to serve the delivery area due to fluctuating demand, e.g., caused by daily or seasonal variations. Consequently, some street segments must be reassigned, leading to changes in the routing of mail carriers and in their visits to the preparation tables at the depot. Recall that the order according to which the letters are sorted on the preparation tables is fixed and cannot be modified to fit the new routing of mail carriers. Accordingly, the letter collection process may become very complex and generate a lot of overhead at the depot, e.g., long walking distances, switching back and forth between preparation tables, or standing in front of a preparation table together with many mail carriers at the same time. Thus, keeping the letter collection operations as simple as possible is vital for DHL and must be taken into consideration when optimizing the routes of mail carriers. To achieve this, we add so-called depot operation constraints to the routing problem described above and refer to the resulting problem as VRP with depot operation constraints (VRPDOC). The VRPDOC is a novel extension of the traditional VRP that explicitly integrates depot operations into routing decisions.

We propose and compare multiple mathematical model formulations for the VRPDOC and further improve the best-performing formulation by adding problem-specific preprocessing techniques and valid inequalities. Theoretical results prove the $\mathcal{NP}$-completeness of finding a feasible solution to the VRPDOC. We show that after relaxing two specific constraints (upper and lower bounds on the route duration), a feasible solution can be found in polynomial time.

Because commercial solvers cannot even find a feasible solution for real-world instances of the VRPDOC in reasonable runtimes, we propose an algorithm capable of efficiently solving real-world instances that relies on the ILS paradigm. To speed up the search, the local search component in ILS-VRPDOC uses granular search that restricts neighborhood exploration using both traditional sparsification methods (e.g., distance-based) and problem-specific sparsification strategies to select additional arcs.

Computational experiments on large-scale real-world instances provided by DHL demonstrate the effectiveness of our approach. ILS-VRPDOC does not only outperform the DHL solutions reducing total travel time by approximately 6% but also provides significantly simpler solutions with regard to the letter collection operations at the preparation tables, highlighting the operational benefits of considering depot operation constraints. We carry out an extensive analysis of the solutions obtained for the real-world DHL instances, assess the added cost of considering the depot operation constraints to the total travel times, and evaluate the effect of neglecting the depot operation constraints on the resulting complexity of the letter collection operations.

The angular-metric traveling salesman problem. The angular-metric traveling salesman problem (AngleTSP) seeks to identify a Hamiltonian cycle that visits a set of vertices in the Euclidean plane while minimizing the total cost defined by the sum of turning angles. When this cost is a combination of the turning angles and the traveled distance, it is referred to as the angular-distance-metric traveling salesman problem (AngleDistanceTSP). Both problems are variants of the classic TSP. Figure 1.4 illustrates how turning angles are determined.

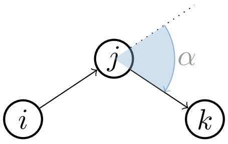


Figure 1.4: Example of a turning angle.

Both problems have practical applications in fields such as robotics and transportation. In transportation, straight movements improve control of heavy vehicles, which is crucial for drivers of machinery like tractors and harvesters operating on uneven terrain, where avoiding sharp turns is essential to prevent accidents (see, e.g., Abubakar et al., 2010). Similarly, semitrailer truck drivers navigating sharp urban corners face risks such as “jackknifing”, where the trailer forms a 90-degree angle with the vehicle, posing danger to surrounding vehicles, pedestrians, and property as reported by the Federal Motor Carrier Safety Administration (2019). Additionally, poor road conditions, bad weather, or improperly secured loads can lead to rollovers, even at low speeds due to the high center of gravity in semitrailer trucks (McKnight and Bahouth, 2009). In all such cases, straight movements are generally preferable to reduce these risks. Consequently, the AngleTSP is highly relevant when planning routes for heavy vehicles.

Prior studies, such as Fischer et al. (2014) or Staněk et al. (2019), proposed several heuristics to solve the problem. However, the best-performing methods are either simple construction heuristics or matheuristics, lacking in solution quality or runtime, respectively. To address the gap in the availability of fast yet effective heuristics, we propose a granular tabu search (GTS) framework, called *GTS-angular*, to address both problem variants. We leverage the geometric properties of good solutions in our construction heuristics and in the sparsification methods used in the local search component of GTS-angular. For the AngleTSP, we refine the convex hull construction heuristic proposed by Staněk et al. (2019). For the AngleDistanceTSP, we introduce a new construction method based on the observation that good solutions rarely include intersecting arcs because they compromise the distance component of the objective function. We strengthen the sparsification method originally proposed by Staněk et al. (2019) of using a lens to limit the size of the neighborhood and introduce two problem-specific sparsification methods which we apply to both the AngleTSP and AngleDistanceTSP. Extensive computational experiments demonstrate the effectiveness of GTS-angular. Compared to the best-performing heuristics developed by Staněk et al. (2019), GTS-angular improves either the solution quality or the runtimes or both, and finds new best-known solutions for around 80% of the benchmark instances for which an optimal solution is not yet available.

The capacitated team orienteering problem with time-dependent and piecewise-linear score functions. The team orienteering problem (TOP) arises in the context of high-quality delivery services, which are essential for customer satisfaction and loyalty. These customer-oriented deliveries provide companies with a competitive edge in today's market. It is particularly relevant in situations where serving all customers is not possible, requiring careful consideration of both collected customer scores and geographic locations.

In the TOP, a fleet of vehicles is available for visiting a set of customers, each associated with a score and a service time. Given a limited tour duration for each vehicle, the decision-maker must determine which customers to visit, which vehicle should serve them, and in what sequence. The objective is to maximize the total score collected by the fleet. Depending on the setting, drivers may be allowed to park their vehicles only for the duration necessary to unload their goods, or may be allowed to park for additional waiting and unloading.

In this thesis, we study the capacitated team orienteering problem with time-dependent and piecewise-linear score functions (C-TOP-TDPLSF). The C-TOP-TDPLSF introduces additional complexity to the standard TOP in three ways: (1) vehicles have limited capacity, (2) each customer has multiple time windows for service, marked as either preferred or less preferred depending on their availability and activity schedules, (3) the score for serving a customer varies based on the time of visit, modeled by a piecewise-linear function. Scores remain constant during preferred time windows but may increase or decrease linearly during less preferred time windows, depending on the customer's availability.

We refer to the C-TOP-TDPLSF in which waiting at a customer is not permitted as the capacitated team orienteering problem with time-dependent and piecewise-linear score functions with no waiting (C-TOP-TDPLSF-nw). To also study the settings in which parking and waiting is allowed, we additionally introduce a variant of the C-TOP-TDPLSF called the capacitated team orienteering problem with time-dependent and piecewise-linear score functions with waiting (C-TOP-TDPLSF-w). In the C-TOP-TDPLSF-w, drivers can park at customer locations and wait for the most advantageous time to begin service. In both, the C-TOP-TDPLSF-nw and the C-TOP-TDPLSF-w, waiting at the depot is allowed.

We propose a mathematical formulation for each variant, supplemented with problem-specific preprocessing techniques and valid inequalities. Because of the limitations of commercial optimization solvers in handling real-world instances of the C-TOP-TDPLSF, we propose two heuristics based on ILS: ILS-noWait for solving the C-TOP-TDPLSF-nw and ILS-cWait-fin for solving the C-TOP-TDPLSF-w. These heuristics differ in the evaluation of waiting decisions. In ILS-noWait, moves that violate time window constraints are rejected. Because the decision about waiting at customers is continuous and thus computationally expensive, ILS-cWait-fin only considers a prespecified set of waiting strategies according to which waiting at a customer is only allowed for a prespecified amount of time. A post-processing finalization phase is applied to the best found solutions of the ILS to optimize customer visiting times while maintaining the sequence of visits.

Computational experiments on small-scale instances show that for C-TOP-TDPLSF-nw instances, ILS-noWait outperforms a commercial optimization solver. However, for some C-TOP-TDPLSF-w instances, ILS-cWait-fin struggles to match the quality of the solutions found with the commercial optimization solver. Although the finalization phase leads to higher runtimes, it is crucial for improving the solutions for the C-TOP-TDPLSF-w. The analysis of solutions of large-scale instances of both problem variants provides valuable managerial insights into the impact of the widths of time windows, vehicle number, fleet size, and of allowing waiting at customer locations.

1.2 Organization

This thesis consists of four articles, introducing several heuristics designed to support decision-making for the optimization problems described above. Our work on the single truck and trailer routing problem with satellite depots (STTRPSD) is presented in Chapter 2. In Chapter 3, we introduce the vehicle routing problem with depot operation constraints (VRPDOC). We examine the angular traveling salesman problem (AngleTSP) and the angular distance traveling salesman problem (AngleDistanceTSP) in Chapter 4. In Chapter 5, our work on the capacitated team orienteering problem with time-dependent and piecewise-linear score functions (C-TOP-TDPLSF) is discussed. Finally, in Chapter 6, we conclude and provide an outlook to further research topics.

Since the four articles emerged from collaborative work, there is a statement at the beginning of each chapter, clarifying my contribution.

References

M. S. Abubakar, D. Ahmad, and F. B. Akande (2010). “A review of farm tractor overturning accidents and safety”. In: *Pertanika Journal of Science and Technology* 18.2, pp. 377–385.

L. Accorsi and D. Vigo (2020). “A hybrid metaheuristic for single truck and trailer routing problems”. In: *Transportation Science* 54.5, pp. 1351–1371. DOI: 10.1287/trsc.2019.0943.

F. Arnold and K. Sörensen (2021). “A progressive filtering heuristic for the location-routing problem and variants”. In: *Computers & Operations Research* 129, pp. 105–166. DOI: 10.1016/j.cor.2020.105166.

L. Bodin and L. Levy (2000). “Scheduling of local delivery carrier routes for the united states postal service”. In: *Arc Routing*. Springer, pp. 419–442. DOI: 10.1007/978-1-4615-4495-1_11.

J. W. Escobar, R. Linfati, and P. Toth (2014). “A hybrid granular tabu search algorithm for the multi-depot vehicle routing problem”. In: *Journal of the Operational Research Society* 65.1, pp. 37–48. DOI: 10.1057/jors.2013.102.

European Commission (2021). *Fit for 55: Delivering the EU’s 2030 Climate Target on the Way to Climate Neutrality*. Accessed: 2024-11-24. URL: https://commission.europa.eu/strategy-and-policy/priorities-2019-2024/european-green-deal/delivering-european-green-deal/fit-55-delivering-proposals_en.

U. D. o. T. Federal Motor Carrier Safety Administration (2019). *Large truck and bus crash facts 2017*. URL: https://www.fmcsa.dot.gov/sites/fmcsa.dot.gov/files/docs/safety/data-and-statistics/461861/ltcbf-2017-final-5-6-2019.pdf.

A. Fischer, F. Fischer, G. Jäger, J. Keilwagen, P. Molitor, and I. Grosse (2014). “Exact algorithms and heuristics for the quadratic traveling salesman problem with an application in bioinformatics”. In: *Discrete Applied Mathematics* 166, pp. 97–114. DOI: 10.1016/j.dam.2013.09.011.

M. Fischetti, J. J. Salazar González, and P. Toth (1997). “A branch-and-cut algorithm for the symmetric generalized traveling salesman problem”. In: *Operations Research* 45.3, pp. 378–394. DOI: 10.1287/opre.45.3.378.

M. Gendreau and J.-Y. Potvin (2019). *Handbook of Metaheuristics*. 2nd. Springer. DOI: 10.1007/978-3-319-91086-4.

A. J. McKnight and G. T. Bahouth (2009). “Analysis of large truck rollover crashes”. In: *Traffic injury prevention* 10.5, pp. 421–426. DOI: 10.1080/15389580903135291.

Postgesetz (PostG) (2024). *Postgesetz vom 15. Juli 2024 (BGBl. 2024 I Nr. 236)*. Accessed: 2024-11-26. URL: https://www.bundesnetzagentur.de/SharedDocs/Downloads/DE/Sachgebiete/Post/Unternehmen_Institutionen/Anbieterverzeichnis/PostG.pdf?__blob=publicationFile&v=3.

S. Reed, A. M. Campbell, and B. W. Thomas (2024). *Does Parking Matter? The Impact of Search Time for Parking on Last-Mile Delivery Optimization.* Tech. rep. DOI: 10.1016/j.tre.2023.103391.

H. Ritchie, M. Roser, and P. Rosado (2020). "CO2 and Greenhouse Gas Emissions". In: *Our World in Data.* Accessed: 2024-11-24. URL: https://ourworldindata.org/co2-and-other-greenhouse-gas-emissions.

R. Staněk, P. Greistorfer, K. Ladner, and U. Pferschy (2019). "Geometric and LP-based heuristics for angular travelling salesman problems in the plane". In: *Computers & Operations Research* 108, pp. 97–111. DOI: 10.1016/j.cor.2019.01.016.

P. Toth and D. Vigo (2003). "The granular tabu search and its application to the vehicle-routing problem". In: *INFORMS Journal on Computing* 15.4, pp. 333–346. DOI: 10.1287/ijoc.15.4.333.24890.

United Nations (2015). *Sustainable Development Goals.* Accessed: 2024-11-26. URL: https://unric.org/en/united-nations-sustainable-development-goals/.

J. G. Villegas, C. Prins, C. Prodhon, A. L. Medaglia, and N. Velasco (2010). "GRASP/VND and multi-start evolutionary local search for the single truck and trailer routing problem with satellite depots". In: *Engineering Applications of Artificial Intelligence* 23.5, pp. 780–794. DOI: 10.1016/j.engappai.2010.01.013.

W. World Trade Organization (2024). *World Trade Outlook and Statistics 2024.* Accessed: 2024-11-26. URL: https://www.wto.org/english/res_e/booksp_e/stat_10oct24_e.pdf.

Chapter 2

The asymmetric single truck and trailer routing problem with satellite depots

Publication status: R. Cavagnini, M. Schneider, and A. Theiß (2024a). “A granular iterated local search for the asymmetric single truck and trailer routing problem with satellite depots at DHL Group”. In: *Networks* 83.1, pp. 3–29. DOI: 10.1002/net.22178

Abstract: To plan the postal deliveries of our industry partner DHL Group (DHL), the single truck and trailer routing problem with satellite depots (STTRPSD) is solved to optimize mail carriers routes. In this application context, instances feature a high number of customers and satellites, and they are based on real street networks. This motivates the study of the asymmetric STTRPSD (ASTTRPSD). The heuristic solution methods proposed in the literature for the STTRPSD can either solve only the symmetric problem variant, or it is unclear whether they can also be used to solve the ASTTRPSD. We introduce an iterated local search, called ILS-ASTTRPSD, which generates different first-level tours in the perturbation phase, and improves the second-level tours in the local search phase. To speed up the search, granular neighborhoods are used. The computational results on instances from the literature prove the capability of ILS-ASTTRPSD to return high-quality solutions. On DHL instances, ILS-ASTTRPSD significantly decreases total travel times of the mail carriers and returns solutions with a different structure compared to the ones provided by DHL. Based on these differences, we give recommendations on how DHL could design more efficient mail carrier practices. Dedicated computational experiments reveal that considering parking and loading times when solving the ASTTRPSD leads to lower travel times, and that ignoring parking times is more counterproductive than ignoring loading times. Moreover, we assess the robustness of our solutions under parking time fluctuations. Finally, we derive properties of instances for which optimal solutions contain multiple second-level tours rooted at the same parking spot and for which the optimal solutions of the ASTTRPSD correspond to the ones of a pure traveling salesman problem.

Contribution of the author: The authors shared efforts in the conceptual development of the research goals, the literature review, the design of the methodology and implementation of the algorithm, the computational experiments and result analysis, and in writing the paper.

2.1 Introduction

Postal routing activities are of major importance for companies like our industry partner DHL Group (DHL). DHL has to serve 54 200 districts on six days a week and has to deliver an average of 57 million letters every day. Therefore, decision support to reduce the travel time of the individual mail carriers is a strong lever to reduce total operating costs.

DHL's postal delivery tours are executed daily, but they are planned at the tactical level, i.e., they remain unchanged for several months. DHL executes the delivery tours using two different means of transportation: a vehicle (car, bike, or carrier) is combined with a mail carrier that walks to carry out the final delivery. This can be modeled as a park and loop problem (PLP, see Bodin and Levy, 2000), in which a mail carrier with a vehicle departs from a main depot to serve the demand of a set of households, which are reachable only by walking (without the vehicle). There is a set of given parking spots, where the mail carrier can park the vehicle and get off. Parking requires a specified amount of time that is independent of the demand to be served from that parking spot. Then, the mail carrier loads their bag with letters to serve a group of households. DHL sorts letters at the depot according to the sequence in which letters will be delivered using dedicated sorting machines. Before leaving the depot, the mail carrier has already packed them into blocks and does not need time to select letters one by one after parking the vehicle. Although letters are already packed, according to DHL, the loading time cannot be considered negligible because the mail carrier still needs to open the vehicle trunk, pick up the correct block of letters, and close the vehicle trunk. Thus, in our problem, the loading time does not depend on the number of letters delivered in a walking tour, but it is considered fixed and non-negligible for each walking tour. The bag has a limited capacity, but the mail carrier can walk one or multiple subtours, all starting and ending at the same parking spot. We refer to the tour traveled by the mail carrier using a vehicle as *first-level tour* and to the walking subtours as *second-level tours*. The aim of DHL is (i) to select a set of parking spots and determine their visiting sequence (first-level tour), and (ii) to choose the household visiting sequence from each selected parking spot (second-level tours) in such a way that each household is visited exactly once. The objective is to minimize the total travel time of all tours of the mail carrier.

The specific PLP that DHL has to solve corresponds to the single truck and trailer routing problem with satellite depots (STTRPSD). Because DHL executes deliveries on a real street network, the presence of one way streets, crossroads, dead-end streets, turning restrictions, and medial strips between lanes makes travel times neither Euclidean nor symmetric. Consequently, DHL has to solve the asymmetric STTRPSD (ASTTRPSD).

Several metaheuristics exist for the symmetric STTRPSD (see Villegas et al., 2010; Accorsi and Vigo, 2020; Arnold and Sörensen, 2021). However, neither the heuristic of Accorsi and Vigo, 2020 nor the one of Arnold and Sörensen, 2021 can solve the asymmetric problem variant. For the method of Villegas et al., 2010, the implementation is not publicly available, and it is not clear whether it can also solve the ASTTRPSD (for more details, see Section 2.2). We contribute by proposing a more general metaheuristic that can be applied to solve both the symmetric and asymmetric STTRPSD variant and that provides our industry partner with an effective solution approach to solve real-world instances. Moreover, apart from their asymmetry, DHL instances differ from the artificial instances used in the literature in two additional aspects. First, while the instances in the literature have at most 20 parking spots and 200 households, the DHL instances are larger with an average of 724 parking spots and 390 households. Second, all DHL instances have more parking spots than households. In each instance, the number of parking spots is approximately double the number of households, representing the possibility, for almost all households, of parking on both sides of the street. An extract of a DHL instance is presented in Figure 2.1. The street network is shown in gray, the households in blue, and the parking spots in magenta.

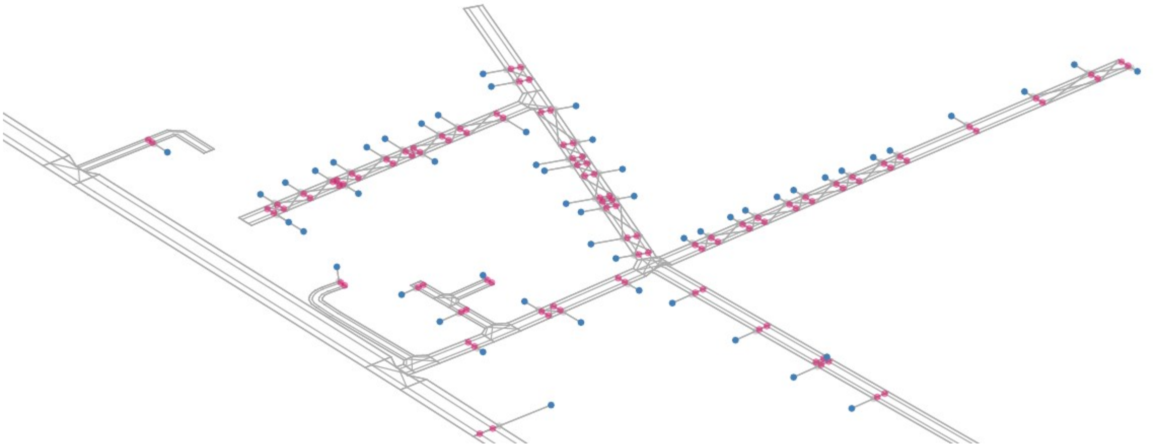


Figure 2.1: Extract of an exemplary DHL instance.

In this paper, we propose an iterated local search (ILS) for the ASTTRPSD, called ILS-ASTTRPSD. The structure of the problem allows us to naturally decompose the problem into a first-level tour which is tackled at the perturbation level, and second-level tours which are improved by the local search phase. This decomposition decreases the computational effort compared to applying the perturbation and the local search phase to the first- and second-level tours simultaneously as done in Villegas et al., 2010 and Accorsi and Vigo, 2020. Our neighborhood operators are specifically designed to generate moves on the second-level tours. However, if we only evaluate these moves based on changes in the second-level tour travel time, we may discard moves that improve the total travel time. For example, if a parking spot is closed, the second-level tour time may increase less than the reduction of the first-level tour time, leading to an overall travel time improvement. To address this issue, the move evaluation considers the travel time savings in the first-level tour that result from closing a parking spot. To speed up the search, our ILS uses granular search (see Toth and Vigo, 2003). This principle is based on a sparsification method which is used to restrict the size of the neighborhoods to explore. Different from the sparsification used in Accorsi and Vigo, 2020 and motivated by the structure of instances based on real street networks, we implement a vertex-based sparsification method. To make the second-level tours compatible with the new parking spot configuration obtained after applying the perturbation move, we also propose two transformation heuristics to be run depending on the selected move. Finally, ILS-ASTTRPSD runs on a starting solution obtained with a clustering algorithm inspired by the one proposed by Fischetti et al., 1997. However, we derive an iterated version of this heuristic and an improvement phase, both tailored to our problem.

In our computational experiments, to test the ILS-ASTTRPSD capability of returning high-quality solutions, we first compare ILS-ASTTRPSD solutions to the state-of-the-art heuristic by Reed et al., 2024 for a similar problem. ILS-ASTTRPSD outperforms this method and finds new best-known solutions for a large number of instances. On DHL instances, ILS-ASTTRPSD significantly improves the total travel times with respect to the solutions provided by DHL. By comparing the structure of ILS-ASTTRPSD and the DHL solutions, we draw insights on more efficient mail carrier practices. For completeness, we also use ILS-ASTTRPSD to solve symmetric instances from the literature. The results show that the solution quality is reasonable compared to the specialized methods from the literature.

Moreover, we derive instance conditions under which: (i) multiple second-level tours from the same parking spot may exist in an optimal solution, and (ii) the optimal solution corresponds to the one of a TSP with one dedicated parking spot for each household. We contribute with managerial insights on the influence of parking and loading times on the solution structure. Results show that considering both parking and loading times leads to shorter travel times and that ignoring parking times returns more costly solutions than ignoring loading times. Finally, we assess the robustness of solutions under parking time fluctuations.

To summarize, our contributions are the following:

- We propose a metaheuristic that solves both the symmetric and asymmetric STTRPSD and that contains new features compared to the methods proposed in the literature. Our metaheuristic solves real-world instances effectively and outperforms the heuristic of Reed et al., 2024 on asymmetric instances from the literature.

- We derive properties of instances that lead to a particular structure in optimal solutions.

- We derive insights on efficient mail carrier practices, on the influence of parking and loading times on the solution structure, and on the robustness of solutions under parking time fluctuations.

The paper is organized as follows. In Section 2.2, we review the literature on the STTRPSD and related problems. Section 2.3 provides a formal description of the problem and introduces the used notation. Section 2.4 derives instance properties that characterize special structures of optimal solutions, and, Section 2.5 describes our ILS-ASTTRPSD. We present the computational experiments and results in Section 2.6. Finally, Section 2.7 concludes the paper.

2.2 Literature review

In this section, we review the literature on the STTRPSD and closely-related problems. The STTRPSD belongs to the class of the truck and trailer routing problems (TTRP), which are discussed in several surveys (Nagy and Salhi, 2007; Drexl, 2012; Prodhon and Prins, 2014; Cuda et al., 2015; Schiffer et al., 2019).

The STTRPSD was first introduced in Villegas et al., 2010. The authors propose a mathematical formulation, a set of symmetric instances, and multiple heuristic methods to solve the problem. The best performing method is a multi-start ILS. However, the authors only report experiments for symmetric instances, and it remains unclear whether their heuristic can also solve asymmetric ones. Belenguer et al., 2016 propose a branch-and-cut algorithm that uses several families of valid inequalities and exact and heuristic separation procedures. Their exact method solves instances with up to 50 households and 10 parking spots to optimality. While the valid inequalities in Belenguer et al., 2016 are specifically tailored to the STTRPSD, Bartolini and Schneider, 2020 solve instances of the STTRPSD of the same size via a branch-and-cut algorithm with valid inequalities developed for the capacitated TTRP. Accorsi and Vigo, 2020 develop the state-of-the-art metaheuristic for the STTRPSD. Their approach is based on an ILS applied within a multistart framework, and the authors store high-quality first- and second-level tours in a solution pool during the execution of the heuristic. The ILS is composed of a perturbation phase based on a ruin and recreate mechanism and a local search component, namely, a randomized variable neighborhood descent (VND) with first improvement. Every time the perturbation phase is called, either a subset of parking spots is randomly selected and closed, or random households are removed from every second-level route until their load is less than a specified threshold. The solution is then rebuilt by selecting households according to different criteria and inserting them in the position which minimizes the insertion cost. At the end, a set-partitioning problem is solved with the tours saved in the solution pool. The algorithm of Accorsi and Vigo, 2020 has been specifically designed for symmetric instances and cannot be used to solve asymmetric instances.

For example, in their algorithm, the authors use, among others, the 2-opt neighborhood operator in intra-route fashion, which causes problems when dealing with asymmetric instances. Arnold and Sörensen, 2021 study the STTRPSD as a variant of location-routing problems and propose a progressive filtering heuristic capable of obtaining solutions competitive to those of Villegas et al., 2010. Arnold and Sörensen, 2021 use the k-means clustering algorithm implementation of the SciKit-learn library, which assumes symmetric instances, they always use the term "edge", and all experiments are based on symmetric instance sets, indicating that their heuristic cannot be used to solve asymmetric instances.

A problem similar to the STTRPSD is the capacitated delivery problem with parking (CDPP) introduced by Reed et al., 2024. The authors explicitly model the time required for a delivery man to search for a parking spot and identify the conditions under which considering the parking time changes the solution structure. Their problem is different from ours in the following aspects. First, they assume that the parking and household locations coincide. Consequently, whenever the parking search time is set to zero, the optimal solution corresponds to that of a TSP, in which every household is served from the parking spot at the same location. In the ASTTRPSD, when no parking times are considered, the optimal solution may differ from the TSP solution because driving on a street network is not always faster than walking, and a household may have more than one associated parking spot. Second, in their paper, the loading time depends on the number of packages to be delivered. Because the sum of all households' demand must be satisfied, the loading time is a constant that can be excluded from the optimization. In the ASTTRPSD, because the mail carrier can directly take the letters in blocks, the loading time is not dependent on the number of letters but fixed per second-level tour. Consequently, the loading time must be included in the optimization because it may influence the number of second-level tours in the solution. Reed et al., 2024 propose a heuristic to address their problem. The computational results show gaps to the optimal solution of at most 5.5% for instances with up to 100 households.

The STTRPSD is also related to PLPs. The practice of combining walking and driving is introduced by Levy and Bodin, 1989 in a location arc routing problem for postal deliveries and formalized as "park-and-loop" problem by Bodin and Levy, 2000. For model formulations of the PLPs proposed by Bodin and Levy, 2000, we refer to Bode, 2013. Additional variants of PLPs are the multi-modal PLP for postal deliveries (Gussmagg-Pfliegl et al., 2011), the doubly open park-and-loop routing problem (Cabrera et al., 2022), and the park-and-loop routing problem with parking selection (Le Colleter et al., 2023).

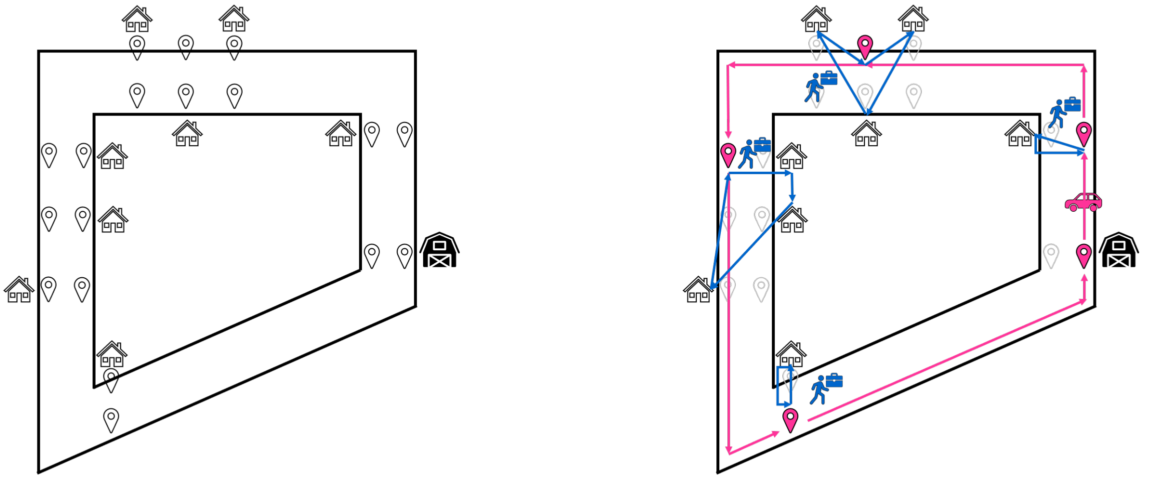
The two-echelon structure of the STTRPSD also relates it to the two-echelon driving and walking distribution problem for last-mile delivery studied in Martinez-Sykora et al., 2020. The authors propose a mathematical formulation of the problem, derive valid inequalities, and solve instances with up to 30 vertices via branch-and-cut. However, different from our problem, the authors do not allow multiple second-level tours starting from the same parking spot.

2.3 Problem description

In this section, we formally describe the ASTTRPSD and introduce all relevant notation. Figure 2.2 shows a small example of the ASTTRPSD.

The ASTTRPSD can be defined on a directed graph $G = (V, A)$, where V is the vertex set and A is the arc set. The vertex set is partitioned into $V = \{0\} \cup V_D \cup V_C$, where 0 is the depot, V_D is the set of parking spots, and V_C is the set of households. Figure 2.2(a) represents the households along the street, the depot in the lower right part of the figure, and the parking spots associated to each household. A nonnegative travel time c_{ij} is associated with each arc $(i, j) \in A$. For walking arcs, i.e., $\{(i,j)|(i \in V_C \land j \in V_C) \lor (i \in V_C \land j \in V_D) \lor (i \in V_D \land j \in V_C)\}$, and driving arcs, i.e., $\{(i,j)|(i \in \{0\} \cup V_D \land j \in \{0\} \cup V_D)\}$, the travel times satisfy the triangle inequality. However, this is not true in general because driving between two locations i and j may be slower than walking from i to k and from k to j. This happens, for example, when two households are located on the opposite sides of a street and crossing the street by walking is possible, but U-turns by vehicles are not allowed. Each household $i \in V_C$ has a nonnegative demand q_i. The capacity of the vehicle is denoted by Q_v, the capacity of the mail carrier bag by Q_b. To guarantee feasibility, we assume that the capacity of the vehicle at least corresponds to the cumulative demand of the households, i.e., $Q_v \geq \sum_{i \in V_C} q_i$. Moreover, we assume that split deliveries are not allowed and that the capacity of the mail carrier bag is at least as large as the maximum household demand, i.e., $Q_b \geq max_{i \in V_C}\{q_i\}$.

A solution S of the STTRPSD consists of (i) a first-level tour t^1, i.e., a cycle starting from the depot, visiting a subset of parking spots denoted by $V_D(t^1)$ and returning to the depot, and (ii) a set of second-level tours T^2, in which each element of this set starts at a parking spot $k \in V_D(t^1)$ contained in the first-level tour, visits one or more households in V_C, and ends at the same parking spot k. In Figure 2.2(b), the selected parking spots and the first-level tour traveled by car are highlighted in magenta, while the second-level tours walked by the mail carrier with a capacitated bag are colored in blue. Multiple second-level tours can be rooted at the same parking spot. The set of households visited in a second-level tour $t^2 \in T^2$ is represented by $V_C(t^2)$. The travel time of the first-level tour is represented by $c(t^1)$, the travel time of a second-level tour is denoted by $c(t^2)$. A solution is feasible if the cumulative demand of the households visited in every second-level tour does not exceed the bag capacity, i.e., $\sum_{i \in V_C(t^2)} q_i \leq Q_b$ for all $t^2 \in T^2$, and each household is visited exactly once. A feasible solution is optimal if it minimizes the sum of the total travel time of the first and the second-level tours $c(t^1) + \sum_{t^2 \in T^2} c(t^2)$. The time required to park the vehicle at a parking spot k is denoted by ρ_k and can be easily included at the instance level by adding the parking time to the travel time of every arc entering the parking spot k from another parking spot, i.e., to the arcs $\{(i,k) | i \in V_D, i \neq k\}$. If the parking time is assumed to be the same for all parking spots, we simply denote it by ρ. Similarly, the time required to load the bag before starting a second-level tour at a parking spot k is denoted by ℓ. When the loading time is assumed to be linearly dependent on the number of letters to be delivered in the second-level tour, loading times are a constant, and they can be excluded from the optimization.



(a) Example of an ASTTRPSD instance

(b) Example of an ASTTRPSD solution

Figure 2.2: A small ASTTRPSD example.

2.4 Conditions for particular optimal solution structures

In this section, we state properties of instances that lead to a particular optimal solution structure. We believe that these properties are useful for the following reasons. First, they allow to gain insights on the structure of optimal solutions by just analyzing the values of the parameters in the instances, i.e., without the need for solving them. As an example, these properties could be considered as features for a machine learning algorithm with the goal of predicting whether a solution is optimal or not. Second, by recognizing that specific properties of instances are fulfilled, tailored solution methods could be implemented. Third, these properties will allow us to verify if the structure of the heuristic solutions obtained by ILS-ASTTRPSD is in line with the structure of optimal solutions.

The first property refers to the presence of multiple second-level tours from the same parking spot. Specifically, there can be multiple second-level tours rooted at the same parking spot only if (i) the cumulative demand of the visited households from the same parking spot exceeds the bag capacity, and (ii) it is more convenient to serve those households from that parking spot than from a different one. This first property can be formalized as follows.

Theorem 2.4.1 (Presence of multiple second-level tours from the same parking spot) In an optimal solution of the ASTTRPSD, multiple second-level tours from the same parking spot may exist only if there exists at least one pair of parking spot vertices k_1, k_2, with $k_1 \neq k_2$, a household vertex h_1, and a subset of household vertices H, with $|H| \geq 1$, such that the following two conditions are both verified:

$C1.$ $q_{h_1} + \sum_{i \in H} q_i > Q_b$

and

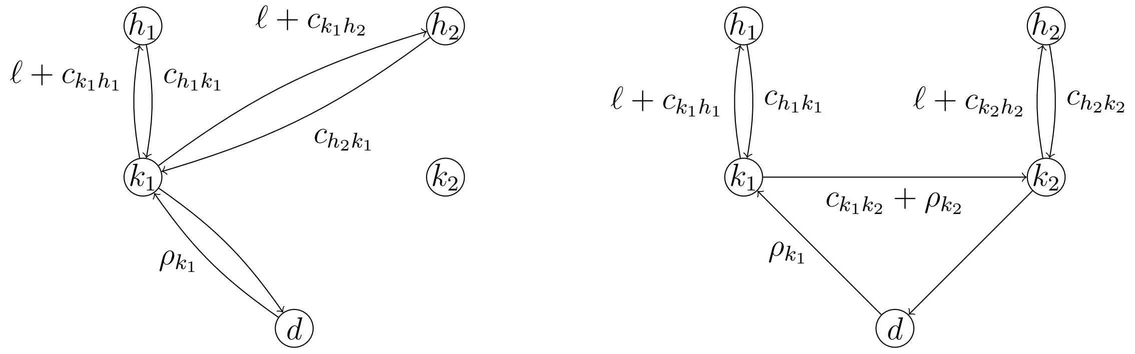
$C2.$ $\rho_{k_1} + \ell + c_{k_1 h_1} + c_{h_1 k_1} + \ell + c_{k_1 H_0} + c_{H_{|H|} k_1}$

$$\leq \rho_{k_1} + \ell + c_{k_1 h_1} + c_{h_1 k_1} + c_{k_1 k_2} + \rho_{k_2} + \ell + c_{k_2 H_0} + c_{H_{|H|} k_2},$$

that can be reduced to:

$$c_{k_1 H_0} + c_{H_{|H|} k_1} \leq c_{k_1 k_2} + \rho_{k_2} + c_{k_2 H_0} + c_{H_{|H|} k_2}.$$

While condition $C1$ is straightforward, Figure 2.3 shows an example of condition $C2$ for two parking spots k_1 and k_2, a household h_1, and, without loss of generality, a subset of household vertices $H = \{h_2\}$. Figure 2.3(a) represents the left-hand side of condition $C2$ in which multiple second-level tours are rooted at the same parking spot, while Figure 2.3(b) shows the right-hand side of condition $C2$ in which each household is served from a different parking spot.



(a) Left-hand side of condition C2.

(b) Right-hand side of condition C2.

Figure 2.3: Example of left-hand and right-hand side of condition C2.

We note that, if the cumulative demand of the visited households from the same parking spot does not exceed the bag capacity, there cannot be multiple second-level tours rooted at that parking spot because of the triangle inequality holding for the walking network.

The second property states the condition under which an optimal solution of the ASTTRPSD corresponds to the one of a TSP in which one parking spot is opened for each household. This situation occurs only if serving a household in a second-level tour already visiting another household is never convenient. This property can be formalized as follows.

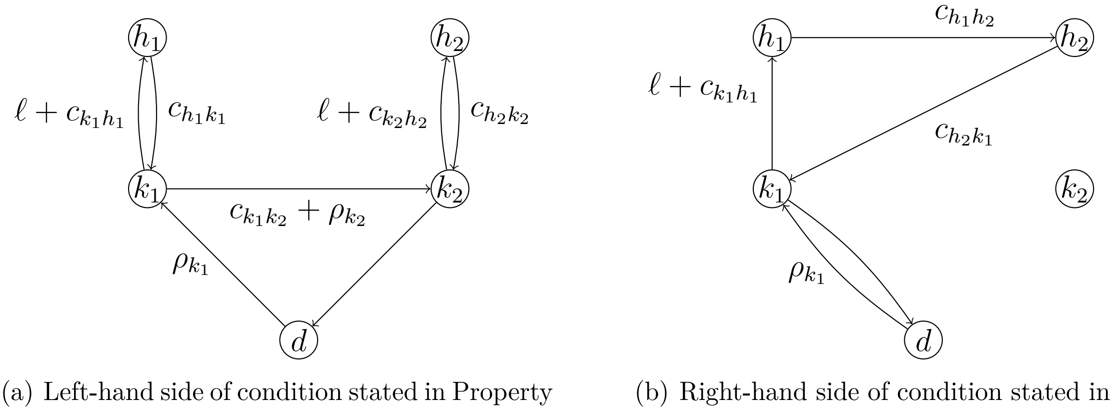
Theorem 2.4.2 (The solution of the ASTTRPSD corresponds to a TSP solution) The optimal solution of the ASTTRPSD corresponds to the one of a TSP in which one parking spot for each household is opened and each household is visited from that parking spot, only if for every pair of parking spot vertices k_1, k_2, with $k_1 \neq k_2$, and of household vertices h_1, h_2, the following condition is verified:

$$\rho_{k_1} + \ell + c_{k_1h_1} + c_{h_1k_1} + c_{k_1k_2} + \rho_{k_2} + \ell + c_{k_2h_2} + c_{h_2k_2} < \rho_{k_1} + \ell + c_{k_1h_1} + c_{h_1h_2} + c_{h_2k_1}$$

that can be reduced to:

$$c_{h_1k_1} + c_{k_1k_2} + \rho_{k_2} + \ell + c_{k_2h_2} + c_{h_2k_2} < c_{h_1h_2} + c_{h_2k_1}.$$

Figure 2.4 shows an example of the condition for two parking spots k_1 and k_2 and two households h_1 and h_2. Figure 2.4(a) represents the left-hand side of the condition in which each household is served from a dedicated parking spot, while Figure 2.4(b) shows the right-hand side of the condition in which multiple households are served in the same second-level tour.



(a) Left-hand side of condition stated in Property 4.2.

(b) Right-hand side of condition stated in Property 4.2.

Figure 2.4: Example of left-hand and right-hand side of the condition stated in Property 4.2.

The solution of such a TSP in which one parking spot for each household is opened and each household is visited from that parking spot is always a feasible solution for the ASTTRPSD. Hence, it provides an upper bound for the optimal objective function value of the ASTTRPSD solution.

2.5 Granular iterated local search

In this section, we introduce the ILS-ASTTRPSD algorithm. The pseudocode is given in Algorithm 1.

After obtaining a starting solution using the two-phase construction heuristic explained in Section 2.5.1, the ILS described in Section 2.5.2 is executed. In each ILS iteration, a local search phase based on a VND with first improvement (Section 2.5.2.1) is applied. Whenever a better solution than the incumbent best is found, the best found solution is updated. The VND terminates when the total travel time of the solution stops improving. Next, the perturbation phase is applied to the best found solution (Section 2.5.2.2) to generate a new solution having different opened parking spots and second-level tours adapted for this new configuration. The ILS terminates after η iterations without improvement.

Algorithm 1: Pseudocode of ILS-ASTTRPSD algorithm

```
$S \leftarrow constructionHeuristic()$
$S^* \leftarrow S$
while $iterations\ without\ improvement < \eta$ do
    while $improvementVND$ do
        $S \leftarrow VND(S)$
        if $c(S) < c(S^*)$ then
            $S^* \leftarrow S$
            $c(S^*) \leftarrow S$
        end
    end
    $S \leftarrow perturbation(S^*)$
end
```

2.5.1 Construction heuristic

Our construction heuristic consists of two phases: an iterated clustering algorithm (Section 2.5.1.1) used to obtain a starting solution and an improvement phase (Section 2.5.1.2) used to optimize the selection of opened parking spots.

2.5.1.1 Iterated clustering algorithm

The iterated clustering algorithm is inspired by the clustering algorithm proposed by Fischetti et al., 1997. In their paper, the authors use this algorithm to generate instances of the group TSP from the instances of the classical TSP contained in the TSPLIB library (Reinelt, 1991). While Fischetti et al., 1997 apply the algorithm to create clusters using vertices from the same vertex set, we use the set of parking spots to select the cluster centroids and the set of households to form clusters. The algorithm takes as input a parameter K that specifies the number of clusters (i.e., open parking spots) to form. Then, it selects K centers (i.e., parking spots) by considering the K vertices located as far as possible from each other. Finally, it assigns each vertex (i.e., household) to its nearest center. The advantage of this algorithm is that it produces well-distributed opened parking spots in the street network. However, it requires to specify a value for K as input, and the optimal value of K is unknown.

To overcome this problem, we propose a new iterated version of this clustering algorithm that is summarized in Algorithm 2. We initialize the number of parking spots to open to the lower bound value of the number of required second-level tours to satisfy the total household demand. Then, we execute the clustering algorithm of Fischetti et al., 1997 for increasing values of K (in steps of one). Once the parking spots to open and the assignment of households to parking spots have been determined, we solve a TSP on the opened parking spots to build the first-level tour t^1. For each parking spot, we build a big temporary second-level tour, in which all the assigned households to that parking spot appear according to their lexicographic order. Next, we check for each parking spot if the sum of the demands of the households assigned to that parking spot exceeds the bag capacity of the mail carrier. If the bag capacity is not exceeded, we leave the second-level tour as it is. Otherwise, by keeping the households in lexicographic order, we cut the big temporary second-level tour into smaller second-level tours as soon as adding the demand of the next household would exceed the bag capacity. Instead of the lexicographic order, a greedy heuristic based on the household demand could be used to reallocate the households to get a feasible solution. However, we use the lexicographic order to keep the runtimes of the algorithm as low as possible because the quality of the solution is guaranteed by the subsequent ILS. Then, we solve a TSP for every second-level tour in T^2. The algorithm terminates after a given number of iterations without improvement.

Algorithm 2: Pseudocode of the iterated clustering algorithm

```
initialize $S = \emptyset,\ c(S) = +\infty, \gamma = 0,\ K = \lceil \sum_{i \in V_C} q_i / Q_b \rceil$
while $\gamma < \gamma_{max}$ do
    $clusteringAlgorithm(K)$
    $S' \leftarrow solveTSP(t^1)$
    $S' \leftarrow solveTSP(t^2)\ \forall t^2 \in T^2$
    $K \mathrel{+}= 1$
    if $c(S') < c(S)$ then
        $\gamma = 0$
        $S \leftarrow S',\ c(S) \leftarrow c(S')$
    else
        $\gamma \mathrel{+}= 1$
    end
end
```

2.5.1.2 Construction heuristic improvement phase

In the improvement phase, we optimize the choice of the opened parking spots by keeping the households visited in each second-level tour $V_C(t^2)$, $t^2 \in T^2$ unchanged and the visiting sequence of the second-level tours fixed.

First, we build the weighted layered graph shown in Figure 2.5. We start by appending a vertex representing the depot 0 at the beginning and at the end of the graph. Each layer of the graph corresponds to a second-level tour $t^2 \in T^2$. Because the visiting sequence of the second-level tours is kept fixed, the layers can be arranged in sequence. The nodes within one layer represent the parking spots associated with the households in that particular second-level tour, i.e., $V_C(t^2)$. For example, k_2^1 represents the parking spot associated to the second household (see subscript) visited in the first second-level tour (see superscript).

For simplicity, the figure shows the situation in which only one parking spot is associated with each household, but each household can potentially have more than one associated parking spot. In this case, we would add an index to the notation used to represent nodes to indicate which of the associated parking spots we refer to. Each arc connects a parking spot of one layer to a parking spot of the successive one. For the sake of representation, the figure does not show the weights on the arcs which represent the travel time between pairs of connected parking spots. By solving a shortest path problem on such a weighted layered graph, we identify the optimal selection of parking spots to open.

Approaches based on solving a shortest path problem on a weighted layered graph are found, for example, in Villegas et al., 2010 and Cabrera et al., 2022. However, these authors follow a route-first cluster-second approach. Consequently, they use a weighted layered graph to optimally cut a household giant tour to determine the assignment of the households to parking spots that then need to be opened. On the contrary, we follow a cluster-first route-second approach that leads to a simplified layered graph. Instead of having as many layers as the number of households, we have as many layers as the number of second-level tours. Within each layer, instead of having as many nodes as the number of parking spots, the number of nodes is at most twice the number of households visited in that second-level tour. Moreover, because in our graph the layers represent the set of feasible second-level tours, no capacity constraints must be considered. To the best of our knowledge, this is the first time that such a layered graph is used to optimize the choice of the opened parking spots while keeping the visiting sequence of the second-level tours fixed.

If our procedure results in a different choice of the opened parking spots with respect to the one of the solution S returned by the iterated clustering algorithm, the second-level tours that were rooted in a parking spot that was previously open need to be adjusted. In each of those second-level tours, the first visited household becomes the closest one to the new opened parking spot. The visiting sequence of the remaining households is determined according to the order in which they appeared after the new first visited household in the previous version of the second-level tour. At the end of this improvement phase, the number of opened parking spots may have changed, e.g., if two second-level tours are now rooted in the same parking spot, or if two second-level tours which were rooted in the same parking spot are now rooted in two different parking spots.

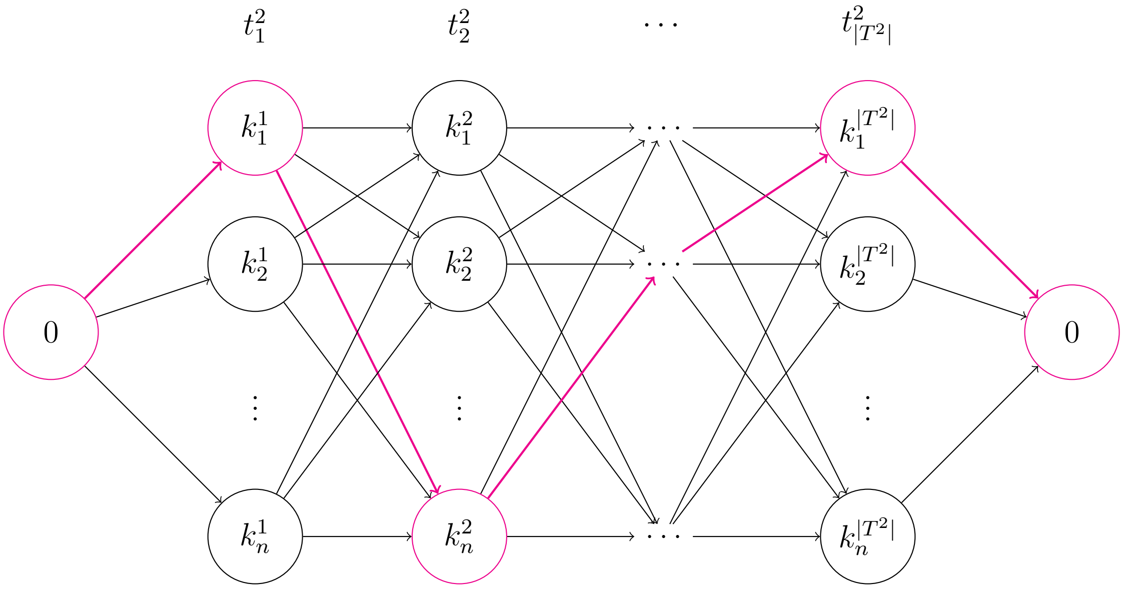


Figure 2.5: Structure of the auxiliary weighted layered graph and shortest path problem solution in magenta.

2.5.2 Iterated local search

Our ILS implements the classical ILS framework originally proposed by Lourenço et al., 2003. Each iteration of ILS is composed of two phases: (i) a VND phase with the goal of improving the travel time of the second-level tours (Section 2.5.2.1), and (ii) a perturbation phase with the goal of generating a new configuration of opened parking spots (Section 2.5.2.2). This problem decomposition differentiates our ILS from the one of Villegas et al., 2010 and Accorsi and Vigo, 2020, who apply their ILS to the first and second-level tours simultaneously.

2.5.2.1 Variable neighborhood descent

VND iteratively evaluates neighboring solutions that are obtained by applying to a solution S a move uniquely defined by a so-called generator arc (i, j) and a neighborhood operator $o \in O$. After the move is applied, the arc (i, j) is contained in the resulting solution. Because, in contrast to Villegas et al., 2010, the pivoting rule of our VND is first improvement, the order according to which the neighborhood operators and the generator arcs are traversed influences the search trajectory. For this reason, the elements in the sets of the neighborhood operators O and of the households V_C (which are used as the origin vertices of the generator arcs) are randomly shuffled at the beginning of each ILS iteration, before calling the VND, resulting in the lists $\tilde{O}$ and $\tilde{V}_C$. This increases the likelihood that different search trajectories are explored and different solutions are obtained in case the same configuration of opened parking spots is considered more than once.

Algorithm 3 shows the pseudocode of the VND. The VND traverses the list $\tilde{O}$

of neighborhood operators described in Section 2.5.2.1.1. For each operator, the generator arc list is traversed. Specifically, each arc is obtained by pairing a vertex i from the list $\tilde{V}_C$ of households with a vertex j from the list of the closest households and parking spots to household i. This latter list is denoted by L_i and obtained as described in Section 2.5.2.1.2. Given a neighborhood operator and a generator arc, a move is applied to a solution S and a new solution $S^{'}$ is obtained. If the total travel time of the new solution $S^{'}$ is lower than the total travel time of S, then the new solution $S^{'}$ becomes the new incumbent in the VND, and the search restarts from the first operator, instead of the current one as in Accorsi and Vigo, 2020.

Algorithm 3: Pseudocode of VND(S)

```
Input: $S$
improvementVND = false
for $o \in \tilde{O}$ do
    for $i \in \tilde{V}_C$ do
        for $j \in L_i$ do
            $S^{'} \leftarrow move(o, i, j, S)$
            if $C(S^{'}) < C(S)$ then
                improvementVND = true
                $S \leftarrow S^{'}$
                $C(S) \leftarrow C(S^{'})$
                break
            end
        end
        if improvementVND = true then
            break
        end
    end
    if improvementVND = true then
        break
    end
end
return $S$
```

2.5.2.1.1 Neighborhood operators

The neighborhood operators contained in set O are defined using the generator arc principle introduced in Section 2.5.2.1. They are depicted in Figure 2.6 and are:

- 2-opt* is used in inter-route and intra-parking spot fashion, i.e., it is not evaluated for second-level tours rooted at different parking spots.

- split creates a new second-level tour containing a single household, and it is defined in both intra- and inter-parking spot fashion.

- exchange swaps one household between second-level tours or within a second-level tour, and it is defined in both intra- and inter-parking spot fashion.

- relocate-1 moves one household between second-level tours or within a second-level tour, and it is defined in both intra- and inter-parking spot fashion.

- relocate-b, where the “b” stands for “backwards”, relocates all predecessors of a household (parking spot excluded) before another one, and it is defined in both intra- and inter-parking spot fashion.

The relocate-1 and split operators reduce the number of opened parking spots by one if the following three conditions apply simultaneously: (i) the second-level tour of household i is composed of only i, (ii) the parking spot of i is used only for the second-level tour of i, and (iii) the second-level tour of j is rooted at a different parking spot (for relocate-1) and j is different from the parking spot of i (for split). Similarly, the relocate-b operator reduces the number of opened parking spots by one if (i) i is the last visited household of a second-level tour, (ii) the parking spot of i is used only for the second-level tour of i, and (iii) the second-level tour of j is rooted at a different parking spot. For this reason, to compute the impact that each move has on the total travel time of a solution, we also consider the saving in the first-level tour travel time if these conditions hold for the above mentioned operators. This prevents discarding moves that may not appear as improving if only their effect on the second-level tour was evaluated.

2.5.2.1.2 Composition of L_i To speed up the search, we consider only a small portion of the total number of arcs in A to be used as generator arcs. This technique has been proposed by Toth and Vigo, 2003 and has later been applied in multiple works (see, e.g., Prins et al., 2007; Escobar et al., 2014; Goeke, 2019). In contrast to Accorsi and Vigo, 2020, who consider a portion of the shortest arcs in A, we propose a per-household sparsification method. Sparsifying on a per-household basis is vital considering that in DHL instances the households are positioned along street segments. This ensures that a given number of arcs incident to isolated households is always included.

In ILS-ASTTRPSD, each generator arc is obtained by pairing a vertex i from the list $\tilde{V_C}$ of households with a vertex j from the list L_i. List L_i contains the $\lceil \kappa_c V_C \rceil$ closest households and the $\lceil \kappa_k V_D \rceil$ closest opened parking spots to the household vertex i, with $0 < \kappa_c < 1$ and $0 < \kappa_k < 1$, respectively. In L_i, the vertices appear sorted according to increasing values of travel time to i. We have also tested the impact of considering a different sorting of the vertices, i.e., first the $\lceil \kappa_c V_C \rceil$ closest households vertices, and then the $\lceil \kappa_k V_D \rceil$ closest parking spot vertices, both sorted according to increasing values of travel times. However, preliminary results have shown a similar solution quality but a slight increase in runtimes.

2.5.2.2 Perturbation

The goal of the perturbation phase is to generate, in each iteration of our ILS, a different configuration of opened parking spots. To this end, in each iteration of ILS, we randomly choose one of the following perturbation moves:

- open a closed parking spot
- close an opened parking spot.

Depending on the chosen perturbation move, one of the following two new transformation heuristics is applied to transform the current solution into a solution with the new parking spot configuration:

- **Transformation heuristic when a closed parking spot is opened**: When a closed parking spot is opened, we adjust the second-level tours by first creating a new second-level tour for each household for which this new parking spot is closer than the currently assigned one. Then, we check if there are opened parking spots with no assigned households, and we close them. If the new opened parking spot is still open, i.e., it has at least one assigned household, we insert it in the first-level tour using a best insertion heuristic.
- **Transformation heuristic when an opened parking spot is closed**: When an opened parking spot is closed, we first solve a TSP on the first-level tour because simply removing the closed parking spot could result in suboptimal solutions. Then, we remove all second-level tours rooted at that parking spot. Finally, we create a new second-level tour that serves each unconnected household from its closest opened parking spot.

2.6 Computational experiments

The goal of the computational experiments is threefold. First, we evaluate the performance of ILS-ASTTRPSD. Second, we evaluate the impact of considering parking and loading times on the solution structure. Third, we assess the robustness of solutions under parking time fluctuations.

Section 2.6.1 describes the instance sets used in our experiments. Section 2.6.2 explains the computational environment and how the parameters for ILS-ASTTRPSD have been set. Finally, Section 2.6.3 presents the ILS-ASTTRPSD performance assessment (Section 2.6.3.1), evaluates the impact of considering parking and loading times (Section 2.6.3.2), and determines how robust solutions calculated assuming parking times equal for all parking spots are if fluctuations occur (Section 2.6.3.3).

2.6.1 Description of the instances

Our computational experiments are mainly based on the DHL instances described in Section 2.6.1.2. However, because for DHL instances optimal solutions are not available, we also test the performance of ILS-ASTTRPSD on the instances of Reed et al., 2024, see Section 2.6.1.1. Finally, to test the ILS-ASTTRPSD capability of returning good solutions also for symmetric instances, we execute ILS-ASTTRPSD on the well-known STTRPSD instances available in the literature. For a detailed description of these instances, we refer to Villegas et al., 2010.

2.6.1.1 Reed et al., 2024 instances

The instances for the CDPP proposed by Reed et al., 2024 are available at `https://doi.org/10.25820/data.006124`. The authors consider three counties, i.e., Cook, Adams, and Cumberland to represent urban, suburban, and rural household geographies, respectively. The authors assume that the driver can park at each household location. To get instances of the ASTTRPSD, we duplicate the household locations to obtain the locations of the parking spots. All travel times are expressed in minutes and taken from real-world data, and the instances are asymmetric. For each county, a different parking time is considered to reflect the time required to find a parking spot. Specifically, $\rho = 9$ for Cook county, $\rho = 5$ for Adams county, and $\rho = 1$ for Cumberland county. The loading time is set to $\ell = 2.1$, and it is charged for each served household. For each combination of county and capacity Q_b varying from one to six, ten instances with $|V_C| = 50$ households and five instances with $|V_C| = 100$ households are considered.

2.6.1.2 DHL instances

The set of real-world, non-Euclidean, asymmetric instances from DHL consists of 35 instances based on a German city of approximately 50 000 inhabitants. The city is divided into postal districts, each one corresponding to one instance and served by a mail carrier. The mean of transport assumed in DHL instances is a car. Without loss of generality, ILS-ASTTRPSD can solve instances with any other mean of transport. The instances contain a number of households $|V_C|$ between 245 and 465 (average 390), and a number of parking spots $|V_D|$ between 412 and 882 (average 724). Each household can be either a single house or an apartment building. For most of the households, there are two associated parking spots, representing a possible parking spot on each side of the street. The bag capacity Q_b is set to 25, and the demand of each household q_i ranges from a minimum of zero units to a maximum of 25. Recalling that DHL is interested in solving the problem at a tactical level, considering households with a demand of zero is meaningful because they represent currently uninhabited houses that could change their status and start having a positive demand before the problem is solved again. If this happens and the capacity of the second-level tour in which that household is visited is not completely exploited, the newly inhabited house can be easily included by keeping the second-level tours of our solutions unchanged. If we ignored these currently uninhabited houses, our solutions would not provide any indication on how to serve them. As a result, a mail carrier could use a greedy approach to decide how to incorporate the visit of these households in the tour with a consequent worse solution quality.

For each instance, two networks are given: (i) a walking network defined on a graph $G_w = (V_w, A_w)$ consisting of walking nodes V_w and walking arcs A_w, and (ii) a driving network defined on a graph $G_d = (V_d, A_d)$ consisting of driving nodes V_d and driving arcs A_d. Figure 2.7 shows an example of the two networks. For each household, there are two vertices in V_w: one representing the house itself and one representing the so-called "walking portal" corresponding to the point on the sidewalk, where the mail carrier enters the driveway of the house. The parking spots associated with the households are included both in V_w and V_d because they are reachable by driving and by walking. Figure 2.8 shows an example of these vertices in V_w and V_d.

To compute the travel time c_{ij} between vertices i and j, we solve a shortest path problem using Dijkstra's algorithm on:

- graph G_d, if i and j are both parking spots,
- graph G_w, if i is a parking spot and j is a household,
- graph G_w, if i is a household and j is a parking spot,
- graph G_w, if i and j are both households.

Because the travel time from the walking portal to the associated household always occurs for the mail carrier, it is a constant and, hence, it is not part of the optimization. Therefore, to compute $\{c_{ij} | (i \in V_C \wedge j \in V_C) \vee (i \in V_C \wedge j \in V_D) \vee (i \in V_D \wedge j \in V_C)\}$, we only consider the travel time from and up to the walking portal. We specify that, in the street network, U-turns are allowed at parking spots in which they are also possible in the real world. If this is the case for a parking spot, we also have a corresponding node in the street network on the other side of the street and they are connected (see, for example, Figure 2.7(b)). Because we solve a shortest path problem on such a street network for each pair of vertices to obtain our distance matrix, in case the U-turn is the cheapest option to connect two vertices, then the shortest path and the distance matrix contain this U-turn.

Finally, because at DHL parking and loading times are not separately known, DHL instances are characterized by a penalty P of 22 seconds every time a second-level tour is performed. This penalty represents the parking and the loading time for the block of letters to be delivered in a second-level tour. Graph G is modified by adding $P = 22$ to the arc travel times $\{c_{ij} | i \in V_D \wedge j \in V_C\}$, i.e., to the travel time of every walking arc connecting a parking spot to a household. If multiple second-level tours are rooted at the same parking spot, the total time required for parking is overestimated because it is added multiple times although parking occurs only once. However, because DHL has considered this graph structure to get their solutions, we use the same structure in the computational experiments of Section 2.6.3.1 to allow for a fair comparison.

The DHL instances are available at `https://data.mendeley.com/datasets/sskwdxcgwt/1`.

2.6.2 Computational environment and parameter tuning

ILS-ASTTRPSD was implemented in C++ and compiled using clang version 12.0.0. The experiments were performed on an Intel(R) Xeon(R) computer with a CPU E5-2430 v2 processor, at 2.50GHz with 64 GB RAM under CentOS GNU/Linux 7. Every time a TSP is solved exactly, we use Gurobi solver version 9.5.0.

The values of the parameters for ILS-ASTTRPSD are summarized in Table 2.1. Because our algorithm contains randomized elements, we performed ten runs of ILS-ASTTRPSD for each instance. Multiple procedures for tuning the parameters of algorithms have been proposed in the literature (see, for example, López-Ibáñez et al., 2016). Because our ILS only includes two parameters to tune, i.e., the values for the termination criterion of ILS-ASTTRPSD and for the sparsification intensity, we have conducted a manual parameter tuning on a subset of the DHL instances which aims at determining a decent parameter setting while avoiding to overfit the setting to the problem instances under consideration. For this parameter tuning, we have randomly selected 15 DHL instances. The following values for the termination criterion of ILS-ASTTRPSD and for the sparsification intensity have been considered: $\eta = [500, 1000, 1500, 2000, 2500]$ and $\kappa_c = \kappa_k = 0.01$, or $\kappa_c = \kappa_k = 0.015$. The results are summarized in Figures 2.9(a) and 2.9(b) which show the impact of an increasing number of iterations without improvement and of different sparsification intensity values on the average percentage gap to the DHL solutions and on the runtime, respectively. The results show that a sparsification intensity of $\kappa_c = \kappa_k = 0.015$ always returns better results than using $\kappa_c = \kappa_k = 0.01$. However, the gain in solution quality reduces and becomes negligible beyond $\eta = 2000$, while the runtime increases considerably for $\eta = 2500$. Consequently, the best configuration ($\kappa_c = \kappa_k = 0.015, \eta = 2000$) in terms of solution quality and runtime tradeoff has been chosen (see Table 2.1).

Nevertheless, preliminary experiments have also shown that ILS-ASTTRPSD runtimes may get long when the number of parking spots is greater than or equal to 100 and the following relationship is fulfilled:

$$\frac{1}{(\rho/\bar{c}_{V_D})|V_D|} \geq 0.04,$$

where ρ is the parking time, $\bar{c}_{V_D}$ is the average inter-parking spot distance, $|V_D|$ is the number of parking spots that is used as an adjustment factor, and 0.04 is an empirical value derived from preliminary experiments. The inequality is satisfied in those situations in which the average inter-parking spot distance exceeds the total parking time when opening all parking spots in the instance multiplied by this empirical value. This relationship is satisfied for big instances representing rural settings with very small parking times compared to the inter-parking spot distance. For these instances, we have observed that by setting the number of iterations without improvement to values larger than 50, the solution quality improves only marginally at the expense of very long runtimes. For this reason, in such settings, the number of iterations without improvement η is set to 50.

Component	Parameter values
termination criterion iterated clustering algorithm	$\gamma_{max} = 10$
termination criterion ILS-ASTTRPSD	$\eta = 2000$
sparsification intensity households	$\kappa_c = 0.015$
sparsification intensity parking spots	$\kappa_k = 0.015$

Table 2.1: ILS-ASTTRPSD parameter values.

2.6.3 Results

In this section, we test the performance of ILS-ASTTRPSD (Section 2.6.3.1), we evaluate the impact of considering parking and loading times when solving the ASTTRPSD (Section 2.6.3.2), and we assess the robustness of solutions under parking time fluctuations (Section 2.6.3.3).

2.6.3.1 ILS-ASTTRPSD performance assessment

To assess the performance of ILS-ASTTRPSD, we compare our solutions to the ones of Reed et al., 2024 and to the ones obtained by the DHL data analytics department for the DHL instances. For completeness, Appendix A compares the results of ILS-ASTTRPSD to the ones of the state-of-the-art heuristics on symmetric STTRPSD instances.

2.6.3.1.1 Comparison on Reed et al., 2024 instances We run ILS-ASTTRPSD on all instances of Reed et al., 2024 and compare to the solutions returned by both their heuristic and the commercial solver Gurobi 9.0.0, which is able to solve some of the instances to optimality.

Table 2.2 reports the results of this comparison. The first three columns identify the instance based on the county, the number of households $|V_C|$, and the bag capacity Q_b. The fourth column reports the average value of the best known solutions (BKS) over ten instances (if $|V_C| = 50$) or over five instances (if $|V_C| = 100$). The BKSs are taken from Reed et al., 2024. If the optimal solutions returned by Gurobi are available, these are used in the computation of the best known solutions (BKSs). Otherwise, the heuristic solutions found by Reed et al., 2024 are used. Combinations of county, number of households, and bag capacity for which all instances have been solved to optimality are highlighted in bold. Concerning the runtimes of Gurobi to solve instances to optimality, Reed et al., 2024 only mention a few examples. When $|V_C| = 50$ and $Q_b = 3$, the average runtime is 21240, 15840, and 11160 seconds for Cook, Adams, and Cumberland counties, respectively. When $|V_C| = 100$ and $Q_b = 2$, the average runtime is 1440 seconds for Cook county, 11160 seconds for Adams county, and 30600 seconds for Cumberland county. The fifth column reports the average gap to the BKS ($\Delta_{BKS}(\%)$) of the heuristic of Reed et al., 2024. The next four columns report the average gap to the BKS reached by ILS-ASTTRPSD in the best ($\Delta^b_{BKS}(\%)$) and average ($\Delta^a_{BKS}(\%)$) run, respectively, the average standard deviation of our ILS-ASTTRPSD runs with respect to the BKS, and the average runtime in seconds ($t(s)$). Underlined values improve on the BKS. Because the parameter tuning for ILS-ASTTRPSD is based on the DHL instances, we also try to solve Reed et al., 2024 instances with ILS-ASTTRPSD using only half the number of iterations without improvement. This ILS-ASTTRPSD variant is called “ILS-ASTTRPSD-fast” and allows us to investigate the tradeoff among solution quality and runtime. The last four columns of Table 2.2 report the same four statistics reported for ILS-ASTTRPSD, but for ILS-ASTTRPSD-fast. A comparison of ILS-ASTTRPSD to the heuristic of Reed et al., 2024 based on the runtimes is not possible due to the lack of information regarding the characteristics of the machine of the University of Iowa’s Argon high performance computing cluster used in Reed et al., 2024. Reed et al., 2024 state that the average runtime of their heuristic is at most 42 seconds by running parallelized experiments using the Gurobi solver and limiting the thread count to 32.

The results show that, in the best run, ILS-ASTTRPSD always finds solutions of better quality than the ones provided by the heuristic of Reed et al., 2024. Specifically, on instances for which optimal solutions are available, ILS-ASTTRPSD returns smaller optimality gaps, and on instances for which only the heuristic solution of Reed et al., 2024 is available, ILS-ASTTRPSD finds new best-known solutions. The same results are obtained in the average run of ILS-ASTTRPSD, except for the instances referring to Cook county with $|V_C| = 50$ and $|V_C| = 100$, and $Q_b = 2$. Comparing the gaps of the best run of ILS-ASTTRPSD to optimal solutions across counties, the best performance is obtained with rural household geographies (Cumberland county), followed by suburban (Adams county), and urban (Cook county) ones. Instead, the comparison of the gaps to the BKSs of the best and average runs of ILS-ASTTRPSD across counties shows that the highest improvements are obtained for the suburban county, followed by the urban, and rural ones.

Looking at the column of Table 2.2 reporting the average runtimes, ILS-ASTTRPSD shows reasonable runtimes. While the runtimes of ILS-ASTTRPSD are roughly equivalent for instances with suburban and urban geographies (Adams and Cook), the ones for the Cumberland county are longer. We note that these instances correspond to big rural instances for which η has been set to 50 (see Section 2.6.2). These longer runtimes are due to very small parking times compared to inter-parking spot distances that cause the opening of a lot of parking spots. This requires, for ILS-ASTTRPSD, to solve a TSP on the first-level tour every time that a perturbation move closing an opened parking spot is applied. These results are in line with the observation of Reed et al., 2024, where some of the Cumberland county instances with 100 households could not be solved to optimality even for small bag capacities.

The comparison of ILS-ASTTRPSD and ILS-ASTTRPSD-fast shows that by decreasing the number of iterations without improvement by half, the runtimes reduce by approximately 24% on average at the expense of a slightly deteriorating solution quality. The biggest difference in solution quality between ILS-ASTTRPSD and ILS-ASTTRPSD-fast is observed for Cook instances with higher capacity values and is equal to 0.2%. The results achieved with ILS-ASTTRPSD-fast suggest that, by properly adjusting the parameter η of ILS-ASTTRPSD for smaller instances than the DHL ones, runtimes can decrease while still achieving a comparable level of solution quality. Finally, the small values of the average standard deviation indicate that the solutions do not vary substantially across the runs of ILS-ASTTRPSD and ILS-ASTTRPSD-fast, which is a signal of algorithmic robustness.

2.6.3.1.2 Comparison on DHL instances We compare the results obtained with ILS-ASTTRPSD to the ones provided by the DHL analytics department on DHL instances. We remark that the DHL solutions have not been obtained by manual planning but by using a local search heuristic. Their local search algorithm takes as input the solution that is currently implemented in practice and applies operators to merge and split second-level tours. A post-processing phase that uses the 2-opt and 3-opt heuristic is finally executed.

Table 2.3 presents the comparison of the DHL solutions to the solutions obtained by ILS-ASTTRPSD. The first columns in the table contain for each instance: the instance name, the number of households ($|V_C|$), the number of parking spots ($|V_D|$), the cumulative demand ($\sum_{i \in V_C} q_i$), and the average inter-household distance ($\overline{c_{ij}}$). The sixth column contains the objective function values of the DHL solution. Because the runtimes of the DHL algorithm are not available, we do not include this information in the table. The last columns of the table contain for each instance: the gap to the DHL solution of the ILS-ASTTRPSD best run ($\Delta^b(\%)$), of the ILS-ASTTRPSD average run ($\Delta^a(\%)$), the standard deviation of the objective function values across ILS-ASTTRPSD runs, and the runtime of the ILS-ASTTRPSD average run ($\bar{t}(s)$). On all instances, ILS-ASTTRPSD improves the solution of DHL in the best run. The same applies for the average run, except for two instances on which we perform slightly worse, and for one instance on which we achieve the same solution quality. Even if an average improvement by 1.95% may seem limited, we remark that this is only for one postal district. Considering that DHL has to serve thousands of districts,

Instance			BKS	Reed et al., 2024	ILS-ASTTRPSD				ILS-ASTTRPSD-fast			
County	$\lvert V_C \rvert$	Q_b		$\Delta_{BKS}(\%)$	$\Delta^b_{BKS}(\%)$	$\Delta^a_{BKS}(\%)$	$\Delta^{std\_dev}_{BKS}(\pm)$	$t(s)$	$\Delta^b_{BKS}(\%)$	$\Delta^a_{BKS}(\%)$	$\Delta^{std\_dev}_{BKS}(\pm)$	$t(s)$
adams	50	1	**296.9**	4.0	0.5	1.3	0.5	61.4	0.5	1.3	0.5	37.3
adams	50	2	**286.7**	2.2	0.3	0.6	0.3	37.1	0.3	0.6	0.3	22.0
adams	50	3	**281.8**	3.1	0.1	0.5	0.3	31.4	0.1	0.5	0.3	18.0
adams	50	4	**280.2**	3.6	0.0	0.2	0.2	30.3	0.0	0.2	0.2	18.3
adams	50	5	290.2	0.0	$\underline{-3.9}$	$\underline{-3.5}$	0.2	25.0	$\underline{-3.9}$	$\underline{-3.5}$	0.2	14.6
adams	50	6	290.2	0.0	$\underline{-4.4}$	$\underline{-4.1}$	0.2	22.9	$\underline{-4.4}$	$\underline{-4.1}$	0.2	13.6
adams	100	1	**538.0**	4.9	1.1	1.9	0.6	517.3	1.1	1.9	0.6	360.8
adams	100	2	**515.2**	2.6	0.7	1.3	0.4	228.7	0.7	1.3	0.4	145.6
adams	100	3	523.2	0.0	$\underline{-3.2}$	$\underline{-2.6}$	0.3	128.8	$\underline{-3.2}$	$\underline{-2.6}$	0.3	81.0
adams	100	4	521.7	0.0	$\underline{-4.7}$	$\underline{-4.4}$	0.3	108.4	$\underline{-4.7}$	$\underline{-4.3}$	0.3	68.0
adams	100	5	521.7	0.0	$\underline{-5.9}$	$\underline{-5.5}$	0.3	81.0	$\underline{-5.8}$	$\underline{-5.5}$	0.3	54.5
adams	100	6	521.3	0.0	$\underline{-6.6}$	$\underline{-6.2}$	0.3	77.4	$\underline{-6.6}$	$\underline{-6.2}$	0.3	47.7
Avg. adams			405.6	1.7	-2.2	-1.7	0.3	112.5	-2.2	-1.7	0.3	73.5
cook	50	1	**328.3**	5.5	1.7	3.5	1.2	18.2	1.7	3.5	1.2	12.7
cook	50	2	**292.4**	1.3	1.0	2.8	1.1	11.2	1.0	2.9	1.2	6.8
cook	50	3	**272.5**	3.4	0.9	2.1	0.8	9.9	1.1	2.2	0.8	5.9
cook	50	4	**261.8**	5.5	0.6	1.6	0.9	20.5	0.6	1.7	0.9	11.8
cook	50	5	273.9	0.0	$\underline{-7.2}$	$\underline{-6.0}$	0.9	26.1	$\underline{-7.1}$	$\underline{-5.9}$	1.0	14.2
cook	50	6	272.4	0.0	$\underline{-9.1}$	$\underline{-8.3}$	0.7	29.5	$\underline{-9.1}$	$\underline{-8.2}$	0.8	16.7
cook	100	1	**621.0**	4.9	1.6	2.9	0.8	254.0	1.6	2.9	0.8	221.1
cook	100	2	**547.9**	0.5	1.5	2.7	0.7	54.8	1.5	2.7	0.7	37.8
cook	100	3	526.1	0.0	$\underline{-1.6}$	$\underline{-0.7}$	0.6	41.4	$\underline{-1.6}$	$\underline{-0.7}$	0.6	27.3
cook	100	4	513.3	0.0	$\underline{-3.5}$	$\underline{-2.8}$	0.4	63.8	$\underline{-3.5}$	$\underline{-2.6}$	0.5	36.4
cook	100	5	507.9	0.0	$\underline{-5.5}$	$\underline{-4.7}$	0.6	72.4	$\underline{-5.4}$	$\underline{-4.6}$	0.6	45.4
cook	100	6	505.2	0.0	$\underline{-7.0}$	$\underline{-6.3}$	0.4	89.6	$\underline{-6.8}$	$\underline{-6.2}$	0.4	50.3
Avg. cook			410.2	1.8	-2.2	-1.1	0.8	57.6	-2.2	-1.0	0.8	40.5
cumberland	50	1	**192.9**	1.6	0.1	0.2	0.1	275.1	0.1	0.2	0.1	146.2
cumberland	50	2	**192.6**	1.5	0.1	0.1	0.0	253.3	0.1	0.1	0.0	131.3
cumberland	50	3	**192.6**	1.5	0.1	0.1	0.0	246.1	0.1	0.1	0.0	123.9
cumberland	50	4	**192.6**	1.5	0.1	0.1	0.0	239.2	0.1	0.1	0.0	122.9
cumberland	50	5	195.5	0.0	$\underline{-1.5}$	$\underline{-1.5}$	0.0	234.5	$\underline{-1.5}$	$\underline{-1.5}$	0.0	119.7
cumberland	50	6	195.5	0.0	$\underline{-1.5}$	$\underline{-1.5}$	0.0	238.0	$\underline{-1.5}$	$\underline{-1.5}$	0.0	120.2
cumberland	100	1	360.6	1.2	0.2	0.5	0.2	7942.5	0.3	1.2	0.6	6432.2
cumberland	100	2	360.1	1.1	$\underline{-0.0}$	0.1	0.0	760.1	0.0	0.2	0.0	589.8
cumberland	100	3	364.0	0.0	$\underline{-1.1}$	$\underline{-1.0}$	0.0	562.8	$\underline{-1.1}$	$\underline{-0.9}$	0.0	455.5
cumberland	100	4	364.0	0.0	$\underline{-1.2}$	$\underline{-1.1}$	0.0	509.6	$\underline{-1.2}$	$\underline{-1.1}$	0.0	433.8
cumberland	100	5	364.0	0.0	$\underline{-1.2}$	$\underline{-1.1}$	0.0	567.1	$\underline{-1.2}$	$\underline{-1.1}$	0.0	482.3
cumberland	100	6	364.0	0.0	$\underline{-1.2}$	$\underline{-1.1}$	0.0	506.6	$\underline{-1.2}$	$\underline{-1.1}$	0.0	416.9
Avg. cumberland			278.2	0.7	-0.6	-0.5	0.0	1027.9	-0.6	-0.4	0.0	797.9
Avg.				1.4	-1.7	-1.1	0.4	399.3	-1.6	-1.1	0.4	304.0

Table 2.2: Comparison of the average results of Reed et al., 2024's heuristic with ILS-ASTTRPSD and ILS-ASTTRPSD-fast. Each row of the table reports the average results over 10 instances if $V_C = 50$, and over 5 instances if $V_C = 100$. Optimality has been proven for the solutions in boldface by Reed et al., 2024. Results that improve on the previous BKS are underlined.

a small percentage improvement entails considerable cost savings. The robustness of ILS-ASTTRPSD across runs is confirmed by the low standard deviation values reported also for the DHL instances.

Because postal delivery tours are planned at the tactical level, runtime is not crucial for DHL. Still, the average runtime of ILS-ASTTRPSD corresponds to approximately 20 minutes, meaning that improvements are found quickly. We observe that the runtimes are not strictly related to the instance dimension. For example, instances 25 and 35 have approximately the same number of households and parking spots, but their average runtimes differ significantly. This might be due to a particular search trajectory of ILS-ASTTRPSD or to the geography of a particular instance. For example, instance 35 has a higher average inter-household distance compared to instance 25. The same can be observed when comparing instances 13, 16, 19, and 26. While these instances have similar dimension, instance 16 with the highest average inter-household distance is the one with the highest runtime. This result is in line with our observation for Reed et al., 2024 instances in Section 2.6.3.1.1, where instances of rural geographies, i.e., with larger inter-household distances, are the most difficult to solve.

					DHL	ILS-ASTTRPSD			
Instance	$\lvert V_C\rvert$	$\lvert V_D\rvert$	$\sum_{i\in V_C} q_i$	$\overline{c_{ij}}$		$\Delta^b(\%)$	$\Delta^a(\%)$	$\Delta^{std\_dev}(\pm)$	$\bar{t}(s)$
1	382	746	646	417.16	11047.67	-1.18	-0.85	0.26	1264.51
2	465	852	713	556.17	11307.95	-2.74	-2.31	0.25	1784.86
3	424	778	703	399.71	10130.74	-0.92	-0.41	0.61	1438.54
4	433	742	711	333.05	8916.46	-4.44	-4.02	0.29	1447.48
5	403	698	826	464.93	9874.41	-3.31	-2.49	0.47	1460.92
6	378	585	887	527.94	9618.68	-4.33	-3.77	0.42	1130.51
7	361	643	884	340.49	8366.41	-1.94	-1.50	0.56	1243.83
8	452	872	791	497.59	10207.59	-1.78	-1.10	0.46	2157.41
9	250	412	483	614.93	7560.05	-2.74	-2.17	0.61	525.97
10	298	481	1092	293.17	8967.08	-1.93	-1.51	0.30	692.75
11	389	758	756	504.81	9900.64	-2.81	-2.17	0.38	1126.18
12	374	729	628	456.74	10774.50	-2.83	-2.36	0.51	1222.89
13	418	762	770	433.92	9539.38	-1.46	-0.88	0.36	1324.16
14	360	697	653	522.93	9910.45	-3.17	-2.19	0.63	975.74
15	441	854	700	537.55	11657.85	-1.85	-0.87	0.51	1584.94
16	416	793	621	834.63	12542.86	-0.90	-0.51	0.25	1448.51
17	427	812	634	375.59	9376.12	-1.06	-0.60	0.23	1570.28
18	452	882	725	600.57	11042.54	-1.36	-0.69	0.45	1787.35
19	414	767	654	409.54	13517.72	-1.40	-0.82	0.32	1368.85
20	378	720	696	444.09	10302.26	-2.43	-1.66	0.53	1078.21
21	458	772	627	475.02	11549.82	-1.46	-0.89	0.39	1393.34
22	447	770	673	327.47	11160.20	-3.39	-2.79	0.43	1397.45
23	430	815	641	353.29	11280.73	-2.04	-0.93	0.64	1414.80
24	334	639	606	647.47	11211.68	-0.75	0.11	0.57	1350.33
25	342	640	601	375.88	9792.03	-1.22	-0.17	0.68	774.70
26	419	814	740	400.45	11573.60	-2.44	-1.77	0.40	1351.10
27	368	700	568	930.46	11439.95	-1.04	-0.77	0.20	1007.06
28	395	762	571	385.66	12231.05	-1.14	-0.55	0.61	1567.54
29	384	746	745	321.71	10173.03	-0.43	-0.00	0.42	1337.85
30	369	722	676	543.72	10765.11	-2.50	-1.35	0.62	762.33
31	410	728	736	413.50	9758.15	-1.95	-1.53	0.40	1168.34
32	429	837	788	696.90	11345.84	-1.76	-1.38	0.34	2006.24
33	245	477	453	1629.02	12999.68	-2.11	-1.98	0.18	1460.67
34	353	695	735	359.09	10254.06	-0.45	0.17	0.35	786.97
35	341	644	498	510.62	9974.97	-1.08	-0.59	0.38	1199.02
Avg.	389.69	724.11	692.31	512.45		-1.95	-1.35	0.43	1303.19

Table 2.3: Comparison of the DHL solutions to the solutions of the best and average run of ILS-ASTTRPSD on DHL instances.

To understand why ILS-ASTTRPSD returns better-quality solutions than the algorithm of DHL, we compare the structure of the solutions of ILS-ASTTRPSD and DHL. Table 2.4 presents this comparison with respect to: the first-level tour travel time $c(t^1)$, the travel time of the second-level tours $\sum_{t^2\in T^2} c(t^2)$, the number of opened parking spots $|V_D(t^1)|$, the number of second-level tours $|T^2|$, and the average number of households served in each second-level tour $\overline{|V_C(t^2)|}$. By looking at the last row of the table reporting the average values, we observe that ILS-ASTTRPSD solutions are characterized by shorter travel times for first- and second-level tours. Considering that in ILS-ASTTRPSD solutions, a lower number of parking spots are opened and that more households are served within each second-level tour, this suggests that ILS-ASTTRPSD performs a better selection of parking spots to open and provides better-quality second-level tours. Moreover, in the DHL solutions, the number of opened parking spots exactly corresponds to the number of second-level tours across all instances. This means that, in the DHL results, there are no second-level tours rooted at the same parking spot. In the ILS-ASTTRPSD solutions, for some instances, the number of second-level tours is higher than the number of opened parking spots, indicating that starting multiple second-level tours from the same parking spot might be beneficial to decrease travel times. By analyzing these instances, we observed that this is useful if, for example, one single household is located very close to others and exhibits a very high demand, or if there are two dead-end streets crossing at a main street where the parking spot is located (see Figures 2.10(a) and 2.10(b)). To increase readability, the figure shows a simplified version of the street network. The vertex highlighted in magenta represents the parking spot at which two second-level tours are rooted, and the first-level tour is represented in magenta. To make the figure more compact, instead of drawing the actual household vertices, we have represented their walking portal vertices in orange or green depending on the second-level tour they belong to. The same color scheme is used for the walked arcs.

Our results suggest that by adopting ILS-ASTTRPSD solutions:

- Mail carriers drive and walk less.
- Mail carriers park less. Allen et al., 2018 noticed that drivers prefer walking instead of continually moving the vehicle for very small distances and trying to find parking spots.
- Performing multiple second-level tours from the same parking spot may be beneficial especially in the presence of households having high demand or in the presence of a particular road network structure.
- Because of the lower number of opened parking spots, mail carriers serve more households in each second-level tour. However, this does not imply longer walking times.

2.6.3.2 Impact of ignoring parking and loading times

When parking does not require difficult maneuvers or when loading does not imply any other operation than grabbing a bag, parking and loading times are negligible. However, if these operations are not immediate, ignoring the time required for parking and loading may result in a solution of bad quality and in increased total travel times.

	$c(t^1)$		$\sum_{t^2 \in T^2} c(t^2)$		$\lvert V_D(t^1)\rvert$		$\lvert T^2\rvert$		$\overline{\lvert V_C(t^2)\rvert}$	
Instance	DHL	ILS-ASTTRPSD	DHL	ILS-ASTTRPSD	DHL	ILS-ASTTRPSD	DHL	ILS-ASTTRPSD	DHL	ILS-ASTTRPSD
1	2902.35	2722.11	8145.32	8195.37	58	55	58	55	6.59	6.95
2	3797.08	3598.03	7510.87	7399.67	55	52	55	52	8.45	8.94
3	2405.94	2274.41	7724.80	7763.23	51	48	51	48	8.31	8.83
4	2357.01	2313.49	6559.45	6207.38	50	46	50	46	8.66	9.41
5	2466.07	2193.87	7408.34	7353.59	58	55	58	57	6.95	7.07
6	2196.52	2156.09	7422.15	7046.51	65	61	65	63	5.82	6.00
7	1866.04	1635.10	6500.37	6568.96	67	59	67	59	5.39	6.12
8	1968.59	1918.29	8238.99	8107.29	63	66	63	67	7.17	6.75
9	2390.87	2309.35	5169.18	5043.64	51	46	51	46	4.90	5.43
10	2059.21	2090.90	6907.87	6703.39	76	73	76	75	3.92	3.97
11	2311.93	2194.77	7588.71	7427.85	56	54	56	54	6.95	7.20
12	3462.30	3329.56	7312.20	7140.46	55	60	55	61	6.80	6.13
13	2588.76	2464.53	6950.62	6935.72	58	54	58	54	7.21	7.74
14	3145.48	2967.53	6764.98	6628.38	50	49	50	49	7.20	7.35
15	3258.84	2958.45	8399.01	8483.83	56	51	56	51	7.88	8.65
16	4168.42	4154.70	8374.43	8274.80	49	52	49	52	8.49	8.00
17	2808.81	2667.19	6567.31	6609.27	37	33	37	34	11.54	12.56
18	3104.88	3098.29	7937.66	7794.34	44	44	44	44	10.27	10.27
19	5589.42	5513.26	7928.30	7814.77	54	50	54	52	7.67	7.96
20	3309.68	3143.57	6992.58	6908.79	57	50	57	50	6.63	7.56
21	3702.90	3509.14	7846.92	7872.37	48	38	48	39	9.54	11.74
22	3002.92	2841.29	8157.28	7940.38	45	42	45	42	9.93	10.64
23	3043.75	2862.57	8236.98	8187.91	51	49	51	49	8.43	8.78
24	3947.10	3894.33	7264.58	7233.02	67	68	67	68	4.99	4.91
25	3369.26	3330.91	6422.77	6341.60	40	39	40	39	8.55	8.77
26	3773.79	3537.61	7799.80	7753.07	49	49	49	50	8.55	8.38
27	4493.87	4408.21	6946.08	6913.10	50	50	50	50	7.36	7.36
28	4103.63	3963.43	8127.42	8128.11	55	44	55	45	7.18	8.78
29	3768.42	3618.65	6404.61	6510.23	46	40	46	42	8.35	9.14
30	4097.62	3966.66	6667.49	6528.79	47	43	47	43	7.85	8.58
31	2018.49	2053.61	7739.66	7514.26	57	59	57	59	7.19	6.95
32	3543.11	3331.90	7802.72	7814.57	76	68	76	69	5.64	6.22
33	6019.41	5705.95	6980.27	7019.42	105	103	105	103	2.33	2.38
34	3452.49	3162.04	6801.58	7045.64	65	54	65	54	5.43	6.54
35	3682.26	3860.12	6292.71	6007.28	64	65	64	65	5.33	5.25
Avg.	3262.21	3135.71	7311.26	7234.77	56.43	53.40	56.43	53.89	7.24	7.64

Table 2.4: Comparison of statistics of the DHL solutions to the solutions of the best run of ILS-ASTTRPSD for DHL instances.

As mentioned in Section 2.1, by adding additional times to the travel times of specific arcs of G, we can obtain different problem settings. To assess the impact of ignoring parking and loading times, we execute ILS-ASTTRPSD on DHL instances considering the following graphs:

1. Graph 1 (G_1) represents an ASTTRPSD only with travel times. In this problem setting, the parking and loading times are neglected.

2. Graph 2 (G_2) represents an ASTTRPSD with parking times, but loading times are neglected. If the vehicle stops at a parking spot, a parking time ρ of 11 seconds is added. Graph G is modified by adding $\rho = 11$ to every $c_{ij}, i \in V_D, j \in V_D, i \neq j$, i.e., to the travel time of every driving arc entering a parking spot. Neglecting loading times represents, for example, problem settings in which the mail carrier's bag already contains the letters of all households.

3. Graph 3 (G_3) represents an ASTTRPSD with loading times, but parking times are neglected. Every time the mail carrier has to load the bag with letters to start a new second-level tour, a loading time ℓ of 11 seconds is added. Graph G is modified by adding $\ell = 11$ to every $c_{ij}, i \in V_D, j \in V_C$, i.e., to the travel time of every walking arc connecting a parking spot to a household. Neglecting

parking times represents problem settings in which parking operations are straightforward.

4. Graph 4 (G_4) represents an ASTTRPSD with travel, parking, and loading times. If the vehicle stops at a parking spot, a parking time ρ of 11 seconds is charged, and every time the mail carrier has to load the bag with letters to start a new second-level tour, a loading time ℓ of 11 seconds is charged. Graph G is modified by adding $\rho = 11$ to every $c_{ij}, i \in V_D, j \in V_D, i \neq j$, i.e., to the travel time of every driving arc entering a parking spot, and $\ell = 11$ to every $c_{ij}, i \in V_D, j \in V_C$, i.e., to the travel time of every walking arc connecting a parking spot to a household.

In the above graphs, when considered, the time values for parking and loading are both set equal to 11 to fairly compare settings in which parking or loading are ignored. The solutions obtained by running ILS-ASTTRPSD on instances based on graphs G_1, G_2, and G_3 represent good solutions for the cases in which parking and loading times, loading times, and parking times are ignored, respectively. For a fair comparison of the objective function values, we use the solutions of the best run of ILS-ASTTRPSD obtained for the instances based on graphs G_1, G_2, and G_3, and evaluate the solutions based on the travel times of graph G_4, i.e., the one including both parking and loading times. This allows us to measure the impact of ignoring parking and/or loading times. Table 2.5 summarizes the average objective function values over the 35 DHL instances decomposed into first $(c(t^1)^{G_4})$ and second-level tours $(c(T^2)^{G_4})$ travel times. The last column contains the average gap to the total objective function value of graph G_4. For the first-level tour, we report driving time, parking time, total time, and the average gap to the first-level tour travel time of graph G_4. For the second-level tours, we report walking time, loading times, total time, and the average gap to the second-level tours travel times of graph G_4. The results of the last column of the table suggest that the highest increase in travel times occurs when both parking and loading times are neglected (graph G_1), followed by the case in which parking times (graph G_3), and loading times (graph G_2) are ignored. The higher increase in travel times in graph G_3 compared to graph G_2 may be even more remarkable in real-world settings in which parking usually requires more time than loading. Ignoring parking and/or loading times (graphs G_1, G_2, and G_3) always results in longer driving times and shorter walking times compared to instances in which they are considered (graph G_4). The absence of parking and/or loading times leads to the opening of a higher number of parking spots and second-level tours, so that the time for parking, for driving between parking spots, and for loading increases, while the walking times decrease.

To understand the reason of the travel time differences of Table 2.5, we compare the structure of the solutions based on some metrics. The average results of these

	$c(t^1)^{G_4}$				$c(T^2)^{G_4}$				
Graph	Driving	Parking	Total	$\overline{\Delta}_{c(t^1)^{G_4}}(\%)$	Walking	Loading	Total	$\overline{\Delta}_{c(T^2)^{G_4}}(\%)$	$\overline{\Delta}_{c^{G_4}}(\%)$
G_1	3541.80	1859.33	5401.12	47.99	4669.00	1862.19	6531.19	-2.40	15.39
G_2	3218.65	841.06	4059.71	11.23	5544.87	863.61	6408.48	-4.23	1.23
G_3	3231.57	863.39	4094.96	12.20	5527.68	867.46	6395.14	-4.43	1.44
G_4	3088.70	561.00	3649.70	-	6102.91	588.39	6691.30	-	-

Table 2.5: Comparison of average objective function values (over 35 instances) obtained by taking the solutions of ILS-ASTTRPSD for the instances of graphs G_1, G_2, G_3, and G_4, and by evaluating them based on the travel times of G_4.

metrics over all instances for each graph are summarized in Table 2.6, while the detailed results are reported in Appendix B. The travel times reported in this table are computed according to the travel times of the graph each line refers to. When parking times are set to zero (graphs G_1 and G_3), the number of opened parking spots $|V_D(t^1)|$ and of second-level tours $|T^2|$ is higher than in the settings where parking times are considered (graphs G_2 and G_4). A consequence of the lower number of opened parking spots in graphs G_2 and G_4 is the larger number of second-level tours originating from the same parking spot compared to their number in graphs G_1 and G_3. The average number of households served in each second-level tours increases as the number of opened parking spots decrease. In general, as more time components are added to the instances, the solution structure gets more consolidated, i.e., fewer parking spots are opened, fewer second-level tours are used, and more households are served within each second-level tour.

	$c(t^1)$		$c(T^2)$		$\|V_D(t^1)\|$	$\|T^2\|$	$\overline{\|V_C(t^2)\|}$
Graph	Driving	Parking	Walking	Loading			
G_1	3541.80	-	4669.44	-	169.03	169.29	2.36
G_2	3218.65	841.06	5544.87	-	76.46	78.51	5.22
G_3	3231.57	-	5527.68	867.46	78.49	78.86	5.16
G_4	3088.70	561.00	6102.91	588.39	51.00	53.49	7.73

Table 2.6: Comparison of objective function values and metrics of the solutions of ILS-ASTTRPSD for the instances of graphs G_1, G_2, G_3, and G_4. The values for each graph type are reported as averages over 35 instances.

The results in Table 2.6 are also in line with the properties stated in Section 2.4. By comparing the number of opened parking spots reported in columns $|V_D(t^1)|$ of Table 2.6, we observe that for graph G_1, the number of opened parking spots is closer to the solution of a TSP, in which each household is served by a dedicated parking spot. Conversely, with graphs G_2, G_3, and G_4, less parking spots are opened. This is a consequence of Property 2.4.2. To understand why, for each graph type, we make explicit the condition stated in Property 2.4.2 that corresponds to the following:

$$\begin{aligned}
&\textit{Graph } G_1: && c_{h_1k_1} + c_{k_1k_2} + c_{k_2h_2} + c_{h_2k_2} < c_{h_1h_2} + c_{h_2k_1}.\\
&\textit{Graph } G_2: && c_{h_1k_1} + c_{k_1k_2} + \rho_{k_2} + c_{k_2h_2} + c_{h_2k_2} < c_{h_1h_2} + c_{h_2k_1}.\\
&\textit{Graph } G_3: && c_{h_1k_1} + c_{k_1k_2} + \ell + c_{k_2h_2} + c_{h_2k_2} < c_{h_1h_2} + c_{h_2k_1}.\\
&\textit{Graph } G_4: && c_{h_1k_1} + c_{k_1k_2} + \rho_{k_2} + \ell + c_{k_2h_2} + c_{h_2k_2} < c_{h_1h_2} + c_{h_2k_1}.
\end{aligned}$$

Across all graphs, this condition contains some common terms. However, for graph G_1, this condition is more likely to be fulfilled than for the other graphs because the left-hand side includes fewer terms than the ones in the other graphs. For graphs G_2, G_3, and G_4, the conditions are more difficult to be satisfied due to the presence of the parking time and/or of the loading time.

Finally, by comparing columns $|V_D(t^1)|$ with $|T^2|$ of Table 2.6, we observe that with graphs G_1 and G_3, the number of opened parking spots is almost equal to the one of the second-level tours. Conversely, with graphs G_2 and G_4, more second-level tours originate from the same parking spot. This is a consequence of Property 2.4.1. To understand why, for each graph type, we make explicit the conditions stated in Property 2.4.1. Because $C1$ is the same for all graphs, we focus only on $C2$, that corresponds to the following:

$$\begin{aligned}
&\textit{Graph } G_1, C2: && c_{k_1H_0} + c_{H_{|H|}k_1} \leq c_{k_1k_2} + c_{k_2H_0} + c_{H_{|H|}k_2}.\\
&\textit{Graph } G_2, C2: && c_{k_1H_0} + c_{H_{|H|}k_1} \leq c_{k_1k_2} + \rho_{k_2} + c_{k_2H_0} + c_{H_{|H|}k_2}.\\
&\textit{Graph } G_3, C2: && c_{k_1H_0} + c_{H_{|H|}k_1} \leq c_{k_1k_2} + c_{k_2H_0} + c_{H_{|H|}k_2}.\\
&\textit{Graph } G_4, C2: && c_{k_1H_0} + c_{H_{|H|}k_1} \leq c_{k_1k_2} + \rho_{k_2} + c_{k_2H_0} + c_{H_{|H|}k_2}.
\end{aligned}$$

Across all graphs, condition $C2$ contains some common terms. However, for graphs G_1 and G_3, condition $C2$ is more difficult to fulfill than for the other graphs because the right-hand side includes fewer terms than the ones in the other graphs. This means that the solution has less second-level tours rooted at the same parking spot. For graphs G_2 and G_4, condition $C2$ is more likely to be satisfied due to the presence of the parking time only on the right-hand side of the inequality. Hence, these solutions contain more second-level tours rooted at the same parking spot.

The fact that our heuristic solutions present structures that are in line with those of optimal solutions is an additional indicator of the quality of our solutions.

2.6.3.3 Robustness of solutions under parking times fluctuations

In this section, we assess how robust solutions calculated assuming parking times equal for all parking spots are if fluctuations occur.

For this analysis, we execute ILS-ASTTRPSD on the DHL instances considering graphs G_3 and G_4 of Section 2.6.3.2 corresponding to the ASTTRPSD with parking times equal to $\rho = 0$ and $\rho = 11$, respectively. Moreover, we create two additional graphs, G_5 and G_6, corresponding to the ASTTRPSD with parking times equal to $\rho = 22$ and $\rho = 33$. In all graphs, the loading time is kept fixed to $\ell = 11$. After obtaining the solutions of ILS-ASTTRPSD on these graphs, we evaluate how these solutions perform under parking times fluctuations. We analyze ten fluctuation intervals from which the parking times are drawn uniformly random for each parking spot. These fluctuation intervals vary by parking time values and width.

Table 2.7 reports the results of this analysis. The first and the second column of the table contain the generated fluctuation intervals and the objective function values (averaged over 35 instances) obtained by solving the ASTTRPSD on DHL instances with parking times taken from the corresponding fluctuation intervals, respectively. Then, the first and the second-level tours of the ASTTRPSD solution returned by assuming the same parking time value (i.e., $\rho = 0$, $\rho = 11$, $\rho = 22$, and $\rho = 33$) for all parking spots are evaluated by considering the parking times from the fluctuation intervals. This new total travel time is compared to the one of the second column of the table. Hence, the last four columns of Table 2.7 report the average percentage loss by assuming the same parking time values for all parking spots when fluctuations instead occur. The values of the expected percentage loss $\mathbb{E}[\overline{loss}(\%)]$ (computed by considering equiprobable fluctuation intervals for ρ) suggest that assuming $\rho = 22$ represents the most robust strategy that allows to better hedge against parking time fluctuations. The least robust strategy is the one which assumes null parking times. By comparing the losses reported in the extreme cases in which the assumed parking time is $\rho = 0$ and the fluctuation interval is $[33, 44]$ and in which the assumed parking time is $\rho = 33$ and the fluctuation interval is $[0, 11]$, we observe that underestimating parking times is more costly than overestimating them.

2.7 Conclusion

Motivated by the challenging task of solving large-scale ASTTRPSD instances for postal deliveries, we propose a new metaheuristic, called ILS-ASTTRPSD. For the asymmetric instances from the literature, ILS-ASTTRPSD provides high-quality solutions in short runtimes. For DHL instances, ILS-ASTTRPSD is always able to reduce total travel times compared to the solutions provided by DHL. This result is due to a different solution structure in which mail carriers spend less time for driving and walking and stop at fewer parking spots. This also contributes to reduce the stress originating from parking activities. Our solutions also exhibit multiple second-level tours rooted at the same parking spot, which is convenient under particular conditions related to the road network structure and to single households with high demands. In our solutions, mail carriers serve more households in each second-level tour without causing longer walking times. Through additional computational experiments, we

Fluctuation intervals	Objective function	Assumed ρ 0	 11	 22	 33
$[0, 11]$	9857.88	1.94	2.05	3.51	5.17
$[0, 22]$	10048.28	4.37	2.93	3.68	4.88
$[6, 16]$	10225.10	2.59	1.15	1.89	3.09
$[11, 22]$	10487.40	4.05	1.27	1.33	2.11
$[11, 33]$	10604.41	7.04	2.83	2.23	2.59
$[17, 27]$	10744.69	5.67	1.48	0.90	1.28
$[22, 33]$	10961.58	7.42	2.01	0.81	0.80
$[22, 44]$	11049.80	10.54	3.76	1.94	1.54
$[28, 38]$	11158.64	9.48	2.74	0.95	0.57
$[33, 44]$	11321.89	11.63	3.72	1.34	0.60
$\mathbb{E}[\overline{loss}(\%)]$		6.47	2.40	1.86	2.26

Table 2.7: Comparison of the average percentage loss (over 35 instances) obtained when evaluating the solution obtained by assuming a parking time value under different fluctuation intervals.

evaluate the impact of not considering parking and loading times. While ignoring both these time components results in higher travel times, the drawbacks of ignoring loading times are more limited than those of ignoring parking times. The results of our computational experiments are in line with the two properties necessary for multiple second-level tours being rooted at the same parking spot, and for the ASTTRPSD solution to correspond to a TSP solution. Finally, we evaluate how robust the solutions are when we assume that all parking spots have the same parking time but fluctuations occur.

Acknowledgments

We thank Sara Reed, Ann Campbell, and Barrett Thomas for providing the detailed solutions of the experiments contained in Reed et al., 2024. We also thank the two anonymous referees for providing valuable comments that helped to improve the paper.

References

L. Accorsi and D. Vigo (2020). “A hybrid metaheuristic for single truck and trailer routing problems”. In: *Transportation Science* 54.5, pp. 1351–1371. DOI: 10.1287/trsc.2019.0943.

J. Allen, M. Piecyk, M. Piotrowska, F. McLeod, T. Cherrett, K. Ghali, T. Nguyen, T. Bektas, O. Bates, A. Friday, et al. (2018). "Understanding the impact of e-commerce on last-mile light goods vehicle activity in urban areas: The case of London". In: *Transportation Research Part D: Transport and Environment* 61, pp. 325–338. DOI: 10.1016/j.trd.2017.07.020.

F. Arnold and K. Sörensen (2021). "A progressive filtering heuristic for the location-routing problem and variants". In: *Computers & Operations Research* 129, pp. 105–166. DOI: 10.1016/j.cor.2020.105166.

E. Bartolini and M. Schneider (2020). "A two-commodity flow formulation for the capacitated truck-and-trailer routing problem". In: *Discrete Applied Mathematics* 275, pp. 3–18. DOI: 10.1016/j.dam.2018.07.033.

J. M. Belenguer, E. Benavent, A. Martínez, C. Prins, C. Prodhon, and J. G. Villegas (2016). "A branch-and-cut algorithm for the single truck and trailer routing problem with satellite depots". In: *Transportation Science* 50.2, pp. 735–749. DOI: 10.1287/trsc.2014.0571.

C. Bode (2013). *Lower bounds for park and loop delivery problems.* Tech. rep. Technical Report LM-2013-02, Chair of Logistics Management, Mainz School of Management and Economics, Johannes Gutenberg University, Mainz, Germany. URL: http://logistik.bwl.uni-mainz.de/158.php.

L. Bodin and L. Levy (2000). "Scheduling of local delivery carrier routes for the united states postal service". In: *Arc Routing.* Springer, pp. 419–442. DOI: 10.1007/978-1-4615-4495-1_11.

N. Cabrera, J.-F. Cordeau, and J. E. Mendoza (2022). "The doubly open park-and-loop routing problem". In: *Computers & Operations Research* 143, pp. 105–761. DOI: 10.1016/j.cor.2022.105761.

R. Cuda, G. Guastaroba, and M. G. Speranza (2015). "A survey on two-echelon routing problems". In: *Computers & Operations Research* 55, pp. 185–199. DOI: 10.1016/j.cor.2014.06.008.

M. Drexl (2012). "Synchronization in vehicle routing—a survey of VRPs with multiple synchronization constraints". In: *Transportation Science* 46.3, pp. 297–316. DOI: 10.1287/trsc.1110.0400.

J. W. Escobar, R. Linfati, and P. Toth (2014). "A hybrid granular tabu search algorithm for the multi-depot vehicle routing problem". In: *Journal of the Operational Research Society* 65.1, pp. 37–48. DOI: 10.1057/jors.2013.102.

M. Fischetti, J. J. Salazar González, and P. Toth (1997). "A branch-and-cut algorithm for the symmetric generalized traveling salesman problem". In: *Operations Research* 45.3, pp. 378–394. DOI: 10.1287/opre.45.3.378.

D. Goeke (2019). "Granular tabu search for the pickup and delivery problem with time windows and electric vehicles". In: *European Journal of Operational Research* 278.3, pp. 821–836. DOI: 10.1016/j.ejor.2019.05.010.

E. Gussmagg-Pfliegl, F. Tricoire, K. F. Doerner, and R. F. Hartl (2011). “Mail-delivery problems with park-and-loop tours: a heuristic approach”. In: *Proceedings of the ORP3 Meeting, Cadiz*. Universidad de Cádiz, pp. 77–81.

T. Le Colleter, D. Dumez, F. Lehuédé, and O. Péton (2023). “Small and Large Neighborhood Search for the Park-and-Loop Routing Problem with Parking Selection”. In: *European Journal of Operational Research* 308.3, pp. 1233–1248. DOI: 10.1016/j.ejor.2023.01.007.

L. Levy and L. Bodin (1989). “The arc oriented location routing problem”. In: *INFOR: Information Systems and Operational Research* 27.1, pp. 74–94. DOI: 10.1080/03155986.1989.11732083.

M. López-Ibáñez, J. Dubois-Lacoste, L. Cáceres Pérez, M. Birattari, and T. Stützle (2016). “The irace package: Iterated racing for automatic algorithm configuration”. In: *Operations Research Perspectives* 3, pp. 43–58. DOI: 10.1016/j.orp.2016.09.002.

H. R. Lourenço, O. C. Martin, and T. Stützle (2003). “Iterated Local Search”. In: *Handbook of Metaheuristics*. Springer US, pp. 320–353. DOI: 10.1007/0-306-48056-5_11.

A. Martinez-Sykora, F. McLeod, C. Lamas-Fernandez, T. Bektaş, T. Cherrett, and J. Allen (2020). “Optimised solutions to the last-mile delivery problem in London using a combination of walking and driving”. In: *Annals of Operations Research* 295.2, pp. 645–693. DOI: 10.1007/s10479-020-03781-8.

G. Nagy and S. Salhi (2007). “Location-routing: Issues, models and methods”. In: *European Journal of Operational Research* 177.2, pp. 649–672. DOI: 10.1016/j.ejor.2006.04.004.

PassMark Software (2022). *Professional CPU benchmarks*. URL: https://www.cpubenchmark.net/singleThread.html (visited on 06/23/2022).

C. Prins, C. Prodhon, A. Ruiz, P. Soriano, and R. Wolfler Calvo (2007). “Solving the Capacitated Location-Routing Problem by a Cooperative Lagrangean Relaxation-Granular Tabu Search Heuristic”. In: *Transportation Science* 41.4, pp. 470–483. DOI: 10.1287/trsc.1060.0187.

C. Prodhon and C. Prins (2014). “A survey of recent research on location-routing problems”. In: *European Journal of Operational Research* 238.1, pp. 1–17. DOI: 10.1016/j.ejor.2014.01.005.

S. Reed, A. M. Campbell, and B. W. Thomas (2024). *Does Parking Matter? The Impact of Search Time for Parking on Last-Mile Delivery Optimization*. Tech. rep. DOI: 10.1016/j.tre.2023.103391.

G. Reinelt (1991). “TSPLIB–A traveling salesman problem library”. In: *ORSA Journal on Computing* 3.4, pp. 376–384. DOI: 10.1287/ijoc.3.4.376.

M. Schiffer, M. Schneider, G. Walther, and G. Laporte (2019). "Vehicle routing and location routing with intermediate stops: A review". In: *Transportation Science* 53.2, pp. 319–343. DOI: `10.1287/trsc.2018.0836`.

P. Toth and D. Vigo (2003). "The granular tabu search and its application to the vehicle-routing problem". In: *INFORMS Journal on Computing* 15.4, pp. 333–346. DOI: `10.1287/ijoc.15.4.333.24890`.

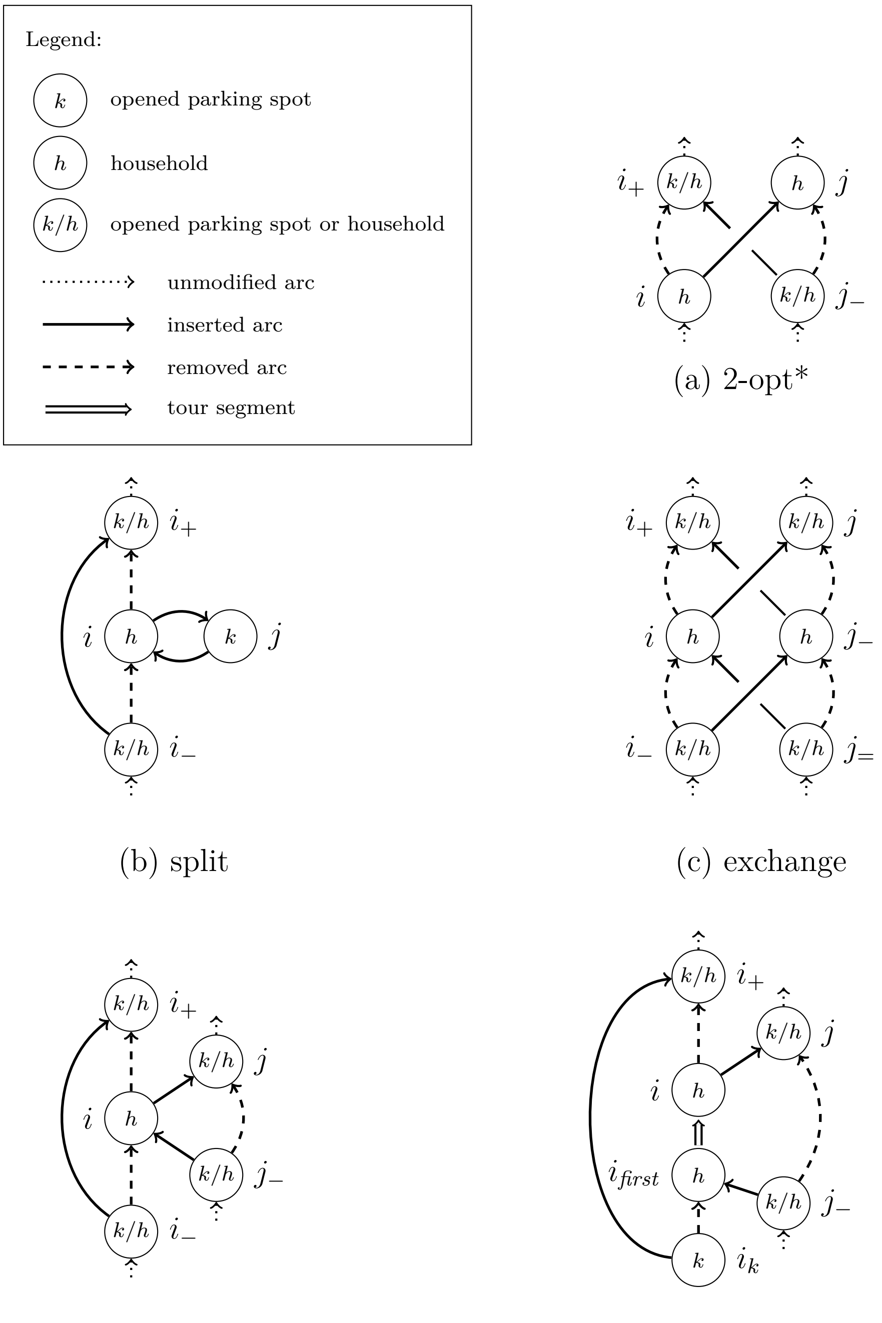
J. G. Villegas, C. Prins, C. Prodhon, A. L. Medaglia, and N. Velasco (2010). "GRASP/VND and multi-start evolutionary local search for the single truck and trailer routing problem with satellite depots". In: *Engineering Applications of Artificial Intelligence* 23.5, pp. 780–794. DOI: `10.1016/j.engappai.2010.01.013`.

2.8 Appendix

A Comparison on symmetric STTRPSD instances

In this section, we report the results obtained by ILS-ASTTRPSD on symmetric STTRPSD instances from the literature. Due to the different structure of the literature instances, we decreased the sparsification intensity for households and parking spots to $\kappa_c = 0.10$ and $\kappa_k = 0.10$, and increased the number of iterations without improvement η to 3000. Moreover, every three iterations of ILS-ASTTRPSD without improvement, we perturb the best found solution in the last three iterations. Table 2.8 shows the comparison of our results to the state-of-the-art approaches in the literature, i.e.: the hybrid metaheuristic of Accorsi and Vigo, 2020 (AVXS), the multi-start iterated local search of Villegas et al., 2010 (MS-ILS), and the progressive filtering heuristic of Arnold and Sörensen, 2021 (PF). Because the algorithm of Accorsi and Vigo, 2020 is available online, we report the runtimes of AVXS executed on our machine. To allow for a fair runtime comparison with MS-ILS and PF, we used the CPU single-thread rating defined by PassMark Software, 2022. The website assigns a score of 1483 to our CPU, and a score of 574 and 2233 to the CPU used by Villegas et al., 2010 and Arnold and Sörensen, 2021, respectively. Thus, we convert the runtimes of Villegas et al., 2010 by dividing them by 2.58 and by multiplying the runtimes of Arnold and Sörensen, 2021 by 1.51. We are aware that a completely precise comparison of runtimes is not possible due to the different programming languages and other factors. However, we follow the procedure adopted by the literature, and we compare with respect to the CPU.

The results show that ILS-ASTTRPSD returns good-quality solutions with a best and an average gap to the BKS significantly below 1%. Also on the symmetric STTRPSD instances, the low standard deviation values suggest that ILS-ASTTRPSD is robust. Our algorithm is competitive to the MS-ILS of Villegas et al., 2010 and to the PF of Arnold and Sörensen, 2021 especially for the large-sized instances. For some of these instances, ILS-ASTTRPSD reaches solutions of a better quality in the best and/or in the average run. The average runtimes of ILS-ASTTRPSD are larger than the ones of the other state-of-the-art heuristics. However, because DHL only plans postal delivery tours at the tactical level, the runtimes are still acceptable.



(a) 2-opt*

(b) split

(c) exchange

(d) relocate-1

(e) relocate-b

Figure 2.6: Neighborhood operators of ILS-ASTTRPSD. The generator arc is denoted by (i, j). The predecessor and successor of i are denoted i_- and i_+, respectively. The first visited household in a second-level tour is denoted by i_{first}. For a more concise representation, in Figure 3(b) and 3(e), the arcs that connect the opened parking spots to the first-level tour are not depicted.

(a) Walking network G_w.

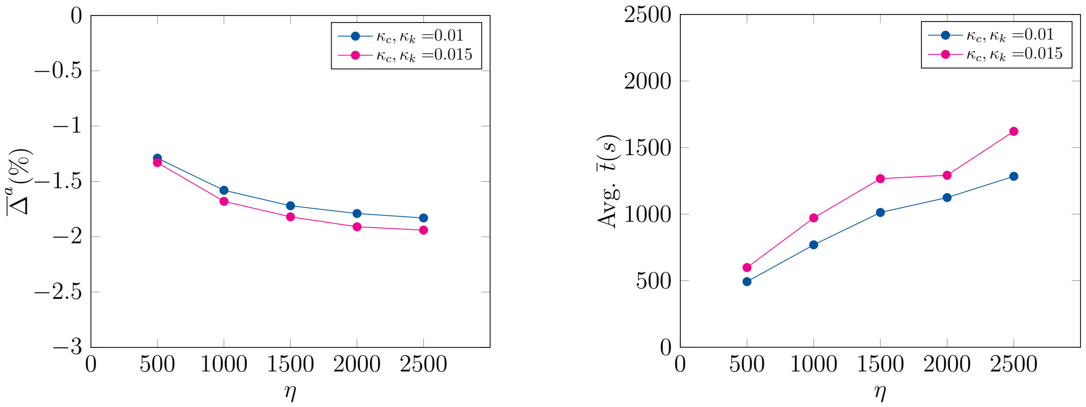
(b) Driving network G_d.

Figure 2.7: Example of walking and driving network.

(a) Vertices representing the parking spots, walking portals, and houses in V_w.

(b) Vertices representing the parking spots in V_d.

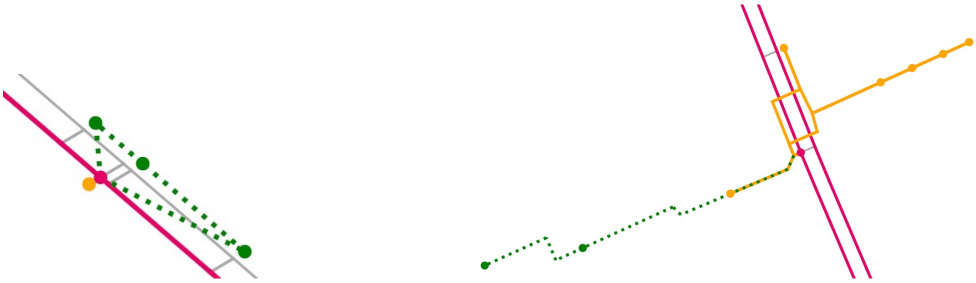
Figure 2.8: Example of vertices in V_w and V_d.



(a) Solution quality.

(b) Runtime.

Figure 2.9: Parameter tuning experiment results.



(a) Instance 10.

(b) Instance 6.

Figure 2.10: Examples of multiple second-level tours from the same parking spot for two instances.

		AVXS			MS-ILS			PF		ILS-ASTTRPSD			
Instance	BKS	$\Delta^b(\%)$	$\Delta^a(\%)$	$\bar{t}(s)$	$\Delta^b(\%)$	$\Delta^a(\%)$	$\bar{t}(s)$	$\Delta(\%)$	$t(s)$	$\Delta^b(\%)$	$\Delta^a(\%)$	$\Delta^{std\_dev}(\pm)$	$\bar{t}(s)$
STTRP-25-5-1-c	405.46	0.00	0.00	1.00	0.00	0.00	4.42	0.00	60.40	0.00	0.00	0.00	11.28
STTRP-25-5-1-rd	584.03	0.00	0.00	1.00	0.00	0.00	4.88	0.00	258.21	0.00	0.00	0.00	10.18
STTRP-25-5-2-c	374.79	0.00	0.00	1.00	0.00	0.00	4.42	0.00	253.68	0.00	0.00	0.01	34.81
STTRP-25-5-2-rd	508.48	0.00	0.00	1.00	0.00	0.00	3.95	0.00	277.84	0.00	0.00	0.00	32.03
STTRP-25-10-1-c	386.45	0.00	0.00	1.00	0.00	0.00	5.12	0.00	134.39	0.00	0.00	0.00	12.0
STTRP-25-10-1-rd	573.96	0.00	0.00	1.00	0.00	0.00	5.35	0.00	66.44	0.00	0.00	0.00	11.25
STTRP-25-10-2-c	380.86	0.00	0.00	1.00	0.00	0.00	4.88	0.00	140.43	0.00	0.12	0.25	22.93
STTRP-25-10-2-rd	506.37	0.00	0.00	1.00	0.00	0.00	4.88	0.02	99.66	0.00	0.01	0.01	19.66
STTRP-50-5-1-c	583.07	0.00	0.00	1.00	0.00	0.00	27.44	0.00	78.52	0.00	0.00	0.00	50.65
STTRP-50-5-1-rd	870.51	0.00	0.00	1.00	0.00	0.00	23.72	0.00	75.50	0.00	0.00	0.00	53.73
STTRP-50-5-2-c	516.98	0.00	0.00	1.00	0.00	0.00	22.09	0.00	327.67	0.00	0.09	0.13	196.14
STTRP-50-5-2-rd	766.03	0.00	0.00	1.00	0.00	0.00	22.79	0.00	146.47	0.00	1.16	0.81	105.65
STTRP-50-10-1-c	387.83	0.00	0.00	1.00	0.00	0.00	31.86	0.32	89.09	0.34	0.47	0.07	36.04
STTRP-50-10-1-rd	811.28	0.00	0.00	1.90	0.00	0.00	27.91	0.00	83.05	0.00	0.06	0.09	29.98
STTRP-50-10-2-c	367.01	0.00	0.00	1.00	0.00	0.00	29.07	0.00	161.57	0.00	0.00	0.00	110.40
STTRP-50-10-2-rd	731.53	0.00	0.00	2.20	0.00	0.00	24.42	0.00	172.14	1.61	1.82	0.21	141.31
STTRP-100-10-1-c	614.02	0.00	0.00	9.50	0.00	0.06	134.88	0.00	107.21	0.00	0.16	0.33	175.58
STTRP-100-10-1-rd	1271.78	0.00	0.00	11.90	0.65	0.87	115.35	0.00	129.86	0.95	1.55	0.37	212.44
STTRP-100-10-2-c	547.44	0.00	0.00	6.00	0.00	0.02	150.93	0.00	209.89	0.00	0.05	0.08	202.58
STTRP-100-10-2-rd	1097.28	0.00	0.00	9.00	0.00	0.06	100.93	0.00	203.85	0.00	0.60	0.38	404.61
STTRP-100-20-1-c	642.61	0.00	0.00	11.20	0.00	0.00	112.09	0.36	123.82	0.00	0.13	0.14	254.75
STTRP-100-20-1-rd	1143.10	0.00	0.00	9.70	0.00	0.31	116.74	0.39	143.45	0.16	0.36	0.13	104.56
STTRP-100-20-2-c	581.56	0.00	0.00	7.80	0.00	0.11	139.53	2.26	205.36	0.00	0.63	0.35	398.84
STTRP-100-20-2-rd	1060.75	0.01	0.24	15.00	0.00	0.20	114.19	0.60	218.95	0.28	1.08	0.39	270.06
STTRP-200-10-1-c	819.96	0.00	0.00	49.30	0.31	1.03	447.44	0.20	226.50	0.53	0.79	0.13	1249.40
STTRP-200-10-1-rd	1755.41	0.00	0.00	66.20	0.45	1.59	399.77	0.21	285.39	0.84	2.05	0.56	1301.08
STTRP-200-10-2-c	710.70	0.00	0.06	44.80	0.51	1.31	452.79	0.22	295.96	1.17	1.59	0.25	1248.26
STTRP-200-10-2-rd	1445.94	0.00	0.00	40.70	0.00	0.84	360.00	0.00	681.01	1.78	2.33	0.48	1462.46
STTRP-200-20-1-c	907.17	0.00	0.00	51.30	0.25	0.70	532.79	2.05	382.03	0.08	0.41	0.29	740.40
STTRP-200-20-1-rd	1610.62	0.00	0.00	53.70	0.22	1.30	496.98	2.58	311.06	0.72	1.57	0.92	1028.85
STTRP-200-20-2-c	814.42	0.00	0.01	38.30	0.77	0.95	567.67	0.18	608.53	0.04	0.72	0.36	1710.74
STTRP-200-20-2-rd	1413.32	0.00	0.00	54.10	0.00	0.81	468.37	1.45	552.66	0.74	0.98	0.22	1390.63
Avg.		0.00	0.01	15.52	0.10	0.32	154.93	0.34	222.21	0.29	0.58	0.22	407.29

Table 2.8: Comparison of the solutions of the heuristics from the literature to the ones of ILS-ASTTRPSD for symmetric instances.

B Detailed results for the impact of ignoring parking and loading times

B.1 Detailed results for graph G_1

Instance	$\lvert V_C\rvert$	$\lvert V_D\rvert$	$\sum_{i\in V_C} q_i$	$\overline{c_{ij}}$	$c(t^1)$	$\sum_{t^2\in T^2} c(t^2)$	$\lvert V_D(t^1)\rvert$	$\lvert T^2\rvert$	$\overline{\lvert V_C(t^2)\rvert}$
1	382	746	646	417.16	3186.72	5471.77	174	174	2.20
2	465	852	713	556.17	3972.45	5156.11	150	150	3.10
3	424	778	703	399.71	2983.57	4839.58	178	178	2.38
4	433	742	711	333.05	2456.27	4381.36	136	136	3.18
5	403	698	826	464.93	2863.95	4414.85	197	198	2.04
6	378	585	887	527.94	2471.13	4410.89	173	175	2.16
7	361	643	884	340.49	1868.59	4294.22	139	139	2.60
8	452	872	791	497.59	2369.00	5215.32	197	197	2.29
9	250	412	483	614.93	2496.43	3370.08	98	98	2.55
10	298	481	1092	293.17	2401.62	3892.34	156	158	1.89
11	389	758	756	504.81	2476.11	4872.41	185	185	2.10
12	374	729	628	456.74	3566.43	4610.93	174	174	2.15
13	418	762	770	433.92	2684.91	4602.22	160	160	2.61
14	360	697	653	522.93	3371.13	3969.06	181	181	1.99
15	441	854	700	537.55	3510.49	5360.06	218	218	2.02
16	416	793	621	834.63	4460.93	5492.73	200	200	2.08
17	427	812	634	375.59	2961.07	4734.62	163	163	2.62
18	452	882	725	600.57	3453.10	5198.55	204	204	2.22
19	414	767	654	409.54	5738.47	5410.20	162	163	2.54
20	378	720	696	444.09	3600.95	4102.34	198	198	1.91
21	458	772	627	475.02	3979.46	5484.10	140	140	3.27
22	447	770	673	327.47	2988.59	6197.99	118	118	3.79
23	430	815	641	353.29	3534.22	5412.74	185	185	2.32
24	334	639	606	647.47	4367.17	4251.95	189	189	1.77
25	342	640	601	375.88	3723.74	4033.70	148	149	2.30
26	419	814	740	400.45	4010.82	4798.18	223	223	1.88
27	368	700	568	930.46	4965.45	4199.24	190	190	1.94
28	395	762	571	385.66	4845.79	4997.82	193	193	2.05
29	384	746	745	321.71	3941.85	4540.14	132	132	2.91
30	369	722	676	543.72	4261.53	4386.76	155	155	2.38
31	410	728	736	413.50	2329.96	5035.29	179	179	2.29
32	429	837	788	696.90	3900.44	4879.95	170	171	2.51
33	245	477	453	1629.02	6321.45	3433.80	172	172	1.42
34	353	695	735	359.09	3762.65	4456.70	153	153	2.31
35	341	644	498	510.62	4136.67	3522.25	126	127	2.69
Avg.	389.69	724.11	692.31	512.45	3541.80	4669.44	169.03	169.29	2.36

Table 2.9: Statistics of the best run of ILS-ASTTRPSD for the DHL instances with graph G_1.

B.2 Detailed results for graph G_2

Instance	$\lvert V_C \rvert$	$\lvert V_D \rvert$	$\sum_{i \in V_C} q_i$	$\overline{c_{ij}}$	$c(t^1)$	$\sum_{t^2 \in T^2} c(t^2)$	$\lvert V_D(t^1) \rvert$	$\lvert T^2 \rvert$	$\overline{\lvert V_C(t^2) \rvert}$
1	382	746	646	417.16	3954.19	6114.57	87	88	4.34
2	465	852	713	556.17	4452.55	5881.19	69	72	6.46
3	424	778	703	399.71	3135.63	6161.82	65	68	6.24
4	433	742	711	333.05	2973.50	4925.16	62	62	6.98
5	403	698	826	464.93	3184.15	5594.19	74	76	5.30
6	378	585	887	527.94	3028.39	5299.49	79	86	4.40
7	361	643	884	340.49	2573.98	4777.37	80	83	4.35
8	452	872	791	497.59	3084.00	6023.88	99	101	4.48
9	250	412	483	614.93	3192.46	3540.12	66	66	3.79
10	298	481	1092	293.17	3076.58	4638.54	88	100	2.98
11	389	758	756	504.81	3021.81	5824.76	79	82	4.74
12	374	729	628	456.74	4318.65	5358.94	86	88	4.25
13	418	762	770	433.92	3321.60	5330.96	75	77	5.43
14	360	697	653	522.93	4006.95	4904.38	80	82	4.39
15	441	854	700	537.55	4175.47	6496.90	93	93	4.74
16	416	793	621	834.63	5160.32	6486.38	79	81	5.14
17	427	812	634	375.59	3184.06	5573.94	44	45	9.49
18	452	882	725	600.57	4095.58	6157.79	74	74	6.11
19	414	767	654	409.54	6380.72	6128.33	81	82	5.05
20	378	720	696	444.09	4041.28	5304.00	75	77	4.91
21	458	772	627	475.02	4489.09	6181.17	79	82	5.59
22	447	770	673	327.47	3311.86	6887.50	48	50	8.94
23	430	815	641	353.29	3456.05	6863.50	66	66	6.52
24	334	639	606	647.47	5359.55	4790.01	105	106	3.15
25	342	640	601	375.88	4157.09	4963.29	62	65	5.26
26	419	814	740	400.45	4517.72	6031.93	87	88	4.76
27	368	700	568	930.46	5171.84	5472.63	67	68	5.41
28	395	762	571	385.66	4671.00	6803.53	62	64	6.17
29	384	746	745	321.71	4314.25	5224.66	54	56	6.86
30	369	722	676	543.72	4624.25	5331.46	57	60	6.15
31	410	728	736	413.50	2899.62	5853.49	77	79	5.19
32	429	837	788	696.90	4299.04	5924.93	78	80	5.36
33	245	477	453	1629.02	7211.50	4232.91	124	124	1.98
34	353	695	735	359.09	4294.94	4956.69	83	85	4.15
35	341	644	498	510.62	4950.03	4029.86	92	92	3.71
Avg.	389.69	724.11	692.31	512.45	4059.71	5544.87	76.46	78.51	5.22

Table 2.10: Statistics of the best run of ILS-ASTTRPSD for the DHL instances with graph G_2.

B.3 Detailed results for graph G_3

Instance	$\lvert V_C\rvert$	$\lvert V_D\rvert$	$\sum_{i\in V_C} q_i$	$\overline{c_{ij}}$	$c(t^1)$	$\sum_{t^2\in T^2} c(t^2)$	$\lvert V_D(t^1)\rvert$	$\lvert T^2\rvert$	$\overline{\lvert V_C(t^2)\rvert}$
1	382	746	646	417.16	3039.75	7076.93	84	84	4.55
2	465	852	713	556.17	3616.64	6692.90	71	72	6.46
3	424	778	703	399.71	2513.42	6769.97	76	76	5.58
4	433	742	711	333.05	2290.97	5593.20	63	63	6.87
5	403	698	826	464.93	2263.03	6517.12	72	72	5.60
6	378	585	887	527.94	2239.93	6112.51	88	90	4.20
7	361	643	884	340.49	1667.22	5719.82	82	82	4.40
8	452	872	791	497.59	1946.48	7147.32	98	99	4.57
9	250	412	483	614.93	2466.46	4275.05	64	64	3.91
10	298	481	1092	293.17	2223.41	5566.27	103	104	2.87
11	389	758	756	504.81	2202.24	6651.75	85	85	4.58
12	374	729	628	456.74	3316.72	6363.21	86	86	4.35
13	418	762	770	433.92	2580.43	6130.92	82	82	5.10
14	360	697	653	522.93	3045.48	5883.47	76	76	4.74
15	441	854	700	537.55	3075.68	7659.97	82	82	5.38
16	416	793	621	834.63	4264.82	7428.49	77	77	5.40
17	427	812	634	375.59	2719.15	6092.18	43	43	9.93
18	452	882	725	600.57	3281.58	6978.94	79	79	5.72
19	414	767	654	409.54	5646.67	6916.10	87	88	4.70
20	378	720	696	444.09	3174.08	6171.94	74	74	5.11
21	458	772	627	475.02	3603.80	7156.08	74	76	6.03
22	447	770	673	327.47	2877.46	7375.79	58	58	7.71
23	430	815	641	353.29	2908.14	7360.18	75	76	5.66
24	334	639	606	647.47	4111.37	6046.87	103	103	3.24
25	342	640	601	375.88	3483.39	5637.83	61	62	5.52
26	419	814	740	400.45	3635.39	6911.00	93	93	4.51
27	368	700	568	930.46	4464.12	6187.95	71	71	5.18
28	395	762	571	385.66	4083.96	7365.45	74	74	5.34
29	384	746	745	321.71	3798.01	5758.49	61	62	6.19
30	369	722	676	543.72	4031.77	5920.57	60	60	6.15
31	410	728	736	413.50	2058.83	6717.91	79	79	5.19
32	429	837	788	696.90	3488.66	6759.55	81	83	5.17
33	245	477	453	1629.02	5909.94	5558.96	126	126	1.94
34	353	695	735	359.09	3291.54	6093.44	72	72	4.90
35	341	644	498	510.62	3784.39	5231.62	87	87	3.92
Avg.	389.69	724.11	692.31	512.45	3231.57	6395.14	78.49	78.86	5.16

Table 2.11: Statistics of the best run of ILS-ASTTRPSD for the DHL instances with graph G_3.

B.4 Detailed results for graph G_4

Instance	$\lvert V_C \rvert$	$\lvert V_D \rvert$	$\sum_{i \in V_C} q_i$	$\overline{c_{ij}}$	$c(t^1)$	$\sum_{t^2 \in T^2} c(t^2)$	$\lvert V_D(t^1) \rvert$	$\lvert T^2 \rvert$	$\overline{\lvert V_C(t^2) \rvert}$
1	382	746	646	417.16	3295.01	7606.96	54	55	6.95
2	465	852	713	556.17	4130.20	6862.38	51	54	8.61
3	424	778	703	399.71	2758.07	7227.29	46	49	8.65
4	433	742	711	333.05	2793.95	5736.55	44	45	9.62
5	403	698	826	464.93	2557.62	6920.93	50	57	7.07
6	378	585	887	527.94	2794.63	6368.91	59	65	5.82
7	361	643	884	340.49	2294.81	5890.15	57	60	6.02
8	452	872	791	497.59	2531.29	7480.28	61	64	7.06
9	250	412	483	614.93	2828.57	4499.06	47	49	5.10
10	298	481	1092	293.17	2733.36	6004.36	65	75	3.97
11	389	758	756	504.81	2654.32	6951.82	52	54	7.20
12	374	729	628	456.74	3928.47	6523.39	56	58	6.45
13	418	762	770	433.92	3026.20	6363.56	53	54	7.74
14	360	697	653	522.93	3452.48	6110.31	48	51	7.06
15	441	854	700	537.55	3421.94	8018.91	46	47	9.38
16	416	793	621	834.63	4780.26	7602.97	51	53	7.85
17	427	812	634	375.59	2967.57	6250.93	33	34	12.56
18	452	882	725	600.57	3469.33	7393.77	39	42	10.76
19	414	767	654	409.54	6001.20	7294.00	47	50	8.28
20	378	720	696	444.09	3684.95	6334.85	48	51	7.41
21	458	772	627	475.02	3809.20	7541.52	35	38	12.05
22	447	770	673	327.47	3157.37	7487.01	40	41	10.90
23	430	815	641	353.29	3244.84	7750.61	44	45	9.56
24	334	639	606	647.47	4670.41	6435.32	70	70	4.77
25	342	640	601	375.88	3640.29	6051.64	32	36	9.50
26	419	814	740	400.45	4042.97	7234.58	47	49	8.55
27	368	700	568	930.46	4974.99	6358.32	47	48	7.67
28	395	762	571	385.66	4240.42	7832.63	41	42	9.40
29	384	746	745	321.71	4091.35	5987.94	40	44	8.73
30	369	722	676	543.72	4520.78	6025.44	45	45	8.20
31	410	728	736	413.50	2480.40	7026.45	58	59	6.95
32	429	837	788	696.90	3908.49	7180.35	63	69	6.22
33	245	477	453	1629.02	6772.28	5881.35	103	103	2.38
34	353	695	735	359.09	3702.97	6474.28	51	53	6.66
35	341	644	498	510.62	4378.48	5486.61	62	63	5.41
Avg.	389.69	724.11	692.31	512.45	3649.70	6691.30	51.00	53.49	7.73

Table 2.12: Statistics of the best run of ILS-ASTTRPSD for the DHL instances with graph G_4.

Chapter 3

The vehicle routing problem with operation simplification constraints

Publication status: A. Theiß, R. Cavagnini, and M. Schneider (2025). *The vehicle routing problem with depot operation constraints at DHL Group.* Working Paper. Chair of Computational Logistics, RWTH Aachen University, Germany

Abstract: In this paper, we study the vehicle routing problem with depot operation constraints (VRPDOC), a new problem arising in the context of postal deliveries at DHL Group. DHL clusters the street segments of a city into districts, and, for each of them, it determines a route traveled by a mail carrier departing from a depot. DHL presorts letters on preparation tables that are located at the depot. The mail carriers have to visit the preparation tables and collect their letters before starting their routes. The sorting of the letters is fixed and cannot be modified. Because the number of letters varies daily, a different number of mail carriers may be needed for delivery depending on the day of the week or the season of the year. The goal of the VRPDOC is to determine the routes for each mail carrier so that the total travel time is minimized. Upper and lower bounds on the duration of each route must be respected, and the letter collection at preparation tables must be kept as simple as possible. We formulate the VRPDOC as an integer program, and we improve the formulation using several preprocessing techniques and valid inequalities. We also derive theoretical results on the complexity of finding a feasible solution for the VRPDOC. Because commercial solvers cannot even find feasible solutions for realistically-sized instances of VRPDOC in reasonable runtimes, we develop an iterated local search (ILS) to solve the VRPDOC. The experiments show that the ILS finds optimal solutions on small-scale instances, and it significantly improves the solutions computed by DHL for real-world instances. More precisely, our solutions do not only decrease the total travel time compared to the DHL solutions, but they simplify letter collection at preparation tables at the same time, leading to depot operations that are fast, generate less confusion and mistakes, and allow for faster learning processes. Additional experiments show that ignoring depot operation constraints leads to shorter travel times but severely complicates the letter collection

at the preparation tables.

Contribution of the author: The authors shared efforts in the conceptual development of the research goals, the literature review, the mathematical formulations of the problem, the design of the methodology and implementation of the algorithm, the computational experiments and result analysis, and in writing the paper.

3.1 Introduction

Cost-efficient routing of mail carriers is of major importance for companies like our industry partner DHL Group. On average, DHL delivers 57 million letters every day in Germany. To keep the problem of routing a given set of mail carriers in a delivery area as simple as possible, DHL has already clustered single households into street segments. For each of these street segments, a so-called service time is computed, which corresponds to the total time a mail carrier needs to serve the households within the street segment, i.e., parking their vehicle (bike/car), walking to the letter boxes of the households, delivering the letters, and returning to the vehicle. With this simplification, the mail delivery problem can be modeled as a vehicle routing problem (VRP) with route duration constraints, in which the mail carriers are the vehicles, the street segments are the customers, and the objective is to minimize the total travel time.

Before starting their route, the mail carriers have to collect the letters of the street segments assigned to their route from so-called preparation tables located at the depot (in the remainder of the paper we will simply refer to them as tables). To avoid wasting time on finding the correct letters while driving the route, mail carriers already collect the letters in the order in which they are delivered on their route. Each table consists of shelves that are divided into sections. Each shelf section is labeled with the names and numbers of the delivery addresses. The delivery addresses are sorted according to their visiting order in the corresponding street segment. An example of a table is shown in Figure 3.1. On the right side of the figure, we see the details of twelve sections from the third level of the shelf, covering three street segments. All households labeled with the same color form one street segment. A street segment can consist either only of households on one side of the road (blue and green segment in the example) or of households on both sides of the road (orange segment). The labels cannot be changed on a daily basis, and, thus, the order in which the letters are sorted on the tables is fixed.

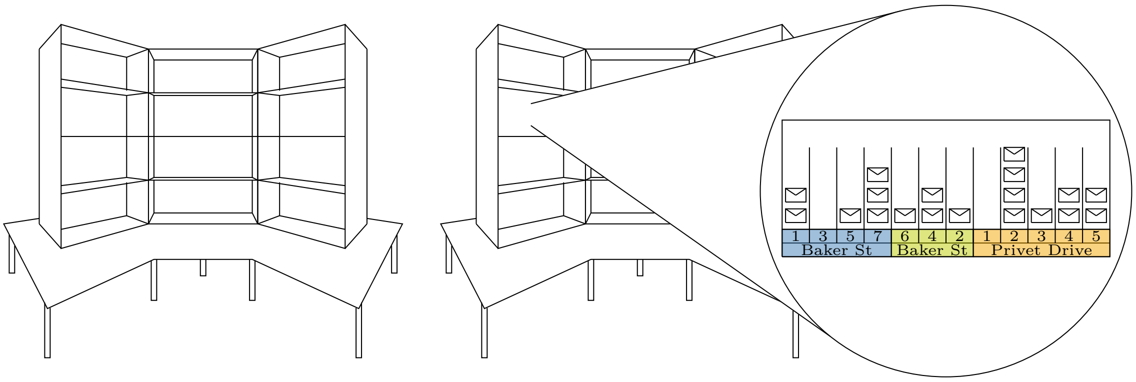
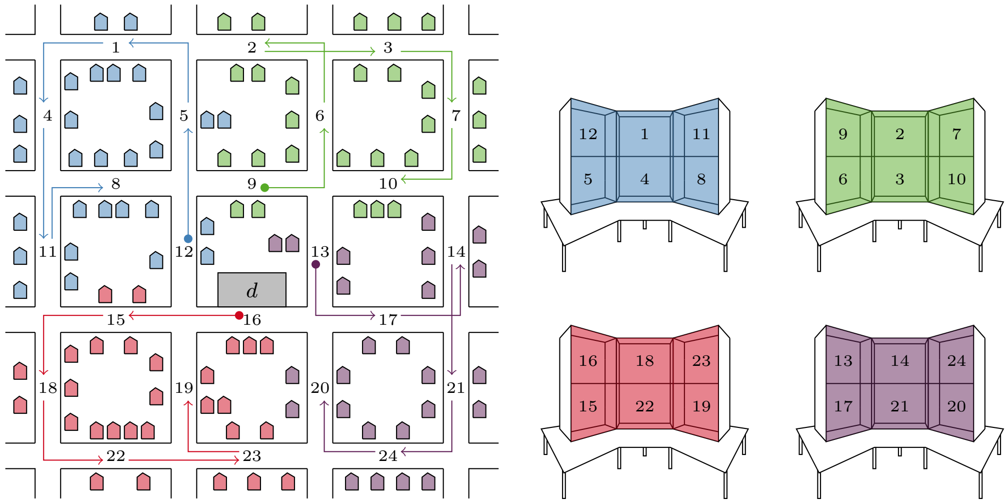


Figure 3.1: Example of a table and detailed view of a shelf.

Figure 3.2 shows an example of a delivery area and the corresponding tables at the depot. The delivery area in Figure 3.2(a) consists of 24 street segments, and the depot is depicted as gray rectangle. To serve the delivery area, four mail carriers are needed. The assignment of street segments to mail carriers (consequently also the assignment of households to mail carriers) is indicated by the coloring of the households. The routes of the mail carriers are given by the colored arcs. To keep the figure as simple as possible, we do not show the arcs from the depot to the first street segment and from the last street segment back to the depot. Figure 3.2(b) depicts the tables at the depot. The street segments are assigned to the tables in such a way that each mail carrier has to visit only one table and that the letters in the shelves are already sorted according to the order of the street segments in the routes of the mail carriers.



(a) delivery area

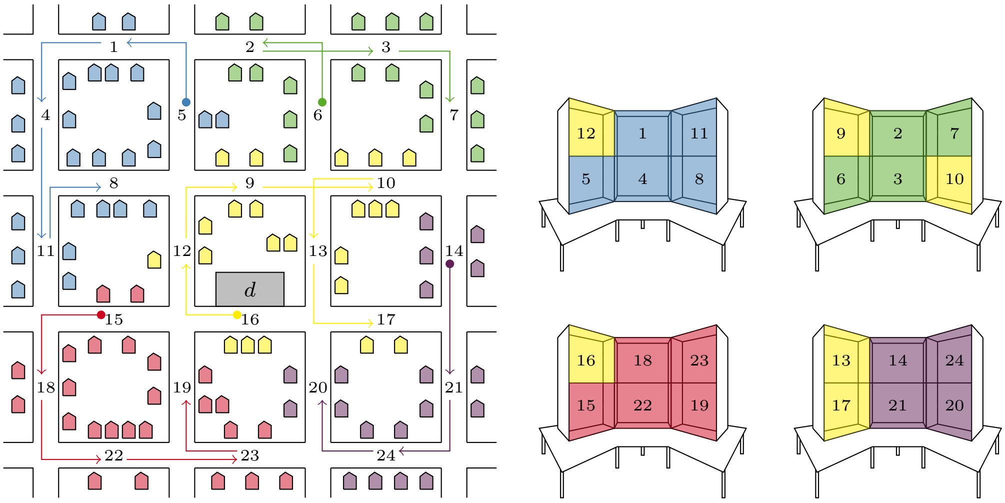
(b) tables at the depot

Figure 3.2: A problem instance with an example solution using four mail carriers

In reality, the demand for mail deliveries varies depending on the day of the week and the season of the year. This implies that the time spent to serve a street segment may reduce or increase depending on the volume of letters to deliver. However, each mail carrier works for a prespecified amount of hours determined by their contract. Because only a limited time can be spent working overtime, and sending workers home early is not economical, DHL changes the number of mail carriers to serve the delivery area on a daily basis. Consequently, some street segments must be reassigned. A new route composition may also be needed because mail carriers can have different working hours depending on their contract, and, for example, one mail carrier working full time is replaced by two mail carriers working part time.

In this case, the reassignment of street segments to other routes results in changes in the routing of the mail carriers and in their visits to the tables at the depot. Recall that the order according to which the letters are sorted on the tables is fixed and cannot be modified to fit the new routing of the mail carriers. Consequently, the letter collection process may become complex and generate a lot of overhead at the depot. This is illustrated by the example in Figure 3.3, using the same instance as in

Figure 3.2. Because of a higher demand, five mail carriers are now needed to serve the delivery area. The most intuitive way of reassigning street segments is to move one or two street segments from the routes of the first four mail carriers to the route of the additional fifth mail carrier (in yellow), resulting in the routing solution in Figure 3.3(a). In Figure 3.3(b) the corresponding situation at the depot is depicted. The "yellow" mail carrier has to walk to every table to collect their letters, a situation that is likely to cause severe problems in reality, when instances have up to 30 tables.



(a) delivery area (b) tables at the depot

Figure 3.3: Example solution for the instance of Figure 3.2 using five mail carriers

Other scenarios that should be avoided are (i) a large number of mail carriers visiting the same table, (ii) the mail carrier having to switch from one side of a table to the other multiple times, or (iii) the mail carrier having to skip blocks of letters to collect the correct ones. The described scenarios are undesirable for DHL for multiple reasons. First, complex depot operations reduce the efficiency and speed because mail carriers need more time to execute them. Second, if multiple mail carriers visit the same table, it is difficult for them to coordinate their collection activities. This increases the risk of making mistakes and collecting the wrong letters. Finally, the learning process for new hires becomes difficult and slow.

Consequently, keeping the depot operations as simple as possible is vital for DHL and must be taken into account when optimizing the routes of mail carriers. To achieve this, we add so-called depot operation constraints to the routing problem described above and refer to the resulting problem as vehicle routing problem with depot operation constraints (VRPDOC). The VRPDOC belongs to the class of NP-hard problems because it generalizes the capacitated VRP (see, e.g., Toth and Vigo, 2002).

The remainder of this section is organized as follows. In Section 3.1.1, we provide a detailed description of the VRPDOC and the depot operation constraints. We review the literature on similar problems in Section 3.1.2 and state our contributions in Section 3.1.3.

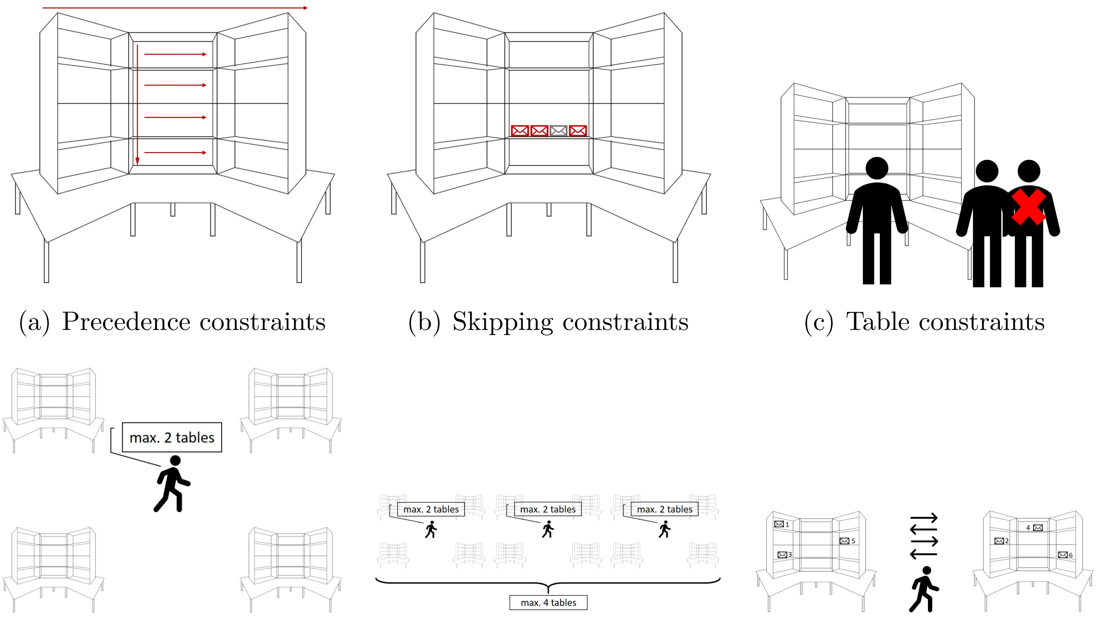
3.1.1 Problem description

The goal of the VRPDOC is to route a given set of mail carriers in a delivery area consisting of street segments under the following constraints: (1) each street segment is visited exactly once, (2) the route of each mail carrier starts and ends at the depot, and (3) the duration of each route stays between a given lower and upper bound. The objective is to minimize the total travel time required by the mail carriers for completing their routes.

To guarantee that letter collection operations at the depot for a given assignment of street segments to tables remain as simple as possible, we add the following depot operation constraints:

1. **Precedence constraints:** If two street segments are assigned to the same table, and if they get assigned to the same mail carrier, then the precedence relationship defined by the order of these two street segments on their table has to be respected in the route of the mail carrier. The effect of these constraints is illustrated in Figure 3.4(a). If the constraints are active, the mail carrier must only move along the table once from left to right, once through each shelf from the top to the bottom row, and through each row from left to right.
2. **Skipping constraints:** Because collecting large blocks of consecutive letters from a shelf is easier than having to skip letters repeatedly, an upper bound on the number of times a mail carrier can skip letters is introduced and represented by $ub_{skipping}$. An example of such a skip is shown in Figure 3.4(b). The mail carrier has to collect the red letters, but because the gray letter is not assigned to their route, the whole row cannot be collected in one go.
3. **Table constraints:** Because the space available to stand in front of a table is limited, an upper bound on the number of mail carriers visiting the same table is introduced and denoted as $ub_{mailcarriers}$. An example in which only two mail carriers can fit in front of a table is depicted in Figure 3.4(c).
4. **Mail carrier constraints:** Because each table is located at a certain distance from the others, an upper bound on the number of tables that a mail carrier can visit is introduced and denoted by ub_{tables}. This avoids that a mail carrier spends too much time moving through the depot. An example in which the mail carrier is only allowed to visit two tables in a depot with four tables is represented in Figure 3.4(d).
5. **Table change constraint:** An upper bound on the total number of table changes, denoted as $ub_{tablechanges}$, is introduced for two reasons. First, to prevent that every mail carrier exploits the upper bound of visited tables imposed by the constraints described in Point 4. An example is shown in Figure 3.4(e): If only the mail carrier constraints of Point 4 with an upper bound of two are

active, each of the three mail carriers could visit two tables. To avoid this, we also impose that the total number of table changes of all mail carriers cannot be greater than four in the example. Second, even if a mail carrier can visit only a few different tables, walking from one to the other may occur several times. An example of such a situation is depicted in Figure 3.4(f): If only the mail carrier constraints of Point 4 with an upper bound of two are active, the mail carrier could visit exactly two tables but move from one to the other multiple times. This is also forbidden by the imposed upper bound on the total number of table changes of four, so that only four switches between the tables are allowed.



(a) Precedence constraints (b) Skipping constraints (c) Table constraints

(d) Mail carrier constraints (e) Table change constraints (1) (f) Table change constraints (2)

Figure 3.4: Motivational examples for the depot operation constraints.

For DHL, solving the VRPDOC is relevant on both the tactical and the operational level. On the tactical level, DHL derives a predefined set of demand scenarios and mail carrier compositions (i.e., the number of mail carriers and their working hours) that repeat across the year, and determine a solution for each of these scenarios only once. Whenever a specific demand realizes, DHL simply implements the corresponding solution. Planning on the operational level becomes necessary when sick notes of mail carriers or unexpected high or low demands cause changes in the available or required number of mail carriers and need to be addressed on a daily basis.

We are aware that considering the assignment of street segments to tables as an additional decision of the optimization problem is an interesting extension of our problem. This would result in the stochastic problem of finding a table assignment that hedges well against demand uncertainty. However, we focus on the non-integrated and deterministic version of the problem in which the assignment of street segments to tables is given and cannot be modified. This has several reasons. From a practical viewpoint, DHL is currently not willing to carry out any change in the assignment of street segments to tables because of organizational effort and the cost incurred. The goal of DHL is to react to a specific demand scenario by optimizing the routes of mail carriers given the current table organization. From a scientific viewpoint, this is the first time that this problem is studied in the literature and starting from its deterministic version is the most natural step. Finally, solving real-world instances of the deterministic problem is already challenging. Thus, we leave the study of a stochastic variant of this problem as future work.

3.1.2 Literature review

To the best of our knowledge, the VRPDOC is a new problem. Although multiple works exist that include one of our additional depot operation constraints (see, e.g., Vidal et al., 2020, for a survey of VRP variants) for a survey of VRP variants), we could not find any work addressing problems that consider all our depot operation constraints at once or at least a relevant subset of them. It rather seems that, apart from loading, depot operations and their interdependency with routing decisions are mostly neglected in the traditional routing literature.

The VRPDOC without the depot operation constraints corresponds to the VRP with lower and upper bounds on the route duration (see, e.g., Laporte et al., 1985). If the bounds on the route duration differ by mail carrier, the problem qualifies as a VRP with heterogeneous fleet (see Koç et al., 2016, for a survey), otherwise it qualifies as a VRP with homogeneous fleet.

One of the depot operations constraints common in the literature are precedence constraints. There are several works that consider VRPs with precedence constraints, (e.g., Irnich, 2008; Dohn et al., 2011; Razali, 2015; Haddadene et al., 2016).

In the VRPDOC, some street segments need to be reassigned to other routes depending on the volume of letters to deliver. Consequently, the VRPDOC also has similarities with rerouting problems. Li et al., 2009 and Mu et al., 2011 study the problem of rerouting the customers of disrupted routes resulting from vehicle breakdowns or accidents. However, different from the VRPDOC, their problems are solved as online problems, i.e., after the vehicles have already started their routes. Nikolić and Teodorović, 2015 study a rerouting problem to be solved if a solution becomes infeasible due to unexpectedly high customer demands. In contrast to our problem, Nikolić and Teodorović, 2015 consider a VRP with customer time windows but no route duration constraints. In the vehicle rescheduling problem, Spliet et al., 2014 consider the joint minimization of routing costs and the costs of deviations of a route from the original schedule. This is similar to our skipping constraints because the assignment of street segments on the tables can be seen as a routing order that we try to keep as good as possible.

The VRPDOC is also related to districting problems because the assignment of street segments to tables could be seen as some sort of districts. Ríos-Mercado and López-Pérez, 2013 and Kalcsics and Ríos-Mercado, 2019 survey this problem class. Different from districting problems, the VRPDOC neither uses graph-based measures nor geometric measures to evaluate solutions. Assis et al., 2014 and Gliesch et al., 2020a study redistricting problems, in which the compactness is maximized while guaranteeing that the homogeneity, workload balance, and conformity of districts is preserved compared to a reference solution. However, in these works, the routing within the districts is absent and is only considered in a later work by Gliesch et al., 2020b.

3.1.3 Contribution and structure of the paper

The contributions of this paper are as follows:

- We address a practically relevant problem that features constraints related to depot operations, which are neglected in many VRP variants.
- We propose different model formulations for the VRPDOC. Moreover, we further improve the best-performing formulation among the proposed ones by adding problem-specific preprocessing techniques and valid inequalities.
- We derive theoretical results by proving that the problem of finding a feasible solution of the VRPDOC is already NP-complete, and that after relaxing two specific constraints, a feasible solution can be found in polynomial time.
- Because commercial solvers cannot even find a feasible solution for realistically-sized instances of the VRPDOC in reasonable runtimes, we propose a solution method, called ILS-VRPDOC, that relies on the iterated local search (ILS) paradigm proposed by Lourenço et al., 2003. To speed up the search, the local search component in ILS-VRPDOC uses granular search (see Toth and Vigo, 2003). This search principle is based on a sparsification method that restricts the size of the neighborhoods to explore. Besides traditional sparsification methods (i.e., distance-based), we propose new problem-specific strategies to select additional arcs.
- In our computational experiments, we compare the performance of the different VRPDOC models on a set of self-generated small-scale instances. In experiments on larger instances provided by DHL, ILS-VRPDOC does not only outperform the DHL solutions regarding the total travel time but also provides significantly simpler solutions with regards to the letter collection operations at the tables. The results show that relevant improvements can be obtained by including our problem-specific sparsification methods. We carry out an extensive analysis of

the solutions obtained for the real-world DHL instances, assess the cost that the consideration of the depot operation constraints has on the total travel times, and evaluate the effect of not considering them on the resulting complexity of the letter collection operations.

The remainder of the paper is organized as follows. We introduce the notation and the mathematical formulation of the VRPDOC in Section 3.2. In Section 3.3, we present complexity results for the VRPDOC. After describing ILS-VRPDOC in Section 3.4, we present the computational experiments, results, and managerial insights in Section 3.5. Finally, Section 3.6 concludes the paper.

3.2 Mathematical formulation

In this section, we present a mathematical formulation of the VRPDOC (Section 3.2.1). Although we tested additional formulations for the VRPDOC (see Appendix A), we introduce only the best-performing formulation called F1a (see Appendix A.7 for an experimental comparison). In Section 3.2.2, we further improve Formulation F1a by proposing preprocessing techniques and valid inequalities.

3.2.1 Formulation F1a

The set of street segments is denoted as I, the depot is represented by d, and the set of all locations is defined as $L = I \cup \{d\}$. The set of arcs connecting the locations is denoted by $A = \{(i,j) : i,j \in L, i \neq j\}$, and the travel time between each pair of locations i and j is given by t_{ij}. Each street segment $i \in I$ is characterized by a service time s_i that represents the time needed by the mail carrier to deliver letters to the households in that street segment.

We denote the set of tables as T, and their organization is defined by the following parameters. The assignment of street segments to tables is described by binary parameters a_i^t, with $a_i^t = 1$ if street segment i is assigned to table t, and $a_i^t = 0$ otherwise. The binary parameters seq_{ij} represent the sequence of street segments on their table. If street segment j is the direct successor of street segment i, then $seq_{ij} = 1$, and $seq_{ij} = 0$ otherwise. The binary parameters $same_{ij}$ indicate if two street segments i and j are assigned to the same table ($same_{ij} = 1$) or not ($same_{ij} = 0$). Additionally, the binary parameters $p_{ij} = 1$ state that street segment i is assigned to the same table as street segment j, and i is located before j in the sequence, otherwise $p_{ij} = 0$.

Set K represents the available mail carriers. We introduce binary variables x_{ij}^k to indicate if mail carrier k goes directly from location i to location j ($x_{ij}^k = 1$) or not ($x_{ij}^k = 0$), binary variables y_i^k to describe whether mail carrier k serves street segment i ($y_i^k = 1$) or not ($y_i^k = 0$), and integer variables u_i^k to represent the position of street segment i in the route of mail carrier k. We use binary variables z^{kt} to decide whether mail carrier k has to visit table t ($z^{kt} = 1$) or not ($z^{kt} = 0$).

The objective is to minimize the total travel time that the mail carriers require to complete their deliveries. The duration of the route traveled by each mail carrier $k \in K$ must be greater than a given lower bound lb^k and smaller than an upper bound ub^k. This guarantees that the mail carrier is neither underutilized nor works too much overtime. The route duration is computed as the sum of the service times and of the travel times between the locations visited in a route. The upper and lower bounds on the route duration may differ for each mail carrier because they depend on the working time imposed by their contract, for example, if a mail carrier has a part-time position.

Table 3.1 summarizes the mathematical notation.

Sets	Description
I	Set of street segments
d	Depot
L	Set of locations $L = I \cup \{d\}$
A	Set of arcs $A = \{(i,j) : i,j \in L, i \neq j\}$
K	Set of mail carriers
T	Set of tables
Parameters	Description
s_i	Service time of location $i \in L$ (service time of the depot is 0)
t_{ij}	Travel times of all arcs $(i,j) \in A$
lb^k	Lower bound on the route duration of mail carrier $k \in K$
ub^k	Upper bound on the route duration of mail carrier $k \in K$
a_i^t	$a_i^t = 1$ if street segment $i \in I$ is assigned to table $t \in T$
$same_{ij}$	$same_{ij} = 1$ if street segment $i \in I$ and street segment $j \in I$ are assigned to the same table
seq_{ij}	$seq_{ij} = 1$ if street segment $i \in I$ and street segment $j \in I$ are assigned to the same table and if j is the direct successor of i
p_{ij}	$p_{ij} = 1$ if street segment $i \in I$ and street segment $j \in I$ are assigned to the same table and i is located before j
$ub_{mailcarriers}$	Upper bound on the number of allowed mail carriers per table
ub_{tables}	Upper bound on the number of allowed tables per mail carrier
$ub_{skipping}$	Upper bound on the number of allowed skips (the total number of times the mail carriers have to skip letters on the table)
$ub_{tablechanges}$	Upper bound on the number of allowed table changes
M	Big M ($M \geq \lvert I \rvert$)
Decision variables	Description
$x_{ij}^k \in \{0,1\}$	$x_{ij}^k = 1$ if mail carrier $k \in K$ goes directly from location i to location j, $x_{ij}^k = 0$ otherwise
$y_i^k \in \{0,1\}$	$y_i^k = 1$ if mail carrier $k \in K$ serves street segment i, $y_i^k = 0$ otherwise
$z^{kt} \in \{0,1\}$	$z^{kt} = 1$ if mail carrier $k \in K$ has to visit table t, $z^{kt} = 0$ otherwise
$u_i^k \in \mathbb{Z}_{\geq 0}$	Position of street segment $i \in I$ in the route of mail carrier $k \in K$

Table 3.1: Summary of the notation for the VRPDOC model.

Formulation F1a represents the VRPDOC as the following integer program:

$$\min \quad \sum_{k \in K} \sum_{(i,j) \in A} t_{ij} x_{ij}^k \tag{3.1}$$

$$\text{s.t.} \quad \sum_{j \in \delta_i^+} x_{ij}^k = y_i^k \qquad i \in I,\ k \in K \tag{3.2}$$

$$\sum_{i \in \delta_j^-} x_{ij}^k = y_j^k \qquad j \in I,\ k \in K \tag{3.3}$$

$$\sum_{j \in \delta_d^+} x_{dj}^k = 1 \qquad k \in K \tag{3.4}$$

$$\sum_{i \in \delta_d^-} x_{id}^k = 1 \qquad k \in K \tag{3.5}$$

$$\sum_{k \in K} y_i^k = 1 \qquad i \in I \tag{3.6}$$

$$\sum_{i \in I} s_i y_i^k + \sum_{(i,j) \in A} t_{ij} x_{ij}^k \leq ub^k \qquad k \in K \tag{3.7}$$

$$\sum_{i \in I} s_i y_i^k + \sum_{(i,j) \in A} t_{ij} x_{ij}^k \geq lb^k \qquad k \in K \tag{3.8}$$

$$y_i^k a_i^t \leq z^{kt} \qquad i \in I,\ k \in K,\ t \in T \tag{3.9}$$

$$\sum_{k \in K} z^{kt} \leq ub_{mailcarriers} \qquad t \in T \tag{3.10}$$

$$\sum_{t \in T} z^{kt} \leq ub_{tables} \qquad k \in K \tag{3.11}$$

$$\sum_{k \in K} \sum_{i \in I} \sum_{j \in I : j \neq i} x_{ij}^k (1 - same_{ij}) \leq ub_{tablechanges} \tag{3.12}$$

$$u_i^k \leq M y_i^k \qquad i \in I,\ k \in K \tag{3.13}$$

$$u_i^k - u_j^k + 1 \leq M(1 - x_{ij}^k) \qquad (i,j) \in A,\ k \in K \tag{3.14}$$

$$u_i^k - u_j^k + 1 \geq -M(1 - x_{ij}^k) \qquad (i,j) \in A,\ k \in K \tag{3.15}$$

$$u_d^k = 1 \qquad k \in K \tag{3.16}$$

$$u_i^k - u_j^k \leq (2 - y_i^k - y_j^k) \cdot ub^k \qquad (i,j) \in A : p_{ij} = 1,\ k \in K \tag{3.17}$$

$$\sum_{i \in I} \sum_{j \in I : j \neq i} \sum_{k \in K} x_{ij}^k \cdot same_{ij} \cdot (1 - seq_{ij}) \leq ub_{skipping} \tag{3.18}$$

$$x_{ij}^k \in \{0,1\} \qquad (i,j) \in A,\ k \in K \tag{3.19}$$

$$y_i^k \in \{0,1\} \qquad i \in I,\ k \in K \tag{3.20}$$

$$u_i^k \in \mathbb{Z}_{\geq 0} \qquad i \in L,\ k \in K \tag{3.21}$$

$$z^{kt} \in \{0,1\} \qquad k \in K,\ t \in T \tag{3.22}$$

The objective of minimizing the total travel time needed by the mail carriers for completing their routes is defined in (3.1). Constraints (3.2) and (3.3) ensure that mail carrier k traverses exactly one outgoing and one ingoing arc of street segment i, respectively, if street segment i is served by mail carrier k. Constraints (3.4) and (3.5) guarantee that every mail carrier leaves and returns to the depot exactly once. Constraints (3.6) state that every street segment is visited by exactly one mail carrier. The upper and lower bounds on the duration of the routes are modeled in constraints (3.7) and (3.8). Constraints (3.9) link variables y and z such that if mail carrier k serves street segment i, and i is located at table t, mail carrier k has to visit table t. The table, the mail carrier, and the table change constraints are modeled in constraints (3.10), (3.11), and (3.12), respectively. Constraints (3.13)–(3.16) define the subtour elimination in the form of the well-known Miller-Tucker-Zemlin constraints. The precedence constraints are given in constraints (3.17). If two street segments $i \in I$ and $j \in I$ that fulfill the precedence constraints (i.e., $p_{ij} = 1$) are assigned to the same mail carrier k, the right-hand-side of the constraint becomes zero, and thus, the index of street segment i in the assigned route is forced to be smaller than the index of street segment j. To model the skipping constraints, a counter keeps track of the number of times that the mail carrier needs to skip letters while collecting them from the shelves, which is the case if the following three conditions are simultaneously satisfied: (i) two street segments $i \in I$ and $j \in I$ are served directly after each other in the route of a mail carrier, (ii) they are assigned to the same table, and (iii) j is not the direct successor of i on the table (see constraints (3.18)). Finally, the domain of the variables is defined in constraints (3.19)–(3.22). Note that the u-variables can also be defined as continuous variables. However, preliminary experiments showed that defining them as continuous variables and removing constraints (3.15) and (3.16) has no impact on the performance of the model. Therefore, we use the integer definition of the variables.

3.2.2 Preprocessing and valid inequalities

To improve Formulation F1a, we restrict the size of the problem by proposing preprocessing techniques in Section 3.2.2.1, and we strengthen the formulation by introducing valid inequalities in Section 3.2.2.2.

3.2.2.1 Preprocessing techniques

We propose two preprocessing techniques to remove infeasible arcs.

Precedence (`Prec`) The first preprocessing technique is based on the precedence constraints. A variable x_{ji}^k, $(j,i) \in A,\ k \in K$ can be fixed to zero (i.e., $x_{ji}^k = 0$) if the following condition holds:

$$p_{ij} = 1. \tag{3.23}$$

Condition (3.23) states that if two street segments $i \in I$ and $j \in I$ visited by the same mail carrier k are assigned to the same table and i is located before j (i.e., $p_{ij} = 1$), j cannot precede i. Then, variable x_{ji}^k can be fixed to zero.

Street segments incompatibility (`pre_SegInc`) The second preprocessing technique is based on the route duration constraints. A variable x_{ij}^k, $(i,j) \in A,\ k \in K$ can be fixed to zero (i.e., $x_{ij}^k = 0$) if the following condition holds:

$$s_i + s_j + t_{ij} \geq ub^k. \tag{3.24}$$

If the sum of the service times of street segments i and j and the travel time between i and j exceeds the upper bound on the route duration of mail carrier k, then only one of the two street segments i and j can be served by mail carrier k and variable x_{ij}^k can be fixed to zero.

If the triangle inequality holds for the travel times, we can strengthen condition (3.24) to:

$$t_{di} + s_i + t_{ij} + s_j + t_{jd} \geq ub^k. \tag{3.25}$$

3.2.2.2 Valid inequalities

To strengthen Formulation F1a, we propose the following four classes of valid inequalities.

Street segments incompatibility (`vi_SegInc`) If the sum of the service times of street segments i and j and the travel time between i and j exceeds the upper bound on the route duration of mail carrier k, at most one of these two street segments can be assigned to mail carrier k. This is described by the following valid inequality:

$$y_i^k + y_j^k \leq 1 \qquad (i,j) \in A : s_i + s_j + t_{ij} \geq ub^k,\ k \in K. \tag{3.26}$$

If the triangle inequality holds for the travel times, we can strengthen the valid inequality to:

$$y_i^k + y_j^k \leq 1 \qquad (i,j) \in A : t_{di} + s_i + t_{ij} + s_j + t_{jd} \geq ub^k,\ k \in K. \tag{3.27}$$

2-cycle and 3-cycle elimination constraints (`Cycle`) To strengthen the subtour elimination, we directly forbid 2- and 3-cycles by adding the following two valid inequalities that are frequently used in the literature (see, e.g., Irnich and Villeneuve, 2006):

$$x_{ij}^k + x_{ji}^k \leq 1 \qquad (i,j) \in A,\ k \in K, \tag{3.28}$$

$$x_{ij}^k + x_{jv}^k + x_{vi}^k \leq 2 \qquad i,j,v \in I : i \neq j \neq v,\ k \in K. \tag{3.29}$$

Bounds on the number of street segments within each route (`NumSegBounds`) We compute the minimum and maximum number of street segments $min_{num\_street\_segments}^k$ and $max_{num\_street\_segments}^k$, respectively, that can be assigned to a mail carrier.

To obtain the minimum number, we compute, for each street segment, the sum of its service time and of the travel time of the arc from the street segment to the location farthest away from the street segment, i.e., we compute

$$s_i + \max_{j \in L}\{t_{ij}\} \tag{3.30}$$

for each street segment $i \in I$. We store these sums in decreasing order in a list called γ^{dec}. Then, we iterate through that list and add up the entries. We stop as soon as the sum exceeds the lower bound lb^k on the duration of the route of mail carrier k. The iteration counter, i.e., the number of entries we have summed up, is the minimum number of street segments that must be assigned to that mail carrier:

$$min_{num\_street\_segments}^k = \min\{j | \sum_{i=1}^{j} \gamma^{dec}[i] > lb^k\}. \tag{3.31}$$

The procedure to obtain the maximum number of street segments that can be assigned to a mail carrier is analogous. We compute, for each street segment, the sum of its service time and of the travel time of the arc from the street segment to the location closest to the street segment, i.e., we compute

$$s_i + \min_{j \in L}\{t_{ij}\} \tag{3.32}$$

for each street segment $i \in I$. We store these sums in increasing order in a list called γ^{inc}. Then, we iterate through that list and add up the entries. We stop as soon as adding the next entry to the sum would exceed the upper bound ub^k on the duration of the route of mail carrier k. The iteration counter, i.e., the number of entries we have summed up, is the maximum number of street segments that can be assigned to that mail carrier:

$$max^k_{num\_street\_segments} = \max\{j | \sum_{i=1}^{j} \gamma^{inc}[i] < ub^k\}. \tag{3.33}$$

Finally, we can add the following valid inequalities to the model:

$$\sum_{i \in I} y_i^k \geq min^k_{num\_street\_segments} \qquad k \in K, \tag{3.34}$$

$$\sum_{i \in I} y_i^k \leq max^k_{num\_street\_segments} \qquad k \in K. \tag{3.35}$$

Symmetry breaking constraints (`SymBreak`) Symmetry breaking constraints are regularly used in models of VRPs with a homogeneous fleet (see, e.g., Archetti et al., 2014; Lahyani et al., 2018; Darvish et al., 2020). Darvish et al., 2020 recommend to use hierarchical constraints because they perform best in terms of solution quality and runtime. The hierarchical formulation of symmetry breaking constraints has the following form:

$$y_i^k \leq \sum_{j=1}^{i-1} y_j^{k-1} \qquad i \in I,\ k \in K \setminus \{0\}. \tag{3.36}$$

Constraints (3.36) ensure that if street segment i is served by mail carrier k, then at least one street segment with an index smaller than i must be served by mail carrier $k-1$.

These constraints are defined for VRPs with a homogeneous fleet and cannot be used for the heterogeneous case. However, for the VRPDOC, it is rarely the case that all mail carriers have different upper and lower bounds on the route duration due to standardized working contracts. Instead, there are groups of mail carriers with the same upper and lower bounds on the route duration. So although we cannot define the symmetry breaking constraints on the whole set of mail carriers, we can still use

them within each group.
Let us assume that the mail carriers are divided into a set of disjoint groups $G = \{K_1, K_2, \dots, K_{|G|}\}$ with $K_1 \cup K_2 \cup \dots \cup K_{|G|} = K$, $K_1 \cap K_2 \cap \dots \cap K_{|G|} = \emptyset$. Each group $K_g = \{k_{g_1}, \dots, k_{g_{|K_g|}}\}$ contains mail carriers that have the same upper and lower bound on their route duration. Then, we can add the following valid inequalities:

$$y_i^{k_{g_l}} \leq \sum_{j=1}^{i-1} y_j^{k_{g_{l-1}}} \qquad i \in I,\ g \in \{1, \dots, |G|\},\ l \in \{2, \dots, |K_g|\}. \tag{3.37}$$

In the special case in which all mail carriers have the same upper and lower bound on their route duration, the VRPDOC is a variant of the VRP with a homogeneous fleet, and because all mail carriers are in the same group, constraints (3.37) exactly correspond to constraints (3.36).

3.3 Complexity results

In this section, we present results about the complexity of the VRPDOC. We refer to the problem of identifying a feasible solution of the VRPDOC as the vehicle routing feasibility problem with depot operation constraints (VRFPDOC). In Theorem 3.3.1, we show that the VRFPDOC is NP-complete. We then define a relaxed version of the VRFPDOC with regard to the route duration constraints, and we call it VRFPDOC$_{rel}$. In Theorem 3.3.2, we show that a solution for the VRFPDOC$_{rel}$ (that is, a solution that is feasible for all the other constraints of the VRPDOC except for the route duration constraints and those forbidding empty routes) can be found in polynomial time.

Theorem 3.3.1 The VRFPDOC is NP-complete.

Proof. To show that the VRFPDOC is NP-complete, we show that the subproblem of assigning street segments to mail carriers in such a way that they fulfill the route duration constraints is NP-complete.

We use a transformation of an instance of the VRFPDOC to an instance of a particular case of the knapsack problem, which is known to be NP-complete (Garey and Johnson, 1979) and is defined as follows:

- Instance: Given a set $P = \{p_1, p_2, \dots, p_n\}$ with $p_i \in \mathbb{R}_{\geq 0}$ and positive integers B and C with $B < C$.
- Question: Does there exist a subset $P' \subseteq P$ such that: $\sum_{p_i \in P'} p_i \geq B$ and $\sum_{p_i \in P'} p_i \leq C$?

We assume an instance of this knapsack problem to be given. We now consider the following instance of the VRFPDOC:

- The number of street segments $|I|$ is given by n.
- The service time s_i of street segment $i \in I$ is given by $s_i = p_i$.
- The travel times are $t_{ij} = 0$ for all $(i, j) \in A$.
- There is one mail carrier $k \in K$ with $lb^k = B$ and $ub^k = C$.
- The upper bounds for the depot operations constraints are given by $ub_{mailcarriers} = ub_{tables} = ub_{skipping} = ub_{tablechanges} = \infty$.

Then, the question of the knapsack problem is equivalent to the question whether there exists a subset of street segments so that the route duration constraint of mail carrier k is fulfilled.

If the knapsack problem has a "yes" answer, then we can transform the solution of the knapsack problem to a solution of the subproblem of assigning street segments to mail carrier k by setting $y_i^k = 1$ if $p_i \in P'$, and $y_i^k = 0$ otherwise. In the same manner, if a subset of street segments $I' \subseteq I$ fulfills $\sum_{i \in I'} s_i <= lb^k$ and $\sum_{i \in I'} s_i >= ub^k$, then by including in P' the items p_i that are represented by the street segments $i \in I'$, the knapsack problem has a "yes" answer. □

Theorem 3.3.2 A feasible solution for an instance of the VRFPDOC$_{rel}$ can be found in polynomial time.

Proof. We perform a proof by cases. The only three cases that can occur depending on the relationship between the desired number of mail carriers $|K|$, and the number of tables $|T|$ are the following:

1. The number of desired mail carriers is equal to the number of tables ($|K| = |T|$). In this case, we assign one table to each mail carrier, and let them visit the street segments according to the sequence in which they are sorted on the table, that we denote by $\overline{u}_i^k$. We obtain a solution identical to the one corresponding to the organization of the tables. Hence, it is trivial to see that the VRFPDOC$_{rel}$ solution is feasible with respect to all constraints (i.e., there is only one mail carrier per table, one table per mail carrier, no skips, and no table changes).

2. The number of desired mail carriers is greater than the number of tables ($|K| > |T|$). In this case, we assign one table to each of the $|T|$ mail carriers, and let them visit the street segments according to the sequence in which they are sorted on the table, that we denote by $\overline{u}_i^k$. Then, we assign an empty route to each of the remaining $|K| - |T|$ mail carriers. Because of the reasons explained in Case 1, such a solution is feasible with respect to all VRFPDOC$_{rel}$ constraints.

3. The number of desired mail carriers is lower than the number of tables ($|K| < |T|$). In this case, we assign the street segments of table t to the route of mail carrier $k = t(\text{mod}|K|)$. Then, each mail carrier visits the street segments according to the sequence in which they are sorted on the tables. For this case, it is necessary to analyze the depot operation constraints one by one:

   - Precedence constraints: Because the street segments are visited according to the sequence in which they are sorted on the tables, the precedence constraints are fulfilled.
   - Skipping constraints: Because the street segments are visited according to the sequence in which they are sorted on the tables, the skipping constraints are fulfilled.
   - Table constraints: Because each table is visited by exactly one mail carrier, the table constraints are fulfilled.
   - Mail carrier constraints: By assigning table t to the route of mail carrier $k = t(\text{mod}|K|)$, we assign the tables as evenly as possible across the mail carriers. This corresponds to the lowest number of tables per mail carrier that can be obtained. If the number of tables per mail carrier is higher than the upper bound imposed by an instance, then the instance is infeasible.
   - Table change constraint: By assigning table t to the route of mail carrier $k = t(\text{mod}|K|)$, we have exactly $|T| - |K|$ table changes in the solution. This corresponds to the minimum number of table changes of any feasible solution.

Because in all three cases the sequence of street segments is fixed and we just assign complete tables to mail carriers, the heuristics described run in $\mathcal{O}(|T|)$ in all three cases. □

3.4 An iterated local search for the VRPDOC

In this section, we introduce our ILS-VRPDOC algorithm (see Figure 4 for a pseudocode overview).

We generate a starting solution S^0 using the construction heuristic described in Section 3.4.1. Then, the ILS presented in Section 3.4.2 is executed. In each ILS iteration, a local search based on a variable neighborhood descent (VND) is applied to improve solution S (see Section 3.4.2.1). In the VND, we allow infeasibilities only with respect to the route duration constraints. At the end of the VND, if the improved solution S is better than the overall best-found solution S^*, S^* is updated. Then, we perturb the best-found solution S^* (Section 3.4.2.2). The ILS terminates if either η iterations without improvement or a given time limit of τ seconds are reached.

Algorithm 4: Pseudocode of the ILS-VRPDOC algorithm

```
$S \leftarrow$ *constructionHeuristic*()
$S^* \leftarrow S,\ S \leftarrow S$
**while** *termination criterion not satisfied* **do**
    $S \leftarrow$ *VND*($S$)
    **if** $c(S) < c(S^*)$ **then**
        $S^* \leftarrow S$
        $c(S^*) \leftarrow S$
    **end**
    $S \leftarrow$ *Perturbation*($S^*$)
**end**
**return** $S^*$
```

3.4.1 Construction heuristic

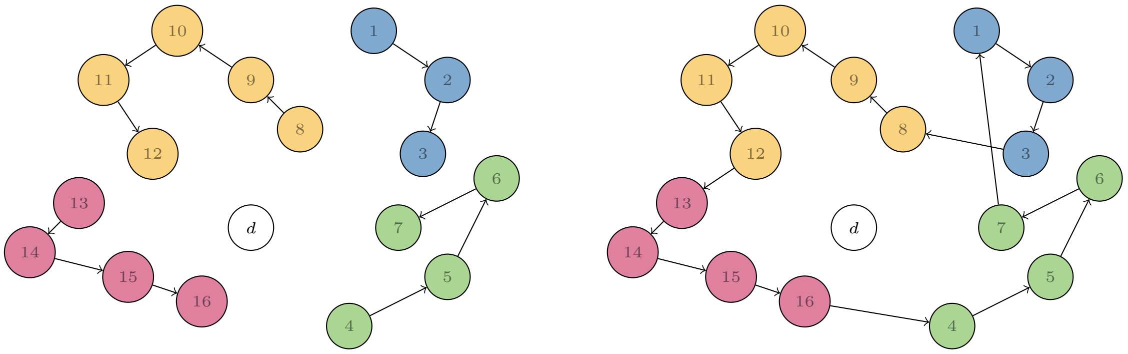
To build an initial solution, we iterate a route-first cluster-second approach. Section 3.4.1.1 describes how to build a giant route consisting of all street segments. Section 3.4.1.2 explains how to cut this giant route to obtain as many routes as the number of desired mail carriers $|K|$. If these two steps don't lead to a feasible solution, we start again by constructing a different giant route and trying to cut it. We stop, as soon as we find a feasible solution.

3.4.1.1 Build giant route

The procedure to build the giant route is explained along the example shown in Figure 3.5. The example contains a depot, 16 street segments, and four tables indicated by four different colors. Street segments depicted in the same color are assigned to the same table, i.e., street segments 1–3 are assigned to table 1, street segments 4–7 to table 2, street segments 8–12 to table 3, and street segments 13–16 to table 4. For the sake of simplicity, we assume that the street segments are sorted on the tables in increasing order of their number. To ensure that we obtain a feasible solution with respect to the precedence and skipping constraints, the order in which the street segments are sorted on the tables is kept. This guarantees that the precedence constraints are always fulfilled, and the number of times a mail carrier skips letters while collecting them is zero. To do this, we build a path (one for each table), in which the street segments are connected according their order on the table (see the arcs in Figure 3.5(a)). In our example, this leads to four paths representing the four different tables. Then, we randomly connect these paths to form the giant route. In Figure 3.5(b), we connect the paths in the following sequence: table 1 – table 3 – table 4 – table 2 – table 1.

3.4.1.2 Cut giant route

To cut the giant route and obtain the desired number of routes, we solve a path problem on an auxiliary layered graph. We describe the construction of this graph in



(a)

(b)

Figure 3.5: Example of the procedure used to build the giant route.

Section 3.4.1.2.1 and the mathematical model of the path problem in Section 3.4.1.2.2.

3.4.1.2.1 Construction of the auxiliary layered graph We denote the auxiliary layered graph by $G_{\text{aux}} = (V_{\text{aux}},\ A_{\text{aux}})$. The set V_{aux} consists of all the vertices in the giant tour (i.e, all street segments $i \in I$) and of a dummy vertex, i.e., we define $V_{\text{aux}} = I \cup \{\text{dummy}\}$. The set of arcs A_{aux} describes the routes of the mail carriers. In particular, an arc $(i, j)^k,\ i, j \in V_{\text{aux}} : i \neq j, k \in K$ represents a route starting at street segment i, and serving all street segments up to j (j excluded) traveled by mail carrier k. The set A_{aux} contains only arcs that are feasible with respect to the route duration constraints and mail carrier constraints. Consequently, for each mail carrier, we add an arc $(i, j)^k$ only if (i) the sum of the service times of all street segments visited between i and the predecessor of j and the travel times for visiting these street segments lies within the lower and upper bound of the route duration of that mail carrier k, and (ii) the route described by $(i, j)^k$ does not visit more than ub_{tables} tables. The construction of such an auxiliary graph is explained in the following example.

Example 3.4.1 Given is a giant route $(s_1, s_2, s_3, s_4, \dots)$ obtained by applying the procedure described in Section 3.4.1.1, where s stands for street segment. Assume that, given a certain demand, DHL decides that three mail carriers are needed with the following lower and upper bounds on their route duration: $(lb^1, ub^1) = (3, 4)$, $(lb^2, ub^2) = (3, 5)$, and $(lb^3, ub^3) = (4, 5)$. Hence, we cut the giant route to obtain three routes. For the sake of simplicity, in this example, we do not include the mail carrier constraint, and we assume that the lower and upper bound on the route duration only consider the service time and avoid the inclusion of the travel times between street segments in these constraints. Moreover, the service time of each street segment is assumed to be equal to one, so that the lower and upper bound on the route duration correspond to a lower and upper bound on the number of street segments per route.

The auxiliary graph has layers that are added one by one. The first layer of the graph represents the first route that has to start at the first street segment s_1 of the giant route. To construct this first layer, we add, for every mail carrier, all arcs with $i = s_1$ that describe feasible routes with respect to the route duration and the mail carrier constraints. Figure 3.6 shows an example of this first layer. Each of the three mail carriers is represented by a different color (green, blue, orange). The first mail carrier k_1 (green) has a lower bound of $lb^1 = 3$ and an upper bound of $ub^1 = 4$ on the route duration. Thus, for this mail carrier, only the two routes described by the green arcs are feasible: the route that includes street segments s_1, s_2, and s_3 and that is represented by the arc $(s_1, s_4)^{k_1}$, and the route that includes the street segments s_1, s_2, s_3, and s_4 and that is represented by the arc $(s_1, s_5)^{k_1}$. Analogously, we add the arcs for the second (blue) and the third (orange) mail carrier. Because each arc is not only defined by its origin and destination but also by the mail carrier, the resulting graph is not simple, and several arcs with the same origin and destination are possible.

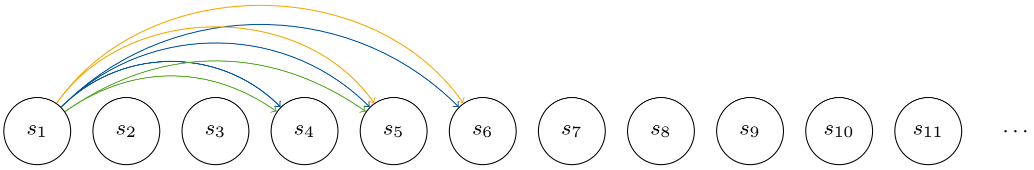


Figure 3.6: First layer of an exemplary auxiliary layered graph built to cut the giant route.

To construct the second layer, the procedure is similar to the one applied to obtain the first layer. The only difference is that we have to consider all street segments that are arc destinations in the first layer as origin vertices. Hence, there may be more origin vertices than the one (s_1) in the first layer. In the example in Figure 3.7, the origin vertices for the second-layer are the street segments s_4, s_5, and s_6. Then, we iterate over all possible origin vertices, and add, for each mail carrier, the arcs representing feasible routes as described for the first layer.

All following layers are build analogously to the second layer. Because we need exactly one route per mail carrier, in total we need $|K|$ layers. ◇

3.4.1.2.2 Path problem After obtaining the auxiliary layered graph, we solve a path problem to find a feasible solution to the problem of cutting the giant route into the desired number of routes. For the model formulation, we consider the set of vertices $V_{aux} = (s_1, s_2, \dots, s_{|I|}, dummy)$ and the set of arcs A_{aux} as they appear in the auxiliary layered graph. We denote the number of table changes implied by the route described by the arc $(i, j)^k$ for all k by tc_{ij}, and the binary parameter tv_{ij}^t indicates

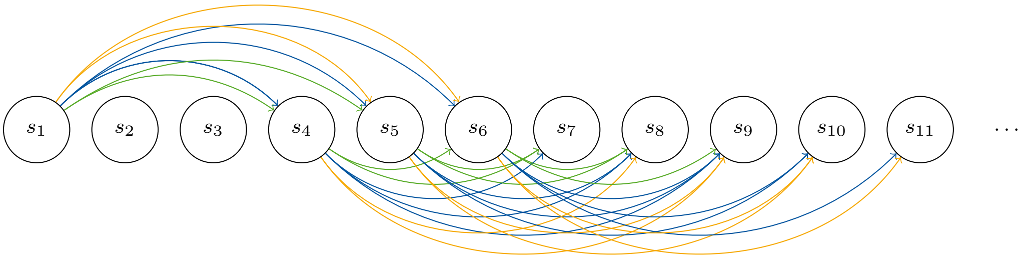


Figure 3.7: First two layers of an exemplary auxiliary layered graph built to cut the giant route.

whether the route described by the arc $(i,j)^k$ for all k implies a visit to table $t \in T$. Finally, the binary variable $x_{ij}^k \in \{0,1\}$ determines if mail carrier k executes the route described by arc $(i,j)^k$ ($x_{ij}^k = 1$) or not ($x_{ij}^k = 0$). We obtain the following binary program for the path problem:

$$
\begin{aligned}
\min \quad & 0 && && (3.38)\\
\text{s.t.} \quad & \sum_{(i,j,k)\in A_{aux}:i=s_1} x_{ij}^k = 1 && && (3.39)\\
& \sum_{(i,j)^k\in A_{aux}:j=dummy} x_{ij}^k = 1 && && (3.40)\\
& \sum_{(i,j)^k\in A_{aux}:j=j'} x_{ij}^k = \sum_{(j,i,k)\in A_{aux}:j=j'} x_{ji}^k && j' \in V_{aux} \setminus \{s_1, dummy\} && (3.41)\\
& \sum_{(i,j)^k\in A_{aux}:k=k'} x_{ij}^k = 1 && k' \in K && (3.42)\\
& \sum_{(i,j)^k\in A_{aux}} tv_{ij}^t \cdot x_{ij}^k \leq ub_{mailcarriers} && t \in T && (3.43)\\
& \sum_{(i,j)^k\in A_{aux}} tc_{ij} \cdot x_{ij}^k \leq ub_{tablechanges} && && (3.44)\\
& x_{ij}^k \in \{0,1\} && (i,j)^k \in A_{aux} && (3.45)
\end{aligned}
$$

Because the goal is only to find a feasible solution, the objective function (3.38) is set to zero. Constraints (3.39), (3.40), and (3.41) ensure that we obtain a path from the first to the last vertex in the giant route. Constraints (3.42) guarantee that exactly one arc is chosen for each mail carrier in the resulting path. The table constraints and table change constraints are modeled in constraints (3.43) and (3.44), respectively. The domain of the variables is defined in constraints (3.45).

We remark that if all mail carriers have the same lower and upper bound on the route duration, the number of arcs in the auxiliary graph drastically decreases and, consequently, also the size of the mathematical model. In fact, the information of which route is performed by which mail carrier is disregarded and, without loss of generality, we randomly assign mail carriers to layers. Referring to Example 3.4.1, suppose that now all three mail carriers have the same lower and upper bound $(lb, ub) = (3, 5)$. The first two layers of the auxiliary layered graph are depicted in Figure 3.8. Because all mail carriers have the same lower and upper bounds on the route duration, we can arbitrarily assign the first layer to the blue mail carrier, and the second layer to the green mail carrier.

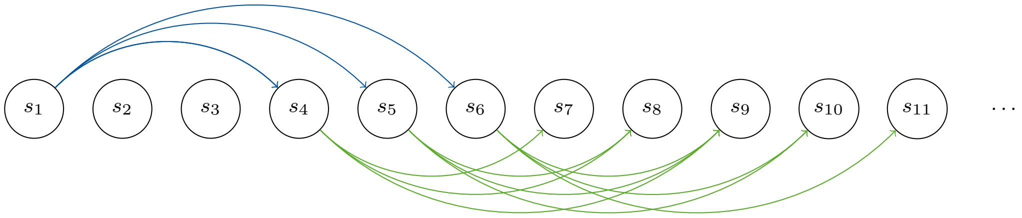


Figure 3.8: Exemplary auxiliary layered graph to cut the giant route if all mail carriers have the same upper and lower bound on the route duration.

This path problem does not necessarily need to be feasible. In this case, we revert to constructing the giant route. Since the order in which the paths representing the tables are connected is randomized, repeating the procedure produces a different giant route. These two steps are iterated until a feasible solution is found. In preliminary studies, we also considered starting from an infeasible initial solution, however, results showed that it is always beneficial to start from a feasible solution.

To summarize, our construction heuristic builds a giant route consisting of all street segments, and cuts it into the desired number of routes by solving a mathematical model. We guarantee that all constraints of the VRPDOC are fulfilled in the construction heuristic, namely:

- The route duration constraints are fulfilled because we only add arcs to the auxiliary layered graph that represent feasible routes with respect to the route duration constraints.

- The precedence constraints are respected because, in the giant route, the street segments are visited in the same sequence in which they are sorted on the tables.

- The skipping constraints are satisfied because, in the giant route, the street segments are visited in the same sequence in which they are sorted on the tables. Consequently, no letters have to be skipped.

- The table constraints are fulfilled due to their inclusion in model (3.38)–(3.45) of the path problem.

- The mail carrier constraints are respected because we only add arcs that represent feasible routes with respect to the mail carrier constraints to the auxiliary layered graph.

- The table change constraint is satisfied due to its inclusion in model (3.38)–(3.45) of the path problem.

3.4.2 Iterated local search

We design our ILS according to the classical ILS framework originally proposed by Lourenço et al., 2003. Each iteration of the ILS consists of two phases. The first phase corresponds to a VND with the goal of improving the solution with respect to the total travel time (Section 3.4.2.1). In the VND, we allow infeasible solutions as described in more detail in Section 3.4.2.1.1. The second phase consists of a perturbation with the goal of reaching new areas of the search space (Section 3.4.2.2).

3.4.2.1 Variable neighborhood descent

In the VND, we iteratively evaluate neighboring solutions with a first improvement search strategy. Each neighboring solution is obtained by applying a move that is uniquely defined by an operator o from the ordered set of neighborhood operators O and a so-called generator arc $(i, j) \in A$. After the execution of the move, the generator arc (i, j) is part of the solution. Because we apply a first improvement search strategy, the order in which we evaluate the moves has an impact on the search trajectory. As soon as we find an improving solution, we accept it, and we restart the search with the newly obtained solution and the first neighborhood operator. During the search process, the VND is allowed to visit solutions that are infeasible with regard to the lower and the upper bounds on the route duration. The solutions are penalized by means of a generalized cost function in which the penalty parameters are determined via a dynamic mechanism.

Algorithm 5 shows the pseudocode of the VND. The VND is always initiated with a feasible solution S. In the first ILS iteration, solution S corresponds to the one returned by the construction heuristic. For the following iterations, solution S is the one returned by the perturbation phase. During the search, the best feasible solution is denoted by S^{best}. At the beginning of each ILS iteration, to increase the likelihood that different search trajectories are explored, the operators from the set O (described in Section 3.4.2.1.2) and the street segments from the set I used as the origin vertices of the generator arcs are saved in ordered sets $\tilde{O}$ and $\tilde{I}$ and randomly shuffled. The penalty parameter α is initialized to α_0. As long as solution S^{best} improves, the VND traverses the neighborhood operator set $\tilde{O}$. The generator arc is obtained by pairing a vertex i from the set of shuffled street segments $\tilde{I}$, and a vertex j from the list L_i in which the possible end vertices for the generator arcs originating in i are saved (see Section 3.4.2.1.3 for the construction of L_i). Given a neighborhood operator and a generator arc, the move is applied to solution S, and a new solution $S^{'}$ is obtained. Solution $S^{'}$ is then evaluated with the generalized cost function as described

in Section 3.4.2.1.1. If the cost of solution $S^{'}$ is lower than that of S, $S^{'}$ becomes the new incumbent solution. Moreover, if $S^{'}$ is a feasible solution and better than the overall best feasible solution, we update S^{best}. The penalty parameter α is updated depending on the amount by which the lower and the upper bounds on the route duration are violated.
If no improvement can be found, the VND terminates.

Algorithm 5: Pseudocode of VND(S)

```
1  $S^{best} = S$
2  $improvement = false$
3  $\tilde{I}, \tilde{O} = shuffle(I, O)$
4  $\alpha = \alpha_0$
5  while $improvement$ do
6      $improvement = false$
7      for $o \in \tilde{O}$ do
8          for $i \in \tilde{I}$ do
9              for $j \in L_i$ do
10                 $S^{'} \leftarrow move(o, i, j, S)$
11                 if $c_{gen}(S^{'}) < c_{gen}(S)$ then
12                     $improvement = true$
13                     $S \leftarrow S^{'}$
14                     $c(S) \leftarrow c(S^{'})$
15                     if $viol(S^{'}) = 0$ and $c(S^{'}) < c(S^{best})$ then
16                         $S^{best} \leftarrow S^{'}$
17                         $c(S^{best}) \leftarrow c(S^{'})$
18                     end
19                     $\alpha \leftarrow updateAlpha(\alpha, viol(S^{'}))$
20                     goto line 5
21             end
22         end
23     end
24 end
25 return $S^{best}$
```

3.4.2.1.1 Evaluation of infeasible solutions By allowing the VND to visit infeasible solutions, it is possible to escape local optima, which leads to more flexibility in the exploration of the solution space. We only allow infeasibility in the route duration constraints because of two reasons: i) As proven in Theorem 3.3.2, a solution to the VRFPDOC$_{rel}$ can be found in polynomial time, which is not true for other constraints. This suggests, that by allowing infeasibility in these constraints in the VND, we gain the most flexibility in the exploration of the solution space. ii) We use a generalized cost function that penalizes the violation of the route duration constraints, which is expressed in the same measurement unit as the objective function of the VRPDOC (i.e., time). Therefore, it is not necessary to calibrate the penalty due to the use of different measurement units.

Let $viol(S)$ be the cumulative route duration violation of a given solution S, $c(S)$ the objective function value of solution S (i.e., the total travel time of solution S), and α the penalty factor. Then, the generalized cost function $c_{gen}(S)$ is:

$$c_{gen}(S) = c(S) + \alpha \cdot viol(S).$$

We apply the dynamic mechanism similar to Schneider and Löffler (2019) to update the penalty parameter α. We initialize the penalty factor α to α_0. If the current solution is infeasible, we multiply α by a factor $\delta > 1$ to increase the penalization of route duration violations. By doing so, the search is guided towards feasible solutions. On the contrary, if the current solution is feasible, we divide α by a factor $\delta > 1$ to decrease the penalization of route duration violations. This brings more diversification in the search as infeasible solutions are penalized less strongly. If α is not bounded, it can increase or decrease strongly, and a lot of iterations are then needed to get back to an adequate value. To avoid this and make the penalty mechanism more reactive, we introduce two additional parameters $\alpha_{\min}$ and $\alpha_{\max}$, and as soon as a feasible solution is found, we set $\alpha = \min(\alpha_{\max}, \frac{\alpha}{\delta})$. Otherwise, we set $\alpha = \max(\alpha_{\min}, \alpha\delta)$.

3.4.2.1.2 Neighborhood operators The neighborhood operators contained in set O are defined such that together with a generator arc (i, j) they uniquely identify a move. The neighborhood operators are presented in Figure 3.9 and are:

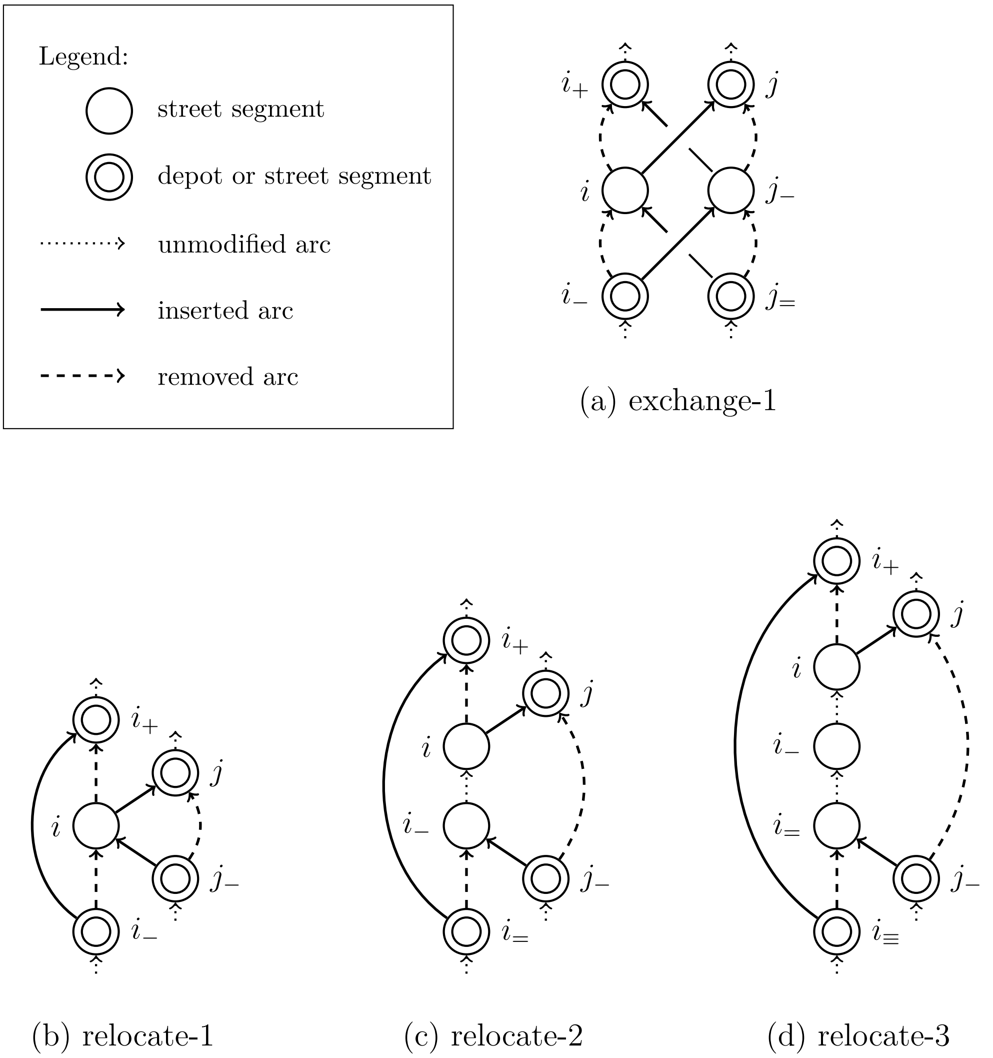
1. exchange-1 swaps the positions of two street segment, and it is defined in both intra- and inter-route fashion.

2. relocate-1 moves one street segment to a different position, and it is defined in both intra- and inter-route fashion.

3. relocate-2 moves one street segment and its predecessor to a different position, and it is defined in both intra- and inter-route fashion.

4. relocate-3 moves one street segment and its two predecessors to a different position, and it is defined in both intra- and inter-route fashion.

If we restrict ourselves to the described traditional VRP operators, it can happen that the depot operations constraints together with the constraints on the route duration of a given instance are so tight that it is not possible to move from one feasible solution to another one. Consequently, we have no possibility to escape from a possibly low-quality local optimum. In the example in Figure 3.10, we are given a solution of an instance with six preparation tables, represented by the different colors, and we have to find a solution with three mail carriers. Assume that in this instance the upper bound of allowed table changes is three, then there is no feasible move that can be applied to this solution because applying any of the moves described above leads to four table changes. There might be other feasible solutions but they cannot be reached with the set of neighborhood operators above.



(a) exchange-1

(b) relocate-1

(c) relocate-2

(d) relocate-3

Figure 3.9: Neighborhood operators of ILS-VRPDOC. The generator arc is given by (i, j). The predecessor and successor of i are denoted as i_- and i_+, respectively. Note that in all of the operators presented, arc (i, j) is always an arc inserted into the solution.

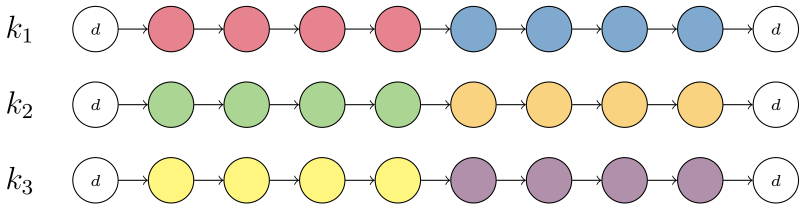


Figure 3.10: The routes of a solution for an instance with six preparation tables represented by the different colors and three mail carriers.

To overcome this problem, we introduce an additional problem-specific neighborhood operator called swap-tables. The operator swaps all street segments of two tables t_m and t_n, $m \neq n$, between two mail carriers k_i and k_j, $i \neq j$. To do this, all street segments in the route of mail carrier k_i that are assigned to table t_m are moved to the end of the route of mail carrier k_j in the order they appear in the route of mail carrier k_i, and vice versa. An example is given in Figure 3.11. At the top of the figure, the original routes of mail carriers k_1 and k_2 are depicted. The colors represent the tables, and the numbers represent the order of the street segments on the tables. We now swap the street segments from the red table of mail carrier k_1 with the street segments from the green table of mail carrier k_2. The resulting solution is given in the middle of the figure. However, in the case in which either mail carrier k_i is already visiting street segments assigned to table t_n or mail carrier k_j is already visiting street segments assigned to table t_m, it can happen that the precedence constraints are violated. In our example, mail carrier k_2 is already visiting two street segments of the red table, and the resulting route does not fulfill the precedence constraints. To overcome this problem, we additionally remove all street segments assigned to table t_m (t_n) from the route of mail carrier k_j (k_i) and reinsert them at the end of the route at the correct position according to the order they are assigned to the table. The final resulting solution is given at the bottom of the figure.

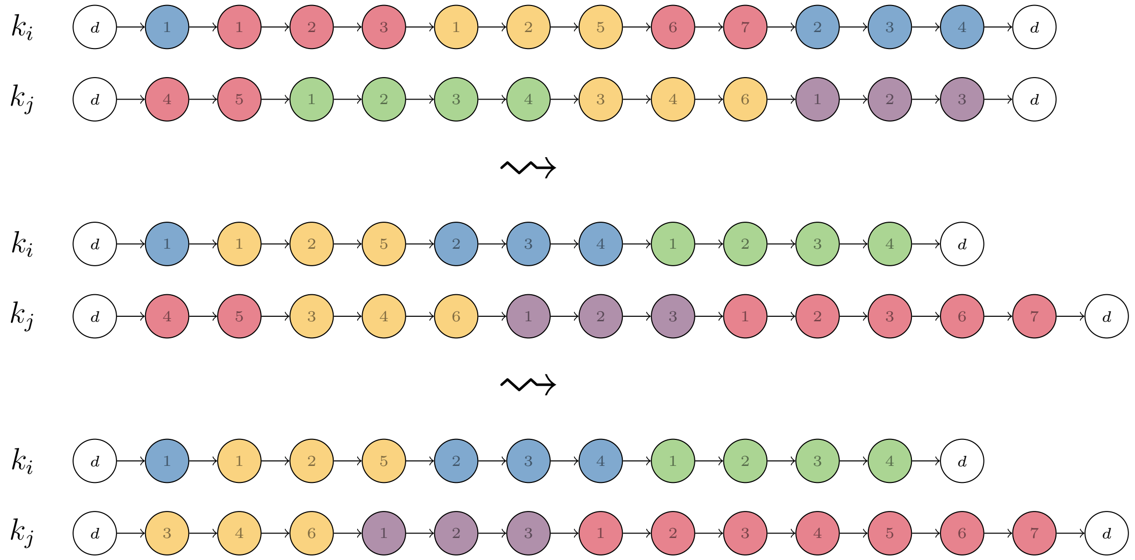

k_i d 1 1 2 3 1 2 5 6 7 2 3 4 d
k_j d 4 5 1 2 3 4 3 4 6 1 2 3 d
k_i d 1 1 2 5 2 3 4 1 2 3 4 d
k_j d 4 5 3 4 6 1 2 3 1 2 3 6 7 d
k_i d 1 1 2 5 2 3 4 1 2 3 4 d
k_j d 3 4 6 1 2 3 1 2 3 4 5 6 7 d

Figure 3.11: Example of the swap-tables operator applied to the routes of mail carriers k_i and k_j.

3.4.2.1.3 Composition of the generator arc set We speed up the search by avoiding to evaluate unpromising moves. Recall that the neighborhood operators are designed in such a way that the generator arc (i, j) is always inserted in the solution. Hence, we identify those arcs that are likely to be part of good-quality solutions, and we only consider these arcs as generator arcs. This process, that is called sparsification and gives rise to granular neighborhoods, was first proposed by Toth and Vigo, 2003 and later applied in multiple works (see, e.g., Prins et al., 2007; Escobar et al., 2014; Goeke, 2019).

To guarantee that for each street segment a given number of incident arcs is always included in the generator arc set (even if a street segment is isolated), we use a street segment-based sparsification method based on distance. For every street segment $i \in I$, we build the list L_i consisting of the κ closest street segments ($\kappa \in \mathbb{N}$) to i. The arcs defined by pairing a street segment $i \in I$ with a street segment $j \in L_i$ are used as generator arcs.

Based on the specific features of the VRPDOC, we also derive two strategies for enriching the generator arc set defined above. The first strategy consists of adding to set L_i all street segments that are assigned to the same table of i and that fulfill the precedence constraints. The arcs added based on this criterion always guarantee that the mail carrier and the table constraints are respected.

The second strategy is motivated by the presence of rural areas that DHL has to serve. In rural areas, there can be one or more villages located far away from each other. Each such village corresponds to a route (i.e., to a table at the depot). For the street segments of these routes, limiting the composition of set L_i as described above, implies that only arcs connecting i to street segments of the same route exist, and no arcs connecting them to other routes exist. Figure 3.12 shows an example of an instance with 13 street segments and three tables represented by three different colors. The arcs between the street segments correspond to the arcs identified by the street segment-based sparsification method with $\kappa = 2$ (i.e., for each street segment, only the two closest street segments are added to the set L_i). We observe that the blue route is isolated with no generator arcs connecting one of its street segments to those of other routes. Adding arcs connecting all street segments on the same table that fulfill the precedence constraints does not change this situation. To overcome this problem, the second strategy consists in enriching L_i with one vertex j_t per table $t \in T$, where j_t is the closest vertex to i on table t.

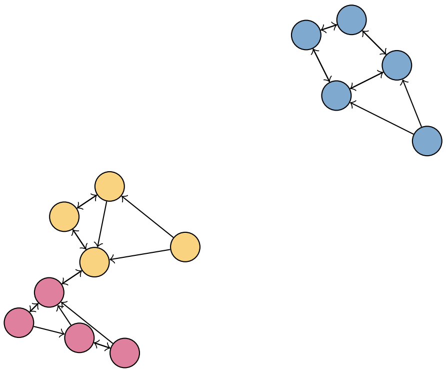


Figure 3.12: Example of the generator arc set for a rural instance obtained by only considering the street segment-based sparsification method based on distance with $\kappa = 2$.

We finally apply the preprocessing techniques `SegInc_p` and `Precedence` (Section 3.2.2.1) to remove from set L_i all those street segments generating infeasible arcs. Because we do not have a separate generator arc set for each mail carrier, for `SegInc_p`, two street segments are incompatible to be in the same route only if condition 3.24 is fulfilled for the maximum upper bound ub^k for $k \in K$.

Contrary to the relocate and exchange neighborhood operators, the swap-tables neighborhood operator is always defined by two mail carriers and two tables (k_1, k_2, t_m, t_n) and not by a generator arc (i, j). Consequently, we cannot apply the sparsification techniques described above. Because the number of mail carriers and the number of tables is usually a lot smaller compared to the number of street segments, and because it is hard to identify pairs of mails carriers leading to good swap-table moves, we search the complete swap-tables neighborhood.

3.4.2.2 Perturbation

To perturbate, we apply a number of random feasible moves defined by the operators presented in Section 3.4.2.1.2 to the overall best-found solution. Because the required perturbation strength depends on the size of an instance, we set the number of applied moves to $\lceil p_{perturb} \cdot |I| \rceil$, with $0 < p_{perturb} < 1$.

3.5 Computational experiments

The goal of our computational study is threefold:

- First, we investigate the performance of different formulations for the VRPDOC. We evaluate the effectiveness of using preprocessing techniques and valid inequalities with the best-performing formulation determined by pretests that are described in Appendix A (see Section 3.5.2).

- Second, we assess the performance of ILS-VRPDOC by comparing its results to the ones of a commercial solver for a set of small-scale instances, and to the solutions computed by DHL for a set of large-scale real-world instances. For the latter, we also compare the structure of the solutions by analyzing routing- and depot-operation-based solution metrics (see Sections 3.5.3 and 3.5.4).

- Third, we study the impact that the depot operation constraints have on the routing decisions and give managerial insights (see Section 3.5.5).

All experiments are performed on an Intel(R) Xeon(R) computer with a CPU E5-2430 v2 processor, at 2.50GHz with 64 GB RAM under Debian 12 (Bookworm) Slim. ILS-VRPDOC is implemented in C++ and compiled using gcc version 12.2. All mathematical models are solved with Gurobi 10.0.0. To allow for a fair comparison, all experiments are run on a single thread. If not stated otherwise, we always use the Gurobi default settings.

The parameter tuning of ILS-VRPDOC is described in Appendix B. We recall that the VRPDOC can be solved at both the tactical and the operational level. Hence, we investigate two different algorithmic variants: version ILS-VRPDOC$_{quality}$ focusing on solution quality and to be used at the tactical level, and version ILS-VRPDOC$_{speed}$ focusing on runtime and to be used at the operational level. The final parameter values for both ILS-VRPDOC$_{speed}$ and ILS-VRPDOC$_{quality}$ are summarized in Table 3.2.

	ILS-VRPDOC$_{quality}$	ILS-VRPDOC$_{speed}$
η	4000	150
τ	3600	60
κ	5	10
Additional generator arcs	included	included
$(\alpha_{\min}, \alpha_{\max}, \alpha_0)$	(0.1,10,1)	(0.1,10,1)
$p_{perturb}$	0.1	0.1

Table 3.2: Final parameter configurations for ILS-VRPDOC$_{quality}$ and ILS-VRPDOC$_{speed}$.

3.5.1 Description of test instances

Our computational experiments use two instance sets. The first set (Section 3.5.1.1) is composed of large-scale real-world instances provided by DHL. We use these instances for the parameter tuning and analysis of the ILS-VRPDOC components (described in Appendix B), to compare the solutions obtained with ILS-VRPDOC to the ones provided by DHL in Section 3.5.4, and to assess the effect of including depot operation constraints on the routing decisions in Section 3.5.5. The second instance set (Section 3.5.1.2) is composed of small-scale instances derived from the first instance set, that are used to analyze the VRPDOC formulations in Appendix A.7, to test the preprocessing techniques and valid inequalities in Section 3.5.2, and to compare the performance of ILS-VRPDOC and Gurobi in Section 3.5.3.

3.5.1.1 Large-scale instances

The instance set provided by DHL consists of 98 real-world instances representing the German city of Hannover and its surrounding area, i.e., the set contains a mix of urban and rural instances. Each instance specifies the complete distance matrix, the service times of the street segments, the assignment of street segments to the tables in the depot, the desired number of mail carriers $|K|$, such that either $|K| < |T|$ or $|K| > |T|$, and the upper and lower bounds on the route duration for each mail carrier. DHL places the latter symmetrically around a given target route duration based on the working hours of the mail carriers. The number of tables in the instances varies between six and 30, and the number of street segments varies between 273 and 1408.

DHL also shared with us the solutions that they computed using dynamic programming and simple heuristics (details cannot be provided here due to confidentiality reasons) and that they currently follow in practice. For 54 out of the 98 instances, the DHL solution violates the precedence constraints for some street segments. Consequently, the comparison is not completely accurate because the DHL solutions are not feasible for the problem described in the paper at hand. Nevertheless, our solutions strictly respect the precedence constraints also for these instances. To make the comparison between ILS-VRPDOC and DHL solutions as fair as possible, we set the values for skipping, mail carrier, table, and table changes of the DHL solution as input parameters of our ILS-VRPDOC.

The same delivery area (depot and street segments) can appear in different instances, e.g., with a different number of desired mail carriers, different bounds on the depot operation constraints, etc.

3.5.1.2 Small-scale instances

Because Gurobi cannot find feasible solutions to the DHL real-world instances in reasonable runtime, we derive a set of smaller instances from the DHL instances. As a basis, we select one urban and one rural instance from the DHL set. For both instances, we randomly choose subsets of 3, 5, and 10 tables, respectively, in such a way that the structure of the instance (i.e., urban or rural) is preserved. For each table, we randomly select subsets of 10, 15, and 20 street segments. For each combination (except for the one with 10 tables and 20 street segments for which Gurobi cannot find a feasible solution in reasonable runtime), the number of mail carriers is set to one mail carrier less and one mail carrier more than the number of tables. This leads to a total of $|\{\text{urban,rural}\}| \cdot (|\{3, 5, 10\}| \cdot |\{10, 15, 20\}| - 1) \cdot |\{\text{\#mail carriers-1,\#mail carriers+1}\}| = 2 \cdot (3 \cdot 3 - 1) \cdot 2 = 32$ instances.

3.5.2 Effectiveness of the preprocessing techniques and valid inequalities

By means of the computational experiments described in Appendix A.7, we established that the best-performing formulation is Formulation F1a. In this section, we evaluate the influence of the preprocessing techniques (Section 3.5.2.1) and valid inequalities (Section 3.5.2.2) on Formulation F1a to determine the final configuration of the model to use for the comparison with ILS-VRPDOC. For these experiments, we use the small-scale instances. The Gurobi time limit is set to two hours.

3.5.2.1 Impact of the preprocessing techniques

To assess the impact of the preprocessing techniques presented in Section 3.2.2.1, we first solve Formulation F1a without preprocessing techniques, and then with the inclusion of the preprocessing techniques `Prec` and `p_SegInc` separately. Table 3.3 shows the results. In the first column of the table, we report the tested configurations.

For each of the three configurations (No preprocessing, Prec, and p_SegInc), we report the number of instances for which at least a feasible solution is found (*#feas*), the average gap (*Gap*) and the average runtime (t) over those instances for which Gurobi finds a feasible solution. The results show that, by activating Prec, the number of instances for which a feasible solution is found within the time limit increases by two and the runtime required for solving the integer program decreases. Thus, we decide to include Prec in the final configuration of F1a.

The inclusion of p_SegInc has no effect on the results compared to those obtained without preprocessing techniques. In fact, in our instances, there are no pairs of street segments with incompatible service times, and all mail carriers have the same lower and upper bounds on the route duration. Nevertheless, these preprocessing techniques can be useful for more heterogeneous instances with, e.g., street segments with very high service times or mail carriers with different lower and upper bounds on the route duration. Because the inclusion of p_SegInc has neither a positive nor negative effect, we also add it to the final configuration of F1a.

Preprocessing technique	*#feas*	*Gap*(%)	$t(s)$
No preprocessing	29/32	6.88	4577.63
Prec	31/32	6.69	4130.06
p_SegInc	29/32	6.88	4577.62

Table 3.3: Results of Formulation F1a with and without preprocessing techniques.

3.5.2.2 Impact of valid inequalities

To evaluate the influence of the valid inequalities, we first solve Formulation F1a with Prec and p_SegInc without valid inequalities, and then, including each of the valid inequalities presented in Section 3.2.2.2 individually. The results are presented in Table 3.4, which has the same structure as Table 3.3. Because in our instances there are no pairs of street segments with incompatible service times, adding vi_SegInc has no effect on the results compared to those obtained with no valid inequalities. The inclusion of Cycle reduces the number of instances for which a feasible solution is found and raises the average gap and runtime. Using NumSegBounds, Gurobi finds a feasible solution within the time limit for all 32 instances. However, the average gap increases compared to the model with no valid inequalities. Including SymBreak cuts down the number of instances for which a feasible solution is found and deteriorates the average gap and runtime.

Because for testing the performance of ILS-VRPDOC we are especially interested in the number of instances for which a feasible solution is found, we refrain from including SymBreak. Including vi_SegInc has no negative effect and could potentially be helpful for other instances, for that reason we decided to include it. Considering that NumSegBounds leads to feasible solutions on all 32 instances while only slightly increasing the average gap, we decide to also include NumSegBounds in Formulation F1a. This leads to the final configuration: Formulation F1a with vi_SegInc, NumSegBounds, Precedence, and p_SegInc.

Valid inequalities	*#feas*	*Gap*(%)	$t(s)$
No valid inequalities	31/32	5.66	4020.58
vi_SegInc	31/32	5.67	4020.64
Cycle	28/32	5.98	4354.54
NumSegBounds	32/32	5.74	4213.18
SymmBreak	30/32	6.13	4204.32

Table 3.4: Results of Formulation F1a with Precedence and p_SegInc with and without valid inequalities.

3.5.3 Comparison between ILS-VRPDOC and Gurobi

In this section, we compare the results of ILS-VRPDOC$_{quality}$ and ILS-VRPDOC$_{speed}$ to the ones of Gurobi on the small-scale instances introduced in Section 3.5.1.2. The results of the comparison are presented in Table 3.5. The first four columns contain instance details: the instance ID, the number of street segments $|I|$, the number of tables at the depot $|T|$, and the number of desired mail carriers $|K|$. Columns 5–8 show the results of Gurobi, i.e., the best objective function value *Obj* and the best lower bound *LB* found in the time limit of two hours, the percentage gap of the objective function value to the lower bound ($\Delta_{LB}(\%) = (Obj - LB)/Obj$), and the average runtime in seconds per run $t(s)$. Instances solved to optimality are marked in bold. Note that because the default value of Gurobi for the MIPGap is 0.0001 (0.01%), an instance can be marked as "solved to optimality" although the percentage gap is still greater than 0. The final six columns contain, for each of our algorithmic variants, the percentage gap of the solution of the best $\Delta_{LB}^{best}(\%)$ and average $\Delta_{LB}^{avg}(\%)$ run to the best lower bound found by Gurobi, and the average runtime $t^a(s)$. In the bottom three rows of the table, we report the average values and the number of instances on which ILS-VRPDOC returns a better or equal solution quality than Gurobi.

Gurobi finds near-optimal solutions for all instances with three tables, but for instances with five and ten tables, the gaps of the best objective value to the best lower bound found by Gurobi after two hours are significant. This indicates that the complexity of an instance depends primarily on the number of tables and not on the number of street segments. These results also show that a commercial solver cannot solve real-world-sized instances (that can reach up to 30 tables) and that a heuristic approach is needed.

ILS-VRPDOC$_{quality}$ and ILS-VRPDOC$_{speed}$ find solutions that are very close to optimal on all instances solved to optimality by Gurobi. On these instances, the runtimes of ILS-VRPDOC$_{speed}$ stay below 15 seconds and the runtimes of ILS-VRPDOC$_{quality}$ below five minutes. On bigger instances, both ILS-VRPDOC$_{quality}$ and ILS-VRPDOC$_{speed}$ obtain better percentage gaps to the lower bounds than Gurobi for both the best and average run.

				Gurobi				ILS-VRPDOC$_{quality}$			ILS-VRPDOC$_{speed}$		
Instance	$\|I\|$	$\|T\|$	$\|K\|$	*Obj*	*LB*	$\Delta_{LB}(\%)$	$t(s)$	$\Delta_{LB}^{best}(\%)$	$\Delta_{LB}^{avg}(\%)$	$t^a(s)$	$\Delta_{LB}^{best}(\%)$	$\Delta_{LB}^{avg}(\%)$	$t^a(s)$
1	30	3	2	9450.37	9450.37	**0.00**	2.39	**0.00**	**0.00**	3.94	**0.00**	**0.00**	0.34
2	30	3	4	11429.57	11429.57	**0.00**	4.32	**0.00**	0.81	6.72	1.69	4.69	0.55
3	30	3	2	14055.00	14055.00	**0.00**	5.26	**0.00**	**0.00**	6.95	**0.00**	0.05	0.77
4	30	3	4	24093.58	24093.58	**0.00**	48.78	0.03	0.03	8.01	0.03	0.40	0.71
5	45	3	2	10904.39	10904.39	**0.00**	3.27	**0.00**	**0.00**	8.11	**0.00**	**0.00**	0.49
6	45	3	4	13405.58	13405.58	**0.00**	31.13	**0.00**	0.09	13.93	0.06	1.86	0.93
7	45	3	2	14861.41	14860.21	**0.01**	56.62	**0.01**	0.19	19.22	**0.01**	0.84	1.16
8	45	3	4	24526.63	24524.23	**0.01**	500.93	**0.01**	0.22	10.51	**0.01**	0.63	0.99
9	60	3	2	11533.07	11533.07	**0.00**	8.65	**0.00**	**0.00**	11.75	**0.00**	**0.00**	0.78
10	60	3	4	13769.60	13769.60	**0.00**	105.22	**0.00**	0.57	14.51	**0.00**	1.52	1.00
11	60	3	2	15798.17	15798.17	**0.00**	109.14	**0.00**	0.34	28.64	0.06	1.65	1.39
12	60	3	4	25561.51	25559.07	**0.01**	1888.10	**0.01**	0.15	21.15	**0.01**	0.72	1.50
13	50	5	4	12691.07	12096.94	4.68	7200.00	4.68	4.68	26.09	4.68	4.68	1.82
14	50	5	5	13727.54	12939.24	5.74	7200.00	4.18	4.18	25.27	4.18	4.24	1.51
15	50	5	4	26774.46	24855.66	7.17	7200.00	6.40	6.48	40.31	6.49	6.72	2.45
16	50	5	5	32359.31	29301.13	9.45	7200.00	6.76	7.13	58.70	6.76	7.30	2.68
17	75	5	4	16118.61	15661.05	2.84	7200.00	2.68	2.91	59.45	2.68	3.09	4.21
18	75	5	5	16999.79	16012.50	5.81	7200.00	6.15	6.41	80.71	6.62	6.68	3.46
19	75	5	4	27967.03	26249.85	6.14	7200.00	6.23	6.66	92.73	6.67	7.11	5.88
20	75	5	5	33080.56	30354.93	8.24	7200.00	7.51	7.93	99.22	7.03	8.42	5.04
21	100	5	4	18238.53	17551.37	3.77	7200.00	3.32	3.36	78.13	3.32	3.67	5.34
22	100	5	5	18329.00	17836.19	2.69	7200.00	2.69	2.69	73.11	2.69	2.69	4.34
23	100	5	4	28350.97	27024.20	4.68	7200.00	4.56	4.88	127.49	5.16	5.76	6.29
24	100	5	5	33485.13	31350.17	6.38	7200.00	4.89	5.50	189.77	5.35	6.39	8.24
25	50	10	9	19350.20	12959.24	33.03	7200.00	30.36	30.36	121.08	30.36	30.63	8.80
26	50	10	11	21457.66	13898.83	35.23	7200.00	30.93	30.93	124.38	30.93	31.05	8.18
27	50	10	9	61246.98	53202.10	13.14	7200.00	10.56	10.71	115.92	10.61	11.10	9.95
28	50	10	11	71858.71	63472.23	11.67	7200.00	10.88	11.06	178.41	11.30	11.45	10.07
29	100	10	9	27001.67	17925.72	33.61	7200.00	21.49	21.69	456.15	21.59	22.06	21.96
30	100	10	11	27815.09	18429.70	33.74	7200.00	24.23	24.44	551.61	24.56	25.23	22.96
31	100	10	9	74290.88	55345.93	25.50	7200.00	11.79	12.58	491.74	12.94	13.38	22.09
32	100	10	11	84598.75	64680.22	23.54	7200.00	12.70	13.05	569.55	12.97	13.66	22.72
Avg						8.66	4586.37	6.66	6.88	116.04	6.84	7.43	5.89
# better solutions								16/32	14/32		15/32	12/32	
# equal solutions								13/32	6/32		10/32	5/32	

Table 3.5: Comparison of the results obtained with ILS-VRPDOC$_{quality}$ and ILS-VRPDOC$_{speed}$ to the results obtained by Gurobi with a time limit of two hours.

3.5.4 Comparison of ILS-VRPDOC solutions and DHL solutions

In this section, we benchmark the results of the two ILS-VRPDOC variants against the ones provided by DHL for the large-scale instances. Table 3.6 shows the results. Columns 2 and 3 report the percentage gap of the objective value of our construction heuristic to the DHL objective value ($\Delta^{init}(\%)$) and the runtime in seconds ($t^{init}(s)$). The remaining columns display, for ILS-VRPDOC$_{quality}$ and ILS-VRPDOC$_{speed}$, respectively, the percentage gap of the solution of the best $\Delta^{best}(\%)$ and average $\Delta^{avg}(\%)$ run to the DHL solution, and the average runtime $t^a(s)$. We report the average values and the number of instances on which ILS-VRPDOC returns a better or equal solution than DHL. The detailed results for each instance can be found in Table 3.17 in Appendix C.

Focusing on the results provided by the construction heuristic, we observe that for 37.76% of instances, the quality of our initial solution is already better than the DHL solution. However, the average gap over all instances is positive (3.08%). The average runtime of our construction heuristic is around 20 seconds, i.e., it finds feasible solutions very quickly, also for large instances.

ILS-VRPDOC$_{quality}$ improves the DHL solution quality on 94.90% of instances, and the average improvement of the best run with respect to the DHL solutions is 6.21%. Comparing the gaps of the best run to those of the average run, ILS-VRPDOC$_{quality}$ shows a robust behavior. As expected, ILS-VRPDOC$_{quality}$ finds better solutions than ILS-VRPDOC$_{speed}$ but exhibits higher runtimes. ILS-VRPDOC$_{speed}$ still improves the DHL solution quality on 92.86% of instances. The average improvement of the best run with respect to the DHL solutions is 5.10%. Again, when comparing the average gaps of the best run to those of the average run, ILS-VRPDOC$_{speed}$ shows a robust behavior. ILS-VRPDOC$_{speed}$ requires less than one minute per run, on average, to deliver good-quality solutions and is therefore a convincing choice to be used at the operational level. Only if runtime is not important, the clearly longer runtimes of ILS-VRPDOC$_{quality}$ are justified to reach some additional solution quality.

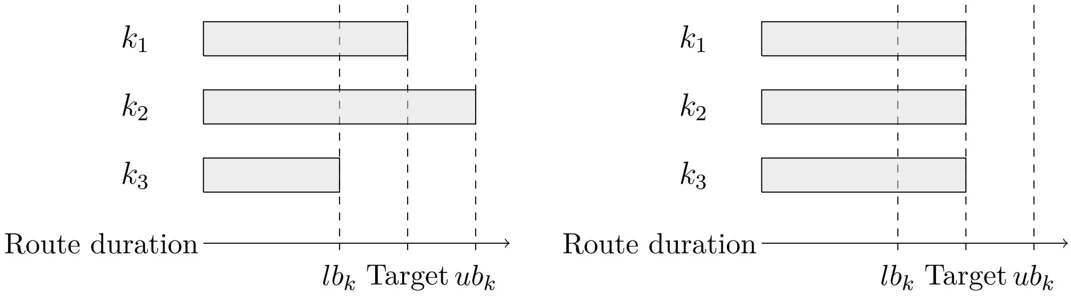
	ILS-VRPDOC$_{init}$		ILS-VRPDOC$_{quality}$			ILS-VRPDOC$_{speed}$		
	$\Delta^{init}(\%)$	$t^{init}(s)$	$\Delta^{b}(\%)$	$\Delta^{a}(\%)$	$t^{a}(s)$	$\Delta^{b}(\%)$	$\Delta^{a}(\%)$	$t^{a}(s)$
Avg	3.08	20.66	-6.21	-5.75	2943.31	-5.10	-4.34	59.12
# better solutions	37/98		93/98	93/98		91/98	89/98	
# equal solutions	1/98		1/98	1/98		1/98	1/98	

Table 3.6: Comparison of the DHL solutions to the solutions of our construction heuristic, and of the best and average run of ILS-VRPDOC$_{quality}$ and ILS-VRPDOC$_{speed}$.

To better understand why ILS-VRPDOC returns better solutions than those of DHL, we compare the structure of the DHL solutions to the structure of the best solutions found by ILS-VRPDOC$_{quality}$. For this comparison, we first focus on the features related to the routing (Section 3.5.4.1), and then on the features related to the depot operation constraints (Section 3.5.4.2).

3.5.4.1 Comparison of the routing in DHL and ILS-VRPDOC solutions

To compare the routes of the DHL to the ILS-VRPDOC$_{quality}$ solutions, we use the maximum and average deviation from the target route duration, i.e., the middle value between the lower and the upper bound on the route duration, and the similarity of the routes. To compute the former, we apply the following three steps: (i) we determine the target for each mail carrier, (ii) for each mail carrier, we compute the deviation of the actual route duration from this middle value, (iii) we report the maximum deviation across all mail carriers, and the average deviation over all mail carriers of a solution. Two exemplary solutions of an instance with three mail carriers (k_1, k_2, and k_3) are shown in Figure 3.13. For the sake of simplicity, we assume that all three mail carriers have the same lower (lb_k) and upper bound (ub_k) on the route duration. Both solutions have the same average route duration over all mail carriers. However, both the maximum and the average deviation from the desired route duration differ. Even if both solutions fulfill the lower and upper bound constraints on the route duration, the solution in Figure 3.13(b) shows a more balanced workload because the amount of overtime and undertime is minimized.



(a) Solution with larger deviations in the route duration.

(b) Solution with lower deviations in the route duration.

Figure 3.13: Comparison of two exemplary solutions 3.13(a) and 3.13(b) with same average route duration over all mail carriers and different route duration deviations.

For the routes similarity metric, we compute the number of arcs that are present both in the DHL and the ILS-VRPDOC solution and divide it by the total number of arcs in the solution. For all metrics, we report the average, maximum and minimum value and the number of instances for which the ILS-VRPDOC solution has an improved or equal value of the considered metric than the DHL solution.

Aggregated results are shown in Table 3.7. The detailed results for each instance can be found in Table 3.18 in Appendix D. In the ILS-VRPDOC solution, the maximum deviation of the route duration is lower while the average deviation is higher than the deviation in the DHL solution. This implies that, in our solutions, even though the route durations deviate more from the middle value, they are more equally balanced among the mail carriers.

Although neither the objective function of the VRPDOC nor its constraints consider the route similarity with respect to the DHL solutions, it is interesting to observe this metric because greater route similarity enables a mail carrier to learn the new route more quickly, reducing resistance to the change. The average similarity over all instances lies above 90%. These high values for route similarity are explained by the need of keeping the sequence in which the street segments are visited as similar as possible to their sorting on the tables.

3.5.4.2 Comparison of the depot operations in DHL and ILS-VRPDOC solutions

To compare the situation at the depot in the DHL and the ILS-VRPDOC$_{quality}$ solutions, we use the following metrics: the maximum number of mail carriers per table, the average number of mail carriers per table, the maximum number of tables over all mail carriers, the number of letter blocks skipped by mail carriers, and the total number of table changes. These metrics are not part of the objective of VRPDOC, but they are considered as constraints. Because we chose as upper bounds of these constraints exactly the values of the DHL solutions, the values of the ILS-VRPDOC solutions can only be equal or better. Nevertheless, it is interesting to analyze the differences between the DHL and our solutions with respect to these metrics to better understand the differences in complexity of the depot operations.

In Table 3.8, we show the results, averaged over all instances. For all metrics we report the average, maximum and minimum value and the number of instances where the ILS-VRPDOC solution has an improved or equal value of the considered metric than the DHL solution. The detailed results for each instance are reported in Table 3.19 in Appendix E.

As explained, in the ILS-VRPDOC solutions, the average values of all metrics are lower than those in the DHL solutions. The most remarkable difference is in the maximum number of mail carriers per table and in the maximum number of tables visited by a mail carrier. On average, in the DHL solutions, there is at least one table that is visited by more than six mail carriers and one mail carrier visiting more than five tables. In our solutions, this numbers are reduced to three mail carriers and three tables, respectively. We observe that in the ILS-VRPDOC solutions, the number of table changes is only smaller than in the DHL solutions for five out of the 98 instances, which is an indicator that this is the constraint limiting the solution space the most.

From the practical point of view, compared to the DHL solutions, ILS-VRPDOC delivers solutions in which:

- Fewer mail carriers visit the same tables on average. As a result, less confusion is generated at each table.

- Each mail carrier visits fewer tables on average. This implies that each mail carrier has to move less between different tables.

- Each mail carrier skips fewer letter blocks while collecting letters from the tables. As a result, the letter collection operations are simplified, leading to a smaller chance of making mistakes and to a faster learning process.

- There are fewer movements of mail carriers between tables at the depot. This decreases potential confusion in the depot during letter collection operations as well as the risk of accidents.

3.5.5 The impact of the depot operation constraints

In this section, we assess the influence of the depot operation constraints on the objective function value, on the routing metrics, and on the metrics describing the situation at the depot. To this end, we compare the solutions of the large-scale instances of the best run of ILS-VRPDOC$_{quality}$ to solutions that we obtain with a modified version of ILS-VRPDOC$_{quality}$ that ignores all depot operation constraints. We call this version of our heuristic "ILS-VRPDOC$^{OFF}_{quality}$". Tables 3.9 and 3.10 are structured like the tables in Section 3.5.4 with the only difference that we add two columns that include the objective function values of ILS-VRPDOC$_{quality}$ and ILS-VRPDOC$^{OFF}_{quality}$ to Table 3.9.

Table 3.9 shows the routing metrics averaged over all instances. The detailed results per instance can be found in Table 3.20 in Appendix F. As expected, ILS-VRPDOC$^{OFF}_{quality}$ returns total travel times lower than ILS-VRPDOC$_{quality}$. By considering the depot operation constraints, on average, the total travel time increases by 10.72%. The increase of the maximum deviation of the route duration is small for ILS-VRPDOC$^{OFF}_{quality}$, but the increase of the average deviation of the route duration is significant. This result is in line with the finding of Matl et al., 2018 reporting that cost-optimal solutions of the CVRP tend to lead to poorly balanced routes. For ILS-VRPDOC$^{OFF}_{quality}$, the similarity of the routes with respect to the DHL solution decreases drastically compared to the similarity of ILS-VRPDOC$_{quality}$. For mail carriers, this would mean that they have to relearn a large share of their routes.

	Service time max deviation		Service time avg deviation		
	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	Similarity
Avg	38.41	36.72	8.39	15.29	92.34
Max	86.56	78.36	23.79	37.99	97.92
Min	8.99	8.99	2.98	2.77	82.95
# improved value		41/98		8/98	
# equal value		57/98		7/98	

Table 3.7: Routing-based comparison of the DHL solutions to the solutions of the best run of ILS-VRPDOC$_{quality}$.

	Max #mail carriers		Avg #mail carriers		Max #tables		Avg #tables		Skips		Table changes	
	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC
Avg	6.96	2.96	1.69	1.61	5.43	2.71	1.82	1.73	8.50	7.56	15.83	15.78
Max	27.00	10.00	2.50	2.42	24.00	9.00	2.83	2.48	53.00	53.00	47.00	47.00
Min	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.00	0.00	0.00	0.00
# improved value		39/98		57/98		35/98		57/98		18/98		5/98
# equal value		59/98		31/98		63/98		31/98		80/98		93/98

Table 3.8: Depot operation-based comparison of the DHL solutions to the solutions of the best run of ILS-VRPDOC$_{quality}$.

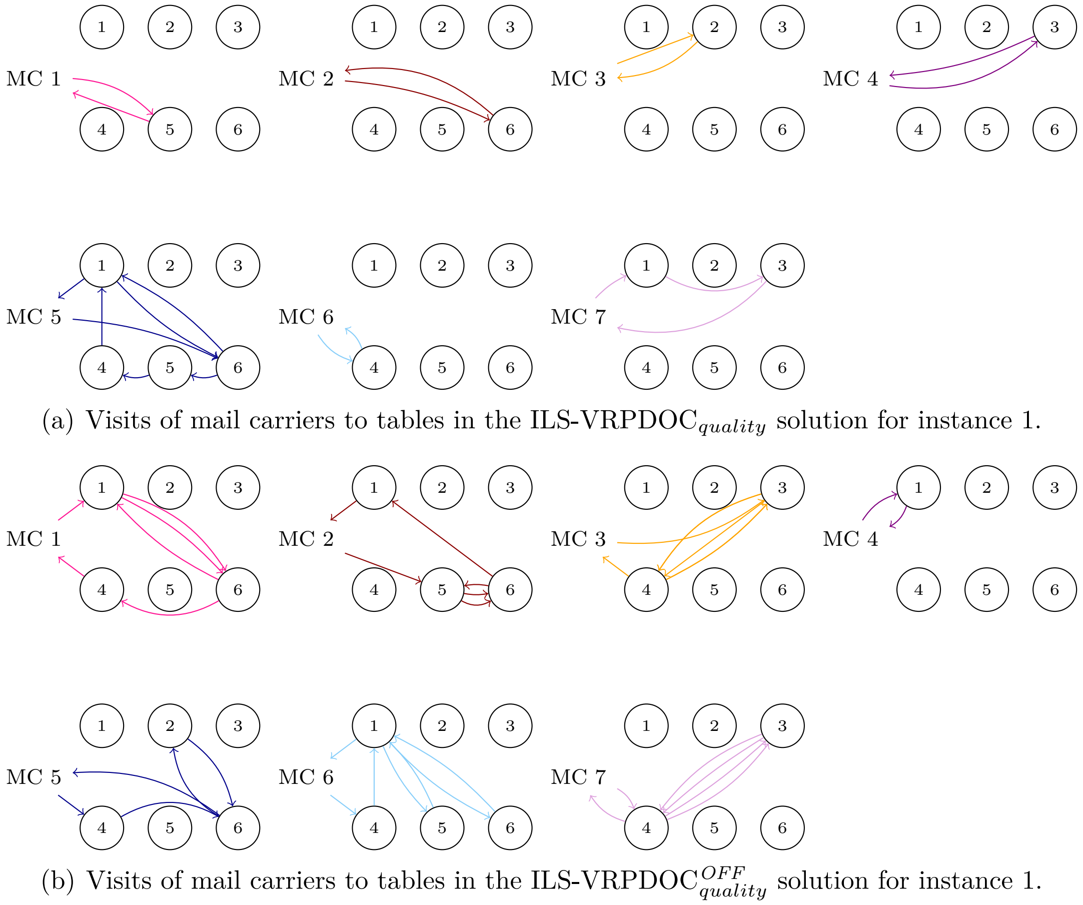
	Objective		Service time max deviation		Service time avg deviation		Similarity to DHL solutions	
	ILS-VRPDOC$_{quality}$	ILS-VRPDOC$^{OFF}_{quality}$	ILS-VRPDOC$_{quality}$	ILS-VRPDOC$^{OFF}_{quality}$	ILS-VRPDOC$_{quality}$	ILS-VRPDOC$^{OFF}_{quality}$	ILS-VRPDOC$_{quality}$	ILS-VRPDOC$^{OFF}_{quality}$
Avg	104570.82	94446.52	36.72	37.89	15.29	22.80	92.34	33.88
Max	209600.72	170253.63	78.36	86.03	37.99	51.30	97.92	63.35
Min	34075.42	29947.85	8.99	8.92	2.77	5.99	82.95	22.20
# improved value		98/98		40/98		11/98		0/98
# equal value		0/98		0/98		0/98		0/98

Table 3.9: Routing-based comparison of the best run of ILS-VRPDOC$_{quality}$ to the solutions of the best run of ILS-VRPDOC$^{OFF}_{quality}$.

	Max #mail carriers		Avg #mail carriers		Max #tables		Avg #tables		Skips		Table changes	
	ILS-VRPDOC$_{quality}$	ILS-VRPDOC$^{OFF}_{quality}$	ILS-VRPDOC$_{quality}$	ILS-VRPDOC$^{OFF}_{quality}$	ILS-VRPDOC$_{quality}$	ILS-VRPDOC$^{OFF}_{quality}$	ILS-VRPDOC$_{quality}$	ILS-VRPDOC$^{OFF}_{quality}$	ILS-VRPDOC$_{quality}$	ILS-VRPDOC$^{OFF}_{quality}$	ILS-VRPDOC$_{quality}$	ILS-VRPDOC$^{OFF}_{quality}$
Avg	2.96	8.65	1.61	4.58	2.71	8.65	1.73	5.00	7.56	437.35	15.78	125.40
Max	10.00	15.00	2.42	6.54	9.00	15.00	2.48	8.38	53.00	776.00	47.00	235.00
Min	1.00	4.00	1.00	2.06	1.00	4.00	1.00	2.71	0.00	88.00	0.00	26.00
# improved value		0/98		0/98		1/98		0/98		0/98		0/98
# equal value		2/98		0/98		2/98		0/98		0/98		0/98

Table 3.10: Depot operation-based comparison of the best run of ILS-VRPDOC$_{quality}$ to the solutions of the best run of ILS-VRPDOC$^{OFF}_{quality}$.

Table 3.10 shows the metrics describing the situation at the depot. Detailed results per instance are reported in Table 3.21 in Appendix G. As expected, all the metrics are clearly higher for ILS-VRPDOC$_{quality}^{OFF}$ than for ILS-VRPDOC$_{quality}$. In particular, the number of skips and the number of table changes drastically increase. This enlarges the chances of mistakes or accidents during the letter collection operations due to the high number of mail carriers moving from one table to another in the depot. To illustrate this Figure 3.14 shows the visits of each mail carrier to tables determined by the solution of ILS-VRPDOC$_{quality}$ (Figure 3.14(a)) and of ILS-VRPDOC$_{quality}^{OFF}$ (Figure 3.14(b)) for instance 1. This is a small instance with six tables represented by the circles in the figures, and a desired number of mail carriers (MC) of seven. We observe that the operations corresponding to the solution provided by ILS-VRPDOC$_{quality}$ are much less complex and error-prone than those of ILS-VRPDOC$_{quality}^{OFF}$. In the latter solution, most mail carriers have to visit several tables and have to come back to the same table multiple times.



(a) Visits of mail carriers to tables in the ILS-VRPDOC$_{quality}$ solution for instance 1.

(b) Visits of mail carriers to tables in the ILS-VRPDOC$_{quality}^{OFF}$ solution for instance 1.

Figure 3.14: Comparison of visits of mail carriers to tables in the different solutions (ILS-VRPDOC$_{quality}$ and ILS-VRPDOC$_{quality}^{OFF}$) for instance 1.

To analyze the letter collection at each table for instance 1 in more detail, Table 3.11

contains, for each mail carrier, tuples consisting of the table ID and of lists representing the blocks of consecutive letters that the mail carrier has to collect from the table. Table 3.10(a) reports the ILS-VRPDOC$_{quality}$ solution, and Table 3.10(b) shows the ILS-VRPDOC$^{OFF}_{quality}$ solution. Table 3.10(a) shows that in the ILS-VRPDOC$_{quality}$ solution big blocks of consecutive letters can be collected by mail carriers in one go. On the contrary, the ILS-VRPDOC$^{OFF}_{quality}$ solution is very complicated as represented by the huge number of non-consecutive blocks of letters that the mail carrier has to collect.

3.6 Conclusion

Motivated by DHL's task to find cost-efficient routing solutions that lead to simplified letter collection processes at the depot, we introduce the VRPDOC. The feasibility problem of the VRPDOC is already NP-complete, however, by relaxing the upper and lower bound on the route duration a feasible solution can be found in polynomial time. We propose two metaheuristic variants to address realistically sized instances—ILS-VRPDOC$_{quality}$ for solving the VRPDOC at the tactical level and ILS-VRPDOC$_{speed}$ for solving it at the operational level. Both heuristics show a convincing performance and are able to find (close to) optimal solutions on small-scale instances and to significantly improve the DHL solutions on larger instances. In many cases, our solutions improve the travel cost and, at the same time, simplify the operations at the depot, i.e., they are faster, generate less confusion and mistakes, decrease the need for coordination, and lead to faster learning processes. Our numerical studies show that travel times could be reduced by 11% on average if depot operation constraints are ignored, however, ignoring them extremely complicates the letter collection at the tables. Future research should, e.g., focus on obtaining a good assignment of the street segments to the tables with the goal of achieving more robust solutions for the different numbers of desired mail carriers.

Mail carrier	Letter collection sequence (table, [street segments])
1	(5,[187-229])
2	(6,[239-275])
3	(2,[48-86])
4	(3,[87-120])
5	(6,[230-233], (5,[183-186]), (4,[136-138]), (1,[3-13]), (6,[234-238]), (1,[14-18])
6	(4,[135, 139-182])
7	(1,[1-2, 19-47]), (3,[121-134])

(a) Letter collection sequence at tables by mail carrier in the ILS-VRPDOC$_{quality}$ solution for instance 1.

Mail carrier	Letter collection sequence (table, [street segments])
1	(1,[1]), (6,[239, 254, 253, 250-252, 262, 255-257, 260-261, 263-266, 269, 273, 270, 267, 268, 271-272, 249, 248, 247, 244-246, 243]), (1,[4, 3]), (6,[242, 240-241]), (4,[138])
2	(5,[183-186]), (6,[234, 236, 238]), (5,[192-214, 216, 220, 219, 221, 218, 217, 215, 222, 228]), (6,[235]), (1,[13])
3	(3,[87]), (4,[140, 151]), (3,[88, 90-91, 94, 96, 95, 97-99, 113]), (4,[157-168]), (3,[100-102, 104-105, 103, 106-110, 112, 111]) (4,[177, 176, 156, 155, 149, 143]), (6,[233, 231])
4	(1,[19-26, 28-29, 32, 34-38, 40-44, 46, 47, 45, 39, 33, 30, 31, 27, 15-16, 18, 17])
5	(4,[139]), (6,[232]), (2,[79, 81, 83-85, 82, 80, 78, 76, 75, 62-64, 56-58, 69, 68, 59, 60-61, 67, 66, 65, 49, 55, 50-54, 48, 70, 71-74, 77, 86]), (6,[274-275, 258-259])
6	(4,[135, 137, 136]), (1,[7, 6, 5, 8-12]), (5,[187-188, 191, 189-190, 223-227, 229, 237]), (1,[14]), (6,[230]), (1,[2])
7	(4,[144, 147, 145-146, 148, 150, 152-153, 170-175, 178-182]), (3,[92-93, 115-118, 120-127, 130, 128-129, 131-134, 119, 114]), (4,[169, 154]), (3,[89]), (4,[142, 141])

(b) Letter collection sequence at tables by mail carrier in the ILS-VRPDOC$^{OFF}_{quality}$ solution for instance 1.

Table 3.11: Letter collection sequence at tables in the different solutions (ILS-VRPDOC$_{quality}$ and ILS-VRPDOC$^{OFF}_{quality}$) for instance 1.

References

C. Archetti, N. Bianchessi, S. Irnich, and M. G. Speranza (2014). “Formulations for an inventory routing problem”. In: *International Transactions in Operational Research* 21.3, pp. 353–374. DOI: 10.1111/itor.12076.

L. S. d. Assis, P. M. Franca, and F. L. Usberti (2014). “A redistricting problem applied to meter reading in power distribution networks”. In: *Computers & Operations Research* 41, pp. 65–75. DOI: 10.1016/j.cor.2013.08.002.

M. Darvish, L. C. Coelho, and R. Jans (2020). *Comparison of symmetry breaking and input ordering techniques for routing problems*. FSA-2020-008. CIRRELT.

A. Dohn, M. S. Rasmussen, and J. Larsen (2011). “The vehicle routing problem with time windows and temporal dependencies”. In: *Networks* 58.4, pp. 273–289. DOI: 10.1002/net.20472.

J. W. Escobar, R. Linfati, and P. Toth (2014). “A hybrid granular tabu search algorithm for the multi-depot vehicle routing problem”. In: *Journal of the Operational Research Society* 65.1, pp. 37–48. DOI: 10.1057/jors.2013.102.

M. R. Garey and D. S. Johnson (1979). *Computers and intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company.

A. Gliesch, M. Ritt, A. H. S. Cruz, and M. C. d. O. Moreira (2020a). “A heuristic algorithm for districting problems with similarity constraints”. In: *2020 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, pp. 1–8. DOI: 10.1109/CEC48606.2020.9185552.

A. Gliesch, M. Ritt, A. H. S. Cruz, and M. C. d. O. Moreira (2020b). “A hybrid heuristic for districting problems with routing criteria”. In: *2020 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, pp. 1–9. DOI: 10.1109/CEC48606.2020.9185863.

D. Goeke (2019). “Granular tabu search for the pickup and delivery problem with time windows and electric vehicles”. In: *European Journal of Operational Research* 278.3, pp. 821–836. DOI: 10.1016/j.ejor.2019.05.010.

S. R. A. Haddadene, N. Labadie, and C. Prodhon (2016). “A GRASP× ILS for the vehicle routing problem with time windows, synchronization and precedence constraints”. In: *Expert Systems with Applications* 66, pp. 274–294. DOI: 10.1016/j.eswa.2016.09.002.

S. Irnich (2008). “A Unified Modeling and Solution Framework for Vehicle Routing and Local Search-Based Metaheuristics”. In: *INFORMS Journal on Computing* 20.2, pp. 270–287. DOI: 10.1287/ijoc.1070.0239.

S. Irnich and D. Villeneuve (2006). “The shortest-path problem with resource constraints and k-cycle elimination for $k \geq 3$”. In: *INFORMS Journal on Computing* 18.3, pp. 391–406. DOI: 10.1287/ijoc.1040.0117.

J. Kalcsics and R. Z. Ríos-Mercado (2019). “Districting problems”. In: *Location science*. Springer, pp. 705–743. DOI: 10.1007/978-3-030-32177-2_25.

Ç. Koç, T. Bektaş, O. Jabali, and G. Laporte (2016). “Thirty years of heterogeneous vehicle routing”. In: *European Journal of Operational Research* 249.1, pp. 1–21. DOI: 10.1016/j.ejor.2015.07.020.

R. Lahyani, L. C. Coelho, and J. Renaud (2018). “Alternative formulations and improved bounds for the multi-depot fleet size and mix vehicle routing problem”. In: *OR Spectrum* 40.1, pp. 125–157. DOI: 10.1007/s00291-017-0494-y.

G. Laporte, Y. Nobert, and M. Desrochers (1985). “Optimal routing under capacity and distance restrictions”. In: *Operations Research* 33.5, pp. 1050–1073. DOI: 10.1287/opre.33.5.1050.

V. Leggieri and M. Haouari (2017). “Lifted polynomial size formulations for the homogeneous and heterogeneous vehicle routing problems”. In: *European Journal of Operational Research* 263.3, pp. 755–767. DOI: 10.1016/j.ejor.2017.05.039.

J.-Q. Li, P. B. Mirchandani, and D. Borenstein (2009). “Real-time vehicle rerouting problems with time windows”. In: *European Journal of Operational Research* 194.3, pp. 711–727. DOI: 10.1016/j.ejor.2007.12.037.

M. López-Ibáñez, J. Dubois-Lacoste, L. Cáceres Pérez, M. Birattari, and T. Stützle (2016). “The irace package: Iterated racing for automatic algorithm configuration”. In: *Operations Research Perspectives* 3, pp. 43–58. DOI: 10.1016/j.orp.2016.09.002.

H. R. Lourenço, O. C. Martin, and T. Stützle (2003). “Iterated Local Search”. In: *Handbook of Metaheuristics*. Springer US, pp. 320–353. DOI: 10.1007/0-306-48056-5_11.

P. Matl, R. F. Hartl, and T. Vidal (2018). “Workload Equity in Vehicle Routing Problems: A Survey and Analysis”. In: *Transportation Science* 52.2, pp. 239–260. DOI: 10.1287/trsc.2017.0744.

Q. Mu, Z. Fu, J. Lysgaard, and R. Eglese (2011). “Disruption management of the vehicle routing problem with vehicle breakdown”. In: *Journal of the Operational Research Society* 62.4, pp. 742–749. DOI: 10.1057/jors.2010.19.

M. Nikolić and D. Teodorović (2015). “Vehicle rerouting in the case of unexpectedly high demand in distribution systems”. In: *Transportation Research Part C: Emerging Technologies* 55, pp. 535–545. DOI: 10.1016/j.trc.2015.03.002.

C. Prins, C. Prodhon, A. Ruiz, P. Soriano, and R. Wolfler Calvo (2007). “Solving the Capacitated Location-Routing Problem by a Cooperative Lagrangean Relaxation-Granular Tabu Search Heuristic”. In: *Transportation Science* 41.4, pp. 470–483. DOI: 10.1287/trsc.1060.0187.

N. Razali (2015). “An Efficient Genetic Algorithm for Large Scale Vehicle Routing Problem Subject to Precedence Constraints”. In: *Procedia - Social and Behavioral Sciences* 195, pp. 1922–1931. DOI: 10.1016/j.sbspro.2015.06.203.

R. Z. Ríos-Mercado and J. F. López-Pérez (2013). “Commercial territory design planning with realignment and disjoint assignment requirements”. In: *Omega* 41.3, pp. 525–535. DOI: 10.1016/j.omega.2012.08.002.

M. Schneider and M. Löffler (2019). “Large composite neighborhoods for the capacitated location-routing problem”. In: *Transportation Science* 53.1, pp. 301–318. DOI: 10.1287/trsc.2017.0770.

R. Spliet, A. F. Gabor, and R. Dekker (2014). “The vehicle rescheduling problem”. In: *Computers & Operations Research* 43, pp. 129–136. DOI: 10.1016/j.cor.2013.09.009.

P. Toth and D. Vigo (2002). *The Vehicle Routing Problem.* Society for Industrial and Applied Mathematics. DOI: 10.1137/1.9780898718515.

P. Toth and D. Vigo (2003). “The granular tabu search and its application to the vehicle-routing problem”. In: *INFORMS Journal on Computing* 15.4, pp. 333–346. DOI: 10.1287/ijoc.15.4.333.24890.

T. Vidal, G. Laporte, and P. Matl (2020). “A concise guide to existing and emerging vehicle routing problem variants”. In: *European Journal of Operational Research* 286.2, pp. 401–416. DOI: 10.1016/j.ejor.2019.10.010.

3.7 Appendix

A Alternative model formulations

In this section, we present multiple formulations for the VRPDOC. In Section A.7 we discuss, the performance of the different formulations and motivate our decision of using the formulation presented in Section 3.2.

A.1 Model Formulation F1a

For the model Formulation F1a and its description, the reader can refer to Section 3.2.

A.2 Model Formulation F1b

Instead of using the decision variable $u_i^k \in \mathbb{Z}, i \in L, k \in K$ representing the position of location i in the route of mail carrier k, we use the decision variable $v_i^k \in \mathbb{R}, i \in L, k \in K$ representing the sum of service times "collected" in the route of mail carrier k after leaving location i. Consequently, we replace constraints (3.13)–(3.17) in Formulation F1a with the following constraints to obtain model Formulation F1b:

$$v_i^k \leq ub^k \qquad i \in I,\ k \in K \tag{3.46}$$

$$v_i^k \geq s_i \qquad i \in I,\ k \in K \tag{3.47}$$

$$v_i^k - v_j^k + ub^k \cdot x_{ij}^k + (ub^k - s_i - s_j) \cdot x_{ji}^k \leq ub^k - s_j \qquad (i,j) \in A,\ k \in K \tag{3.48}$$

$$v_d^k = 0 \qquad k \in K \tag{3.49}$$

$$v_i^k - v_j^k \leq (2 - y_i^k - y_j^k) \cdot ub^k \qquad (i,j) \in A : p_{ij} = 1,\ k \in K \tag{3.50}$$

Because now $v_i^k \in \mathbb{R}, i \in L, k \in K$ represents the sum of service times "collected" in the route of mail carrier k before reaching location i, we have to adapt the upper bound on v_i^k to ub^k in constraints (3.46) and we can introduce a lower bound in constraints (3.47). In constraints (3.48), we fix $v_j^k \geq v_i^k + s_j$ in case $x_{ij}^k = 1$ and $v_i^k \leq v_j^k + s_i$ in case $x_{ji}^k = 1$, and we fix v_d^k to be zero in constraints (3.49). The precedence constraints in constraints (3.50) are modeled exactly the same way as in Formulation F1a.

A.3 Model Formulation F2a

To reduce the number of variables in the model, we remove the y-variables from Formulation F1a and replace them with $y_i^k = \sum_{j \in \delta_i^+} x_{ij}^k$. We obtain the following Formulation F2a:

$$\min \quad \sum_{k \in K} \sum_{(i,j) \in A} t_{ij} x_{ij}^k \tag{3.51}$$

$$\text{s.t.} \quad \sum_{k \in K} \sum_{j \in \delta_i^+} x_{ij}^k = 1 \qquad i \in I \tag{3.52}$$

$$\sum_{k \in K} \sum_{i \in \delta_j^-} x_{ij}^k = 1 \qquad j \in I \tag{3.53}$$

$$\sum_{j \in \delta_i^-} x_{ji}^k = \sum_{j \in \delta_i^+} x_{ij}^k \qquad i \in I,\ k \in K \tag{3.54}$$

$$\sum_{j \in \delta_d^+} x_{dj}^k = 1 \qquad k \in K \tag{3.55}$$

$$\sum_{i \in \delta_d^-} x_{id}^k = 1 \qquad k \in K \tag{3.56}$$

$$\sum_{i \in I} s_i \sum_{j \in \delta_i^+} x_{ij}^k + \sum_{(i,j) \in A} t_{ij} x_{ij}^k \leq ub^k \qquad k \in K \tag{3.57}$$

$$\sum_{i \in I} s_i \sum_{j \in \delta_i^+} x_{ij}^k + \sum_{(i,j) \in A} t_{ij} x_{ij}^k \geq lb^k \qquad k \in K \tag{3.58}$$

$$a_i^t \sum_{j \in \delta_i^+} x_{ij}^k \leq z^{kt} \qquad i \in I,\ k \in K,\ t \in T \tag{3.59}$$

$$\sum_{k \in K} z^{kt} \leq ub_{mailcarriers} \qquad t \in T \tag{3.60}$$

$$\sum_{t \in T} z^{kt} \leq ub_{tables} \qquad k \in K \tag{3.61}$$

$$\sum_{i \in I} \sum_{j \in I : j \neq i} \sum_{k \in K} x_{ij}^k (1 - same_{ij}) \leq ub_{tablechanges} \tag{3.62}$$

$$u_i^k \leq M \sum_{(i,j) \in \delta_i^+} x_{ij}^k \qquad i \in I,\ k \in K \tag{3.63}$$

$$u_i^k - u_j^k + 1 \leq M(1 - x_{ij}^k) \qquad (i,j) \in A,\ k \in K \tag{3.64}$$

$$u_i^k - u_j^k + 1 \geq -M(1 - x_{ij}^k) \qquad (i,j) \in A,\ k \in K \tag{3.65}$$

$$u_d^k = 1 \qquad k \in K \tag{3.66}$$

$$u_i^k - u_j^k \leq (2 - \sum_{m \in \delta_i^+} x_{im}^k - \sum_{m \in \delta_j^+} x_{jm}^k) \cdot M \qquad (i,j) \in A : p_{ij} = 1,\ k \in K \tag{3.67}$$

$$\sum_{i \in I} \sum_{j \in I : j \neq i} \sum_{k \in K} x_{ij}^k \cdot same_{ij} \cdot (1 - seq_{ij}) \leq ub_{skipping} \tag{3.68}$$

$$x_{ij}^k \in \{0, 1\} \qquad (i,j) \in A,\ k \in K \tag{3.69}$$

$$u_i^k \in \mathbb{Z} \qquad i \in L,\ k \in K \tag{3.70}$$

A.4 Model Formulation F2b

We replace the decision variable $u_i^k \in \mathbb{Z}, i \in L, k \in K$ representing the position of location i in the route of mail carrier k with the decision variable $v_i^k \in \mathbb{R}, i \in L, k \in K$ representing the sum of service times "collected" in the route of mail carrier k after leaving location i, as done in F1b. Consequently, we can replace constraints (3.63)–(3.67) in Formulation F2a with the following constraints to obtain model Formulation F2b:

$$v_i^k \leq ub^k \qquad i \in I,\ k \in K \tag{3.71}$$

$$v_i^k \geq s_i \qquad i \in I,\ k \in K \tag{3.72}$$

$$v_i^k - v_j^k + ub^k \cdot x_{ij}^k + (ub^k - s_i - s_j) \cdot x_{ji}^k \leq ub^k - s_j \qquad (i,j) \in A,\ k \in K \tag{3.73}$$

$$v_d^k = 0 \qquad k \in K \tag{3.74}$$

$$v_i^k - v_j^k \leq (2 - \sum_{m \in \delta_i^+} x_{im}^k - \sum_{m \in \delta_j^+} x_{jm}^k) \cdot M \qquad (i,j) \in A : p_{ij} = 1,\ k \in K \tag{3.75}$$

A.5 Model Formulation F2c

The following model formulations are based on the formulation of the CVRP presented in Leggieri and Haouari, 2017. Instead of using the variable $y_i^k, i \in I, k \in K$ and $u_i^k, i \in I, k \in K$, we use the binary variable $y_{ij}^k, i \in I, j \in I, k \in K$ determining whether mail carrier k visits both street segments i and j, and i precedes j in the route. We can then replace the MTZ-constraints (3.63)–(3.66) with the following subtour elimination constraints (3.76)–(3.78) and the precedence constraints (3.67) with constraints (3.79) in Formulation F2a in order to obtain Formulation F2c.

$$x_{ij}^k \leq y_{ij}^k \qquad i,j \in I,\ k \in K \tag{3.76}$$

$$-(1 - x_{iv}^k) \leq y_{ij}^k - y_{vj}^k \leq 1 - x_{iv}^k \qquad i,j,v \in I, i \neq j \neq v,\ k \in K \tag{3.77}$$

$$y_{ij}^k + y_{ji}^k \leq 1 \qquad i,j \in I,\ k \in K \tag{3.78}$$

$$y_{ji}^k \leq 1 - p_{ij} \qquad i,j \in I,\ k \in K \tag{3.79}$$

Constraints (3.76) and (3.77) link the x-variables with the y-variables. In constraints (3.78) subtours are eliminated because for two street segments $i \in I$ and $j \in I$, i can either precede j or j precedes i, but not both at the same time, which would be the case in a subtour.

A.6 Model Formulation F3a

Leggieri and Haouari, 2017 propose an even stronger model formulation that is obtained by lifting some constraints in Formulation F2c. The additional variables $f_{ijv}^k = x_{ij}^k y_{jv}^k, i,j,v \in L, k \in K$ and $g_{ijv}^k = y_{ij}^k x_{jv}^k, i,j,v \in L, k \in K$ are introduced, and we obtain Formulation F3a by replacing constraints (3.76)–(3.78) in Formulation F2c by the following constraints:

$$\sum_{v \in I \setminus \{i,j\}} f_{ivj}^k + x_{ij}^k = y_{ij}^k \qquad (i,j) \in A,\ k \in K \tag{3.80}$$

$$f_{dij}^k + \sum_{v \in I \setminus \{i,j\}} f_{vij}^k = y_{ij}^k \qquad (i,j) \in A,\ k \in K \tag{3.81}$$

$$\sum_{v \in I \setminus \{i,j\}} g_{ijv}^k + g_{ijd}^k = y_{ij}^k \qquad (i,j) \in A,\ k \in K \tag{3.82}$$

$$f_{ijv}^k + g_{vij}^k \leq x_{ij}^k \qquad i,j,v \in I, i \neq j \neq v,\ k \in K \tag{3.83}$$

$$\sum_{v \in I \setminus \{i,j\}} f_{ivj}^k - \sum_{v \in I \setminus \{i,j\}} g_{ivj}^k = 0 \qquad (i,j) \in A,\ k \in K \tag{3.84}$$

$$f_{dvj}^k \leq x_{dv}^k \qquad j, v \in I,\ k \in K \tag{3.85}$$

$$g_{ivd}^k \leq x_{vd}^k \qquad i, v \in I,\ k \in K \tag{3.86}$$

A.7 Comparison of the different model formulations

In this section, we compare the proposed model formulations. For this, we solve the small-scale instances introduced in Section 3.5.1.2 for each of the above formulations with Gurobi with a time limit of two hours. We compare the results based on the number of instances for which a feasible solution is found within the time limit (*#feas*), and the average percentage gap (*Gap*) and runtime (*t*) over these instances. The results in Table 3.12 show that the best formulations are Formulation F1a and Formulation F1b which return feasible solutions for the highest number of instances within the time limit.

	IP		
Formulation	*#feas*	*Gap*(%)	*t*(*s*)
F1a	29	6.88	4577.63
F1b	28	5.82	4273.28
F2a	18	9.15	4529.50
F2b	24	4.22	4182.97
F2c	16	3.62	3542.21
F3a	3	0.00	2493.16

Table 3.12: Results of the different mathematical model formulations.

For these formulations, Table 3.13 shows the average percentage gap and runtime only on the 28 instances for which a feasible solution is found in the time limit. The results show that on these 28 instances, Formulation F1b is slightly better than Formulation F1a. Because we use the solution of the mathematical model to assess the solution quality provided by our heuristic, considering the formulation that is able to solve more instances within the time limit is more relevant, as the solution quality for both formulations is nearly the same. Hence, in our computational experiments, we consider Formulation F1a.

	IP		
Formulation	*#feas*	*Gap*(%)	*t*(*s*)
F1a	28	5.95	4483.97
F1b	28	5.82	4273.28

Table 3.13: Comparison of model Formulation F1a and Formulation F1b.

B Parameter tuning and analysis of the ILS-VRPDOC components

In this section, we describe how we determine the parameters of ILS-VRPDOC. Although various sophisticated methods for tuning the parameters of algorithms have been proposed in the literature (see, for example, López-Ibáñez et al., 2016), we perform a very basic tuning of the parameters because ILS-VRPDOC has only a few parameters to be tuned, i.e., the stopping criterion (time and number of iterations without improvement, see Section B.1), the generator arc set (sparsification intensity and additional generator arcs, see Section B.2), and the infeasibility penalty factor and its bounds (see Section B.3).

To avoid overfitting, we perform the parameter tuning experiments on a subset of DHL instances. We first divide the DHL set into six groups, depending on the number of tables (below 10, between 11 and 20, and above 21) and on whether more or less mail carriers are needed compared to the number of tables in the depot. Then, we randomly choose between four and six instances from each group, resulting in a final subset of 30 instances.

Because our algorithm contains randomized elements, we perform ten runs of ILS-VRPDOC for each instance. In the parameter tuning experiments, we always compare to the solution provided by DHL.

B.1 Stopping criteria

The first two parameters to tune define the stopping criteria of ILS-VRPDOC, and are: the iterations without improvement η, and the time limit in seconds τ.

We recall that the VRPDOC can be solved at both tactical and operational level. Hence, we choose the values to test for the parameters η and τ based on the possibility of deriving two versions of ILS-VRPDOC, that are: version ILS-VRPDOC$_{quality}$ focusing on solution quality and to be used at the tactical level, and version ILS-VRPDOC$_{speed}$ focusing on runtime and to be used at the operational level. Because the runtime is more critical at operational level, we start by choosing a value for the time limit of 60 seconds per run so that, in the worst case, the runtime of ILS-VRPDOC is at most ten minutes. Then, we tested higher values for the time limit. The largest considered value is 3600 seconds per run because even if at the tactical level the runtime is not critical, ILS-VRPDOC must be run for a larger number of instances, each with a different number of desired mail carriers. For each considered value of τ, we determined a corresponding value of η through preliminary experiments, aiming to retain a balance between the two stopping criteria.

Table 3.14 shows the results of ILS-VRPDOC for varying values of η and τ. The first two columns display the number of iterations without improvement and the time limit in seconds. All other parameters are fixed. The sparsification intensity is set such that only the five closest locations to each street segment are considered when deriving the generator arcs, additional arcs are not included in the generator arc set, and the penalty term values are set to standard values $(\alpha_{\min}, \alpha_{\max}, \alpha_0) = (0.01, 100, 1)$ used in the literature (see, e.g., Schneider and Löffler, 2019). For every parameter configuration, we report the average over the 30 instances, for each of the following metrics: gap of the best solution out of the ten runs to the DHL solution($\Delta^b(\%)$), the average gap over the ten runs ($\Delta^a(\%)$), and the average runtime over the ten runs ($t^a(s)$). The results show that already with the fastest variant of ILS-VRPDOC, our solutions are better than those of DHL both in the best and average run. By increasing η and τ, the solution quality keeps improving. As expected, the results show a tradeoff between runtime and solution quality. We set $\eta = 150$ and $\tau = 60$ for ILS-VRPDOC$_{speed}$ and $\eta = 4000$ and $\tau = 3600$ for ILS-VRPDOC$_{quality}$.

η	τ	$\Delta^b(\%)$	$\Delta^a(\%)$	$t^a(s)$
150	60	-5.47	-4.57	42.14
500	300	-5.54	-4.37	165.15
1000	600	-5.75	-4.69	328.04
4000	3600	-6.58	-5.98	1762.20

Table 3.14: Results of the parameter tuning for the stopping criteria of ILS-VRPDOC.

B.2 Size and composition of the generator arc set

In this section, we tune the parameters that affect the size and composition of the generator arc set. For both of the final termination criteria from Section B.1, we test two sparsification intensity values, i.e., $\kappa = 5$ and $\kappa = 10$. Moreover, for each of the four obtained combinations, we test the effect of including or not additional arcs according to the strategies presented in Section 3.4.2.1.3.

Table 3.15 shows the results, and is organized like Table 3.14 with the additional columns describing the size and composition of the generator arc set. For combination ($\eta = 150$, $\tau = 60$), we focus on the average solution quality, because ILS-VRPDOC$_{speed}$ is used on the operational level, where time could be a critical factor and there might not be enough time to do ten runs of the algorithm. A better average solution quality is returned when considering a lower sparsification intensity (i.e., $\kappa = 10$) compared to the corresponding variant with $\kappa = 5$. By enriching the generator arc set with the additional arcs described in Section 3.4.2.1.3, we obtain further improvements in the best and average solution quality. Using a higher sparsification intensity translates into slightly faster runtimes. For ILS-VRPDOC$_{quality}$ (i.e., $\eta = 4000$ and $\tau = 3600$), we put more emphasis on the best run because time is not a critical factor here, and we assume that it is always possible to do ten runs. A better solution quality in the best run is reached considering a higher sparsification intensity (i.e., $\kappa = 5$).

Including the additional arcs leads to higher runtimes, but we again obtain further improvements in the best and average solution quality.

Consequently, for ILS-VRPDOC$_{speed}$, we choose the parameter combination ($\kappa =$ 10, Additional generator arcs=included) represented by line 2 in Table 3.15 while for ILS-VRPDOC$_{quality}$, we choose the combination ($\kappa = 5$, Additional generator arcs=included) of line 8 in Table 3.15. These results are a clear indicator that the additional arcs are vital to find solutions of good quality.

η	τ	κ	Additional generator arcs	$\Delta^b(\%)$	$\Delta^a(\%)$	$t^a(s)$
150	60	5	not included	-5.47	-4.57	42.14
150	60	5	included	-5.61	-4.51	50.14
150	60	10	not included	-5.47	-4.62	44.11
150	60	10	included	-5.56	-4.65	51.12
4000	3600	5	not included	-6.58	-5.98	1762.20
4000	3600	5	included	-6.77	-6.08	2361.87
4000	3600	10	not included	-6.56	-6.13	1930.56
4000	3600	10	included	-6.71	-6.14	2425.72

Table 3.15: Results of the parameter tuning for the generator arc set size and composition for ILS-VRPDOC.

B.3 Infeasiblity penalty factor

The last set of parameters to tune defines the minimum, maximum, and initial values ($\alpha_{\min}$, $\alpha_{\max}$, and α_0) of the penalty factor α that is used in the VND. The parameter δ is given by $\exp\left({}^{\log(\alpha_{\max}/\alpha_{\min})}/_{\alpha_{\max}}\right)$. For each of the two algorithmic versions established before, we test the values $(\alpha_{\min}, \alpha_{\max}, \alpha_0) = (0.01, 1, 0.1)$ proposed by Schneider and Löffler, 2019. Then, we increase all three α-values by a factor of ten until the quality of the solution stops improving.

Table 3.16 shows the results and is organized as Table 3.15 with an additional column for the α-values. Increasing $\alpha_{\max}$ to 10 000 for ILS-VRPDOC$_{speed}$ and to 100 000 for ILS-VRPDOC$_{quality}$ leads to the overall best solution quality. Increasing the values ($\alpha_{\min}$, $\alpha_{\max}$, α_0) = (0.01,1,0.1) by a factor of ten improves the solution quality for both combinations ($\eta = 150$, $\tau = 60$, $\kappa = 10$, Additional generator arcs=included) and ($\eta = 4000$, $\tau = 3600$, $\kappa = 5$, Additional generator arcs=included). However, further increasing the values ($\alpha_{\min}$, $\alpha_{\max}$, α_0) by a factor of ten provides results with a solution quality that is similar to the one obtained for the starting values.

η	τ	κ	Additional generator arcs	$(\alpha_{\min}, \alpha_{\max}, \alpha_0)$	$\Delta^b(\%)$	$\Delta^a(\%)$	$t^a(s)$
150	60	10	included	(0.01,1,0.1)	-5.56	-4.65	51.12
150	60	10	included	(0.1,10,1)	-6.08	-5.36	53.40
150	60	10	included	(1,100,10)	-5.88	-5.15	53.81
4000	3600	5	included	(0.01,1,0.1)	-6.77	-6.08	2361.87
4000	3600	5	included	(0.1,10,1)	-7.21	-6.70	2563.76
4000	3600	5	included	(1,100,10)	-6.94	-6.44	2602.75

Table 3.16: Results of the parameter tuning for the infeasibility penalty factor of ILS-VRPDOC.

C Detailed results for the comparison of the DHL solutions to ILS-VRPDOC$_{quality}$ solutions

				DHL	ILS-VRPDOC$_{init}$		ILS-VRPDOC$_{quality}$			ILS-VRPDOC$_{speed}$		
Instance	$\|I\|$	$\|T\|$	$\|K\|$	*Obj*	$\Delta^{init}(\%)$	$t^{init}(s)$	$\Delta^{b}(\%)$	$\Delta^{a}(\%)$	$t^{a}(s)$	$\Delta^{b}(\%)$	$\Delta^{a}(\%)$	$t^{a}(s)$
1	275	6	7	80306.27	6.97	0.12	-14.40	-13.96	337.68	-14.12	-13.56	20.62
2	278	6	7	52383.28	23.67	0.10	-2.65	-1.24	171.99	-1.72	1.23	13.78
3	300	7	8	61147.58	35.14	0.14	-5.31	-4.63	409.08	-4.70	-3.71	23.74
4	273	7	6	36080.66	2.79	0.10	-5.56	-3.98	339.39	-3.46	-3.36	14.38
5	376	9	8	113977.07	8.40	0.86	-17.69	-15.71	1195.06	-12.96	-12.70	35.68
6	376	9	10	111441.89	17.40	2.23	-9.84	-8.97	1261.64	-6.50	-5.77	42.11
7	471	10	9	51611.83	15.35	0.63	-3.67	-3.58	1252.16	-3.45	-3.38	51.69
8	471	10	11	52937.57	11.10	0.61	-5.05	-4.93	1463.39	-5.03	-4.85	58.90
9	475	10	5	48070.73	3.07	0.13	1.99	1.99	410.23	1.99	1.99	16.84
10	475	10	9	51828.73	5.54	1.00	-5.96	-5.25	1864.72	-4.94	-4.38	54.44
11	475	10	11	52691.95	10.08	0.73	-4.57	-4.38	1743.46	-4.41	-3.85	57.77
12	475	10	12	55253.52	2.45	2.41	-8.23	-7.80	2196.88	-7.92	-5.65	59.01
13	480	10	11	54135.82	5.12	1.22	-4.88	-4.79	1432.91	-4.63	-4.21	49.84
14	514	12	11	209600.72	-0.00	0.91	-0.00	-0.00	186.04	-0.00	-0.00	8.00
15	514	12	13	230777.27	4.18	1.78	-25.27	-21.02	2093.26	-17.67	-14.88	59.95
16	582	12	6	66697.42	4.77	0.29	1.01	1.22	1532.63	1.04	1.30	55.98
17	582	12	10	70990.00	7.04	1.78	-5.51	-5.16	2993.39	-4.62	-3.51	60.19
18	582	12	11	71366.72	11.57	0.98	-5.50	-4.58	2978.43	-3.78	-3.38	60.41
19	582	12	13	71383.10	0.88	3.35	-4.20	-3.50	2914.09	-2.64	-1.80	60.26
20	582	12	14	71544.74	13.12	1.38	-4.08	-2.95	3213.96	-2.45	-1.81	60.56
21	625	13	12	47652.46	6.33	1.80	-8.88	-8.67	3589.74	-8.33	-7.60	60.56
22	625	13	14	47041.95	28.90	1.36	-5.27	-4.53	3527.31	-3.59	-3.08	60.27
23	622	13	12	56328.01	4.94	0.69	-4.76	-4.44	2871.02	-4.19	-3.78	59.11
24	622	13	14	58871.79	-1.82	4.43	-8.75	-8.48	3095.95	-7.62	-6.78	60.47
25	663	14	13	79206.96	-3.84	7.63	-8.91	-7.78	2695.29	-8.23	-6.95	60.19
26	663	14	15	75749.86	2.52	7.09	-2.61	-2.30	3400.34	-2.15	-1.49	60.31
27	700	14	7	99048.40	0.31	0.34	-3.25	-3.25	1296.59	-3.25	-3.25	55.07
28	700	14	13	107648.24	6.45	1.37	-9.51	-9.04	3054.55	-9.08	-8.09	60.26
29	700	14	15	104876.38	0.07	5.88	-5.04	-4.59	3571.28	-4.32	-3.72	60.25
30	726	15	8	49052.81	3.61	2.65	-4.71	-4.71	1794.68	-4.71	-4.71	60.30
31	726	15	14	57841.67	-6.45	7.65	-15.38	-14.72	3177.93	-14.58	-13.85	60.36
32	701	15	14	86149.90	4.05	1.32	-4.15	-3.98	3047.55	-3.69	-3.47	60.23
33	701	15	16	88584.53	-1.81	5.48	-6.60	-6.17	3562.92	-5.72	-4.87	60.77
34	760	16	15	66775.56	9.98	2.68	-8.63	-8.14	3507.19	-7.68	-6.95	60.42
35	751	16	15	105504.51	-3.77	5.46	-6.01	-5.88	3356.22	-5.80	-5.42	60.56
36	751	16	17	111828.22	2.49	3.10	-7.45	-6.91	3397.01	-6.85	-6.31	60.53
37	829	17	9	82907.27	6.27	3.82	1.03	1.03	3031.88	1.03	1.04	60.55
38	829	17	15	93979.54	11.66	4.41	-7.54	-6.92	3600.39	-6.50	-5.95	60.64
39	829	17	16	94282.40	1.18	11.81	-6.71	-6.30	3528.51	-6.06	-5.05	60.80
40	829	17	18	93599.51	13.21	5.43	-4.54	-4.38	3601.21	-4.06	-2.84	60.77
41	829	17	19	95578.75	-1.55	11.77	-5.54	-4.97	3600.63	-4.35	-3.43	60.64
42	925	19	18	99827.69	2.16	8.57	-5.27	-4.86	3482.59	-4.82	-4.10	60.41

				DHL	ILS-VRPDOC$_{init}$		ILS-VRPDOC$_{quality}$			ILS-VRPDOC$_{speed}$		
Instance	$\lvert I\rvert$	$\lvert T\rvert$	$\lvert K\rvert$	*Obj*	$\Delta^{init}(\%)$	$t^{init}(s)$	$\Delta^{b}(\%)$	$\Delta^{a}(\%)$	$t^{a}(s)$	$\Delta^{b}(\%)$	$\Delta^{a}(\%)$	$t^{a}(s)$
43	925	19	20	98594.66	3.16	25.97	-2.73	-2.56	3600.52	-2.28	-1.32	60.85
44	942	19	10	90774.77	0.09	8.29	-2.69	-2.69	3222.51	-2.69	-2.69	60.58
45	942	19	17	94097.24	-0.46	21.68	-1.47	-1.47	1615.37	-1.47	-1.47	60.28
46	942	19	18	93705.88	-0.15	8.75	-0.17	-0.17	988.26	-0.17	-0.17	48.75
47	908	19	10	111950.33	0.44	30.65	-6.09	-6.02	3600.67	-5.89	-4.95	60.84
48	908	19	18	121663.85	-0.83	7.50	-6.92	-6.19	3477.04	-5.86	-4.66	60.38
49	908	19	20	120329.49	-0.25	26.20	-5.28	-5.12	3554.57	-4.64	-3.44	60.59
50	908	19	21	124268.87	-3.26	35.11	-8.73	-8.52	3600.96	-6.41	-5.96	61.01
51	925	19	18	149507.13	-2.51	19.32	-13.22	-12.75	3600.72	-12.12	-11.35	60.34
52	925	19	20	141349.27	-2.38	13.81	-7.73	-7.26	3601.10	-6.22	-5.38	61.14
53	925	19	21	141449.26	0.08	19.98	-6.97	-6.68	3601.02	-5.40	-4.45	61.07
54	923	20	10	113765.56	2.46	23.64	-5.45	-5.45	2917.00	-5.45	-5.44	60.40
55	923	20	18	127691.87	-6.50	9.34	-10.62	-10.36	3600.62	-9.43	-8.69	60.51
56	923	20	19	128271.91	-5.69	9.09	-11.32	-10.67	3600.69	-10.12	-9.34	60.84
57	923	20	21	127353.23	8.74	6.01	-9.43	-9.21	3601.00	-8.71	-7.70	60.89
58	923	20	22	127232.23	0.72	14.55	-7.66	-7.29	3600.66	-6.63	-5.84	60.86
59	929	21	19	150342.68	2.18	5.98	-5.19	-4.63	3574.17	-4.67	-4.20	60.41
60	929	21	20	153093.73	-1.63	7.21	-6.98	-6.62	3600.77	-6.38	-5.48	60.86
61	929	21	23	154417.54	2.87	25.31	-6.33	-5.90	3601.01	-5.05	-4.01	60.90
62	934	21	12	88196.75	1.15	8.46	-2.22	-2.22	3600.77	-2.22	-2.22	61.01
63	1114	23	21	134146.07	-0.35	52.13	-1.77	-1.77	2611.56	-1.77	-1.61	60.43
64	1114	23	22	134170.55	-0.17	34.49	-0.50	-0.50	1492.39	-0.50	-0.50	60.25
65	1051	23	12	95717.08	3.20	8.00	-2.08	-2.08	3600.88	-2.08	-2.08	61.15
66	1051	23	21	108528.78	-5.13	28.76	-10.29	-9.99	3600.79	-9.34	-8.70	61.09
67	1051	23	22	105518.97	-1.81	28.18	-6.99	-6.55	3601.12	-6.15	-5.54	61.10
68	1051	23	24	103758.82	-0.50	30.57	-4.22	-3.85	3601.45	-3.04	-2.34	61.69
69	1051	23	25	106000.89	-1.57	33.40	-6.70	-6.26	3602.39	-5.15	-3.94	61.83
70	965	24	23	119080.63	-2.49	21.88	-8.48	-7.80	3600.97	-7.34	-6.31	60.79
71	965	24	25	118442.36	-0.02	28.45	-6.63	-6.40	3601.47	-5.68	-4.81	62.59
72	1142	24	23	127830.99	7.12	16.97	-8.61	-8.29	3601.09	-7.62	-6.68	62.42
73	1142	24	25	119689.31	6.23	61.54	-2.14	-1.64	3601.40	2.38	3.74	68.05
74	1200	25	14	119604.29	2.71	43.86	-7.06	-6.89	3602.76	-6.51	-5.41	62.07
75	1200	25	24	136155.86	-3.40	76.13	-7.86	-7.30	3602.35	-6.09	-4.75	76.66
76	1200	25	26	138261.97	-1.96	42.21	-7.22	-6.72	3602.43	-4.68	-4.20	62.98
77	1200	25	28	139005.59	-0.93	68.63	-5.64	-5.12	3601.17	-2.67	-1.59	70.08
78	1225	25	24	128255.07	-1.48	157.72	-7.48	-7.03	3601.76	-3.89	-3.30	158.97
79	1119	26	25	141018.97	0.62	39.34	-6.17	-5.11	3601.72	-4.64	-3.88	60.74
80	1119	26	27	143749.07	-1.30	52.37	-5.96	-5.52	3601.43	-3.83	-2.97	61.74
81	1080	26	25	123149.40	0.64	34.10	-8.68	-7.52	3602.02	-6.14	-4.77	61.61
82	1080	26	27	116985.82	6.69	42.93	-1.47	-1.25	3600.82	0.07	0.98	61.26
83	1275	26	25	172465.52	-9.89	57.09	-14.04	-13.64	3601.94	-12.63	-11.47	62.93
84	1275	26	27	155416.88	-0.25	14.28	-0.27	-0.27	1487.09	-0.27	-0.27	60.22
85	1275	26	28	155798.72	1.92	14.74	1.84	1.84	1650.74	1.84	1.84	60.27
86	1221	26	13	144936.73	9.59	9.71	-0.30	-0.30	3602.60	-0.27	0.06	61.47
87	1221	26	22	166087.78	5.37	37.39	-7.84	-7.20	3601.39	-6.30	-4.88	61.22
88	1221	26	23	171720.73	-3.61	85.35	-10.59	-9.87	3603.41	-8.86	-6.57	86.44
89	1221	26	24	165108.85	1.09	34.88	-5.73	-4.86	3602.05	-4.36	-3.32	61.02

				DHL	ILS-VRPDOC$_{init}$		ILS-VRPDOC$_{quality}$			ILS-VRPDOC$_{speed}$		
Instance	$\|I\|$	$\|T\|$	$\|K\|$	Obj	$\Delta^{init}(\%)$	$t^{init}(s)$	$\Delta^{b}(\%)$	$\Delta^{a}(\%)$	$t^{a}(s)$	$\Delta^{b}(\%)$	$\Delta^{a}(\%)$	$t^{a}(s)$
90	1221	26	25	163742.70	-1.12	69.29	-3.90	-3.28	3600.99	-2.30	-1.59	70.61
91	1221	26	27	168394.38	3.43	66.57	-5.39	-4.94	3603.15	-3.79	-1.90	71.02
92	1221	26	28	171411.25	20.12	28.55	-7.19	-6.52	3605.58	-3.97	-1.59	63.15
93	1199	29	27	144566.45	-0.69	70.61	-5.86	-5.43	3601.26	-3.21	-2.18	70.50
94	1199	29	28	145994.48	-0.11	66.42	-6.13	-5.31	3600.91	-4.09	-2.61	67.10
95	1199	29	32	150366.94	-1.74	44.48	-7.06	-6.57	3603.44	-5.25	-4.62	63.16
96	1408	30	29	183708.28	-6.98	49.71	-9.62	-9.36	3601.91	-8.55	-8.09	62.65
97	1282	30	15	174091.53	4.64	8.94	-5.03	-5.03	3601.56	-5.03	-4.98	62.71
98	1282	30	28	203951.23	2.15	12.83	-10.72	-10.37	3603.46	-9.53	-9.07	61.61

Table 3.17: Comparison of the DHL solutions to the solutions of our construction heuristic, and of the best and average run of ILS-VRPDOC$_{quality}$ and ILS-VRPDOC$_{speed}$.

D Detailed results for the routing-based comparison of the DHL solutions to ILS-VRPDOC$_{quality}$ solutions

				Service time max deviation		Service time avg deviation		
Instance	$\|I\|$	$\|T\|$	$\|K\|$	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	Similarity
1	275	6	7	20.54	18.87	6.57	13.34	89.72
2	278	6	7	15.61	14.81	8.85	11.97	90.18
3	300	7	8	24.27	21.83	8.69	12.77	91.88
4	273	7	6	11.79	9.34	3.93	5.52	93.55
5	376	9	8	27.20	26.78	6.80	12.64	93.49
6	376	9	10	27.03	27.61	9.20	18.19	91.19
7	471	10	9	28.11	28.79	7.02	12.61	93.75
8	471	10	11	29.26	29.15	10.62	19.37	91.70
9	475	10	5	19.90	19.59	12.63	11.46	97.08
10	475	10	9	55.54	54.09	17.27	28.26	93.60
11	475	10	11	31.20	31.27	6.67	18.94	90.53
12	475	10	12	31.58	31.91	7.93	22.91	89.32
13	480	10	11	17.01	16.55	5.68	10.41	91.04
14	514	12	11	30.06	21.74	7.99	6.47	97.14
15	514	12	13	51.27	50.16	11.09	25.78	90.32
16	582	12	6	14.62	13.53	7.45	8.28	93.54
17	582	12	10	34.96	33.01	7.42	12.84	90.20
18	582	12	11	32.29	31.68	6.56	14.44	89.71
19	582	12	13	34.47	33.73	6.96	17.57	89.41
20	582	12	14	34.18	34.29	7.76	25.49	87.42
21	625	13	12	23.39	23.08	6.63	9.10	86.66
22	625	13	14	25.22	25.62	6.98	15.56	91.86
23	622	13	12	23.88	23.73	9.67	10.64	93.06
24	622	13	14	47.29	47.85	13.96	29.05	90.72
25	663	14	13	40.86	40.01	7.00	16.44	93.49
26	663	14	15	31.34	30.99	5.71	12.33	89.53
27	700	14	7	14.04	14.04	6.21	6.21	97.60
28	700	14	13	24.22	24.19	7.44	13.12	94.25
29	700	14	15	22.34	21.61	7.91	13.72	93.71
30	726	15	8	64.96	46.15	16.24	14.47	97.41
31	726	15	14	33.97	33.16	5.41	9.92	93.78
32	701	15	14	12.58	12.67	5.63	7.47	93.15
33	701	15	16	39.16	39.97	9.74	18.97	86.19
34	760	16	15	21.27	20.18	4.86	11.72	92.52
35	751	16	15	26.18	22.32	5.12	11.38	92.95
36	751	16	17	31.93	29.62	5.08	12.73	91.80
37	829	17	9	65.73	47.79	14.61	12.79	97.61
38	829	17	15	38.15	38.35	7.50	18.11	90.40
39	829	17	16	36.76	31.62	7.11	9.71	91.83
40	829	17	18	41.63	41.99	8.60	19.87	88.43

				Service time max deviation		Service time avg deviation		
Instance	$\|I\|$	$\|T\|$	$\|K\|$	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	Similarity
41	829	17	19	42.68	42.24	8.46	24.06	91.04
42	925	19	18	35.15	35.92	5.24	15.18	93.85
43	925	19	20	32.30	32.91	5.95	15.68	92.06
44	942	19	10	75.62	47.52	16.91	14.14	97.48
45	942	19	17	80.51	61.65	18.60	11.72	97.81
46	942	19	18	40.92	40.84	6.15	6.15	97.92
47	908	19	10	62.32	62.60	12.58	37.28	97.06
48	908	19	18	20.24	20.43	7.34	7.41	95.90
49	908	19	20	37.58	36.59	7.04	14.01	93.97
50	908	19	21	34.50	34.70	7.90	16.00	90.85
51	925	19	18	39.39	39.28	5.26	10.97	93.74
52	925	19	20	36.66	35.56	4.56	11.48	87.83
53	925	19	21	33.80	33.97	8.11	15.70	87.42
54	923	20	10	8.99	8.99	2.98	2.77	97.43
55	923	20	18	26.07	26.55	4.58	15.19	93.62
56	923	20	19	25.73	20.96	4.75	11.09	93.42
57	923	20	21	29.05	29.24	5.66	13.76	89.94
58	923	20	22	23.55	23.55	4.49	13.82	92.06
59	929	21	19	24.31	24.31	4.64	9.20	93.78
60	929	21	20	27.46	23.78	4.80	9.86	93.15
61	929	21	23	26.09	26.09	8.65	16.82	91.07
62	934	21	12	61.74	60.57	23.79	24.41	97.04
63	1114	23	21	86.56	76.59	17.03	14.05	95.95
64	1114	23	22	43.88	42.13	7.45	7.45	96.04
65	1051	23	12	55.09	44.44	12.62	10.28	97.27
66	1051	23	21	41.69	39.89	8.63	13.90	93.94
67	1051	23	22	38.88	38.88	8.59	10.66	93.29
68	1051	23	24	33.82	31.26	7.44	14.25	89.58
69	1051	23	25	39.07	39.90	8.49	17.06	85.87
70	965	24	23	35.12	35.69	5.00	13.53	92.11
71	965	24	25	34.50	34.95	4.67	17.07	85.25
72	1142	24	23	40.36	40.64	9.50	14.93	92.02
73	1142	24	25	34.95	34.95	8.82	15.42	93.83
74	1200	25	14	72.10	67.85	23.68	36.53	92.26
75	1200	25	24	53.26	51.68	8.85	19.35	88.64
76	1200	25	26	50.04	49.86	7.87	22.02	82.95
77	1200	25	28	63.67	63.36	20.76	28.74	90.15
78	1225	25	24	78.36	78.36	10.01	37.99	93.92
79	1119	26	25	37.86	37.77	7.09	9.09	92.31
80	1119	26	27	31.99	31.99	4.30	12.18	91.54
81	1080	26	25	57.18	57.69	8.85	19.96	90.14
82	1080	26	27	53.65	53.65	7.82	14.53	94.49
83	1275	26	25	41.60	40.79	4.38	13.89	92.69
84	1275	26	27	51.73	49.23	10.75	11.40	97.77
85	1275	26	28	50.54	50.97	15.86	16.46	97.54
86	1221	26	13	48.21	46.49	10.55	10.63	94.57
87	1221	26	22	54.10	54.77	7.14	17.09	92.20

				Service time max deviation		Service time avg deviation		
Instance	$\|I\|$	$\|T\|$	$\|K\|$	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	Similarity
88	1221	26	23	51.89	51.89	6.66	18.51	88.50
89	1221	26	24	49.95	49.98	6.56	19.48	92.29
90	1221	26	25	51.01	51.86	9.98	19.02	94.38
91	1221	26	27	45.11	45.66	5.97	19.90	86.94
92	1221	26	28	43.36	43.38	6.13	23.22	83.67
93	1199	29	27	54.69	54.03	4.72	17.05	92.41
94	1199	29	28	50.79	50.80	4.43	18.10	92.34
95	1199	29	32	47.59	47.59	6.65	22.59	88.63
96	1408	30	29	29.61	27.97	6.76	9.09	93.46
97	1282	30	15	38.23	36.54	6.84	6.85	97.30
98	1282	30	28	13.29	13.62	4.42	6.79	92.82

Table 3.18: Routing-based comparison of the DHL solutions to the solutions of the best run of ILS-VRPDOC$_{quality}$.

E Detailed results for the depot operation-based comparison of the DHL solutions to ILS-VRPDOC$_{quality}$ solutions

				Max #mail carriers		Avg #mail carriers		Max #tables		Avg #tables		Skips		Table changes	
Instance	$\|I\|$	$\|T\|$	$\|K\|$	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC
1	275	6	7	2.00	2.00	2.00	1.83	6.00	4.00	1.71	1.57	3.00	2.00	6.00	6.00
2	278	6	7	2.00	2.00	2.00	1.83	6.00	5.00	1.71	1.57	5.00	5.00	5.00	5.00
3	300	7	8	2.00	2.00	2.00	1.86	7.00	4.00	1.75	1.62	3.00	3.00	6.00	6.00
4	273	7	6	6.00	4.00	1.71	1.57	2.00	2.00	2.00	1.83	0.00	0.00	6.00	6.00
5	376	9	8	8.00	3.00	1.78	1.78	2.00	2.00	2.00	2.00	0.00	0.00	8.00	8.00
6	376	9	10	2.00	2.00	2.00	1.78	9.00	2.00	1.80	1.60	2.00	2.00	9.00	9.00
7	471	10	9	9.00	4.00	1.80	1.70	2.00	2.00	2.00	1.89	0.00	0.00	9.00	9.00
8	471	10	11	2.00	2.00	2.00	1.70	9.00	2.00	1.82	1.55	4.00	4.00	9.00	8.00
9	475	10	5	1.00	1.00	1.00	1.00	2.00	2.00	2.00	2.00	1.00	0.00	5.00	5.00
10	475	10	9	8.00	4.00	1.70	1.60	2.00	2.00	1.89	1.78	1.00	1.00	8.00	8.00
11	475	10	11	2.00	2.00	1.90	1.80	6.00	3.00	1.73	1.64	8.00	8.00	8.00	8.00
12	475	10	12	2.00	2.00	1.90	1.90	4.00	4.00	1.58	1.58	10.00	9.00	9.00	9.00
13	480	10	11	2.00	2.00	1.90	1.90	9.00	5.00	1.73	1.73	5.00	5.00	8.00	8.00
14	514	12	11	1.00	1.00	1.00	1.00	2.00	2.00	1.09	1.09	1.00	0.00	1.00	1.00
15	514	12	13	2.00	2.00	1.83	1.75	10.00	3.00	1.69	1.62	8.00	8.00	9.00	9.00
16	582	12	6	1.00	1.00	1.00	1.00	2.00	2.00	2.00	2.00	10.00	0.00	14.00	13.00
17	582	12	10	6.00	3.00	1.67	1.58	2.00	2.00	2.00	1.90	13.00	7.00	14.00	14.00
18	582	12	11	11.00	3.00	1.83	1.75	2.00	2.00	2.00	1.91	13.00	10.00	12.00	12.00
19	582	12	13	2.00	2.00	1.92	1.67	9.00	3.00	1.77	1.54	12.00	12.00	11.00	11.00
20	582	12	14	3.00	3.00	2.17	2.17	6.00	3.00	1.86	1.86	17.00	12.00	14.00	14.00
21	625	13	12	12.00	5.00	1.85	1.85	2.00	2.00	2.00	2.00	12.00	12.00	23.00	23.00
22	625	13	14	2.00	2.00	2.00	2.00	13.00	3.00	1.86	1.86	7.00	7.00	13.00	13.00
23	622	13	12	11.00	3.00	1.77	1.62	2.00	2.00	1.92	1.75	3.00	3.00	11.00	11.00
24	622	13	14	2.00	2.00	2.00	1.77	13.00	3.00	1.86	1.64	5.00	5.00	14.00	14.00
25	663	14	13	12.00	4.00	1.79	1.71	2.00	2.00	1.92	1.85	0.00	0.00	15.00	15.00
26	663	14	15	2.00	2.00	1.93	1.93	10.00	7.00	1.80	1.80	10.00	10.00	20.00	20.00
27	700	14	7	1.00	1.00	1.00	1.00	2.00	2.00	2.00	2.00	0.00	0.00	7.00	7.00
28	700	14	13	13.00	6.00	1.86	1.64	2.00	2.00	2.00	1.77	0.00	0.00	13.00	13.00
29	700	14	15	2.00	2.00	1.86	1.71	12.00	5.00	1.73	1.60	4.00	4.00	11.00	11.00
30	726	15	8	1.00	1.00	1.00	1.00	2.00	2.00	1.88	1.88	0.00	0.00	7.00	7.00
31	726	15	14	14.00	6.00	1.87	1.67	2.00	2.00	2.00	1.79	0.00	0.00	14.00	14.00
32	701	15	14	12.00	6.00	1.73	1.60	2.00	2.00	1.86	1.71	3.00	3.00	12.00	12.00
33	701	15	16	2.00	2.00	2.00	1.80	12.00	3.00	1.88	1.69	25.00	25.00	17.00	17.00
34	760	16	15	12.00	3.00	1.69	1.56	2.00	2.00	1.80	1.67	11.00	11.00	12.00	12.00
35	751	16	15	2.00	2.00	1.88	1.56	2.00	2.00	2.00	1.67	0.00	0.00	15.00	14.00
36	751	16	17	2.00	2.00	2.00	1.56	2.00	2.00	1.88	1.47	0.00	0.00	15.00	15.00
37	829	17	9	1.00	1.00	1.00	1.00	2.00	2.00	1.89	1.89	1.00	0.00	9.00	9.00
38	829	17	15	9.00	4.00	1.82	1.88	3.00	3.00	2.07	2.13	13.00	13.00	18.00	18.00
39	829	17	16	15.00	3.00	1.82	1.76	2.00	2.00	1.94	1.88	4.00	4.00	19.00	19.00
40	829	17	18	3.00	3.00	2.06	2.12	10.00	4.00	1.94	2.00	17.00	17.00	21.00	21.00

				Max #mail carriers		Avg #mail carriers		Max #tables		Avg #tables		Skips		Table changes	
Instance	$\|I\|$	$\|T\|$	$\|K\|$	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC
41	829	17	19	2.00	2.00	1.94	1.76	8.00	3.00	1.74	1.58	9.00	9.00	16.00	16.00
42	925	19	18	17.00	4.00	1.84	1.74	2.00	2.00	1.94	1.83	0.00	0.00	17.00	17.00
43	925	19	20	2.00	2.00	1.89	1.79	17.00	5.00	1.80	1.70	10.00	10.00	18.00	18.00
44	942	19	10	1.00	1.00	1.00	1.00	2.00	2.00	1.90	1.90	0.00	0.00	9.00	9.00
45	942	19	17	1.00	1.00	1.00	1.00	2.00	2.00	1.12	1.12	0.00	0.00	2.00	2.00
46	942	19	18	1.00	1.00	1.00	1.00	2.00	2.00	1.06	1.06	0.00	0.00	1.00	1.00
47	908	19	10	2.00	2.00	1.05	1.05	3.00	3.00	2.00	2.00	1.00	0.00	10.00	10.00
48	908	19	18	8.00	3.00	1.37	1.32	2.00	2.00	1.44	1.39	0.00	0.00	8.00	8.00
49	908	19	20	2.00	2.00	1.58	1.58	11.00	3.00	1.50	1.50	6.00	6.00	10.00	10.00
50	908	19	21	2.00	2.00	1.95	1.89	11.00	3.00	1.76	1.71	14.00	14.00	16.00	16.00
51	925	19	18	18.00	5.00	1.89	1.63	2.00	2.00	2.00	1.72	0.00	0.00	18.00	18.00
52	925	19	20	2.00	2.00	1.89	1.89	17.00	5.00	1.80	1.80	32.00	28.00	16.00	16.00
53	925	19	21	2.00	2.00	1.89	1.84	9.00	3.00	1.71	1.67	29.00	29.00	15.00	15.00
54	923	20	10	1.00	1.00	1.00	1.00	2.00	2.00	2.00	2.00	0.00	0.00	11.00	10.00
55	923	20	18	9.00	4.00	1.80	1.60	2.00	2.00	2.00	1.78	0.00	0.00	18.00	18.00
56	923	20	19	19.00	6.00	1.90	1.70	2.00	2.00	2.00	1.79	0.00	0.00	19.00	19.00
57	923	20	21	2.00	2.00	2.00	1.75	20.00	3.00	1.90	1.67	13.00	13.00	19.00	19.00
58	923	20	22	2.00	2.00	1.95	1.70	10.00	3.00	1.77	1.55	5.00	5.00	17.00	17.00
59	929	21	19	9.00	4.00	1.76	1.62	2.00	2.00	1.95	1.79	0.00	0.00	18.00	18.00
60	929	21	20	19.00	7.00	1.86	1.67	2.00	2.00	1.95	1.75	0.00	0.00	19.00	19.00
61	929	21	23	2.00	2.00	1.86	1.71	11.00	3.00	1.70	1.57	12.00	12.00	16.00	16.00
62	934	21	12	1.00	1.00	1.00	1.00	2.00	2.00	1.75	1.75	2.00	0.00	9.00	9.00
63	1114	23	21	1.00	1.00	1.00	1.00	2.00	2.00	1.10	1.10	19.00	0.00	2.00	2.00
64	1114	23	22	1.00	1.00	1.00	1.00	2.00	2.00	1.05	1.05	19.00	0.00	1.00	1.00
65	1051	23	12	1.00	1.00	1.00	1.00	2.00	2.00	1.92	1.92	0.00	0.00	11.00	11.00
66	1051	23	21	10.00	4.00	1.74	1.57	2.00	2.00	1.90	1.71	0.00	0.00	19.00	19.00
67	1051	23	22	20.00	6.00	1.83	1.65	2.00	2.00	1.91	1.73	1.00	1.00	20.00	20.00
68	1051	23	24	2.00	2.00	1.78	1.78	18.00	4.00	1.71	1.71	22.00	22.00	19.00	19.00
69	1051	23	25	3.00	3.00	2.43	2.17	18.00	4.00	2.24	2.00	30.00	30.00	32.00	32.00
70	965	24	23	21.00	10.00	1.83	1.71	2.00	2.00	1.91	1.78	2.00	2.00	21.00	21.00
71	965	24	25	2.00	2.00	1.96	1.79	12.00	3.00	1.88	1.72	26.00	26.00	29.00	29.00
72	1142	24	23	21.00	5.00	1.83	1.88	2.00	2.00	1.91	1.96	6.00	6.00	26.00	26.00
73	1142	24	25	2.00	2.00	1.92	1.88	22.00	7.00	1.84	1.80	9.00	9.00	24.00	24.00
74	1200	25	14	2.00	2.00	1.08	1.20	3.00	3.00	1.93	2.14	31.00	21.00	16.00	16.00
75	1200	25	24	20.00	5.00	1.88	1.92	3.00	3.00	1.96	2.00	31.00	31.00	25.00	25.00
76	1200	25	26	2.00	2.00	1.92	1.88	4.00	3.00	1.85	1.81	53.00	53.00	38.00	38.00
77	1200	25	28	2.00	2.00	1.44	1.48	3.00	3.00	1.29	1.32	31.00	31.00	10.00	10.00
78	1225	25	24	20.00	5.00	1.76	1.76	2.00	2.00	1.83	1.83	3.00	3.00	20.00	20.00
79	1119	26	25	18.00	4.00	1.65	1.58	2.00	2.00	1.72	1.64	9.00	9.00	18.00	18.00
80	1119	26	27	2.00	2.00	1.69	1.65	13.00	3.00	1.63	1.59	9.00	9.00	21.00	21.00
81	1080	26	25	22.00	6.00	1.81	1.92	2.00	2.00	1.88	2.00	3.00	3.00	38.00	38.00
82	1080	26	27	2.00	2.00	1.92	1.88	24.00	9.00	1.85	1.81	4.00	4.00	23.00	23.00
83	1275	26	25	24.00	4.00	1.88	1.77	2.00	2.00	1.96	1.84	6.00	6.00	24.00	24.00
84	1275	26	27	2.00	2.00	1.04	1.04	1.00	1.00	1.00	1.00	0.00	0.00	0.00	0.00
85	1275	26	28	2.00	2.00	1.08	1.08	1.00	1.00	1.00	1.00	0.00	0.00	0.00	0.00
86	1221	26	13	1.00	1.00	1.00	1.00	2.00	2.00	2.00	2.00	6.00	0.00	29.00	28.00
87	1221	26	22	7.00	4.00	1.65	1.54	2.00	2.00	1.95	1.82	15.00	14.00	21.00	21.00

				Max #mail carriers		Avg #mail carriers		Max #tables		Avg #tables		Skips		Table changes	
Instance	$\|I\|$	$\|T\|$	$\|K\|$	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC	DHL	ILS-VRPDOC
88	1221	26	23	21.00	7.00	2.50	2.19	3.00	3.00	2.83	2.48	14.00	14.00	43.00	43.00
89	1221	26	24	11.00	6.00	1.77	1.77	2.00	2.00	1.92	1.92	12.00	12.00	23.00	23.00
90	1221	26	25	11.00	4.00	1.38	1.42	2.00	2.00	1.44	1.48	6.00	6.00	13.00	13.00
91	1221	26	27	2.00	2.00	1.88	1.92	9.00	4.00	1.81	1.85	30.00	30.00	33.00	33.00
92	1221	26	28	3.00	3.00	2.19	2.42	9.00	4.00	2.04	2.25	43.00	42.00	47.00	47.00
93	1199	29	27	15.00	5.00	1.83	1.66	2.00	2.00	1.96	1.78	3.00	3.00	26.00	26.00
94	1199	29	28	26.00	5.00	1.86	1.72	2.00	2.00	1.93	1.79	0.00	0.00	26.00	26.00
95	1199	29	32	2.00	2.00	1.93	1.62	10.00	3.00	1.75	1.47	26.00	26.00	24.00	24.00
96	1408	30	29	25.00	5.00	1.80	1.70	2.00	2.00	1.86	1.76	0.00	0.00	27.00	27.00
97	1282	30	15	1.00	1.00	1.00	1.00	2.00	2.00	2.00	2.00	0.00	0.00	15.00	15.00
98	1282	30	28	27.00	5.00	1.87	1.63	2.00	2.00	2.00	1.75	0.00	0.00	28.00	28.00

Table 3.19: Depot operation-based comparison of the DHL solutions to the solutions of the best run of ILS-VRPDOC$_{quality}$.

F Detailed results for the routing-based comparison of the ILS-VRPDOC$_{quality}$ solutions to the ILS-VRPDOC$^{OFF}_{quality}$ solutions

				Objective		Service time max deviation		Service time avg deviation		Similarity to DHL solutions	
Instance	$\|I\|$	$\|T\|$	$\|K\|$	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}
1	275	6	7	68740.90	62801.91	18.87	19.63	13.34	15.73	89.72	45.74
2	278	6	7	50993.70	46704.53	14.81	15.89	11.97	12.95	90.18	46.67
3	300	7	8	57901.85	49982.84	21.83	24.75	12.77	18.95	91.88	51.95
4	273	7	6	34075.42	29947.85	9.34	11.99	5.52	9.98	93.55	40.86
5	376	9	8	93816.72	82443.89	26.78	27.53	12.64	20.17	93.49	44.79
6	376	9	10	100481.24	90682.77	27.61	27.81	18.19	21.53	91.19	50.00
7	471	10	9	49719.36	43638.34	28.79	27.82	12.61	16.29	93.75	32.29
8	471	10	11	50262.42	44511.48	29.15	29.33	19.37	23.06	91.70	37.34
9	475	10	5	49025.43	40724.75	19.59	14.03	11.46	8.23	97.08	35.21
10	475	10	9	48737.83	43116.50	54.09	53.15	28.26	26.98	93.60	33.26
11	475	10	11	50283.28	44930.79	31.27	31.70	18.94	22.79	90.53	33.54
12	475	10	12	50708.49	45147.68	31.91	31.54	22.91	20.87	89.32	34.91
13	480	10	11	51495.36	45892.44	16.55	17.57	10.41	12.73	91.04	34.62
14	514	12	11	209600.72	138273.76	21.74	30.93	6.47	20.31	97.14	23.62
15	514	12	13	172467.93	152854.81	50.16	51.74	25.78	35.20	90.32	22.20
16	582	12	6	67373.45	57002.42	13.53	13.35	8.28	8.69	93.54	33.84
17	582	12	10	67079.17	59928.71	33.01	30.17	12.84	18.32	90.20	34.12
18	582	12	11	67442.05	61054.54	31.68	25.41	14.44	9.99	89.71	32.72
19	582	12	13	68381.81	61649.80	33.73	34.81	17.57	15.02	89.41	32.94
20	582	12	14	68623.11	62577.09	34.29	28.78	25.49	12.54	87.42	32.21
21	625	13	12	43420.61	39302.15	23.08	23.79	9.10	15.31	86.66	49.14
22	625	13	14	44563.03	41202.67	25.62	25.91	15.56	18.67	91.86	49.77
23	622	13	12	53645.91	43954.54	23.73	22.46	10.64	14.13	93.06	32.49
24	622	13	14	53719.78	44971.56	47.85	47.21	29.05	24.87	90.72	27.20
25	663	14	13	72151.27	65051.33	40.01	40.41	16.44	25.64	93.49	29.44
26	663	14	15	73772.15	67192.25	30.99	31.26	12.33	18.49	89.53	29.20
27	700	14	7	95826.72	85384.75	14.04	12.48	6.21	7.25	97.60	52.05
28	700	14	13	97409.55	91256.59	24.19	23.13	13.12	9.44	94.25	51.19
29	700	14	15	99591.85	93745.98	21.61	22.68	13.72	16.31	93.71	52.59
30	726	15	8	46743.05	36557.86	46.15	63.81	14.47	39.35	97.41	52.86
31	726	15	14	48943.76	42563.29	33.16	33.64	9.92	21.31	93.78	52.84
32	701	15	14	82572.73	74594.75	12.67	12.65	7.47	8.30	93.15	32.31
33	701	15	16	82736.75	76762.85	39.97	39.64	18.97	18.96	86.19	37.10
34	760	16	15	61011.39	56424.07	20.18	21.63	11.72	15.54	92.52	63.35
35	751	16	15	99159.71	91747.66	22.32	25.84	11.38	13.28	92.95	35.25
36	751	16	17	103501.30	96162.32	29.62	31.88	12.73	15.86	91.80	31.51
37	829	17	9	83764.56	69906.90	47.79	65.70	12.79	44.58	97.61	38.42
38	829	17	15	86895.58	77053.91	38.35	37.66	18.11	20.53	90.40	36.73
39	829	17	16	87959.53	78487.27	31.62	36.41	9.71	20.36	91.83	27.57
40	829	17	18	89347.42	81969.05	41.99	40.97	19.87	27.50	88.43	27.74

				Objective		Service time max deviation		Service time avg deviation		Similarity to DHL solutions	
Instance	$\|I\|$	$\|T\|$	$\|K\|$	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}
41	829	17	19	90282.71	81545.10	42.24	42.71	24.06	28.75	91.04	27.71
42	925	19	18	94565.20	87467.86	35.92	35.93	15.18	29.45	93.85	28.42
43	925	19	20	95907.39	89419.68	32.91	32.69	15.68	21.14	92.06	29.52
44	942	19	10	88330.87	68874.72	47.52	67.60	14.14	43.37	97.48	28.78
45	942	19	17	92713.46	78351.90	61.65	80.20	11.72	45.94	97.81	23.88
46	942	19	18	93542.82	79484.96	40.84	39.64	6.15	22.61	97.92	28.54
47	908	19	10	105134.08	89312.40	62.60	62.40	37.28	44.55	97.06	35.08
48	908	19	18	113248.93	100498.56	20.43	20.52	7.41	10.19	95.90	32.40
49	908	19	20	113977.13	103619.46	36.59	37.14	14.01	22.67	93.97	30.93
50	908	19	21	113417.08	104678.12	34.70	32.81	16.00	20.75	90.85	30.68
51	925	19	18	129736.31	116936.20	39.28	35.57	10.97	21.12	93.74	28.00
52	925	19	20	130423.66	119932.59	35.56	34.32	11.48	20.22	87.83	28.25
53	925	19	21	131597.14	122945.22	33.97	33.76	15.70	23.45	87.42	31.50
54	923	20	10	107567.55	93737.83	8.99	8.92	2.77	5.99	97.43	44.27
55	923	20	18	114129.51	106782.51	26.55	26.40	15.19	15.92	93.62	37.19
56	923	20	19	113749.54	109832.87	20.96	24.11	11.09	13.91	93.42	35.24
57	923	20	21	115343.65	109906.61	29.24	29.37	13.76	19.55	89.94	32.20
58	923	20	22	117492.02	107559.07	23.55	23.72	13.82	15.49	92.06	43.07
59	929	21	19	142544.76	133540.72	24.31	24.23	9.20	13.74	93.78	33.54
60	929	21	20	142402.56	134184.86	23.78	27.25	9.86	16.72	93.15	37.83
61	929	21	23	144643.26	137104.37	26.09	26.83	16.82	16.22	91.07	37.29
62	934	21	12	86240.44	65433.78	60.57	61.94	24.41	42.07	97.04	31.71
63	1114	23	21	131766.76	116649.61	76.59	86.03	14.05	51.30	95.95	23.70
64	1114	23	22	133506.39	121693.90	42.13	43.49	7.45	26.22	96.04	23.86
65	1051	23	12	93727.56	74681.86	44.44	54.04	10.28	40.06	97.27	29.44
66	1051	23	21	97360.82	83251.70	39.89	41.65	13.90	23.95	93.94	25.28
67	1051	23	22	98146.01	86494.15	38.88	38.83	10.66	20.30	93.29	23.58
68	1051	23	24	99382.19	87724.38	31.26	33.60	14.25	23.13	89.58	24.19
69	1051	23	25	98897.03	87281.41	39.90	38.12	17.06	20.61	85.87	22.49
70	965	24	23	108984.91	99914.10	35.69	35.65	13.53	23.30	92.11	31.88
71	965	24	25	110589.18	100706.53	34.95	34.53	17.07	22.11	85.25	25.86
72	1142	24	23	116821.71	101479.20	40.64	40.08	14.93	23.38	92.02	29.53
73	1142	24	25	117123.22	104477.06	34.95	32.44	15.42	16.38	93.83	28.88
74	1200	25	14	111157.56	98846.79	67.85	72.78	36.53	33.93	92.26	34.43
75	1200	25	24	125456.71	115068.56	51.68	53.67	19.35	30.61	88.64	28.68
76	1200	25	26	128275.41	119528.84	49.86	50.87	22.02	27.82	82.95	26.35
77	1200	25	28	131164.45	119911.50	63.36	63.54	28.74	37.96	90.15	27.52
78	1225	25	24	118660.41	109189.65	78.36	78.47	37.99	47.76	93.92	29.94
79	1119	26	25	132312.31	121274.17	37.77	36.33	9.09	20.80	92.31	35.66
80	1119	26	27	135184.95	126663.34	31.99	31.90	12.18	18.98	91.54	33.86
81	1080	26	25	112458.17	100580.55	57.69	57.97	19.96	32.30	90.14	25.43
82	1080	26	27	115260.39	103425.49	53.65	53.88	14.53	33.14	94.49	24.84
83	1275	26	25	148247.83	139948.74	40.79	41.86	13.89	27.14	92.69	29.00
84	1275	26	27	155004.65	140786.88	49.23	51.04	11.40	32.20	97.77	32.49
85	1275	26	28	158665.70	144263.40	50.97	50.93	16.46	25.29	97.54	35.61
86	1221	26	13	144496.74	130905.98	46.49	44.76	10.63	20.21	94.57	30.55
87	1221	26	22	153066.71	144137.87	54.77	54.15	17.09	23.80	92.20	34.03

				Objective		Service time max deviation		Service time avg deviation		Similarity to DHL solutions	
Instance	$\|I\|$	$\|T\|$	$\|K\|$	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}
88	1221	26	23	153527.34	148993.25	51.89	51.34	18.51	28.10	88.50	29.34
89	1221	26	24	155644.23	150039.35	49.98	48.24	19.48	24.72	92.29	35.58
90	1221	26	25	157353.58	150250.68	51.86	51.89	19.02	32.79	94.38	28.73
91	1221	26	27	159322.28	154204.09	45.66	45.29	19.90	26.40	86.94	32.85
92	1221	26	28	159080.51	155315.27	43.38	43.95	23.22	27.82	83.67	30.18
93	1199	29	27	136095.96	125769.70	54.03	54.87	17.05	28.43	92.41	28.06
94	1199	29	28	137041.84	124529.27	50.80	50.91	18.10	29.94	92.34	28.77
95	1199	29	32	139756.01	126973.19	47.59	47.94	22.59	28.35	88.63	28.84
96	1408	30	29	166029.52	161423.37	27.97	29.99	9.09	19.28	93.46	26.86
97	1282	30	15	165339.03	149813.66	36.54	38.91	6.85	18.74	97.30	30.07
98	1282	30	28	182095.37	170253.63	13.62	13.90	6.79	7.41	92.82	45.95

Table 3.20: Routing-based comparison of the solutions of the best run of ILS-VRPDOC$_{quality}$ (Q) to the solutions of the best run of ILS-VRPDOC$^{OFF}_{quality}$ (Q^{OFF}).

G Detailed results for the depot operation-based comparison of the ILS-VRPDOC$_{quality}$ solutions to the ILS-VRPDOC$^{OFF}_{quality}$ solutions

				Max #mail carriers		Avg #mail carriers		Max #tables		Avg #tables		Skips		Table changes	
Instance	$\|I\|$	$\|T\|$	$\|K\|$	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}
1	275	6	7	2.00	5.00	1.83	3.17	4.00	4.00	1.57	2.71	2.00	109.00	6.00	27.00
2	278	6	7	2.00	6.00	1.83	3.67	5.00	4.00	1.57	3.14	5.00	112.00	5.00	26.00
3	300	7	8	2.00	7.00	1.86	4.43	4.00	5.00	1.62	3.88	3.00	88.00	6.00	44.00
4	273	7	6	4.00	4.00	1.57	3.14	2.00	5.00	1.83	3.67	0.00	126.00	6.00	28.00
5	376	9	8	3.00	6.00	1.78	5.00	2.00	8.00	2.00	5.62	0.00	123.00	8.00	67.00
6	376	9	10	2.00	8.00	1.78	5.78	2.00	8.00	1.60	5.20	2.00	107.00	9.00	65.00
7	471	10	9	4.00	6.00	1.70	3.20	2.00	5.00	1.89	3.56	0.00	259.00	9.00	43.00
8	471	10	11	2.00	6.00	1.70	3.10	2.00	4.00	1.55	2.82	4.00	245.00	8.00	31.00
9	475	10	5	1.00	4.00	1.00	3.20	2.00	8.00	2.00	6.40	0.00	246.00	5.00	55.00
10	475	10	9	4.00	6.00	1.60	3.90	2.00	7.00	1.78	4.33	1.00	239.00	8.00	64.00
11	475	10	11	2.00	5.00	1.80	3.90	3.00	5.00	1.64	3.55	8.00	250.00	8.00	51.00
12	475	10	12	2.00	5.00	1.90	3.80	4.00	4.00	1.58	3.17	9.00	237.00	9.00	53.00
13	480	10	11	2.00	6.00	1.90	4.10	5.00	6.00	1.73	3.73	5.00	251.00	8.00	48.00
14	514	12	11	1.00	9.00	1.00	5.50	2.00	12.00	1.09	6.00	0.00	238.00	1.00	141.00
15	514	12	13	2.00	9.00	1.75	5.83	3.00	11.00	1.62	5.38	8.00	239.00	9.00	144.00
16	582	12	6	1.00	5.00	1.00	3.33	2.00	9.00	2.00	6.67	0.00	298.00	13.00	79.00
17	582	12	10	3.00	9.00	1.58	4.00	2.00	9.00	1.90	4.80	7.00	286.00	14.00	83.00
18	582	12	11	3.00	10.00	1.75	4.50	2.00	8.00	1.91	4.91	10.00	289.00	12.00	86.00
19	582	12	13	2.00	8.00	1.67	4.75	3.00	9.00	1.54	4.38	12.00	289.00	11.00	84.00
20	582	12	14	3.00	9.00	2.17	4.58	3.00	6.00	1.86	3.93	12.00	291.00	14.00	82.00
21	625	13	12	5.00	8.00	1.85	4.46	2.00	7.00	2.00	4.83	12.00	193.00	23.00	89.00
22	625	13	14	2.00	12.00	2.00	5.54	3.00	7.00	1.86	5.14	7.00	186.00	13.00	102.00
23	622	13	12	3.00	6.00	1.62	3.46	2.00	5.00	1.75	3.75	3.00	343.00	11.00	60.00
24	622	13	14	2.00	9.00	1.77	4.85	3.00	7.00	1.64	4.50	5.00	360.00	14.00	72.00
25	663	14	13	4.00	9.00	1.71	4.79	2.00	8.00	1.85	5.15	0.00	345.00	15.00	107.00
26	663	14	15	2.00	8.00	1.93	4.21	7.00	8.00	1.80	3.93	10.00	354.00	20.00	95.00
27	700	14	7	1.00	6.00	1.00	2.93	2.00	7.00	2.00	5.86	0.00	243.00	7.00	82.00
28	700	14	13	6.00	8.00	1.64	3.71	2.00	7.00	1.77	4.00	0.00	251.00	13.00	71.00
29	700	14	15	2.00	10.00	1.71	4.21	5.00	8.00	1.60	3.93	4.00	225.00	11.00	81.00
30	726	15	8	1.00	6.00	1.00	2.80	2.00	8.00	1.88	5.25	0.00	243.00	7.00	88.00
31	726	15	14	6.00	9.00	1.67	3.60	2.00	6.00	1.79	3.86	0.00	233.00	14.00	81.00
32	701	15	14	6.00	7.00	1.60	3.73	2.00	6.00	1.71	4.00	3.00	377.00	12.00	77.00
33	701	15	16	2.00	7.00	1.80	3.20	3.00	5.00	1.69	3.00	25.00	356.00	17.00	54.00
34	760	16	15	3.00	7.00	1.56	3.25	2.00	6.00	1.67	3.47	11.00	179.00	12.00	66.00
35	751	16	15	2.00	9.00	1.56	3.81	2.00	7.00	1.67	4.07	0.00	402.00	14.00	64.00
36	751	16	17	2.00	8.00	1.56	3.88	2.00	7.00	1.47	3.65	0.00	421.00	15.00	64.00
37	829	17	9	1.00	5.00	1.00	2.06	2.00	6.00	1.89	3.89	0.00	423.00	9.00	78.00
38	829	17	15	4.00	6.00	1.88	3.12	3.00	7.00	2.13	3.53	13.00	416.00	18.00	81.00
39	829	17	16	3.00	9.00	1.76	4.65	2.00	8.00	1.88	4.94	4.00	466.00	19.00	111.00
40	829	17	18	3.00	10.00	2.12	4.76	4.00	7.00	2.00	4.50	17.00	457.00	21.00	117.00

				Max #mail carriers		Avg #mail carriers		Max #tables		Avg #tables		Skips		Table changes	
Instance	$\|I\|$	$\|T\|$	$\|K\|$	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}
41	829	17	19	2.00	9.00	1.76	4.76	3.00	7.00	1.58	4.26	9.00	467.00	16.00	108.00
42	925	19	18	4.00	9.00	1.74	5.58	2.00	8.00	1.83	5.89	0.00	470.00	17.00	163.00
43	925	19	20	2.00	10.00	1.79	5.53	5.00	11.00	1.70	5.25	10.00	470.00	18.00	154.00
44	942	19	10	1.00	7.00	1.00	3.74	2.00	12.00	1.90	7.10	0.00	518.00	9.00	141.00
45	942	19	17	1.00	9.00	1.00	5.47	2.00	11.00	1.12	6.12	0.00	533.00	2.00	165.00
46	942	19	18	1.00	9.00	1.00	4.95	2.00	10.00	1.06	5.22	0.00	500.00	1.00	151.00
47	908	19	10	2.00	7.00	1.05	3.84	3.00	13.00	2.00	7.30	0.00	447.00	10.00	129.00
48	908	19	18	3.00	10.00	1.32	5.84	2.00	10.00	1.39	6.17	0.00	444.00	8.00	145.00
49	908	19	20	2.00	10.00	1.58	5.21	3.00	8.00	1.50	4.95	6.00	457.00	10.00	143.00
50	908	19	21	2.00	11.00	1.89	6.00	3.00	11.00	1.71	5.43	14.00	446.00	16.00	152.00
51	925	19	18	5.00	11.00	1.63	6.37	2.00	10.00	1.72	6.72	0.00	458.00	18.00	185.00
52	925	19	20	2.00	11.00	1.89	6.47	5.00	10.00	1.80	6.15	28.00	461.00	16.00	177.00
53	925	19	21	2.00	10.00	1.84	6.11	3.00	11.00	1.67	5.52	29.00	432.00	15.00	169.00
54	923	20	10	1.00	4.00	1.00	2.25	2.00	7.00	2.00	4.50	0.00	433.00	10.00	67.00
55	923	20	18	4.00	7.00	1.60	3.90	2.00	7.00	1.78	4.33	0.00	452.00	18.00	98.00
56	923	20	19	6.00	9.00	1.70	4.95	2.00	9.00	1.79	5.21	0.00	444.00	19.00	123.00
57	923	20	21	2.00	7.00	1.75	4.20	3.00	9.00	1.67	4.00	13.00	488.00	19.00	100.00
58	923	20	22	2.00	6.00	1.70	3.05	3.00	7.00	1.55	2.77	5.00	417.00	17.00	70.00
59	929	21	19	4.00	7.00	1.62	4.52	2.00	8.00	1.79	5.00	0.00	457.00	18.00	127.00
60	929	21	20	7.00	10.00	1.67	4.90	2.00	9.00	1.75	5.15	0.00	428.00	19.00	119.00
61	929	21	23	2.00	10.00	1.71	5.05	3.00	10.00	1.57	4.61	12.00	431.00	16.00	115.00
62	934	21	12	1.00	7.00	1.00	3.62	2.00	12.00	1.75	6.33	0.00	488.00	9.00	136.00
63	1114	23	21	1.00	12.00	1.00	6.35	2.00	13.00	1.10	6.95	0.00	603.00	2.00	220.00
64	1114	23	22	1.00	12.00	1.00	6.52	2.00	10.00	1.05	6.82	0.00	596.00	1.00	222.00
65	1051	23	12	1.00	7.00	1.00	3.09	2.00	13.00	1.92	5.92	0.00	574.00	11.00	154.00
66	1051	23	21	4.00	10.00	1.57	4.52	2.00	9.00	1.71	4.95	0.00	594.00	19.00	165.00
67	1051	23	22	6.00	10.00	1.65	5.57	2.00	9.00	1.73	5.82	1.00	579.00	20.00	198.00
68	1051	23	24	2.00	9.00	1.78	5.17	4.00	8.00	1.71	4.96	22.00	580.00	19.00	179.00
69	1051	23	25	3.00	12.00	2.17	5.52	4.00	11.00	2.00	5.08	30.00	585.00	32.00	191.00
70	965	24	23	10.00	11.00	1.71	4.71	2.00	10.00	1.78	4.91	2.00	491.00	21.00	131.00
71	965	24	25	2.00	13.00	1.79	5.29	3.00	10.00	1.72	5.08	26.00	518.00	29.00	165.00
72	1142	24	23	5.00	9.00	1.88	5.21	2.00	9.00	1.96	5.43	6.00	601.00	26.00	174.00
73	1142	24	25	2.00	13.00	1.88	6.04	7.00	11.00	1.80	5.80	9.00	597.00	24.00	192.00
74	1200	25	14	2.00	8.00	1.20	3.40	3.00	8.00	2.14	6.07	21.00	635.00	16.00	124.00
75	1200	25	24	5.00	8.00	1.92	5.56	3.00	11.00	2.00	5.79	31.00	651.00	25.00	169.00
76	1200	25	26	2.00	10.00	1.88	6.08	3.00	11.00	1.81	5.85	53.00	658.00	38.00	177.00
77	1200	25	28	2.00	9.00	1.48	5.88	3.00	12.00	1.32	5.25	31.00	667.00	10.00	166.00
78	1225	25	24	5.00	10.00	1.76	5.68	2.00	11.00	1.83	5.92	3.00	651.00	20.00	169.00
79	1119	26	25	4.00	7.00	1.58	3.54	2.00	6.00	1.64	3.68	9.00	551.00	18.00	133.00
80	1119	26	27	2.00	7.00	1.65	3.73	3.00	6.00	1.59	3.59	9.00	572.00	21.00	130.00
81	1080	26	25	6.00	11.00	1.92	5.88	2.00	13.00	2.00	6.12	3.00	557.00	38.00	218.00
82	1080	26	27	2.00	15.00	1.88	6.54	9.00	13.00	1.81	6.30	4.00	552.00	23.00	235.00
83	1275	26	25	4.00	11.00	1.77	5.65	2.00	11.00	1.84	5.88	6.00	661.00	24.00	205.00
84	1275	26	27	2.00	12.00	1.04	5.54	1.00	11.00	1.00	5.33	0.00	643.00	0.00	184.00
85	1275	26	28	2.00	11.00	1.08	5.92	1.00	10.00	1.00	5.50	0.00	598.00	0.00	187.00
86	1221	26	13	1.00	8.00	1.00	4.19	2.00	13.00	2.00	8.38	0.00	640.00	28.00	191.00
87	1221	26	22	4.00	10.00	1.54	4.58	2.00	9.00	1.82	5.41	14.00	597.00	21.00	169.00

				Max #mail carriers		Avg #mail carriers		Max #tables		Avg #tables		Skips		Table changes	
Instance	$\|I\|$	$\|T\|$	$\|K\|$	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}	Q	Q^{OFF}
88	1221	26	23	7.00	13.00	2.19	5.35	3.00	10.00	2.48	6.04	14.00	615.00	43.00	207.00
89	1221	26	24	6.00	9.00	1.77	5.08	2.00	11.00	1.92	5.50	12.00	577.00	23.00	168.00
90	1221	26	25	4.00	12.00	1.42	5.12	2.00	9.00	1.48	5.32	6.00	651.00	13.00	186.00
91	1221	26	27	2.00	11.00	1.92	4.88	4.00	9.00	1.85	4.70	30.00	616.00	33.00	164.00
92	1221	26	28	3.00	10.00	2.42	5.12	4.00	9.00	2.25	4.75	42.00	638.00	47.00	168.00
93	1199	29	27	5.00	11.00	1.66	4.76	2.00	9.00	1.78	5.11	3.00	634.00	26.00	189.00
94	1199	29	28	5.00	11.00	1.72	4.72	2.00	9.00	1.79	4.89	0.00	624.00	26.00	186.00
95	1199	29	32	2.00	9.00	1.62	4.55	3.00	9.00	1.47	4.12	26.00	618.00	24.00	185.00
96	1408	30	29	5.00	12.00	1.70	6.10	2.00	11.00	1.76	6.31	0.00	776.00	27.00	210.00
97	1282	30	15	1.00	8.00	1.00	4.10	2.00	15.00	2.00	8.20	0.00	669.00	15.00	208.00
98	1282	30	28	5.00	5.00	1.63	2.63	2.00	5.00	1.75	2.82	0.00	555.00	28.00	92.00

Table 3.21: Depot operation-based comparison of the solutions of the best run of ILS-VRPDOC$_{quality}$ (Q) to the solutions of the best run of ILS-VRPDOC$^{OFF}_{quality}$ (Q^{OFF}).

Chapter 4

The angular traveling salesman problem

Publication status: R. Cavagnini, M. Schneider, and A. Theiß (2024b). “A tabu search with geometry-based sparsification methods for angular traveling salesman problems”. In: *Networks* 83.1, pp. 30–52. DOI: `10.1002/net.22180`

Abstract: The angular-metric traveling salesman problem (AngleTSP) aims to find a tour visiting a given set of vertices in the Euclidean plane exactly once while minimizing the cost given by the sum of all turning angles. If the cost is obtained by combining the sum of all turning angles and the traveled distance, the problem is called angular-distance-metric traveling salesman problem (AngleDistanceTSP). In this work, we study the symmetric variants of these problems. Because both the AngleTSP and AngleDistanceTSP are NP-hard, multiple heuristic approaches have been proposed in the literature. Nevertheless, a good tradeoff between solution quality and runtime is hard to find. We propose a granular tabu search (GTS) that considers the geometric features of the two problems in the design of starting solutions and sparsification methods. We further enrich the GTS with components that guarantee both intensification and diversification during the search. The computational results on benchmark instances from the literature show that (i) for the AngleTSP, our GTS lies on the Pareto frontier of the best performing-heuristics, and (ii) for the AngleDistanceTSP, our GTS provides the best solution quality across all existing heuristics in competitive runtimes. In addition, new best-known solutions are found for most benchmark instances for which an optimal solution is not available.

Contribution of the author: The authors shared efforts in the conceptual development of the research goals, the literature review, the design of the methodology and implementation of the algorithm, the computational experiments and result analysis, and in writing the paper.

4.1 Introduction

The angular-metric traveling salesman problem (AngleTSP) aims to find a Hamiltonian cycle visiting a given set of vertices in the Euclidean plane while minimizing the cost given by the sum of all turning angles. If the cost is obtained by combining the sum of all turning angles and the traveled distance, the problem is called angular-distance-metric traveling salesman problem (AngleDistanceTSP). The two problems have applications in, e.g., robotics and transportation. In robotics, keeping the movements of a robot as straight as possible is key to avoiding energy-consuming changes in driving direction. In transportation, straight movements make heavy vehicles more controllable. For example, drivers of farm machinery such as tractors and harvesters often have to execute operations on an irregular ground. In this situation, sharp corners should be avoided as much as possible to avoid accidents (see, e.g., Abubakar et al., 2010). This is also true for drivers of semitrailer trucks that deliver goods in city centers in which corners are typically sharp and tight. By driving sharp corners (even at a low speed), the trailer may end up at a 90-degree angle relative to the vehicle it is attached to (a phenomenon called "jackknifing", see Leng et al., 2022). This is an unpleasant situation for the driver, and it increases the hazard to nearby vehicles, properties, and pedestrians as reported by the Federal Motor Carrier Safety Administration, 2019. Moreover, poor road conditions, bad weather conditions, and improperly secured load may cause rollovers (even at slow speed) because of the high center of gravity of semitrailer trucks (McKnight and Bahouth, 2009). In all these situations, straight movements are preferable to turning maneuvers.

In this work, we consider the symmetric variants of these problems, which are formally defined as follows. Let $G = (V, A)$ be a complete directed graph with vertex set $V = \{1, 2, \ldots, n\}$ in the Euclidean plane and arc set $A = \{(i, j) : i, j \in V, i \neq j\}$. The Euclidean distance d_{ij} is associated with each arc and a non-negative turning angle α_{ijk} with each vertex triplet $(i, j, k), i, j, k \in V$. The cost for each triplet (i, j, k) is defined as $c_{ijk} = \rho_1 \alpha_{ijk} + \rho_2 (d_{ij} + d_{jk})/2$, where ρ_1 and ρ_2 are the weights associated with the turning angles and the distances, respectively. For the AngleTSP, ρ_2 is equal to 0. A solution of the AngleTSP or AngleDistanceTSP is denoted by S, and it is a tour visiting each vertex in V exactly once. A solution is optimal if it minimizes the cost of the tour $C(S)$.

Because both problems are NP-hard and exact solution methods can only solve small instances in reasonable runtimes (see Fischer et al., 2014), Staněk et al., 2019 proposed multiple heuristics. The best-performing methods are either simple construction heuristics or matheuristics: the former are fast but provide unsatisfying solution quality, the latter achieve superior solution quality but are expensive in terms of runtime. There is evidence of a gap in the literature concerning the availability of fast but effective heuristics for both problem variants.

This paper contributes to filling this gap by presenting a granular tabu search (GTS) framework, called *GTS-angular*, to address both problem variants. To obtain decent starting solutions for GTS-angular, we exploit geometric properties of good solutions. For the AngleTSP, we present a faster version of the convex hull construction heuristic proposed by Staněk et al., 2019. For the AngleDistanceTSP, we develop a new construction method based on our empirical observation that good solutions typically do not contain arcs that intersect. A tour containing intersections is not optimal with regard to the traveled distance, which is one of the components to be minimized in the objective function of the AngleDistanceTSP.

To speed up the search, our tabu search adopts the granular search principle introduced by Toth and Vigo, 2003. This principle relies on sparsification methods which are used to restrict the size of the neighborhoods to explore. GTS frameworks provide high-quality solutions within reduced runtimes for several routing-related problems like the capacitated vehicle routing problem (CVRP, Toth and Vigo, 2003), the vehicle routing problem (VRP) with time windows (Schneider et al., 2017), the multi-depot VRP (Escobar et al., 2014), the pickup and delivery problem with time windows and electric vehicles (Goeke, 2019), the dial-a-ride problem (Kirchler and Wolfler Calvo, 2013), and the capacitated location routing problem (CLRP, Prins et al., 2007; Schneider and Löffler, 2019).

In this work, we strengthen the sparsification method originally proposed by Staněk et al., 2019 of using a lens to limit the size of the neighborhood to search. Our contribution is to design two problem-specific sparsification methods which are applied to both the AngleTSP and AngleDistanceTSP and which give rise to two algorithmic variants of GTS-angular.

We run numerical studies on benchmark instances from the literature to test the performance of the two GTS-angular variants differing with respect to the sparsification method. Compared with the best-performing heuristics developed by Staněk et al., 2019, GTS-angular improves either the solution quality or the runtimes, sometimes both, and finds new best-known solutions for many benchmark instances.

The remainder of the paper is organized as follows. In Section 4.2, we review the literature related to the AngleTSP and the AngleDistanceTSP. Section 4.3 describes GTS-angular in detail. We discuss the experimental setting and the computational results in Section 4.4. Finally, Section 4.5 presents our conclusion.

4.2 Literature review

The AngleTSP has been introduced by Aggarwal et al., 2000, who show that this problem is NP-hard. Savla et al., 2008 are the first authors to study the AngleDistanceTSP, which is later formalized by Medeiros and Urrutia, 2010 as an approximation of the TSP in which the turn radius of the car is limited.

In both problem variants, the cost arises for the successive use of two arcs. Hence, most research has treated these problems as generalizations of the quadratic TSP (QTSP). Polyhedral properties for the symmetric QTSP are studied in Fischer and Helmberg, 2013, who derive classes of strengthened subtour elimination constraints. Fischer et al., 2014 study the symmetric QTSP and propose both exact and heuristic approaches to solve it. The exact approaches include: a transformation of the problem to the TSP, then solved by standard methods, a branch-and-bound algorithm, and a branch-and-cut algorithm. These methods solve instances with up to 40 vertices in reasonable runtimes. The heuristic approaches are based on modifications of heuristics commonly applied to the TSP. While these methods are faster than the exact ones, they show significant optimality gaps even for relatively small instances with up to 50 vertices. Fischer et al., 2015 investigate the performance of three exact methods for solving QTSP instances with bioinformatic applications.

Apart from Aggarwal et al., 2000, who exclusively study the AngleTSP, the only paper in the literature focusing exclusively on the AngleTSP and AngleDistanceTSP is the one of Staněk et al., 2019. The authors propose a large number of heuristic approaches that can be classified into the following categories: construction heuristics, geometric heuristics (applied only to the AngleTSP), and linear programming-based approaches. The authors extend these methods by applying improvement algorithms, such as 2-opt and 3-opt heuristics, a destroy-and-repair matheuristic, and a simulated annealing metaheuristic. Their fastest methods are the simple construction heuristics, and the methods returning the best solution quality are the linear programming-based approaches improved by the destroy-and-repair matheuristic. A good trade-off between solution quality and runtime is difficult to find. Moreover, the authors highlight that the quite notable optimality gaps make the development of metaheuristics a promising research topic.

4.3 Granular tabu search for the angular metric traveling salesman problem

In this section, we describe the general idea of tabu search (TS) and we introduce the GTS-angular framework. Section 4.3.1 provides details on how the starting solution is obtained, Section 4.3.2 describes the used neighborhoods, Section 4.3.3 introduces the sparsification methods, and Section 4.3.4 presents the continuous diversification strategy.

The TS framework was introduced by Glover, 1989. In its basic form, it includes a local search that iteratively explores neighboring solutions. Two main features of TS are (i) the presence of a memory, which contains information on which moves are tabu to avoid cycling, and (ii) the application of the best non-tabu move in each iteration—which may be deteriorating—to escape from local optima.

Algorithm 6 provides a pseudocode overview of GTS-angular and its main components are detailed in the remainder of this section. After obtaining a starting solution according to the construction heuristic described in Section 4.3.1 (line 1) and initializing the current solution S^{curr} and the overall best-found solution S^* with it, the TS component is initialized (line 2) and executed (lines 3–23). The tabu criterion forbids to reinsert an arc that has been removed for a tenure of τ iterations. As aspiration criterion, we always permit a move that is tabu but improves on the overall best-found solution S^*.

At the beginning of each iteration, we define the best solution found in this iteration by S, we initialize it to an empty tour and set its cost to infinity (line 4). In each iteration, GTS-angular evaluates neighboring solutions that are obtained by applying a move defined by a neighborhood operator and a so-called generator arc. The neighborhood operator defines the type of modification applied to a solution (e.g., if one vertex is relocated to a new position or if two vertex positions are exchanged). All neighborhood operators are designed in such a way that the generator arc is always an arc inserted in the solution. For a given neighborhood operator, the generator arc unambiguously defines the move, that is, all the other arcs involved in the move are implicitly defined. Our algorithm first traverses the generator arc set A_g (line 5). To speed up the search, the dimension of A_g is reduced to the arcs selected by applying the sparsification methods described in Section 4.3.3. For each generator arc, we loop over the relocate, exchange, and 2-opt operators contained in the set $\mathcal{O}$ (line 6) and presented in Section 4.3.2, and the move applied to S^{curr} (resulting in $S^{'}$) is evaluated with respect to its cost $C(S^{'})$ (line 7-8). Moves deteriorating the cost of S^{curr} are evaluated according to the extended objective function $C_{div}(S^{'})$ that considers the number of times the inserted arcs were included in previously carried-out moves (Section 4.3.4). This continuous diversification mechanism (lines 9–11) guides the search to new areas of the solution space by favoring solutions with arcs that have rarely been included in previous iterations. Within each iteration of the TS, we keep track of the best non-tabu solution S (lines 12–18). Note that, we have also tested the first-improvement variant of GTS-angular during our computational experiments. However, the gained runtime advantage was not high enough to counterbalance the experienced loss in solution quality. Consequently, we only focus on the best-improvement variant of GTS-angular. If the move is tabu and does not satisfy the aspiration criterion (lines 12–14), TS omits the cost check and the eventual update of the best solution S of the current iteration (lines 15–18). Finally, S^{curr} is updated with S, i.e., the best found solution in this iteration (line 21), and the overall best solution S^*, the tabu list, and the generator arc set are updated (line 22). GTS-angular terminates after η iterations without improvement.

Algorithm 6: Pseudocode of GTS-angular

```
$S^{curr} \leftarrow constructionHeuristic()$
initialize $S^* \leftarrow S^{curr}$, $tabuList$, and generator arc set $A_g$
while $termination\ criterion\ not\ satisfied$ do
    $S = \emptyset$, $cost^S = \infty$
    for $(i, j) \in A_g$ do
        for $o \in \mathcal{O}$ do
            $S^{'} \leftarrow move((i, j), o, S^{curr})$
            $cost^{S^{'}} = C(S^{'})$
            if $cost^{S^{'}} \geq C(S^{curr})$ then
                $cost^{S^{'}} = C_{div}(S^{'})$
            end
            if $S^{'} \in tabuList\ and\ notAspiration$ then
                continue
            end
            if $cost^{S^{'}} < cost^S$ then
                $S \leftarrow S^{'}$
                $cost^S = cost^{S^{'}}$
            end
        end
    end
    $S^{curr} \leftarrow S$
    update $S^*$, $tabuList$, $A_g$
end
```

4.3.1 Construction heuristics

To generate an initial solution, we use separate algorithms for the AngleTSP (Section 4.3.1.1) and the AngleDistanceTSP (Section 4.3.1.2).

4.3.1.1 Initial solution for the AngleTSP

For the AngleTSP, the literature has shown that optimal tours have a snail shell shape (see, e.g., Aichholzer et al., 2017; Staněk et al., 2019). Therefore, we use a construction heuristic based on convex hulls similar to the one described in Staněk et al., 2019, and we call it convex hull heuristic (CHH). Our construction heuristic works as follows. We first determine the convex hull considering the complete vertex set V. While Staněk et al., 2019 does not specify how the convex hulls are computed, we use a modified version of Graham's scan algorithm (Graham, 1972). This modified version is described by Cormen et al., 2009: instead of computing the actual polar angles via trigonometric functions and sorting the vertices accordingly in increasing order, vertices are sorted by only comparing the polar angles via cross products. Given are a reference vertex v_0, which is the vertex with the minimum y-coordinate and a pair of vertices v_1 and v_2. To decide whether the polar angle of v_1 with respect to v_0 is larger or smaller than the polar angle of v_2 with respect to v_0, a single cross product can be computed without computing the actual angle. If the sign of the cross product is negative, then a left turn must be taken to move from v_0 to v_1 to v_2, i.e., the polar angle is wider for v_2 than for v_1. If the sign is positive, a right turn must be taken, i.e., the polar angle is smaller for v_2 than for v_1. The special case of a cross product equal to zero represents collinearity of vertices v_0, v_1 and v_2. In this case, the vertex farther away from v_0 is chosen first.

Once the outermost convex hull is found, we remove the vertices defining this convex hull from V, and we compute the convex hull on the remaining vertices. We repeat the procedure until at most one vertex is left. Finally, these convex hulls (and the final vertex, if any) are connected to form a tour in the cheapest insertion fashion from the innermost to the outermost convex hull. We have also tested a variant of this construction heuristic in which the convex hulls are merged in the opposite order, i.e., from the outermost to the innermost. However, the results reported in Section 4.4.3 suggest that merging the convex hulls from the innermost to the outermost convex hull provides better results. Specifically, given a pair of convex hulls, the inner one is inserted in the outer one at the position causing the minimal cost increase. If the innermost "convex hull" consists of just two vertices, they are inserted at once. Our insertion procedure makes our construction heuristic faster than the procedure described in Staněk et al., 2019. In their approach, every time two convex hulls must be merged, the authors check, for all pairs of convex hulls and all edge pairs, the connection returning the smallest sum of objective function values over all subtours. Figure 4.1 shows an example of constructing an initial solution for the AngleTSP according to our approach. In step 1, we obtain three convex hulls corresponding to three subtours. In step 2, we connect the two innermost convex hulls in cheapest insertion fashion, and two subtours are left. Finally, in step 3, the innermost subtour is connected to the outermost convex hull. Because this last step results in a tour visiting all vertices in V, the algorithm terminates.

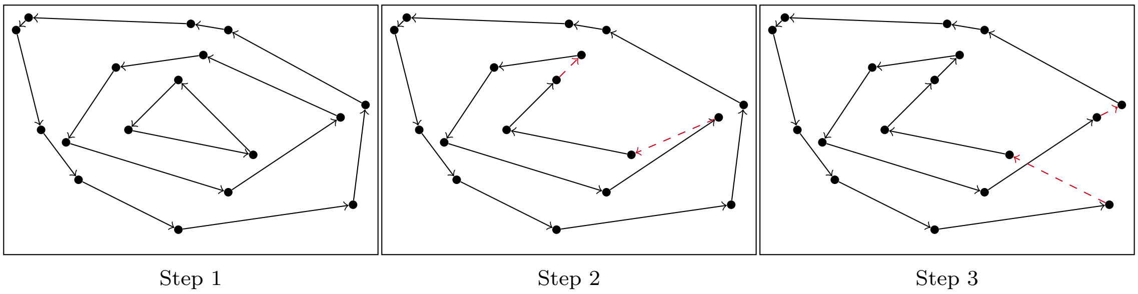
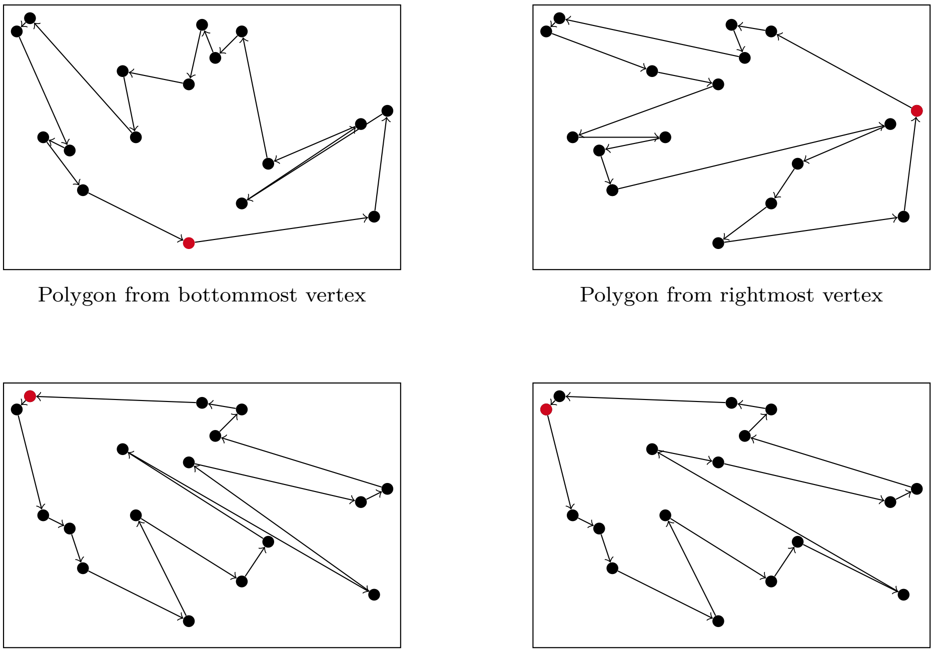


Figure 4.1: Example of constructing an initial solution for the AngleTSP.

4.3.1.2 Initial solution for the AngleDistanceTSP

By plotting several optimal solutions of the AngleDistanceTSP in which the turning angle and distance component are relatively balanced, we observed that optimal tours usually contain no intersections. This empirical observation is in line with the fact that, in the Euclidean plane, in a minimal TSP tour no arcs intersect. To obtain such a closed path with no intersections, we designed a construction heuristic that has similarities to the sweep algorithm of Gillett and Miller, 1974, and we call it no-intersections heuristic (NIH). We first select a starting vertex v_0. Then, the remaining vertices in $V \setminus \{v_0\}$ are sorted in increasing order by comparing the polar angles via cross products (see above). This results in a list of vertices arranged in counterclockwise order. Finally, we obtain a tour by connecting v_0 with the first vertex of the list, all the vertices in the list according to the order in which they appear, and the last vertex of the list to v_0. Because the resulting tour depends on the choice of the starting vertex, we repeat the procedure for all possible starting vertices, and we choose the tour with the lowest cost. Figure 4.2 shows an example of four tours obtained by selecting different starting vertices highlighted in red.



Polygon from bottommost vertex

Polygon from rightmost vertex

Polygon from topmost vertex

Polygon from leftmost vertex

Figure 4.2: Example of four tours obtained from different starting vertices for the AngleDistanceTSP.

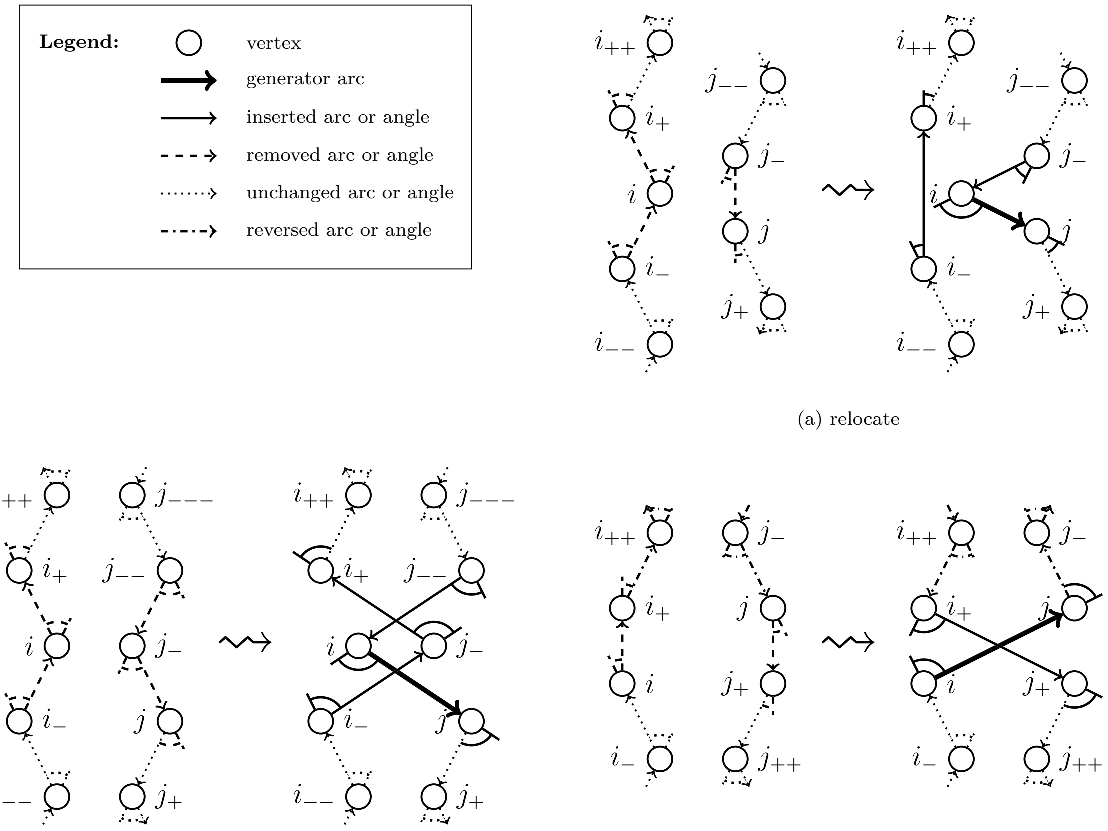
4.3.2 Neighborhoods

In each iteration, GTS-angular applies the best non-tabu move in a composite neighborhood defined by the set $\mathcal{O}$ containing three operators: relocate, exchange, and 2-opt (Toth and Vigo, 2003). Differently from the standard TSP, to compute the impact that each move has on the cost of a solution, we have to take into account that the cost is defined for triplets of vertices. This implies that each modification of a vertex position causes a change not only of the cost corresponding to that vertex but also on the cost associated with its predecessor and successor.

All operators are defined using the generator arc principle introduced in Section 4.3.

Figure 4.3 presents the operators:

- The relocate operator moves one vertex from its current position to another one in the tour.
- The exchange operator swaps the positions of two vertices in the tour.
- The 2-opt operator removes two non-consecutive arcs from the tour, inverts the vertex segment between the two removed arcs, and reconnects the inverted segment using two new arcs.



(a) relocate

(b) exchange

(c) 2-opt

Figure 4.3: Neighborhood operators of GTS-angular. The generator arc is denoted by (i, j) and highlighted in bold. The predecessor and successor of i are denoted by i_- and i_+, respectively.

4.3.3 Construction of the generator arc set

To reduce the neighborhood size, we recall that the generator arc set A_g contains a subset of the complete arc set A and that this subset is determined by applying sparsification methods. Sparsification methods were introduced by Toth and Vigo, 2003 and have later been used in multiple works (see, e.g., Prins et al., 2007; Escobar et al., 2014; Goeke, 2019).

In our case, A_g is composed of the arcs contained in the arc sets A_a and A_s. A_a includes the arcs accepted by the most recent moves, and we explain how it is obtained in Section 4.3.3.1. A_s contains the arcs selected by one of our two sparsification strategies, and, in Section 4.3.3.2, we describe how it is derived. Finally, in Section 4.3.3.3, we provide the details on how the arcs in A_a and A_s are merged into set A_g.

4.3.3.1 Composition of A_a

Different strategies have been suggested for guaranteeing that the generator arc set contains the arcs belonging to high-quality solutions. Toth and Vigo, 2003, Schneider et al., 2017, and Becker et al., 2021 include the arcs that were inserted in the best move selected in each iteration. In the first two papers, the authors remove these arcs repetitively after a given number of iterations, while, in the third paper, these arcs are always kept. Neverthess, all these papers agree that the inclusion of the arcs inserted by the best move in the last iteration is fundamental for achieving good solution quality.

In GTS-angular, we add the arcs inserted by the most recently accepted moves to the generator arc set. Initially, A_a is an empty set, and, during the search, its cardinality cannot exceed the limit $\lceil \kappa_a |A| \rceil$, with $0 < \kappa_a < 1$. At the end of each iteration, after the move is carried out, the set of arcs inserted by the applied move is represented by A_{ins}, and each of its elements is added to A_a with a time stamp representing the moment in which they have been inserted. If, in an iteration, an arc in A_{ins} has already been added by previous accepted moves, the time stamp of that arc is updated. When the cardinality limit of A_a is reached, the arcs with the oldest time stamp are removed from the set and those added by the last move are inserted. By definition, a generator arc defines a valid move if it is not currently contained in the tour. Hence, the arcs in A_a can only become generator arcs after they have been removed in one of the previous GTS-angular iterations. However, this is not in contrast with the idea of TS that tries to prevent the reinsertion of recently removed arcs. First, we recall that the tabu tenure of TS forbids to reinsert removed arcs only for a limited number of iterations. Consequently, a removed arc can always be reinserted in the tour at a later point of the search. Second, the same generator arc can generate a different move from iteration to iteration because the second arc that is inserted depends on the current solution. Hence, considering a recently removed arc as generator arc does not necessarily lead to obtaining the same solution again. Finally, if a tabu yet promising arc appears in A_a, it can be used coupled with a different neighborhood operator and generate a different solution. Including such an arc in A_a guarantees that this arc is considered for insertion in the tour, and it increases the likelihood that the aspiration criterion is satisfied.

4.3.3.2 Composition of A_s

Even if most of the papers using sparsification strategies rely on the arc cost-based sparsification method originally proposed by Toth and Vigo, 2003, other authors have presented alternative methodologies (Labadie et al., 2012; Schneider et al., 2017). Due to the special features of the AngleTSP and AngleDistanceTSP, we propose two sparsification strategies.

Both approaches rely on a modification of the general lens procedure (LENS) introduced by Staněk et al., 2019. In LENS, a lens with thickness γ is positioned between two consecutive vertices i and i_+ in a tour such that the lens curvature intersects at these vertices. Only the vertices inside this lens are considered for insertion in the tour between i and i_+. This procedure guarantees that any inserted vertex generates: (1) an additional angle that is smaller than the angle of the lens curvature at the tangent point with the vertices, and (2) a limited increase in the traveled distance. The left part of Figure 4.4 shows an example of LENS applied to the arc (i, i_+). In this example, only vertex k is within the lens and hence considered for insertion between i and i_+. The right part of Figure 4.4 shows the resulting insertion and updated tour section.

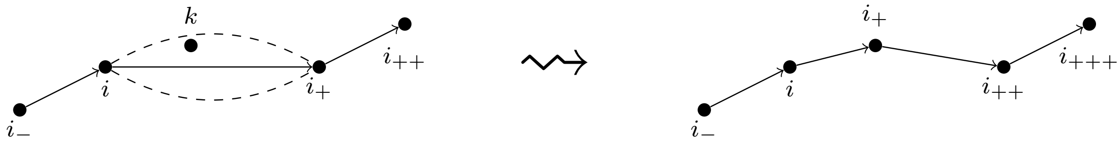


Figure 4.4: Example of LENS applied to the arc (i, i_+).

In contrast to Staněk et al., 2019, who apply LENS to every arc of a tour and include all the resulting arcs in A_s, we further intensify the sparsification by limiting the cardinality of A_s to $\lceil \kappa_s |A| \rceil$, with $0 < \kappa_s < 1$. The parameter κ_s represents the sparsification intensity: the lower κ_s, the more intense the sparsification, i.e., only a few arcs are included in A_s. This implies that a criterion to select the arcs on which the lens should be positioned has to be defined. To this end, we propose the following two strengthened LENS-based sparsification strategies:

- **Random-based lens procedure (r-LENS)**: In each iteration, the lens is initially positioned on the arc originating from a vertex i, and then applied to the subsequent arcs according to their order in the tour. However, across iterations, if the lens is always initially positioned on the arc originating from the same vertex i, there may be little variety in A_s due to its restricted cardinality. To prevent this, r-LENS starts positioning the lens on the arc of a random vertex in each iteration. This ensures that different parts of the tour are targeted across iterations.

- **Cost-based lens procedure (c-LENS)**. According to this approach, the turning angles of the tour are first sorted in non-increasing order. Then, the lens is applied sequentially, from the pair of arcs defining the largest angle to the smallest one, until the cardinality limit of A_s is reached. This sparsification procedure is greedier than r-LENS, and it aims to improve the most expensive parts of the tour.

Contrary to the sparsification methods proposed in the literature, which are applied only once before the search starts, our sparsification methods depend on the current solution and are therefore called at the end of each iteration of GTS-angular, and applied to the tour resulting after carrying out the best non-tabu move. Arcs which have already been included in A_a are not added to A_s.

To introduce more diversification, both sparsification strategies are dynamic. In contrast to common practice in the literature, in which the dynamic component is based on variations of the sparsification intensity κ_s (e.g., Toth and Vigo, 2003; Schneider and Löffler, 2019), in r-LENS and c-LENS, the sparsification intensity remains unchanged. Instead, we modify the criterion according to which the arcs are selected, i.e., we vary the lens thickness in the interval $[\gamma_{min},\ 2\gamma_{min}\ ]$. More precisely, the lens thickness is initially set equal to γ_{min}. Every η^{κ} iterations without improvement, the lens thickness is increased by γ_{min}/μ, where $\mu = \eta/\eta^{\kappa}$-1. In the last η^{κ} iterations of GTS-angular, the lens thickness is doubled with respect to its initial value γ_{min}. Enlarging the lens thickness γ when improvements are hard to find allows inserting more diverse arcs in the generator arc set while keeping its size and the resulting neighborhood restricted. Whenever an improvement with respect to the overall best solution is found, the lens thickness is reset to γ_{min}.

Based on the used sparsification method, we derive two algorithmic variants, namely GTS-angular r-LENS and GTS-angular c-LENS.

4.3.3.3 Composition of A_g

Algorithm 7 provides an overview of how the generator arc set A_g is updated during the search depending on the sets A_a and A_s. At the beginning of GTS-angular, A_s contains the arcs added by the sparsification strategy applied to the tour of the current solution S^{curr} resulting from the construction heuristic, and A_a and A_g are initialized to empty sets (line 1). All arcs contained in A_s are added to A_g (line 2). Then, lines 4–20 of the pseudocode in Figure 6 are executed. At the end of every iteration of GTS-angular (see Figure 6), after the best move is carried out, the tour of the new current solution S^{curr} and the set A_{ins} of the arcs inserted by the move are obtained (line 4). Next, A_g is cleaned from all arcs (line 5). The set A_a is updated by adding the arcs contained in A_{ins} (line 6). Subsequently, A_s is obtained by applying the sparsification strategy to the tour of the current solution S^{curr} and by discarding those arcs which are already in A_a (line 7). If these arcs would be included in A_s, they would increase the cardinality of A_s unnecessarily (and consequently prevent other arcs to be added to A_s) because they would be part of A_g in any case. Finally, the arcs contained in A_a and A_s are merged into A_g (line 8). The set A_g is then used for the next iteration of GTS-angular.

Algorithm 7: Pseudocode for the composition of the generator arc set A_g.

```
1 initialize $A_s \leftarrow sparsificationStrategy(S^{curr})$, $A_a = \emptyset$, $A_g = \emptyset$
2 Add arcs in $A_s$ to $A_g$
3 while termination criterion not satisfied do
4     $S^{curr}$, $A_{ins} \leftarrow$ apply best move {lines 4–20 of pseudocode in Figure 6}
5     Delete all arcs from $A_g$
6     Add arcs in $A_{ins}$ to $A_a$
7     $A_s \leftarrow sparsificationStrategy(S^{curr}, A_a)$
8     Add arcs in $A_a$ and $A_s$ to $A_g$
9 end
```

4.3.4 Continuous diversification

We incorporate the continuous diversification strategy originally proposed by Cordeau et al., 1997 into GTS-angular. This diversification technique has been successfully applied in a number of TS metaheuristics developed for different problem classes (e.g., the CLRP and the multi-depot VRP). The goal of continuous diversification is to guide the search towards unexplored regions of the solution space.

Whenever a move does not improve the cost value, i.e., a move is changing S^{curr} into $S^{'}$ with $C(S^{'}) \geq C(S^{curr})$, we penalize the cost of that solution based on its similarity with respect to the previously visited solutions. To measure this similarity, we keep a counter $\Lambda(a)$ for every arc $a \in A$. Every time GTS-angular carries out a move that inserts the set of arcs A_{ins}, we increase $\Lambda(a)$ by one for every arc $a \in A_{ins}$. The resulting formula to evaluate non-improving solutions corresponds to an extended objective function that considers the number of times the inserted arcs were included in previously carried-out moves. The extended objective function is denoted by $C_{div}(S^{'})$ and is computed as follows:

$$C_{div}(S^{'}) = C(S^{'}) \left(1 + \sigma \cdot \sum_{a \in A_{ins}} \Lambda(a) \right), \tag{4.1}$$

where $C(S^{'})$ is the cost of solution $S^{'}$ (i.e., the neighboring solution of S^{curr}), σ is a parameter that controls the intensity of the diversification, and $\sum_{a \in A_{ins}} \Lambda(a)$ is the number of times that the arcs inserted by the applied move were contained in previously carried-out moves. Whenever a new best solution is found, $\Lambda(a)$ is reset to zero for every arc $a \in A$; thus, $C_{div}(S^{'}) = C(S^{'})$.

4.4 Computational experiments

The goal of the computational experiments is twofold. First, we assess the quality of the solutions returned by the construction heuristics described in Section 4.3.1 compared to the quality of a random starting solution, and we evaluate the influence of these starting solutions on the final objective function values and runtimes of GTS-angular. Second, we compare our two algorithmic variants (GTS-angular r-LENS and GTS-angular c-LENS) to the state-of-the-art algorithms from the literature.

Section 4.4.1 presents the instances on which the two algorithmic variants are tested and the computational environment. In Section 4.4.2, we describe the parameter setting for GTS-angular. Section 4.4.3 presents the study on the construction heuristics. Finally, Section 4.4.4 presents the comparison of the results to the state-of-the-art heuristics.

4.4.1 Benchmark instances and computational environment

To test the performance of GTS-angular, we run computational experiments on the benchmark set proposed by Staněk et al., 2019 and available at `https://arxiv.org/abs/1803.03681`. This set is composed of ten symmetric instances for each of the 40 different sizes $n = 5, 10, \ldots, 200$. In total, 400 instances for the AngleTSP and 400 instances for the AngleDistanceTSP are considered. Each AngleDistanceTSP instance has been obtained taking the same vertex coordinates as the corresponding AngleTSP instance. The difference between the two instance sets lies in how the cost of the AngleTSP and of the AngleDistanceTSP instance is computed. To guarantee the comparability of the results, we adopted the same cost computation as done in the literature (see Staněk et al., 2019; Fischer et al., 2014; Aichholzer et al., 2017). In the AngleTSP, the cost for each triplet of vertices $i, j, k \in V$ corresponds to the turning angle α_{ijk} multiplied by 1000. In the AngleDistanceTSP, the cost for each triplet of vertices $i, j, k \in V$ is obtained by combining the turning angle and the Euclidean distance as follows:

$$c_{ijk} = 100\left(40 \cdot \alpha_{ijk} + \frac{d_{ij} + d_{jk}}{2}\right). \tag{4.2}$$

In both cases, the resulting costs are rounded to 12 decimal places.

GTS-angular was implemented in C++ and compiled using Clang version 12.0.0. The experiments were performed on an Intel(R) Xeon(R) computer with a CPU E5-2430 v2 processor, at 2.50GHz with 64 GB RAM under CentOS GNU/Linux 7. Because our algorithm contains randomized elements, we performed ten runs for each instance.

4.4.2 Parameter setting

During the development of the algorithm, parameter tuning experiments have been performed to adequately set the parameters for GTS-angular. To avoid overfitting the parameters to the instance type, we set all GTS-angular parameters to the same values for solving both the AngleTSP and the AngleDistanceTSP instances, except for the minimum lens thickness parameter γ_{min}. Because of the different computation of the costs for each triplet of vertices performed in the AngleTSP and AngleDistanceTSP instances (see Equation 4.2), a different value for the minimum lens thickness for each problem type, namely γ^{A}_{min} for the AngleTSP and γ^{AD}_{min} for the AngleDistanceTSP, should be properly tuned. Consequently, we considered the same lens thickness value of 0.6981 suggested by Staněk et al., 2019 for solving the AngleTSP instances and, consistently with the computation of the turning angle (see Section 4.4.1), we multiply that by 1000, that is, γ^{A}_{min} =698.1. For the AngleDistanceTSP instances, we computed γ^{AD}_{min} coherently with the cost computation in Equation (4.2) as γ^{AD}_{min} =100$(40 \cdot 0.6981 + \overline{d})$, where $\overline{d}$ is the average distance over all arcs in A. Table 4.1 summarizes all GTS-angular parameters.

Component	Notation	Parameter values
tabu tenure	$\tau = [\tau_{min}, \tau_{max}]$	$\tau = [10, 20]$
termination criterion	η	$\eta = 30\,000$
continuous diversification	σ	$\sigma = 2e^{-6}$
dynamic sparsification	$\gamma = [\gamma^{A}_{min},\ 2\gamma^{A}_{min}\ ]$	$\gamma\ = [698.1, 2 \cdot 698.1] = [698.1, 1\,396.2]$
	$\gamma = [\gamma^{AD}_{min},\ 2\gamma^{AD}_{min}\ ]$	$\gamma\ = [100(40 \cdot 0.6981 + \overline{d}), 200(40 \cdot 0.6981 + \overline{d})]$
		$= [100(2\,792.4 + \overline{d}), 200(2\,792.4 + \overline{d})]$
accepted arc list length limit	κ_a	$\kappa_a = 0.05$
sparsification intensity	κ_s	$\kappa_s = 0.05$
sparsification iterations	η^{κ}	$\eta^{\kappa} = 1\,000$

Table 4.1: GTS-angular parameter values.

4.4.3 Performance of GTS-angular with different construction heuristics

To assess the value of our construction heuristics, we compare the performance of GTS-angular r-LENS and GTS-angular c-LENS using the construction heuristics of Section 4.3.1 (i.e., CHH for AngleTSP instances and NIH for AngleDistanceTSP instances) to GTS-angular r-LENS and GTS-angular c-LENS when using a randomly generated starting solution. As a benchmark for our comparison, we consider the results of Staněk et al., 2019, which include:

- The solution (objective function value) obtained by each of their algorithmic variants for each instance.

- The optimal objective function value for each of the AngleTSP instances with size up to 75 vertices, and for each of the AngleDistanceTSP instances with size up to 100 vertices. The number of AngleTSP and AngleDistanceTSP instances for which an optimal solution is available is reported in Table 4.2. These optimal solutions are obtained by solving the integer linear models for the AngleTSP and AngleDistanceTSP via an off-the-shelf solver (for more details, see Staněk et al., 2019).

	Number of instances	
	Optimal solution available	Optimal solution not available
AngleTSP	150	250
AngleDistanceTSP	200	200
Total	350	450

Table 4.2: Number of AngleTSP and AngleDistanceTSP instances for which an optimal solution is available from Staněk et al., 2019.

The best-known solution (BKS) in the following comparisons corresponds to the optimal objective function value for the instances for which an optimal solution is available, and to the best found objective function value across all the algorithmic variants of Staněk et al., 2019 for those instances for which optimality is not proven.

Tables 4.3 and 4.4 report the results of the construction heuristics for the AngleTSP and the AngleDistanceTSP, respectively. For each method, we report the gap of the starting solution to the BKS averaged over all instances ($\overline{\Delta}^{start}_{BKS}(\%)$), the runtimes of the approach ($\overline{t}^{start}(s)$) averaged over all runs and instances, the gap of the best run to the BKS averaged over all instances ($\overline{\Delta}^{best}_{BKS}(\%)$), the gap to the BKS averaged over all runs and instances ($\overline{\Delta}^{avg}_{BKS}(\%)$), and the runtimes $\overline{t}(s)$ averaged over all runs and instances. In the tables, we highlight in bold the best result for each of the reported measures.

Comparing the quality of the starting solutions, we observe that both our construction heuristics return a clearly better starting solution quality than the quality of a random starting solution. As expected, randomly generating solutions is faster than our construction heuristics. However, the runtimes of our construction heuristics only amount to around 1.2% or less of the total GTS-angular runtime, and therefore the speed difference is rather irrelevant.

Focusing on the final objective values, for the AngleTSP instances, GTS-angular r-LENS and GTS-angular c-LENS return a notably better solution quality when using CHH compared to using a random starting solution. However, this happens at the expense of a higher runtime. For the AngleDistanceTSP instances, the superior quality of the starting solution returned by NIH is less relevant for the performance of GTS-angular r-LENS and GTS-angular c-LENS. This result could be caused by the quality of the NIH solution that is not improving as much as the CHH solution with respect to a random starting solution.

	Random starting solution					CHH				
	$\overline{\Delta}^{start}_{BKS}(\%)$	$\overline{t}^{start}(s)$	$\overline{\Delta}^{best}_{BKS}(\%)$	$\overline{\Delta}^{avg}_{BKS}(\%)$	$\overline{t}(s)$	$\overline{\Delta}^{start}_{BKS}(\%)$	$\overline{t}^{start}(s)$	$\overline{\Delta}^{best}_{BKS}(\%)$	$\overline{\Delta}^{avg}_{BKS}(\%)$	$\overline{t}(s)$
GTS-angular r-LENS	378.50	**0.02**	-0.35	1.34	**262.82**	**45.12**	0.55	**-0.36**	**1.28**	290.28
GTS-angular c-LENS	378.50	**0.02**	0.07	1.55	**330.37**	**45.12**	0.55	**-0.13**	**1.29**	350.65

Table 4.3: Comparison of GTS-angular r-LENS and GTS-angular c-LENS using the starting solution from CHH and a random starting solution for the AngleTSP instances.

	Random starting solution					NIH				
	$\overline{\Delta}^{start}_{BKS}(\%)$	$\overline{t}^{start}(s)$	$\overline{\Delta}^{best}_{BKS}(\%)$	$\overline{\Delta}^{avg}_{BKS}(\%)$	$\overline{t}(s)$	$\overline{\Delta}^{start}_{BKS}(\%)$	$\overline{t}^{start}(s)$	$\overline{\Delta}^{best}_{BKS}(\%)$	$\overline{\Delta}^{avg}_{BKS}(\%)$	$\overline{t}(s)$
GTS-angular r-LENS	352.43	**0.02**	-0.20	**0.11**	**186.18**	**89.55**	2.10	-0.20	0.12	198.09
GTS-angular c-LENS	352.43	**0.02**	-0.11	0.33	177.79	**89.55**	2.10	-0.11	**0.18**	**174.03**

Table 4.4: Comparison of GTS-angular r-LENS and GTS-angular c-LENS using the starting solution from NIH and a random starting solution for the AngleDistanceTSP instances.

Due to the positive impact of the starting solution obtained with CHH, we perform additional experiments in which we merge the convex hulls in the opposite order, i.e., from the outermost to the innermost one. We call this construction heuristic CHH^o, where the "o" stands for "opposite". The results reported in Table 4.5 show that also CHH^o returns a clearly better starting solution quality than the quality of a random starting solution. However, the comparison of CHH and CHH^o suggests that the quality of the starting solution returned by CHH is better than the one of CHH^o. Moreover, on average, even if the runtime of GTS-angular r-LENS and GTS-angular c-LENS is slightly lower when using the starting solution of CHH^o, a better solution quality is returned using the starting solution of CHH. Hence, in the remainder of the paper, we use the results of GTS-angular r-LENS and GTS-angular c-LENS with the starting solution of CHH.

	CHH					CHH^o				
	$\overline{\Delta}^{start}_{BKS}(\%)$	$\overline{t}^{start}(s)$	$\overline{\Delta}^{best}_{BKS}(\%)$	$\overline{\Delta}^{avg}_{BKS}(\%)$	$\overline{t}(s)$	$\overline{\Delta}^{start}_{BKS}(\%)$	$\overline{t}^{start}(s)$	$\overline{\Delta}^{best}_{BKS}(\%)$	$\overline{\Delta}^{avg}_{BKS}(\%)$	$\overline{t}(s)$
GTS-angular r-LENS	**45.12**	**0.55**	-0.36	**1.28**	290.28	47.05	0.68	**-0.38**	1.32	**275.67**
GTS-angular c-LENS	**45.12**	**0.55**	**-0.13**	**1.29**	350.65	47.05	0.68	-0.06	1.33	**347.76**

Table 4.5: Comparison of GTS-angular r-LENS and GTS-angular c-LENS using the starting solution from CHH that merges the convex hulls from the innermost to the outermost and CHH^o that merges the convex hulls from the outermost to the innermost.

4.4.4 Comparison to the literature

In this section, we compare the performance of GTS-angular r-LENS and GTS-angular c-LENS to the state-of-the-art heuristics from the literature. In addition to the reported measures of Staněk et al., 2019 described in Section 4.4.3, we now also consider the runtime of each of their algorithmic variants for each instance.

For each algorithmic variant of Staněk et al., 2019, we first determine whether it is dominated or not (see Tables 4.10 and 4.11 in Appendix A). For this analysis, we consider the results as reported by Staněk et al., 2019. Specifically, we compute, for each algorithmic variant the average gap (over all instances) between the obtained objective function value and the BKS and the average runtimes. The status of an algorithm is "not dominated" if there is no other algorithm that provides: (i) a better solution quality and faster runtimes simultaneously, (ii) the same solution quality but lower runtimes, (iii) the same runtimes but better solution quality. Otherwise, it is "dominated". For the comparison with GTS-angular, we consider only the non-dominated algorithmic variants that show an average gap to the BKS below 2%. This results in the selection of four algorithmic variants (namely, $LPP^R + M^{15}$, $LPP^R + M^{20}$, $LPC_1^R + M^{15}$, $LPC_1^R + M^{20}$) for the AngleTSP, and six algorithmic variants (namely, $CIF + M^{15}$, $CIF + M^{20}$, $NN_S^2 + M^{15}$, $NN_S^2 + M^{20}$, $LPC_1^R + M^{20}$, $LPC_2^R + M^{20}$) for the AngleDistanceTSP. The algorithmic variants with names starting with "LP" are methods based on the solution of the linear programming relaxation of the AngleTSP and AngleDistanceTSP models. They differ with respect to the rounding procedure applied to the variables representing the selection of the arcs in a tour. The algorithmic variants with names starting with "CIF" are methods based on the cheapest insertion heuristic proposed by Fischer et al., 2014. The algorithmic variants with names starting by "NN_S^2" are methods based on the nearest neighbor heuristic run from all possible starting vertices and coupled with the 2-opt improvement heuristic. The suffixes "M^{15}" and "M^{20}" refer to an improvement heuristic similar to a large neighborhood search, in which part of the tour is destroyed by removing all arcs adjacent to a set of vertices in a geographical neighborhood. The numbers 15 and 20 represent the expected number of vertices in such a set. Then, the tour is repaired by optimally reconnecting the paths and isolated points by solving a quadratic programming model.

To make runtimes comparable, we have rerun the original code of all non-dominated algorithms of Staněk et al., 2019 on our test computer described in Section 4.4.1. The algorithms of Staněk et al., 2019 are implemented in Python; GTS-angular is implemented in C++. Depending on the test setting, multiple sources have reported speedup factors of C++ compared to Python in the range from 10 to 100. To ensure a fair comparison of runtimes, we convert the runtimes of the algorithms of Staněk et al., 2019 using the most optimistic speedup factor of 100 (thus guaranteeing that we are definitely not giving any runtime advantage to our GTS-angular). Because all non-dominated algorithms of Staněk et al., 2019 make calls to the solver Gurobi, whose runtimes are independent of the programming language used for the main algorithm, we convert the runtimes as follows:

$$t_{conv} = t_{Gurobi} + \frac{t_{tot} - t_{Gurobi}}{100},$$

where t_{conv} denotes the converted runtime, t_{Gurobi} the runtime spent in the Gurobi (version 10.0.0) calls with the parameter OutputFlag set to zero, and t_{tot} is the total runtime of the respective algorithm.

Tables 4.6 and 4.7 report the described runtime measures for all non-dominated algorithms of Staněk et al., 2019 averaged over the AngleTSP and the AngleDistanceTSP instances, respectively. The total runtime t_{tot} and the converted runtime t_{conv} are reported in seconds, the runtime for Gurobi calls t_{Gurobi} as percentage of the total runtime. The tables show that for all but two of the algorithms, most of the runtime is spent in Gurobi. Consequently, for the large majority of algorithms, the runtime differences between C++ and Python are nearly irrelevant for the comparison. Nevertheless, we base the following comparisons on the converted runtimes t_{conv}.

Algorithm	$\widetilde{t}_{tot}(s)$	$\widetilde{t}_{Gurobi}(\%)$	$\bar{t}_{conv}(s)$
$LPP^R + M^{15}$	340.09	94.41	321.28
$LPP^R + M^{20}$	411.69	94.68	390.02
$LPC_1^R + M^{15}$	351.44	94.41	331.98
$LPC_1^R + M^{20}$	405.30	94.46	383.08

Table 4.6: AngleTSP: conversion of the runtimes of the non-dominated algorithms of Staněk et al., 2019.

Algorithm	$\widetilde{t}_{tot}(s)$	$\widetilde{t}_{Gurobi}(\%)$	$\bar{t}_{conv}(s)$
$CIF + M^{15}$	45.37	82.32	37.43
$CIF + M^{20}$	74.36	87.60	65.23
$NN_S^2 + M^{15}$	133.51	27.17	37.24
$NN_S^2 + M^{20}$	160.08	38.32	62.33
$LPC_1^R + M^{20}$	261.35	94.40	246.86
$LPC_2^R + M^{20}$	275.68	89.66	247.47

Table 4.7: AngleDistanceTSP: conversion of the runtimes of the non-dominated algorithms of Staněk et al., 2019.

Tables 4.8 and 4.9 summarize the comparison of GTS-angular r-LENS and GTS-angular c-LENS for the AngleTSP and AngleDistanceTSP, respectively, to the non-dominated algorithms of Staněk et al., 2019. For each solution method, we report the gap of the best run to the BKS averaged over all instances ($\overline{\Delta}^{best}_{BKS}(\%)$), the gap to the BKS averaged over all runs and instances ($\overline{\Delta}^{avg}_{BKS}(\%)$), and the runtimes $\bar{t}(s)$ (that correspond to $\bar{t}_{conv}(s)$ for the algorithms of Staněk et al., 2019) averaged over all runs and instances. In the tables, we highlight in bold the best result for each of these three measures. Tables 4.12–4.15 in Appendix B contain more detailed results grouped according to the instance size. In Figure 4.5, we represent the Pareto frontiers of the algorithms with respect to the average runtime $\bar{t}(s)$ and the average gap to the BKS $\overline{\Delta}^{avg}_{BKS}(\%)$ for the AngleTSP and AngleDistanceTSP, respectively. Every time the performance of the algorithms is discussed in terms of solution quality and runtimes, we use the average solution quality and average runtimes of GTS-angular for fair comparison. The solution of the best run is only used to carry out comparisons based exclusively on solution quality. We note that, by using the converted runtimes, some of the algorithms that are non-dominated according to the results reported by Staněk et al., 2019 are now dominated (see $LPP^R + M^{20}$ for the AngleTSP, and $CIF + M^{15}$ and $CIF + M^{20}$ for the AngleDistanceTSP). However, we keep them in our comparison, and we refrain from rerunning the code of Staněk et al., 2019 for all 66 algorithmic variants due to the immense computational effort.

On AngleTSP instances, GTS-angular r-LENS and $LPC_1^R + M^{20}$ dominate GTS-angular c-LENS. The position of GTS-angular r-LENS on the Pareto frontier (Figure 4.5(a)) shows that GTS-angular r-LENS achieves a good compromise between solution quality and runtime, and dominates three out of the four algorithms of Staněk et al., 2019 (namely, $LPP^R + M^{15}$, $LPC_1^R + M^{15}$, and $LPC_1^R + M^{20}$).

For the AngleDistanceTSP, both GTS-angular variants achieve a better solution quality than the algorithms of Staněk et al., 2019 in competitive runtimes. GTS-angular r-LENS returns a better solution quality than GTS-angular c-LENS, while the latter is faster. The position of GTS-angular r-LENS and GTS-angular c-LENS in Figure 4.5(b) shows that they lie not only on the Pareto frontier, but they also dominate two state-of-the-art algorithms (namely, $LPC_1^R + M^{20}$ and $LPC_2^R + M^{20}$).

Analyzing the results with regard to the average gap to the BKS obtained in the best run, i.e., $\overline{\Delta}_{BKS}^{best}(\%)$, and reported in Tables 4.8 and 4.9, our algorithmic variants achieve a better solution quality than the algorithms of Staněk et al., 2019 both for the AngleTSP and AngleDistanceTSP instances. Specifically, GTS-angular r-LENS finds new BKSs for 177 AngleTSP instances, i.e., 70.8% of the instances for which an optimal solution is not available (see Table 4.2), and for 183 AngleDistanceTSP instances, i.e., 91.5% of the instances for which an optimal solution is not available. Moreover, the heuristics of Staněk et al., 2019 returning the best solution quality (i.e., $LPC_1^R + M^{20}$ for AngleTSP instances, and $LPC_2^R + M^{20}$ for AngleDistanceTSP instances) reported, for instances with cardinality between 105 and 200, average gaps to the best lower bounds of 15.75% and 8.26% for the AngleTSP and AngleDistanceTSP, respectively. For the same instances, our average gaps to the best lower bounds are 13.89% for the AngleTSP instances and 7.30% for the AngleDistanceTSP instances.

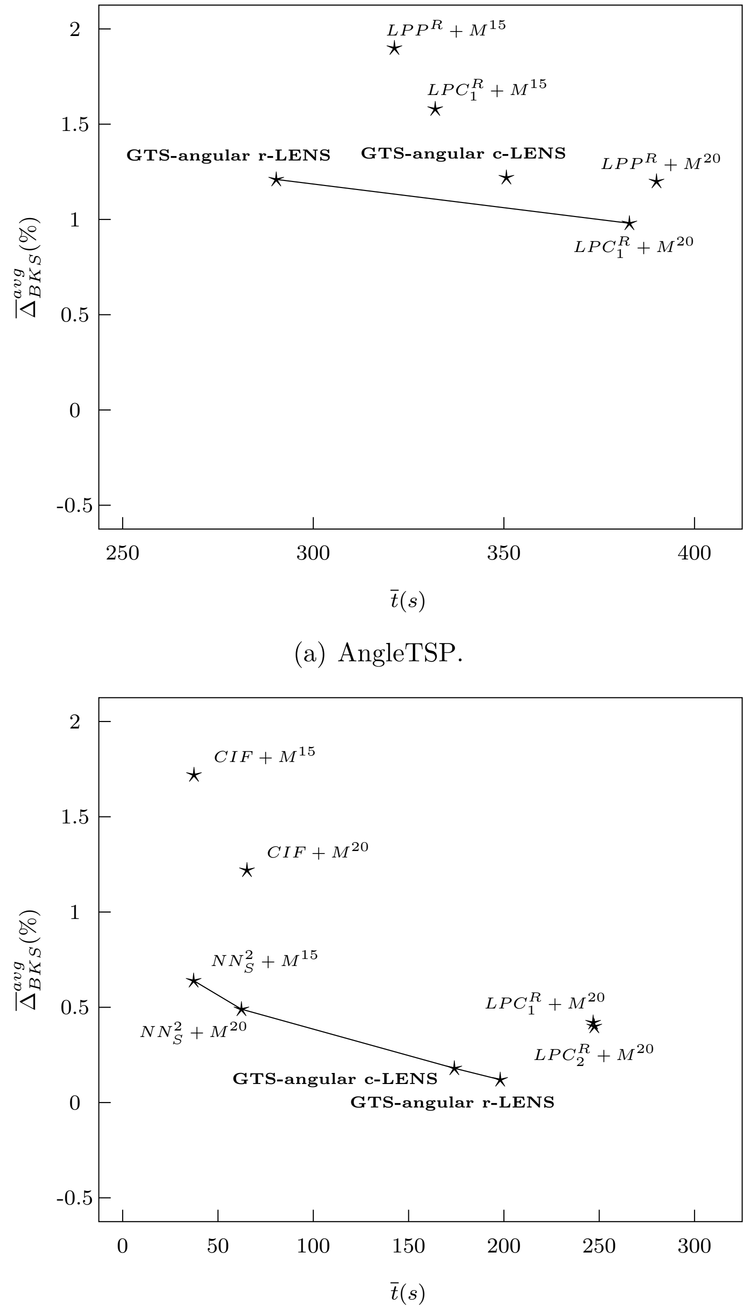
	$LPP^R + M^{15}$	$LPP^R + M^{20}$	$LPC_1^R + M^{15}$	$LPC_1^R + M^{20}$	GTS-angular r-LENS	GTS-angular c-LENS
$\overline{\Delta}_{BKS}^{best}(\%)$	1.90	1.20	1.58	0.98	**-0.36**	-0.13
$\overline{\Delta}_{BKS}^{avg}(\%)$	1.90	1.20	1.58	**0.98**	1.28	1.29
$\bar{t}(s)$	321.28	390.02	331.98	382.91	**290.28**	350.65

Table 4.8: AngleTSP results: comparison of the average results of the non-dominated algorithms of Staněk et al., 2019 with GTS-angular r-LENS and GTS-angular c-LENS.

To provide more details on our comparison, we analyze the following indicators computed by grouping instances by size: (1) the percentage of instances for which each of the algorithms is able to find the optimal solution (Section 4.4.4.1), and (2) the percentage of instances on which our GTS-angular variants provide better or

	$CIF + M^{15}$	$CIF + M^{20}$	$NN_S^2 + M^{15}$	$NN_S^2 + M^{20}$	$LPC_1^R + M^{20}$	$LPC_2^R + M^{20}$	GTS-angular r-LENS	GTS-angular c-LENS
$\overline{\Delta}_{BKS}^{best}(\%)$	1.72	1.22	0.64	0.49	0.42	0.40	**-0.20**	-0.11
$\overline{\Delta}_{BKS}^{avg}(\%)$	1.72	1.22	0.64	0.49	0.42	0.40	**0.12**	0.18
$\bar{t}(s)$	37.43	65.23	**37.24**	62.33	246.86	247.47	198.09	174.03

Table 4.9: AngleDistanceTSP results: comparison of the average results of the non-dominated algorithms of Staněk et al., 2019 with GTS-angular r-LENS and GTS-angular c-LENS.



(a) AngleTSP.

(b) AngleDistanceTSP.

Figure 4.5: Pareto frontiers of the compared algorithmic variants.

equal solution quality than the one provided by the best-performing heuristics of Staněk et al., 2019 (Section 4.4.4.2). For the sake of representation, the figures in the remainder of this section show aggregated data. Specifically, we first obtain groups of instances by dividing them according to their size aggregated in increasing size intervals of 20. For example, the instance group 5–25 contains all instances with size between 5 and 25 vertices, the instance group 30–50 contains all instances with size between 30 and 50 vertices, and so on. Then, we compute the percentage of instances within each group satisfying the evaluated indicator, and we report that value.

4.4.4.1 Percentage of optimal solutions found

According to this indicator, we compute the percentage of instances for which our GTS-angular variants and the heuristics proposed by Staněk et al., 2019 are able to find the optimal solution. To derive this indicator, we refer to the optimal objective function values computed by Staněk et al., 2019. We recall that these optimal values were provided only for the instances with sizes 5–75 for the AngleTSP, and instances with sizes 5–100 for the AngleDistanceTSP. Because their heuristics do not contain random components, the authors executed only one run for each heuristic. Hence, we compute this indicator for their algorithms by using the objective function value of that run. For our GTS-angular variants, we consider the objective function value of the best run. Figure 4.6 shows, for each heuristic, the percentage of AngleTSP and AngleDistanceTSP instances for which the optimal solution is found.

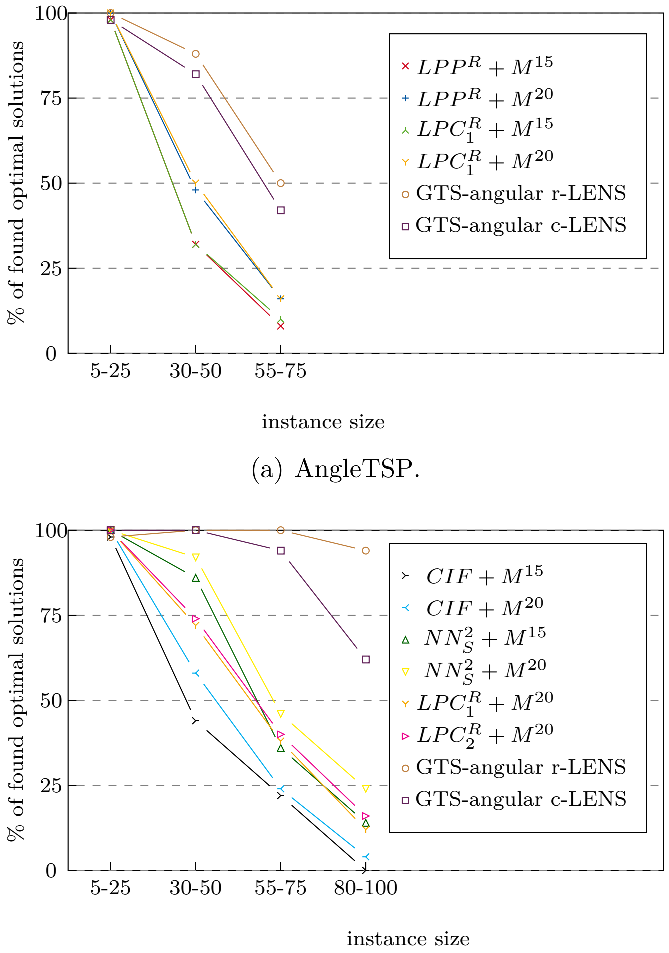
For the AngleTSP instances (Figure 4.6(a)), GTS-angular r-LENS and GTS-angular c-LENS are the heuristics returning the highest percentage of optimal solutions. For instances with size ranging between 30 and 75 vertices, our algorithmic variants always find at least 25% more optimal solutions compared with the variants of Staněk et al., 2019.

For the AngleDistanceTSP instances (Figure 4.6(b)), our GTS-angular variants find optimal solutions for most of the instances with up to 100 vertices. Conversely, the heuristics of Staněk et al., 2019 are not able to find optimal solutions for most of the instances with sizes starting from 55 vertices.

Comparing our two algorithmic variants, GTS-angular r-LENS is in general superior to GTS-angular c-LENS. An exception is represented by the case of AngleDistanceTSP instances with sizes ranging from 5 to 25. In this instance group, GTS-angular c-LENS performs slightly better.

4.4.4.2 Percentage of instances with better or equal solution quality

According to this indicator, we compute the percentage of instances for which our GTS-angular variants find a better or equal solution than the one provided by each of the non-dominated heuristics of Staněk et al., 2019. For each of our algorithmic variants, we count the number of instances for which the returned objective function value of the best run is lower or equal to the one provided by each of the non-dominated



(a) AngleTSP.

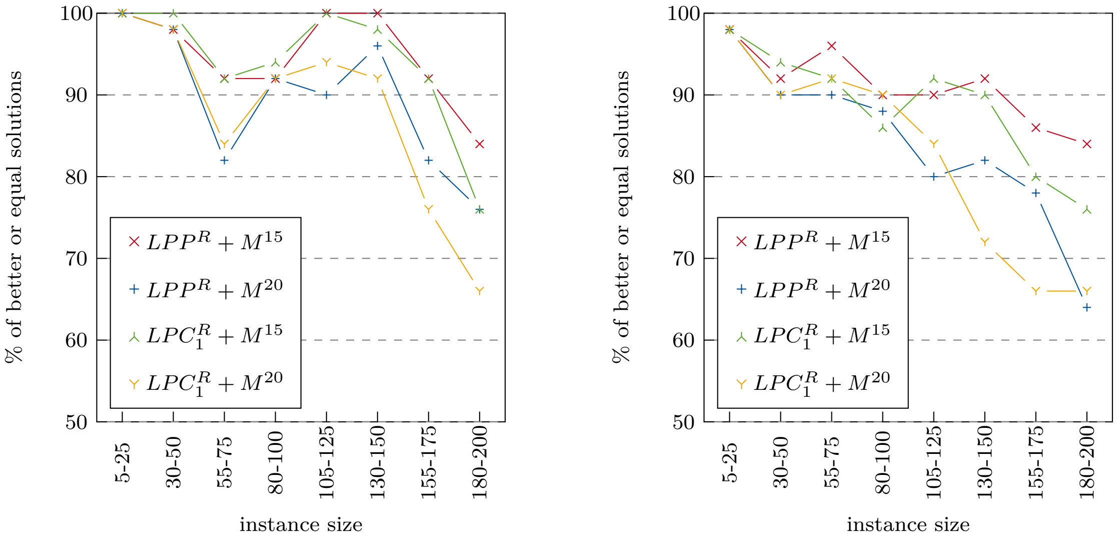
(b) AngleDistanceTSP.

Figure 4.6: Percentage of optimal solutions found by all compared heuristics.

heuristics of Staněk et al., 2019. Figures 4.7 and 4.8 show the percentage of these instances for GTS-angular r-LENS and GTS-angular c-LENS for AngleTSP and AngleDistanceTSP instances, respectively.

For the AngleTSP, both GTS-angular variants (Figures 4.7(a) and 4.7(b)) return better or equal solutions than the non-dominated variants of Staněk et al., 2019 for more than 60% of the instances, independent of the instance size. Compared to $LPC_1^R + M^{20}$ (the best-performing heuristic with respect to solution quality by Staněk et al., 2019), GTS-angular r-LENS finds better or equal solutions for more than 75% of the instances with up to 175 vertices and for more than 65% of the instances with 180 to 200 vertices.

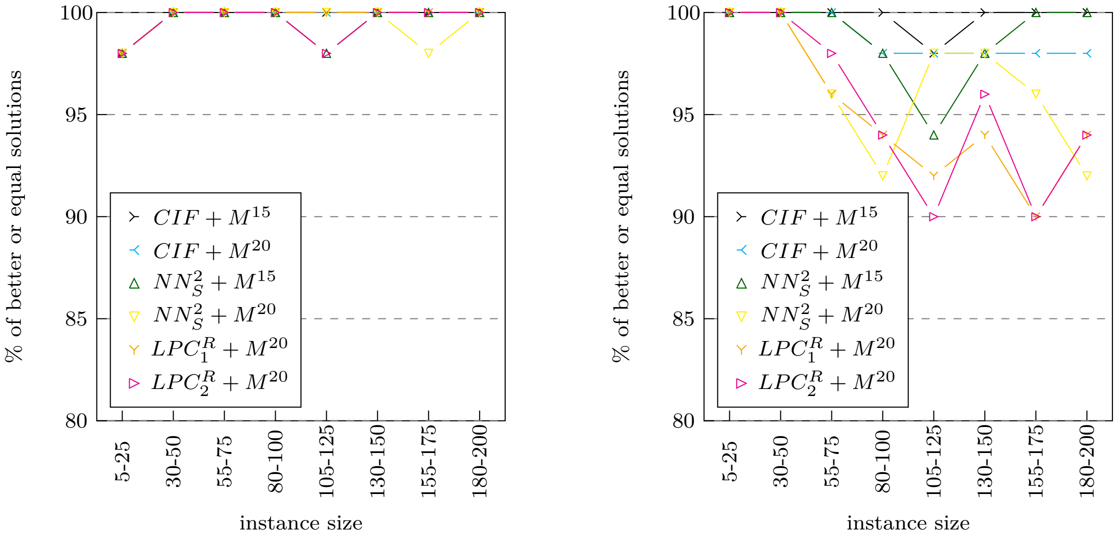
For the AngleDistanceTSP, the performance of our GTS-angular variants is convincing: for all but three instance size groups, GTS-angular r-LENS finds better or equal solutions for all instances. In those three instance size groups, GTS-angular r-LENS returns better or equal solutions for more than 97% of the instances (Figure 4.8(a)). Across all instance size groups, GTS-angular c-LENS finds better or equal solutions for at least 90% of the instances (Figure 4.8(b)). In addition, GTS-angular r-LENS always returns a better or equal solution than the one found by the algorithms of Staněk et al., 2019 for the biggest instances with sizes 180-200.



(a) GTS-angular with r-LENS.

(b) GTS-angular with c-LENS.

Figure 4.7: Percentage of AngleTSP instances for which GTS-angular finds a better or equal solution than the ones of the heuristics of Staněk et al., 2019.



(a) GTS-angular with r-LENS.

(b) GTS-angular with c-LENS.

Figure 4.8: Percentage of AngleDistanceTSP instances for which GTS-angular finds a better or equal solution than the ones of the heuristics of Staněk et al., 2019.

4.5 Conclusion

We propose a GTS framework that considers the geometric features of the AngleTSP and AngleDistanceTSP in the design of the starting solutions and the sparsification methods. The comparison of two sparsification methods proves that r-LENS returns solutions of better quality than c-LENS on AngleTSP and AngleDistanceTSP instances, but c-LENS is faster on AngleDistanceTSP instances. The computational results on AngleTSP instances show that the performance of GTS-angular lies on the Pareto frontier of heuristic AngleTSP and AngleDistanceTSP methods and it qualifies as a valuable alternative to the heuristics proposed by Staněk et al., 2019. For the AngleDistanceTSP instances, GTS-angular provides the best solution quality across the best-performing heuristics of Staněk et al., 2019 in competitive runtimes. Moreover, it dominates two of the heuristics proposed by Staněk et al., 2019. Finally, it finds new best-known solutions on around 81% of AngleTSP and AngleDistanceTSP instances for which an optimal solution is not available.

References

M. S. Abubakar, D. Ahmad, and F. B. Akande (2010). "A review of farm tractor overturning accidents and safety". In: *Pertanika Journal of Science and Technology* 18.2, pp. 377–385.

A. Aggarwal, D. Coppersmith, S. Khanna, R. Motwani, and B. Schieber (2000). "The angular-metric traveling salesman problem". In: *SIAM Journal on Computing* 29.3, pp. 697–711. DOI: 10.1137/S0097539796312721.

O. Aichholzer, A. Fischer, F. Fischer, J. F. Meier, U. Pferschy, A. Pilz, and R. Staněk (2017). "Minimization and maximization versions of the quadratic travelling salesman problem". In: *Optimization* 66.4, pp. 521–546. DOI: 10.1080/02331934.2016.1276905.

C. Becker, R. Cavagnini, S. Irnich, and M. Schneider (2021). *Rule-based design of a granular tabu search for the multi-depot vehicle routing problem.* Working Paper DPO-2021-01. Aachen, Germany: Deutsche Post Chair – Optimization of Distribution Networks, RWTH Aachen University.

J.-F. Cordeau, M. Gendreau, and G. Laporte (1997). "A tabu search heuristic for periodic and multi-depot vehicle routing problems". In: *Networks* 30.2, pp. 105–119. DOI: 10.1002/(SICI)1097-0037(199709)30:2%3C105::AID-NET5%3E3.0.CO;2-G.

T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein (2009). *Introduction to algorithms.* MIT press.

J. W. Escobar, R. Linfati, and P. Toth (2014). "A hybrid granular tabu search algorithm for the multi-depot vehicle routing problem". In: *Journal of the Operational Research Society* 65.1, pp. 37–48. DOI: 10.1057/jors.2013.102.

U. D. o. T. Federal Motor Carrier Safety Administration (2019). *Large truck and bus crash facts 2017.* URL: https://www.fmcsa.dot.gov/sites/fmcsa.dot.gov/

files/docs/safety/data-and-statistics/461861/ltcbf-2017-final-5-6-2019.pdf.

A. Fischer and C. Helmberg (2013). “The symmetric quadratic traveling salesman problem”. In: *Mathematical Programming* 142.1, pp. 205–254. DOI: 10.1007/s10107-012-0568-1.

A. Fischer, F. Fischer, G. Jäger, J. Keilwagen, P. Molitor, and I. Grosse (2014). “Exact algorithms and heuristics for the quadratic traveling salesman problem with an application in bioinformatics”. In: *Discrete Applied Mathematics* 166, pp. 97–114. DOI: 10.1016/j.dam.2013.09.011.

A. Fischer, F. Fischer, G. Jäger, J. Keilwagen, P. Molitor, and I. Grosse (2015). “Computational recognition of RNA splice sites by exact algorithms for the quadratic traveling salesman problem”. In: *Computation* 3.2, pp. 285–298. DOI: 10.3390/computation3020285.

B. E. Gillett and L. R. Miller (1974). “A heuristic algorithm for the vehicle-dispatch problem”. In: *Operations research* 22.2, pp. 340–349. DOI: 10.1287/opre.22.2.340.

F. Glover (1989). “Tabu search – Part 1”. In: *ORSA Journal on Computing* 1.3, pp. 190–206. DOI: 10.1287/ijoc.1.3.190.

D. Goeke (2019). “Granular tabu search for the pickup and delivery problem with time windows and electric vehicles”. In: *European Journal of Operational Research* 278.3, pp. 821–836. DOI: 10.1016/j.ejor.2019.05.010.

R. L. Graham (1972). “An efficient algorithm for determining the convex hull of a finite planar set”. In: *Information Processing Letters* 1, pp. 132–133.

D. Kirchler and R. Wolfler Calvo (2013). “A granular tabu search algorithm for the dial-a-ride problem”. In: *Transportation Research Part B* 56, pp. 120–135. DOI: 10.1016/j.trb.2013.07.014.

N. Labadie, R. Mansini, J. Melechovskỳ, and R. W. Calvo (2012). “The team orienteering problem with time windows: An LP-based granular variable neighborhood search”. In: *European Journal of Operational Research* 220.1, pp. 15–27. DOI: 10.1016/j.ejor.2012.01.030.

Z. Leng, Y. Wang, M. Xin, and M. A. Minor (2022). “The Effect of Sideslip on Jackknife Limits during Low Speed Trailer Operation”. In: *Robotics* 11.6, p. 133. DOI: 10.3390/robotics11060133.

A. J. McKnight and G. T. Bahouth (2009). “Analysis of large truck rollover crashes”. In: *Traffic injury prevention* 10.5, pp. 421–426. DOI: 10.1080/15389580903135291.

A. C. Medeiros and S. Urrutia (2010). “Discrete optimization methods to determine trajectories for Dubins’ vehicles”. In: *Electronic Notes in Discrete Mathematics* 36, pp. 17–24. DOI: 10.1016/j.endm.2010.05.003.

C. Prins, C. Prodhon, A. Ruiz, P. Soriano, and R. Wolfler Calvo (2007). “Solving the Capacitated Location-Routing Problem by a Cooperative Lagrangean Relaxation-

Granular Tabu Search Heuristic". In: *Transportation Science* 41.4, pp. 470–483. DOI: 10.1287/trsc.1060.0187.

K. Savla, E. Frazzoli, and F. Bullo (2008). "Traveling salesperson problems for the Dubins vehicle". In: *IEEE Transactions on Automatic Control* 53.6, pp. 1378–1391. DOI: 10.1109/TAC.2008.925814.

M. Schneider and M. Löffler (2019). "Large composite neighborhoods for the capacitated location-routing problem". In: *Transportation Science* 53.1, pp. 301–318. DOI: 10.1287/trsc.2017.0770.

M. Schneider, F. Schwahn, and D. Vigo (2017). "Designing granular solution methods for routing problems with time windows". In: *European Journal of Operational Research* 263.2, pp. 493–509. DOI: 10.1016/j.ejor.2017.04.059.

R. Staněk, P. Greistorfer, K. Ladner, and U. Pferschy (2019). "Geometric and LP-based heuristics for angular travelling salesman problems in the plane". In: *Computers & Operations Research* 108, pp. 97–111. DOI: 10.1016/j.cor.2019.01.016.

P. Toth and D. Vigo (2003). "The granular tabu search and its application to the vehicle-routing problem". In: *INFORMS Journal on Computing* 15.4, pp. 333–346. DOI: 10.1287/ijoc.15.4.333.24890.

4.6 Appendix

A Dominated and non-dominated algorithmic variants by Staněk et al., 2019

Tables 4.10 and 4.11 report the dominated and non-dominated algorithmic variants proposed by Staněk et al., 2019 for the AngleTSP and AngleDistanceTSP instances, respectively.

Algorithmic variant	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}(s)$	dominated/not dominated
CIF	56.5	3.49	dominated
$CIF + 2O$	22.75	3.85	dominated
$CIF + L$	48.89	43.70	dominated
$CIF + M^{15}$	7.67	66.19	dominated
$CIF + M^{20}$	5.33	104.65	dominated
$CIF + 3O$	11.74	88.61	dominated
NN_S	39.57	146.51	dominated
$NN_S + 2O$	35.92	146.64	dominated
$NN_S + L$	38.60	187.77	dominated
$NN_S + M^{15}$	12.30	253.81	dominated
$NN_S + M^{20}$	8.64	317.54	dominated
$NN_S + 3O$	26.93	215.02	dominated
NN_S^L	36.03	215.33	dominated
$NN_S^L + 2O$	19.49	215.62	dominated
$NN_S^L + L$	32.06	254.86	dominated
$NN_S^L + M^{15}$	3.97	278.38	dominated
$NN_S^L + M^{20}$	2.59	329.23	dominated
$NN_S^L + 3O$	9.00	296.41	dominated
NN_S^2	28.93	35.61	dominated
$NN_S^2 + 2O$	28.93	35.67	dominated
$NN_S^2 + L$	28.02	76.84	dominated
$NN_S^2 + M^{15}$	11.25	138.75	dominated
$NN_S^2 + M^{20}$	7.91	198.77	dominated
$NN_S^2 + 3O$	23.89	102.17	dominated
CH	41.07	0.81	dominated
$CH + 2O$	39.35	0.92	dominated
$CH + L$	40.28	42.20	dominated
$CH + M^{15}$	12.22	106.32	dominated
$CH + M^{20}$	8.98	175.15	dominated
$CH + 3O$	29.13	72.91	dominated
CH_C	35.87	1.22	dominated
$CH_C + 2O$	35.45	1.30	dominated
$CH_C + L$	35.48	42.63	dominated
$CH_C + M^{15}$	12.72	109.70	dominated
$CH_C + M^{20}$	8.95	176.71	dominated
$CH_C + 3O$	29.97	59.39	dominated
CH_C^L	32.90	0.73	not dominated
$CH_C^L + 2O$	21.76	0.98	not dominated
$CH_C^L + L$	29.64	40.34	dominated
$CH_C^L + M^{15}$	3.92	66.17	not dominated
$CH_C^L + M^{20}$	2.68	120.13	not dominated
$CH_C^L + 3O$	9.01	86.96	dominated
LPP	25.69	107.78	dominated
$LPP + 2O$	13.25	108.05	dominated
$LPP + L$	22.15	148.14	dominated
$LPP + M^{15}$	4.18	162.68	dominated
$LPP + M^{20}$	2.65	213.59	not dominated
$LPP + 3O$	6.43	178.42	dominated
LPP^R	10.39	192.41	dominated
$LPP^R + 2O$	7.20	192.57	dominated
$LPP^R + L$	8.66	233.06	dominated
$LPP^R + M^{15}$	1.90	245.95	not dominated
$LPP^R + M^{20}$	1.20	290.26	not dominated
$LPP^R + 3O$	4.29	252.79	dominated
LPC_1^R	8.56	204.32	dominated
$LPC_1^R + 2O$	6.44	204.46	dominated
$LPC_1^R + L$	7.17	244.93	dominated
$LPC_1^R + M^{15}$	1.58	256.32	not dominated
$LPC_1^R + M^{20}$	0.98	299.53	not dominated
$LPC_1^R + 3O$	3.71	262.42	dominated
LPC_2^R	8.38	212.48	dominated
$LPC_2^R + 2O$	6.40	212.62	dominated
$LPC_2^R + L$	6.99	253.05	dominated
$LPC_2^R + M^{15}$	1.62	264.87	dominated
$LPC_2^R + M^{20}$	1.01	306.34	dominated
$LPC_2^R + 3O$	3.70	271.11	dominated

Table 4.10: Dominated and non-dominated algorithmic variants proposed by Staněk et al., 2019 for the AngleTSP instances.

Algorithmic variant	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}(s)$	dominated/not dominated
CIF	12.62	3.49	dominated
$CIF + 2O$	5.61	3.79	not dominated
$CIF + L$	12.13	40.48	dominated
$CIF + M^{15}$	1.72	25.13	not dominated
$CIF + M^{20}$	1.22	42.00	not dominated
$CIF + 3O$	3.29	67.35	dominated
NN_S	13.28	147.56	dominated
$NN_S + 2O$	5.66	147.89	dominated
$NN_S + L$	12.07	187.50	dominated
$NN_S + M^{15}$	1.45	175.55	dominated
$NN_S + M^{20}$	1.03	195.07	dominated
$NN_S + 3O$	3.29	213.71	dominated
NN_S^L	12.46	241.49	dominated
$NN_S^L + 2O$	5.21	241.83	dominated
$NN_S^L + L$	11.19	281.43	dominated
$NN_S^L + M^{15}$	1.36	268.74	dominated
$NN_S^L + M^{20}$	0.94	288.29	dominated
$NN_S^L + 3O$	3.03	307.01	dominated
NN_S^2	2.34	69.36	dominated
$NN_S^2 + 2O$	2.34	69.42	dominated
$NN_S^2 + L$	2.13	107.61	dominated
$NN_S^2 + M^{15}$	0.64	90.97	not dominated
$NN_S^2 + M^{20}$	0.49	106.27	not dominated
$NN_S^2 + 3O$	1.66	120.77	dominated
CH	52.32	0.74	dominated
$CH + 2O$	8.41	1.30	dominated
$CH + L$	42.62	40.84	dominated
$CH + M^{15}$	2.29	39.76	dominated
$CH + M^{20}$	1.50	65.52	dominated
$CH + 3O$	4.22	74.77	dominated
CH_C	47.85	0.90	dominated
$CH_C + 2O$	7.99	1.44	dominated
$CH_C + L$	39.24	40.89	dominated
$CH_C + M^{15}$	2.26	38.58	dominated
$CH_C + M^{20}$	1.51	63.96	dominated
$CH_C + 3O$	4.26	74.54	dominated
CH_C^L	17.86	0.56	not dominated
$CH_C^L + 2O$	7.89	0.95	not dominated
$CH_C^L + L$	17.21	33.70	dominated
$CH_C^L + M^{15}$	1.74	30.50	dominated
$CH_C^L + M^{20}$	1.23	51.05	dominated
$CH_C^L + 3O$	4.34	68.42	dominated
LPP	5.89	79.29	dominated
$LPP + 2O$	2.65	79.52	dominated
$LPP + L$	5.30	117.33	dominated
$LPP + M^{15}$	0.76	99.76	dominated
$LPP + M^{20}$	0.59	115.46	dominated
$LPP + 3O$	1.57	132.85	dominated
LPP^R	6.45	145.14	dominated
$LPP^R + 2O$	2.45	145.38	dominated
$LPP^R + L$	5.42	184.40	dominated
$LPP^R + M^{15}$	0.70	166.58	dominated
$LPP^R + M^{20}$	0.46	182.29	dominated
$LPP^R + 3O$	1.37	197.83	dominated
LPC_1^R	5.62	144.70	dominated
$LPC_1^R + 2O$	2.37	144.94	dominated
$LPC_1^R + L$	4.78	183.76	dominated
$LPC_1^R + M^{15}$	0.64	165.91	dominated
$LPC_1^R + M^{20}$	0.42	180.77	not dominated
$LPC_1^R + 3O$	1.33	197.69	dominated
LPC_2^R	5.25	154.52	dominated
$LPC_2^R + 2O$	2.27	154.75	dominated
$LPC_2^R + L$	4.49	193.57	dominated
$LPC_2^R + M^{15}$	0.61	175.70	dominated
$LPC_2^R + M^{20}$	0.40	190.72	not dominated
$LPC_2^R + 3O$	1.27	206.42	dominated

Table 4.11: Dominated and non-dominated algorithmic variants proposed by Staněk et al., 2019 for the AngleDistanceTSP instances.

B GTS-angular detailed results

Tables 4.12 and 4.13 report the comparison of the results grouped by instance size of the non dominated algorithms of Staněk et al., 2019 with GTS-angular r-LENS for the AngleTSP and AngleDistanceTSP instances, respectively. Tables 4.14 and 4.15 report the comparison of the results grouped by instance size of the non dominated algorithms of Staněk et al., 2019 with GTS-angular c-LENS for the AngleTSP and AngleDistanceTSP instances, respectively.

	$LPP^R + M^{15}$		$LPP^R + M^{20}$		$LPC_1^R + M^{15}$		$LPC_1^R + M^{20}$		GTS-angular r-LENS		
Size	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}^{best}(\%)$	$\overline{\Delta}_{BKS}^{avg}(\%)$	$\bar{t}^{avg}(s)$
5	0.00	0.18	0.00	0.19	0.00	0.18	0.00	0.21	0.00	0.00	0.03
10	0.00	0.41	0.00	0.36	0.00	0.38	0.00	0.39	0.00	0.00	0.10
15	0.00	1.02	0.00	1.05	0.00	1.07	0.00	1.03	0.00	0.03	0.66
20	0.00	28.08	0.00	9.35	0.00	27.76	0.00	9.46	0.00	0.11	1.52
25	0.03	13.92	0.00	22.30	0.03	13.74	0.00	22.66	0.00	0.28	2.82
30	2.46	16.54	0.29	44.06	2.44	15.91	0.29	44.13	0.00	0.09	4.12
35	1.75	12.62	1.17	29.36	1.75	16.97	0.77	28.56	0.00	0.29	5.48
40	0.52	81.22	0.39	92.44	0.55	20.16	0.39	89.55	0.11	0.60	8.15
45	1.33	43.02	0.79	44.85	1.57	129.75	1.13	43.78	0.24	0.51	10.78
50	2.81	40.24	3.03	33.52	2.94	40.78	3.07	38.28	0.12	0.48	13.30
55	1.15	47.17	1.40	50.05	1.56	52.78	1.42	45.82	0.18	0.86	16.70
60	2.27	64.22	1.10	143.14	2.14	62.91	1.12	110.51	0.51	1.30	23.06
65	2.00	61.86	1.37	97.47	2.03	61.12	1.21	88.50	0.07	0.80	23.12
70	3.54	59.69	2.49	226.42	3.26	61.66	2.15	106.22	0.30	1.20	30.94
75	3.68	78.52	3.16	122.97	2.91	77.17	2.22	119.98	0.52	1.62	39.13
80	1.99	71.62	1.52	221.62	2.06	71.35	1.45	122.24	-0.42	1.03	48.84
85	1.89	129.11	0.94	194.28	1.77	118.94	1.74	158.34	-0.47	0.64	55.61
90	1.84	175.57	1.13	145.07	2.04	123.88	1.47	148.39	-0.44	0.59	68.53
95	2.65	152.03	1.40	178.23	2.30	262.39	0.90	193.36	-0.28	1.29	87.80
100	1.78	164.27	1.85	206.52	0.96	159.14	1.24	208.47	-1.25	0.34	98.16
105	1.49	265.97	0.73	209.27	1.29	437.74	0.98	192.58	-1.32	-0.11	116.38
110	1.23	192.95	0.78	330.04	1.08	204.76	1.37	294.92	-1.14	0.48	142.25
115	3.12	212.34	2.05	348.61	2.13	207.68	1.06	336.06	-1.42	1.32	168.12
120	1.69	265.23	2.00	377.40	2.51	259.66	0.86	353.04	-0.42	1.33	174.16
125	1.65	289.47	0.82	393.66	1.57	257.50	1.19	339.65	-1.20	1.06	219.04
130	2.04	310.34	0.78	505.92	1.08	321.42	0.71	467.38	-1.27	1.48	242.56
135	2.61	412.56	0.74	421.58	1.31	384.64	1.45	468.69	-1.14	0.75	276.24
140	2.35	431.83	1.33	505.78	1.81	435.93	0.90	662.40	-1.02	1.09	326.16
145	2.74	428.14	1.69	656.21	1.90	512.38	0.77	547.82	-0.63	2.63	367.78
150	3.13	513.04	2.03	606.98	1.77	483.46	0.26	559.07	-0.91	1.60	447.62
155	1.86	561.82	1.27	618.75	1.71	521.27	1.10	602.04	-0.22	1.93	527.07
160	1.62	563.99	0.96	634.72	2.01	558.81	1.11	637.95	-0.69	1.78	507.06
165	2.12	644.94	1.36	751.54	2.02	790.36	1.10	769.64	-0.36	2.74	613.91
170	2.35	665.16	1.46	738.15	0.69	759.51	0.50	853.14	-0.85	1.98	728.70
175	1.67	861.34	1.28	895.28	1.50	846.18	0.77	913.91	-0.80	2.54	790.32
180	2.16	881.03	1.00	896.17	1.14	809.29	0.84	901.22	0.40	3.61	818.04
185	2.56	923.65	1.39	1095.27	2.10	910.28	0.96	1117.72	0.47	3.91	998.30
190	3.10	960.09	1.44	1136.95	1.58	927.81	1.24	1054.27	0.02	3.43	1121.57
195	2.73	1067.69	1.92	1201.70	2.15	1167.62	0.97	1313.30	-0.76	2.48	1149.00
200	1.96	1158.22	1.04	1413.78	1.60	1164.71	0.51	1351.58	-0.40	3.27	1338.27
Avg per inst	1.90	321.28	1.20	390.02	1.58	331.98	0.98	382.91	-0.36	1.28	290.28

Table 4.12: AngleTSP results: comparison of the results averaged by instance size of the non-dominated algorithms of Staněk et al., 2019 with GTS-angular r-LENS.

	$CIF + M^{15}$		$CIF + M^{20}$		$NN_S^2 + M^{15}$		$NN_S^2 + M^{20}$		$LPC_1^R + M^{20}$		$LPC_2^R + M^{20}$		GTS-angular r-LENS		
Size	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}^{best}_{BKS}(\%)$	$\overline{\Delta}^{avg}_{BKS}(\%)$	$\bar{t}^{avg}(s)$
5	0.00	0.10	0.00	0.11	0.00	0.11	0.00	0.12	0.00	0.22	0.00	0.22	0.00	0.00	0.06
10	0.00	0.19	0.00	0.23	0.00	0.19	0.00	0.20	0.00	0.42	0.00	0.43	0.00	0.00	0.26
15	0.00	0.32	0.00	0.32	0.00	0.31	0.00	0.33	0.00	0.62	0.00	0.66	0.00	1.82	0.63
20	0.00	3.09	0.00	0.80	0.00	3.15	0.00	0.80	0.00	1.41	0.00	1.46	0.06	1.49	1.19
25	0.18	4.12	0.00	5.93	0.00	4.10	0.00	6.26	0.00	6.96	0.00	6.95	0.00	0.70	2.13
30	0.47	3.74	0.31	7.63	0.00	3.37	0.00	7.33	0.25	10.25	0.25	10.17	0.00	0.45	3.27
35	0.89	4.60	0.37	8.56	0.05	4.25	0.00	8.27	0.05	10.32	0.05	10.40	0.00	0.24	4.75
40	1.05	4.57	0.21	8.67	0.06	4.42	0.06	9.45	0.00	13.06	0.00	13.20	0.00	0.18	6.48
45	0.60	11.53	0.32	10.74	0.00	10.28	0.00	9.08	0.12	15.59	0.10	16.02	0.00	0.15	8.70
50	1.26	13.96	0.84	17.23	0.17	11.98	0.05	12.25	0.29	19.99	0.29	20.56	0.00	0.10	11.09
55	1.19	13.47	1.47	13.03	0.39	12.63	0.37	10.43	0.50	19.21	0.44	19.22	0.00	0.15	14.38
60	0.65	15.17	0.34	29.41	0.52	12.00	0.59	26.30	0.11	37.45	0.11	37.26	0.00	0.12	17.70
65	1.35	20.34	0.89	34.94	0.25	14.74	0.25	30.07	0.83	50.16	0.53	49.63	0.00	0.05	21.21
70	1.35	16.45	1.83	37.29	0.20	15.28	0.25	30.07	0.63	58.53	0.56	57.73	0.00	0.09	26.62
75	1.82	16.94	1.07	47.70	0.65	17.77	0.42	45.37	0.45	63.94	0.55	63.63	0.00	0.17	34.97
80	2.36	19.24	1.49	39.51	0.56	17.04	0.39	32.27	0.49	68.74	0.47	68.37	0.00	0.20	40.57
85	2.15	24.19	1.77	46.49	0.52	27.12	0.49	37.00	0.38	82.23	0.36	86.37	0.01	0.15	45.82
90	1.83	40.65	1.48	30.99	0.79	27.89	0.68	37.98	1.33	90.58	1.36	90.05	0.00	0.13	53.41
95	1.82	43.04	1.72	40.49	0.61	42.80	0.47	37.94	1.00	99.52	0.88	99.58	0.00	0.13	65.19
100	1.35	29.58	1.48	50.68	0.64	27.56	0.48	39.32	0.69	106.04	0.55	105.29	0.00	0.20	80.21
105	2.21	37.59	1.89	37.62	0.47	33.83	0.27	52.59	0.52	121.08	0.52	120.20	-0.14	0.02	89.99
110	2.36	33.63	2.10	71.36	0.56	34.87	0.46	71.78	0.75	170.66	0.57	175.73	-0.11	0.03	109.35
115	2.59	33.04	1.65	69.19	1.05	37.74	1.09	65.04	0.56	181.22	0.59	178.05	-0.32	-0.13	125.96
120	1.61	37.97	1.41	105.53	0.80	39.19	0.65	64.27	0.44	208.95	0.43	212.20	-0.20	-0.03	134.88
125	1.51	36.63	0.99	85.85	1.01	35.57	0.88	73.75	0.62	217.11	0.64	223.51	-0.42	-0.15	159.93
130	2.44	41.34	1.61	101.86	1.06	53.08	0.69	105.04	0.65	280.79	0.57	287.00	-0.40	-0.19	165.13
135	1.77	55.95	0.82	93.19	0.79	41.46	0.40	83.95	0.50	297.16	0.51	295.41	-0.36	-0.10	208.88
140	2.90	69.66	1.41	109.08	0.81	52.83	0.39	107.32	0.65	315.16	0.56	320.38	-0.24	0.05	242.54
145	2.55	53.45	1.86	92.89	1.27	51.01	1.00	100.71	0.24	338.56	0.25	333.11	-0.28	0.12	270.61
150	2.26	69.40	1.25	118.01	0.89	63.34	0.68	90.98	0.27	385.14	0.30	386.91	-0.61	-0.24	287.81
155	2.92	64.93	1.86	108.53	0.75	62.22	0.69	91.11	0.25	418.69	0.31	416.14	-0.44	-0.10	348.43
160	2.27	89.10	1.83	117.95	1.03	111.02	1.00	103.91	0.74	500.20	0.60	501.84	-0.39	-0.05	374.42
165	2.78	68.97	1.86	104.30	1.58	66.02	0.97	97.86	0.31	467.59	0.29	472.23	-0.41	0.03	427.46
170	2.13	69.20	1.97	94.21	1.16	80.54	0.71	101.35	0.25	521.77	0.25	522.79	-0.43	-0.03	456.83
175	2.29	63.57	1.78	107.72	1.30	67.86	0.95	110.81	0.29	525.27	0.23	526.13	-0.50	-0.01	514.18
180	2.53	77.16	1.62	105.10	1.40	74.96	1.15	121.08	0.50	670.68	0.52	667.02	-0.65	-0.27	566.67
185	3.10	77.88	2.12	143.94	0.92	83.62	0.64	148.36	0.50	792.41	0.50	798.71	-0.65	-0.14	604.57
190	2.49	75.87	1.23	160.65	1.16	75.50	0.80	167.56	0.27	889.03	0.27	884.59	-0.61	-0.18	736.33
195	2.97	71.34	2.19	165.60	1.06	83.88	1.10	171.9	0.77	859.39	0.78	855.71	-0.53	-0.11	791.15
200	2.79	85.16	1.70	185.85	1.08	84.15	0.62	182.97	0.78	958.31	0.78	963.86	-0.51	-0.11	869.75
Avg per inst	1.72	37.43	1.22	65.23	0.64	37.24	0.49	62.33	0.42	246.86	0.40	247.47	-0.20	0.12	198.09

Table 4.13: AngleDistanceTSP results: comparison of the results averaged by instance size of the non-dominated algorithms of Staněk et al., 2019 with GTS-angular r-LENS.

	$LPP^R + M^{15}$		$LPP^R + M^{20}$		$LPC_1^R + M^{15}$		$LPC_1^R + M^{20}$		GTS-angular c-LENS		
Size	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}^{best}(\%)$	$\overline{\Delta}_{BKS}^{avg}(\%)$	$\bar{t}^{avg}(s)$
5	0.00	0.18	0.00	0.19	0.00	0.18	0.00	0.21	0.00	0.00	0.07
10	0.00	0.41	0.00	0.36	0.00	0.38	0.00	0.39	0.00	0.00	0.16
15	0.00	1.02	0.00	1.05	0.00	1.07	0.00	1.03	0.00	0.00	0.78
20	0.00	28.08	0.00	9.35	0.00	27.76	0.00	9.46	0.00	0.04	1.79
25	0.03	13.92	0.00	22.30	0.03	13.74	0.00	22.66	0.06	0.29	3.19
30	2.46	16.54	0.29	44.06	2.44	15.91	0.29	44.13	0.00	0.13	4.42
35	1.75	12.62	1.17	29.36	1.75	16.97	0.77	28.56	0.13	0.30	6.44
40	0.52	81.22	0.39	92.44	0.55	20.16	0.39	89.55	0.08	0.49	9.12
45	1.33	43.02	0.79	44.85	1.57	129.75	1.13	43.78	0.10	0.45	12.53
50	2.81	40.24	3.03	33.52	2.94	40.78	3.07	38.28	0.06	0.46	15.81
55	1.15	47.17	1.40	50.05	1.56	52.78	1.42	45.82	0.16	0.85	20.67
60	2.27	64.22	1.10	143.14	2.14	62.91	1.12	110.51	0.40	1.27	25.56
65	2.00	61.86	1.37	97.47	2.03	61.12	1.21	88.50	0.12	0.93	28.73
70	3.54	59.69	2.49	226.42	3.26	61.66	2.15	106.22	0.25	0.86	37.36
75	3.68	78.52	3.16	122.97	2.91	77.17	2.22	119.98	0.59	1.51	48.58
80	1.99	71.62	1.52	221.62	2.06	71.35	1.45	122.24	-0.42	1.02	58.64
85	1.89	129.11	0.94	194.28	1.77	118.94	1.74	158.34	-0.17	0.62	66.23
90	1.84	175.57	1.13	145.07	2.04	123.88	1.47	148.39	-0.46	0.79	84.13
95	2.65	152.03	1.40	178.23	2.30	262.39	0.90	193.36	0.28	1.24	111.32
100	1.78	164.27	1.85	206.52	0.96	159.14	1.24	208.47	-1.11	0.06	120.61
105	1.49	265.97	0.73	209.27	1.29	437.74	0.98	192.58	-1.36	-0.16	155.82
110	1.23	192.95	0.78	330.04	1.08	204.76	1.37	294.92	-0.88	0.33	171.55
115	3.12	212.34	2.05	348.61	2.13	207.68	1.06	336.06	-0.58	1.00	216.59
120	1.69	265.23	2.00	377.40	2.51	259.66	0.86	353.04	-0.22	1.41	211.29
125	1.65	289.47	0.82	393.66	1.57	257.50	1.19	339.65	-0.77	1.08	244.25
130	2.04	310.34	0.78	505.92	1.08	321.42	0.71	467.38	-0.41	1.83	296.55
135	2.61	412.56	0.74	421.58	1.31	384.64	1.45	468.69	-0.88	0.95	305.28
140	2.35	431.83	1.33	505.78	1.81	435.93	0.90	662.40	-0.20	1.92	360.58
145	2.74	428.14	1.69	656.21	1.90	512.38	0.77	547.82	0.58	2.39	441.66
150	3.13	513.04	2.03	606.98	1.77	483.46	0.26	559.07	-0.73	1.32	531.41
155	1.86	561.82	1.27	618.75	1.71	521.27	1.10	602.04	-0.23	1.45	640.09
160	1.62	563.99	0.96	634.72	2.01	558.81	1.11	637.95	-0.72	1.68	654.21
165	2.12	644.94	1.36	751.54	2.02	790.36	1.10	769.64	-0.58	1.94	743.64
170	2.35	665.16	1.46	738.15	0.69	759.51	0.50	853.14	-0.07	2.94	775.97
175	1.67	861.34	1.28	895.28	1.50	846.18	0.77	913.91	-0.14	2.57	935.95
180	2.16	881.03	1.00	896.17	1.14	809.29	0.84	901.22	-0.07	3.07	1085.87
185	2.56	923.65	1.39	1095.27	2.10	910.28	0.96	1117.72	1.10	4.22	1234.51
190	3.10	960.09	1.44	1136.95	1.58	927.81	1.24	1054.27	0.75	3.82	1372.88
195	2.73	1067.69	1.92	1201.70	2.15	1167.62	0.97	1313.30	0.30	3.43	1278.44
200	1.96	1158.22	1.04	1413.78	1.60	1164.71	0.51	1351.58	-0.09	3.07	1713.43
Avg per inst	1.90	321.28	1.20	390.02	1.58	331.98	0.98	382.91	-0.13	1.29	350.65

Table 4.14: AngleTSP results: comparison of the the results averaged by instance size of the non-dominated algorithms of Staněk et al., 2019 with GTS-angular c-LENS.

	$CIF + M^{15}$		$CIF + M^{20}$		$NN_S^2 + M^{15}$		$NN_S^2 + M^{20}$		$LPC_1^R + M^{20}$		$LPC_2^R + M^{20}$		GTS-angular c-LENS		
Size	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}_{BKS}(\%)$	$\bar{t}_{conv}(s)$	$\overline{\Delta}^{best}_{BKS}(\%)$	$\overline{\Delta}^{avg}_{BKS}(\%)$	$\bar{t}^{avg}(s)$
5	0.00	0.10	0.00	0.11	0.00	0.11	0.00	0.12	0.00	0.22	0.00	0.22	0.00	0.00	0.08
10	0.00	0.19	0.00	0.23	0.00	0.19	0.00	0.20	0.00	0.42	0.00	0.43	0.00	0.00	0.32
15	0.00	0.32	0.00	0.32	0.00	0.31	0.00	0.33	0.00	0.62	0.00	0.66	0.00	0.61	0.71
20	0.00	3.09	0.00	0.80	0.00	3.15	0.00	0.80	0.00	1.41	0.00	1.46	0.00	0.58	1.39
25	0.18	4.12	0.00	5.93	0.00	4.10	0.00	6.26	0.00	6.96	0.00	6.95	0.00	0.53	2.35
30	0.47	3.74	0.31	7.63	0.00	3.37	0.00	7.33	0.25	10.25	0.25	10.17	0.00	0.12	3.53
35	0.89	4.60	0.37	8.56	0.05	4.25	0.00	8.27	0.05	10.32	0.05	10.40	0.00	0.15	5.15
40	1.05	4.57	0.21	8.67	0.06	4.42	0.06	9.45	0.00	13.06	0.00	13.20	0.00	0.25	6.47
45	0.60	11.53	0.32	10.74	0.00	10.28	0.00	9.08	0.12	15.59	0.10	16.02	0.00	0.23	8.84
50	1.26	13.96	0.84	17.23	0.17	11.98	0.05	12.25	0.29	19.99	0.29	20.56	0.00	0.21	11.01
55	1.19	13.47	1.47	13.03	0.39	12.63	0.37	10.43	0.50	19.21	0.44	19.22	0.00	0.20	13.93
60	0.65	15.17	0.34	29.41	0.52	12.00	0.59	26.30	0.11	37.45	0.11	37.26	0.00	0.14	17.70
65	1.35	20.34	0.89	34.94	0.25	14.74	0.25	30.07	0.83	50.16	0.53	49.63	0.00	0.07	21.66
70	1.35	16.45	1.83	37.29	0.20	15.28	0.25	30.07	0.63	58.53	0.56	57.73	0.01	0.12	27.54
75	1.82	16.94	1.07	47.70	0.65	17.77	0.42	45.37	0.45	63.94	0.55	63.63	0.02	0.26	31.47
80	2.36	19.24	1.49	39.51	0.56	17.04	0.39	32.27	0.49	68.74	0.47	68.37	0.04	0.15	41.29
85	2.15	24.19	1.77	46.49	0.52	27.12	0.49	37.00	0.38	82.23	0.36	86.37	0.03	0.17	45.87
90	1.83	40.65	1.48	30.99	0.79	27.89	0.68	37.98	1.33	90.58	1.36	90.05	0.03	0.28	55.05
95	1.82	43.04	1.72	40.49	0.61	42.80	0.47	37.94	1.00	99.52	0.88	99.58	0.06	0.24	63.98
100	1.35	29.58	1.48	50.68	0.64	27.56	0.48	39.32	0.69	106.04	0.55	105.29	0.06	0.29	66.10
105	2.21	37.59	1.89	37.62	0.47	33.83	0.27	52.59	0.52	121.08	0.52	120.20	-0.09	0.16	86.78
110	2.36	33.63	2.10	71.36	0.56	34.87	0.46	71.78	0.75	170.66	0.57	175.73	-0.06	0.17	101.40
115	2.59	33.04	1.65	69.19	1.05	37.74	1.09	65.04	0.56	181.22	0.59	178.05	-0.23	0.02	116.99
120	1.61	37.97	1.41	105.53	0.80	39.19	0.65	64.27	0.44	208.95	0.43	212.20	-0.17	-0.05	124.12
125	1.51	36.63	0.99	85.85	1.01	35.57	0.88	73.75	0.62	217.11	0.64	223.51	-0.28	-0.03	158.57
130	2.44	41.34	1.61	101.86	1.06	53.08	0.69	105.04	0.65	280.79	0.57	287.00	-0.24	0.09	151.28
135	1.77	55.95	0.82	93.19	0.79	41.46	0.40	83.95	0.50	297.16	0.51	295.41	-0.28	0.11	170.82
140	2.90	69.66	1.41	109.08	0.81	52.83	0.39	107.32	0.65	315.16	0.56	320.38	-0.15	0.11	204.80
145	2.55	53.45	1.86	92.89	1.27	51.01	1.00	100.71	0.24	338.56	0.25	333.11	-0.12	0.30	224.38
150	2.26	69.40	1.25	118.01	0.89	63.34	0.68	90.98	0.27	385.14	0.30	386.91	-0.44	0.02	267.56
155	2.92	64.93	1.86	108.53	0.75	62.22	0.69	91.11	0.25	418.69	0.31	416.14	-0.37	0.01	303.88
160	2.27	89.10	1.83	117.95	1.03	111.02	1.00	103.91	0.74	500.20	0.60	501.84	-0.27	0.14	332.20
165	2.78	68.97	1.86	104.30	1.58	66.02	0.97	97.86	0.31	467.59	0.29	472.23	-0.22	0.21	376.23
170	2.13	69.20	1.97	94.21	1.16	80.54	0.71	101.35	0.25	521.77	0.25	522.79	-0.33	0.06	425.68
175	2.29	63.57	1.78	107.72	1.30	67.86	0.95	110.81	0.29	525.27	0.23	526.13	-0.19	0.19	489.41
180	2.53	77.16	1.62	105.10	1.40	74.96	1.15	121.08	0.50	670.68	0.52	667.02	-0.39	0.18	444.55
185	3.10	77.88	2.12	143.94	0.92	83.62	0.64	148.36	0.50	792.41	0.50	798.71	-0.27	0.12	514.50
190	2.49	75.87	1.23	160.65	1.16	75.50	0.80	167.56	0.27	889.03	0.27	884.59	-0.29	0.12	615.34
195	2.97	71.34	2.19	165.60	1.06	83.88	1.10	171.90	0.77	859.39	0.78	855.71	-0.12	0.31	681.15
200	2.79	85.16	1.70	185.85	1.08	84.15	0.62	182.97	0.78	958.31	0.78	963.86	-0.19	0.39	747.09
Avg per inst	1.72	37.43	1.22	65.23	0.64	37.24	0.49	62.33	0.42	246.86	0.40	247.47	-0.11	0.18	174.03

Table 4.15: AngleDistanceTSP results: comparison of the results averaged by instance size of the non-dominated algorithms of Staněk et al., 2019 with GTS-angular c-LENS.

Chapter 5

The capacitated team orienteering problem with multiple time windows and time-dependent score functions

Publication status: A. Theiß, R. Cavagnini, and D. Gellert (2025). *An iterated local search for the capacitated team orienteering problem with time-dependent and piecewise linear score functions.* Working Paper. Chair of Computational Logistics, RWTH Aachen University, Germany

Abstract: In this paper, we study two versions of the capacitated team orienteering problem with time-dependent and piecewise linear score functions (C-TOP-TDPLSF). In the C-TOP-TDPLSF, a fleet of capacitated vehicles is available for visiting a set of customers, each associated with a demand, a service time, a set of time windows, and a score that varies based on the visiting time and that is modeled by a piecewise linear function. Given a maximum tour duration for each vehicle, the company must determine which customers to visit, which vehicle should serve them, and in which sequence. The objective is to maximize the total score collected by the fleet. Depending on the setting, parking and waiting at customer locations may or may not be allowed. We refer to the C-TOP-TDPLSF in which waiting at customers is not permitted as C-TOP-TDPLSF-nw and the variant in which waiting is allowed as C-TOP-TDPLSF-w. We propose a mathematical formulation for both variants, and improve them with problem-specific preprocessing techniques and valid inequalities. Because commercial optimization solvers cannot even find feasible solutions for realistically-sized instances of C-TOP-TDPLSF in reasonable runtimes, we propose two heuristics based on an iterated local search framework: ILS-noWait for solving the C-TOP-TDPLSF-nw and ILS-cWait-fin for solving the C-TOP-TDPLSF-w. These heuristics differ in the evaluation of the waiting decision. In ILS-cWait-fin, a finalization phase is applied to a subset of best-found solutions from the ILS to optimize customer visiting times while maintaining the sequence of visits fixed. Computational experiments on small-scale instances show that for C-TOP-TDPLSF-nw instances, ILS-noWait outperforms a commercial optimization

solver. However, for some C-TOP-TDPLSF-w instances, ILS-cWait-fin struggles to match the quality of the solutions found with the commercial optimization solver. By analyzing large-scale instances, we draw managerial insights into the impact of the time window width, vehicle number, fleet size, and of allowing waiting at customer locations.

Contribution of the author: The authors shared efforts in the conceptual development of the research goals, the design of the methodology and implementation of the algorithm, the computational experiments and result analysis, and in writing the paper.

5.1 Introduction

High-quality delivery services are crucial for the success of companies and form a central component of their customer service strategies. Customer-oriented deliveries improve satisfaction and loyalty, providing companies with a competitive advantage in today's market.

We study the problem of a company that delivers goods and wants to maximize the quality of the service offered to customers. The quality of the service depends on whether and when customers are visited. Each customer has a demand and a service time. Given a time horizon, each customer has multiple time windows during which they can be visited. Customers indicate these time windows as either preferred or less preferred. This depends on their primary activity that customers must interrupt to receive goods. Depending on the urgency of the primary activity, a time window may be less favored than another. The quality of the delivery service offered by the company depends on whether and on the time at which customers are visited. In fact, the quality of the service is high if the visit falls in a preferred time window, and decreases for less preferred time windows as the visit time gets farther from the preferred time windows. A fleet of capacitated vehicles for visiting customers is available, and for each vehicle, the total time spent traveling and serving customers cannot exceed a maximum duration. Finally, depending on the setting, drivers may be allowed to park their vehicles only for the duration necessary to unload their goods and visit a customer, or may be allowed to park and wait at a customer location.

This problem has multiple real-world applications, especially for customer-service oriented companies. A typical example is deliveries to restaurants. Throughout the day, restaurants have open and close hours. Within their open hours, they may experience peak hours in which they must serve many people. When a restaurant is open but experiences low occupancy, waiters can easily interrupt their activities to manage goods receipt, making these time windows preferred. The quality of the service provided by the company is high by visiting that restaurant during such time windows due to the convenience for the restaurant. As the delivery time approaches peak hours, the waiters' idleness decreases and consequently, also the restaurant's preference and the quality of the service. This reflects the inconvenience for the restaurant to be visited in that time window.

To study this problem, we introduce a variant of the team orienteering problem (TOP) and we call it capacitated team orienteering problem with time-dependent and piecewise linear score functions (C-TOP-TDPLSF). We refer to the C-TOP-TDPLSF in which waiting at a customer is not permitted as the capacitated team orienteering problem with time-dependent and piecewise linear score functions with no waiting (C-TOP-TDPLSF-nw). In C-TOP-TDPLSF-nw, a waiting period, i.e., a time span between the arrival of a vehicle at a customer and the start of service, is not allowed. However, to also study those settings in which parking and waiting is allowed, we introduce a variant of the C-TOP-TDPLSF called the capacitated team orienteering problem with time-dependent and piecewise linear score functions with waiting (C-TOP-TDPLSF-w). In the C-TOP-TDPLSF-w, drivers can park at customer locations and wait for the most advantageous time to begin service. Both in the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w, waiting at the depot is allowed.

The C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w differ from the TOP in three significant ways. First, the vehicles have a limited capacity. Second, each customer has multiple time windows during which they can be visited and they can be either preferred or less preferred. Third, the score that can be collected at each customer depends on the visiting time of a vehicle at that customer. To represent these possibly non-continuous opening times and different preferences, multiple time windows with time-dependent scores described by a piecewise linear score function must be considered. Specifically, our score function exhibits constant scores for preferred time windows and linear functions with positive or negative slopes for less preferred time windows.

In the literature, the TOP with multiple time windows and time-dependent score functions is studied by Yu et al. (2019). While in their work, the score of each time window is represented by a constant function, the piecewise linear score functions in the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w allow for a more realistic representation of customer's preferences. In Yu et al. (2019), similar to the C-TOP-TDPLSF-w, if a vehicle arrives at a customer before the opening of any of its time windows, waiting at a customer is allowed. However, in their work, waiting longer than the opening time of any time windows is never convenient because the score within each time window is constant. In fact, waiting longer only consumes time that could be used in a more profitable manner. On the contrary, in the C-TOP-TDPLSF-w, waiting longer than the opening time of a less preferred time window may lead to a higher collected score. Yu et al. (2019) also propose a heuristic to solve their problem. However, because of this difference in the customers' score function, applying their heuristic to the C-TOP-TDPLSF-w leads to suboptimal solutions.

We fill the gaps in the literature by addressing two practically relevant problems (i.e., the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w) that feature an accurate representation of customers' preferences and account for the option of waiting at customers. For each of these two problems, we propose a mathematical formulation, problem-specific preprocessing techniques, and valid inequalities. Because commercial optimization solvers cannot find a feasible solution for realistically-sized instances of the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w in reasonable runtimes, we propose an iterated local search (ILS) to solve them. In particular, we present two ILS variants: ILS-noWait for solving the C-TOP-TDPLSF-nw, and ILS-cWait-fin for solving the C-TOP-TDPLSF-w. These two ILS variants differ in how the waiting decisions are evaluated. On the one hand, for the C-TOP-TDPLSF-nw, because waiting is not allowed, ILS-noWait considers all moves in which the arrival time of a vehicle at a customer falls outside a time window as infeasible and does not accept them. On the other hand, solving the C-TOP-TDPLSF-w requires to make decisions about the length of the waiting time of vehicles at customers. Because these decisions are continuous, evaluating them within a metaheuristic framework is computationally expensive. Hence, during the local search phase, ILS-cWait-fin only considers a prespecified set of waiting strategies. These waiting strategies depend on whether the upcoming time window with respect to the arrival time at a customer is a preferred or a less preferred one. During the ILS, a given number of best solutions is saved. In ILS-cWait-fin, after the ILS terminates, each of the saved best solutions undergoes a finalization phase. This finalization phase keeps the customer visiting sequence of the solution fixed but their visiting times can be adapted. Hence, the goal of this finalization phase is to search for better customer visiting times (and hence, better waiting strategies) to improve the collected score.

For our computational experiments, we generate a new set of instances based on the ones in Souffriau et al. (2013). On small-scale instances, the results show the effectiveness of our preprocessing techniques and that our ILS-noWait outperforms a commercial optimization solver on C-TOP-TDPLSF-nw small-scale instances. However, for some C-TOP-TDPLSF-w small-scale instances, ILS-cWait-fin struggles to consistently achieve a good solution quality. The results show that althought the finalization phase leads to higher runtimes, it is crucial for improving the solutions for the C-TOP-TDPLSF-w. Moreover, we show that existing algorithms from the literature are inadequate for producing high-quality solutions for instances of the C-TOP-TDPLSF-w due to their simplistic waiting strategy. The solutions obtained by ILS-noWait and ILS-cWait-fin for large-scale instances are used to draw managerial insights into the effects of the time window width, vehicle fleet size, and of allowing waiting at customer locations.

This paper is organized as follows. Section 5.2 presents a review of the literature of related problems. In Section 5.3, we provide a detailed description of the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w and their mathematical formulations. In Section 5.4, we describe the ILS variants that we designed to solve the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w. Section 5.5 presents the computational experiments and results. Finally, we conclude in Section 5.6.

5.2 Literature review

To the best of our knowledge, the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w have not been studied in the literature yet. In this section, we review the extant literature with respect to the main features of our problems.

In the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w, we consider a limited number of capacitated homogeneous vehicles with route duration constraints. The TOP with a homogeneous and limited vehicle fleet and tour duration constraints is introduced by Chao et al. (1996) who develop a two-step heuristic for this problem and propose benchmark instances. Archetti et al. (2009) are the first to extend the TOP by considering a capacity constraint for each vehicle in addition to the time limit on the tour duration. They propose a branch-and-price exact algorithm (later improved in Archetti et al., 2013), and three metaheuristics.

An important feature of our problems is the presence of time windows. Kantor and Rosenwein (1992) are the first to study the orienteering problem with time windows. Vansteenwegen et al. (2009) study the team orienteering problem with time windows (TOPTW) and tour duration constraints and design a fast iterated local search (ILS) to solve it. In their local search procedure, only one neighborhood operator that tries to insert one customer at a time is used. The perturbation procedure removes one or more consecutive customers and it is designed to encourage every customer to be removed at least once. After a new solution is obtained, the search continues from the current solution instead of the best-one found. The distinguishing feature of this metaheuristic relies on the efficient time window feasibility evaluation of insertion moves and on the efficient shifting of arrival times at customers after one or more customers are removed from a route. Gunawan et al. (2017) propose an improved version of this ILS by properly tuning the algorithm's parameters and Gavalas et al. (2019) improve the ILS of Vansteenwegen et al. (2009) by proposing two extensions that group closely located customers together. Amarouche et al. (2020) propose a multistart ILS that alternates between two search spaces, the route representation of a solution and the giant tour representation of a solution. Additional metaheuristics have been proposed (see, e.g., Labadie et al., 2012; Labadie et al., 2010; Labadie et al., 2011; Lin and Vincent, 2012; Hu and Lim, 2014; Guibadj and Moukrim, 2014; Schmid and Ehmke, 2017). However, the ILS of Vansteenwegen et al. (2009) stands out for speed, while the multistart ILS of Amarouche et al. (2020) outperforms the other methods for solution quality. Different from C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w, in the TOPTW, only one time window per customer is considered.

Tricoire et al. (2010) introduce the multi-period orienteering problem with multiple time windows (MuPOPTW). In the MuPOPTW, there are mandatory and optional customers to visit, and each customer has individual service times. In each period, one route is planned that cannot exceed a maximum duration for that period, and a maximum driver's working time over the whole time horizon. Each customer may have up to two time windows per day that can also vary depending on the day. Time windows of the same customer cannot overlap. Due to the consideration of multiple time windows, the MuPOPTW is related to the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w. Multiple time windowsand multiple periods are also included in the multi-constraint team orienteering problem with multiple time windows (MCTOPMTW) studied by Souffriau et al. (2013). This problem has multiple constraints on a number of attributes referring both to intra-route features (e.g., route duration), and inter-tour features (e.g., a maximum number of vertices of a type visited by all routes). Apart from the presence of multiple attributes, the MCTOPMTW differentiates from the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w because in the MCTOPMTW, time windows are not only vertex-dependent but also vehicle-dependent.

All papers discussed above consider a constant score for each customer. Nevertheless, real-world problems may require to consider time-dependent scores. Yu et al. (2022) classify (T)OP variants with time-dependent scores into two categories: the orienteering problem with arrival-time-varying score (OPATP) and the orienteering problem with service-time-varying score (OPSTP, see, e.g., Erdoğan and Laporte, 2013; Gunawan et al., 2018; Khodadadian et al., 2022). Because the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w belong to the OPATP, we only focus on this category of time-dependent scores.

The score function used in the team orienteering problem with time windows and time-dependent scores (TOPTW-TDS) investigated by Yu et al. (2019) is the most similar to the one of the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w. The authors consider multiple time windows in which a vertex can be visited and waiting at a vertex for the opening of any of its time windows is allowed. The collected score when visiting a vertex is the product of the vertex basic score and a recommendation factor of the time window the vertex is visited in. Consequently, the score of a vertex in the TOPTW-TDS is described by a step function. Yu et al. (2019) develop an artificial bee colony heuristic to solve the TOPTW-TDS. To guarantee diversification, the authors consider a simulated annealing-based acceptance rule. Despite of the good performance achieved by their heuristic when solving the TOPTW-TDS, the design of this artificial bee colony heuristic is not suitable for solving the C-TOP-TDPLSF-w due to two features of this metaheuristic. First, in the heuristic by Yu et al. (2019), if the vehicle arrives at a vertex before the opening time of one of its time windows, the visiting time is delayed until the opening time of one of its time windows. In fact, in their problem, waiting longer than the time window opening is not advantageous because it does not lead to a higher collected score and, instead, it only consumes time for the route duration. Recalling that in our problem, the score within less preferred time windows is determined by a piecewise linear function, waiting only for the opening of the upcoming time window may result in a lower collected score than the one that could be achieved by visiting the customer in the middle of that time window, for example. Second, to consider the presence of multiple time windows with different scores in the neighborhood search, Yu et al. (2019) design an operator that randomly selects a vertex with at least two available time windows and changes its current visiting time with the opening time of one of the next time windows selected randomly. For our problem, this may lead to collecting again a suboptimal score if the visiting time is changed with the opening time of a less preferred time window. Both cases potentially lead to bad-quality solutions for the C-TOP-TDPLSF-w because part of the time budget for the route duration is spent without a gain in the collected score as counterpart. Hence, a properly-designed waiting strategy for the C-TOP-TDPLSF-w is necessary. Moreover, apart from extending the TOPTW-TDS by considering scores determined by a piecewise linear function, we also consider capacitated vehicles. This makes impossible to use Yu et al. (2019) algorithm to solve instances of C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w.

Kim et al. (2020) study a version of the TOPTW-TDS, in which only one vehicle is considered and customers do not have time windows but can be visited at any time during the time horizon. The time horizon is divided into periods with each period having a different but constant score. Yu et al. (2022) study the TOP-TTP in which the score collected by visiting a vertex decreases with the arrival time and increases with the service time spent at a vertex. Because the score depends on two variables, the score function of a vertex is non-linear. We differentiate from the TOP-TTP by considering capacitated vehicles and constant service times. In our problem, each customer's score function is piecewise linear.

Scores depending on the arrival time are often considered in papers studying problems with applications in catastrophe management, in which the probability of finding survivors decreases with the length of the time interval between the disaster and the rescuing teams arriving at destination (see, e.g., Erkut and Zhang, 1996). Arrival-time dependent scores also characterize technician routing applications, in which a larger score is given depending on the urgency of a task (see, e.g., Tang et al., 2007; Ekici and Retharekar, 2013). In these works, neither time windows nor capacity restrictions are considered. Peng et al. (2019) describe the scheduling problem for an agile observation satellite (AEOS), that has to perform a subset of observation tasks while the satellite is within a visible time window of an observation target. Within this time window, the best image quality can be achieved when the AEOS is directly above the target (that is, the Nadir Point). The larger the angle of observation, the worse the image quality. This problem falls in the category of arrival time-dependent scores with time windows and, due to the AEOS characteristics, travel times are time-dependent. However, differently from most of the other papers in the literature, the scores do not decrease monotonically over time, but increase up to the Nadir Point and then decrease again. Therefore, similar to our problem, they consider a piecewise linear function. However, the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w consider multiple vehicles and static traveling times. Recently, Barrena et al. (2023) investigate the selective traveling salesperson problem with time-dependent profits. In this problem, the score function of a vertex can be time-dependent in two ways: It can be monotonically increasing, also called cumulative, or it can completely depend on the visiting time. In the latter case, it resembles a piecewise linear function that can take on positive or negative values. Negative values correspond to a loss. The authors distinguish between the cases in which a vertex can be visited only once, or multiple times. The case with only one visit is a common characteristic to the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w. We extend the problem of Barrena et al. (2023) by considering multiple routes that are constrained not only by the duration, but also by the capacity of a vehicle.

The OP is a special case of the traveling salesperson problem (TSP) with profits introduced by Feillet et al. (2005). A solution of this problem is a tour that visits a subset of vertices exactly once and returns to the origin. When the objective function aims to simultaneously maximizing the total revenue minus the travel costs, the TSP with profits corresponds to the profitable tour problem introduced by Archetti et al. (2009). However, more often, only one of the two objective function components is considered in the objective function, and the other in the constraints. When the travel cost is interpreted as travel times, and the objective is to only optimize the collected profit while respecting a maximum tour duration constraint, the problem is equal to the OP. When the objective is to minimize travel costs while collecting at least a minimum amount of profit, the problem is referred to as the prize-collecting traveling salesperson problem. An overview of the variants of the TSP with profits and the VRP with profits and a comparison of exact, approximate, and heuristic algorithms to solve them can be found in Archetti et al. (2014).

Finally, the OP is also related to the tourist trip design problem (TTDP, Gavalas et al., 2014; Ruiz-Meza and Montoya-Torres, 2022). In the TTDP, the goal is to optimize the tour planning of an individual tourist by considering multiple means of transport to move between points of interest (e.g., museums and parks). Each mean of transport is characterized by different travel times and the score of each point of interest is partially based on the subjective importance for the tourist. A TTDP in which there is only one mean of transport reduces to the OP.

5.3 Problem description and model formulation

In this section, we describe the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w, and we provide their mathematical formulations. In Section 5.3.1, we present the C-TOP-TDPLSF-nw in which waiting is only allowed at the depot. The C-TOP-TDPLSF-w, i.e., a variant of the C-TOP-TDPLSF-nw in which waiting at the depot and at customers is allowed, is described in Section 5.3.2. Finally, Section 5.3.3 presents preprocessing techniques and valid inequalities that are applied to strengthen the formulations.

5.3.1 C-TOP-TDPLSF with no waiting at customers (C-TOP-TDPLSF-nw)

We formulate the C-TOP-TDPLSF-nw as a mixed integer program. The C-TOP-TDPLSF-nw can be defined on a complete directed graph $G = (V, A)$, where V is the vertex set with $|V| = N$ and A is the set of arcs. The vertex set is partitioned into $V = \{0\} \cup V_C \cup \{N + 1\}$, where 0 and $N + 1$ represent the depot (every route starts at 0 and ends at $N + 1$), and V_C is the set of customers. For convenience, we represent the set of customers plus the origin depot by $V_0 = \{0\} \cup V_C$ and the set of customers plus the arrival depot by $V_{N+1} = V_C \cup \{N + 1\}$. A nonnegative travel time d_{ij} is associated to each arc $(i, j) \in A$ and we assume that the travel times satisfy the triangle inequality.

The set of homogeneous vehicles is represented by K. Each vehicle has capacity Q_k and is initially located at the depot. The vehicles are loaded at the depot at the beginning of the tour and return to the depot empty at the end of the tour. Each vehicle is subject to a maximum tour duration D_{max} that is identical for all vehicles. Each customer $i \in V_C$ has a nonnegative demand q_i, a service time s_i, and a maximum score p_i. Split deliveries are not allowed, i.e., the demand can only be served in one visit by one vehicle.

Given a time horizon of length T, every customer $i \in V_C$ has a given list W_i of chronologically-ordered hard time windows that describe the time periods within T in which customer i can be visited. The opening and the closing time of time window $w \in W_i$ are represented by a_{iw} and b_{iw}. Each customer has a set of preferred time windows $P_i \subseteq W_i$ and less preferred time windows $LP_i \subseteq W_i$, with $P_i \cup LP_i = W_i$ and $P_i \cap LP_i = \emptyset$. We additionally split the set of less preferred time windows LP_i into two sets LP_i^+ and LP_i^-, which indicate the sign of the slope. Two consecutive time windows within W_i can only overlap at the edge (i.e., $b_{i,w-1} = a_{iw}$) and we assume there cannot be two consecutive time windows of the same type. For all periods outside the preferred and less preferred time windows, no visit can take place.

The decision variables $x_{ij}^k, i \in V_0, j \in V_{N+1} : i \neq j, k \in K$ are binary and equal to one if arc (i, j) is traversed by vehicle k, and zero otherwise. Variable $t_i^k, i \in V, k \in K$ represents the visiting time of vehicle k at vertex i. Note that in the C-TOP-TDPLSF-nw, the visiting time corresponds to the arrival time. Moreover, variables $y_{iw}^k, w \in W_i, i \in V_C, k \in K$ are binary and equal to one if customer i is visited in time window w by vehicle k and zero otherwise.

The objective of C-TOP-TDPLSF-nw is to maximize the sum of the scores collected at the visited customer. The score collected at each customer is defined by $f_i(t_i^k)$ and depends on the time at which a vehicle arrives at the customer. Specifically, $f_i : T \to \mathbb{R}$ is the piecewise linear function for vertex $i \in V_C$ that assigns a score to each point t in the time horizon T. Based on t_i^k, the score function is defined as

$$f_i(t_i^k) = \begin{cases} \beta_{iw} t_i^k + \alpha_{iw} & \text{if } \exists w \in W_i, a_{iw} < t_i^k < b_{iw}, \\ \max\{\beta_{i,w-1} t_i^k + \alpha_{i,w-1}, \beta_{iw} t_i^k + \alpha_{iw}\} & \text{if } \exists w \in W_i, w \neq 1, b_{i,w-1} = t_i^k = a_{iw}, \\ 0 & \text{otherwise.} \end{cases}$$

The coefficients α_{iw} and β_{iw} represent the intercept and the slope of the score function, respectively. If t_i^k falls on the edge between two time windows (e.g., $t_i^k = b_{i,w-1} = a_{iw}$), we assume that the highest score is collected. In preferred time windows $w \in P_i$, the slope of the score function is equal to zero ($\beta_{iw} = 0$) and the intercept is equal to p_i ($\alpha_{iw} = p_i$), i.e., the maximum nonnegative score.

Figure 5.1 shows an example of the score function for a customer, which opens 6:00 a.m. in the morning and closes at 10:00 p.m., and is additionally closed in time intervals $(8, 10)$ and $(16, 18)$. The maximum score is $p_i = 100$. The customer has eight time windows: three preferred time windows ($[10, 12]$, $[14, 16]$, and $[20, 22]$), and five less preferred time windows ($[6, 7]$, $(13, 14)$, and $[18, 20)$ with positive slope and $(7, 8]$ and $(12, 13]$ with negative slope).

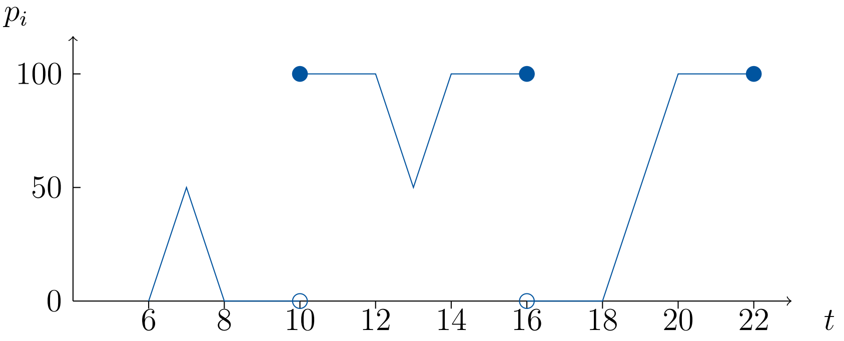


Figure 5.1: Exemplary score function for a customer with eight time windows.

In the C-TOP-TDPLSF-nw, waiting is only allowed at the depot, i.e., before a tour starts. By departing at the beginning of the time horizon, if the arrival time of the vehicle at the first customer i visited in a tour falls in one of its preferred time windows W_i, then the arrival time σ_i at customer i corresponds to d_{0i}, i.e., the travel time between the depot and customer i. In this case, there is no waiting time at the depot. On the opposite, waiting at the depot occurs if departing at time zero leads to an arrival time at the first visited customer that is outside any of its time windows. The waiting time is dependent on the next upcoming time window. For a point in time t, where visiting customer i is not possible, the upcoming time window is defined as

$$w_{\text{upcoming},i}(t) = \underset{w \in W_i}{\text{argmin}}\{a_{iw} > t\}.$$

The arrival time σ_i at the first customer $i \in V_C$ visited in a tour is defined as follows:

$$\sigma_i = \begin{cases} d_{0i} & \text{if } \exists w \in W_i, a_{iw} > d_{0i} > b_{iw}, \\ a_{iw_{\text{upcoming},i}(d_{0i})} & \text{if } w_{\text{upcoming},i}(d_{0i}) \in P_i \cup LP_i^-, \\ \frac{a_{iw_{\text{upcoming},i}(d_{0i})}+b_{iw_{\text{upcoming},i}(d_{0i})}}{2} & \text{if } w_{\text{upcoming},i}(d_{0i}) \in LP_i^+. \end{cases}$$

If the upcoming time window $w = w_{\text{upcoming},i}(t)$ is a preferred or a less preferred one with negative slope, i.e., $w \in P_i \cup LP_i^-$, then the arrival time σ_i at customer i corresponds to a_{iw}, i.e., the opening time of time window w. If the next upcoming time window w is a less preferred time window with positive slope, i.e., $w \in LP_i^+$, then the arrival time σ_i at customer i corresponds to $\frac{a_{iw}+b_{iw}}{2}$, i.e., the middle of time window w.

The mathematical model of C-TOP-TDPLSF-nw is formulated as a mixed-integer program as follows:

$$\max \sum_{k\in K}\sum_{i\in V_C} f_i(t_i^k) \tag{5.1}$$

$$\text{s.t.} \sum_{w\in W_i} y_{iw}^k = \sum_{j\in V_{N+1}:j\neq i} x_{ij}^k \qquad i \in V_C, k \in K \tag{5.2}$$

$$\sum_{k\in K}\sum_{w\in W_i} y_{iw}^k \leq 1 \qquad i \in V_C \tag{5.3}$$

$$\sum_{j\in V_{N+1}} x_{0j}^k = \sum_{j\in V_0} x_{jN+1}^k = 1 \qquad k \in K \tag{5.4}$$

$$\sum_{j\in V_{N+1}:j\neq i} x_{ij}^k = \sum_{j\in V_0:j\neq i} x_{ji}^k \qquad i \in V_C, k \in K \tag{5.5}$$

$$D_{max} - d_{0i}x_{0i}^k - (d_{jN+1}+s_j)x_{jN+1}^k - (t_j^k - t_i^k) \geq 0 \qquad i \in V_C, j \in V_0, k \in K \tag{5.6}$$

$$\sum_{i\in V}\sum_{j\in V:j\neq i} q_i x_{ij}^k \leq Q_k \qquad k \in K \tag{5.7}$$

$$t_i^k \geq \sigma_i x_{0i}^k \qquad i \in V_C, k \in K \tag{5.8}$$

$$t_i^k \leq \sigma_i + M(1 - x_{0i}^k) \qquad i \in V_C, k \in K \tag{5.9}$$

$$t_i^k + s_i + d_{ij} \leq t_j^k + M(1 - x_{ij}^k) \qquad i \in V_C, j \in V_{N+1} : j \neq i, k \in K \tag{5.10}$$

$$t_i^k + s_i + d_{ij} \geq t_j^k - M(1 - x_{ij}^k) \qquad i \in V_C, j \in V_{N+1} : j \neq i, k \in K \tag{5.11}$$

$$a_{iw}y_{iw}^k \leq t_i^k \leq M(1 - y_{iw}^k) + b_{iw} \qquad w \in W_i, i \in V_C, k \in K \tag{5.12}$$

$$t_i^k \leq M \sum_{j\in V_{N+1}:j\neq i} x_{ij}^k \qquad i \in V_C, k \in K \tag{5.13}$$

$$t_{N+1}^k \leq T \qquad k \in K \tag{5.14}$$

$$t_i^k \geq 0 \qquad i \in V, k \in K \tag{5.15}$$

$$x_{ij}^k \in \{0,1\} \qquad i \in V_0, j \in V_{N+1} : j \neq i, k \in K \tag{5.16}$$

$$y_{iw}^k \in \{0,1\} \qquad w \in W_i, i \in V_C, k \in K \tag{5.17}$$

The objective function of maximizing the collected score by all vehicles is defined in (5.1). We denote the objective function of this formulation with $\Theta^{C-TOP-TDPLSF-nw}$.

The collected score at customer i depends only on the arrival time t_i^k. Constraints (5.2) link the variables x_{ij}^k and y_{iw}^k. If customer i is visited (i.e., $\sum\limits_{j \in V_{N+1}: j \neq i} x_{ij}^k = 1$), then the visit must fall in one of its time windows. Otherwise, if customer i is not visited (i.e., $\sum\limits_{j \in V_{N+1}: j \neq i} x_{ij}^k = 0$), no time window for that customer is chosen. Constraints (5.3) guarantee that each customer can be visited at most once. Constraints (5.4) ensure that there are as many tours as the number of available vehicles. This constraint allows empty tours, i.e., tours including no vertices but the depot. Constraints (5.5) ensure flow conservation, and constraints (5.6) restrict every tour duration to the limit D_{max}. Constraints (5.7) guarantee that the load never exceeds the capacity of the vehicle. Constraints (5.8) and (5.9) define the arrival time at the first visited customer in a tour and allow waiting at the depot. Constraints (5.10) and (5.11) determine the arrival time at each customer. These constraints do not allow waiting at the customers and they also serve as subtour elimination constraints. For these constraints, the large number M is substituted with $T + s_i + d_{ij}$ to tighten the formulation. Constraints (5.12) are inspired by Tricoire et al. (2010), in which the time windows are not only dependent on the vertex, but also on the period of the visit. These constraints link variables y_{iw}^t and t_i^k. Constraints (5.13) force the arrival time at a non-visited customer to be zero. For constraints (5.12) and (5.13), the large number M is substituted with $\min\{T - d_{i0}, b_i^{max}\}$, where $b_i^{max} = \max\{b_{iw} : w \in W_i\}$ to tighten the formulation. Constraints (5.14) impose that the arrival time at the depot cannot be greater than the time horizon length T. Finally, constraints (5.15)–(5.17) define the domain of the variables.

5.3.2 C-TOP-TDPLSF with flexible waiting strategy (C-TOP-TDPLSF-w)

In the C-TOP-TDPLSF-w, if the arrival time of a vehicle at a customer falls outside one of its time windows, waiting is allowed. Note that, because of the shape of customers' score functions in less preferred time windows with positive slope, waiting longer than the opening time of the upcoming time window may be beneficial.

To allow for a flexible waiting strategy at customers, we modify the model formulation (5.1)–(5.17) by removing constraints (5.9) and (5.11), and by modifying constraints (5.9) as follows:

$$t_i^k \geq d_{0i} x_{0i}^k \qquad i \in V_C, k \in K. \tag{5.18}$$

In the C-TOP-TDPLSF-w formulation, the variable t_i^k represents the visiting time (instead of the arrival time) at customer i because it may include the time that the vehicle k waits at customer i. We denote the objective function of this formulation with $\Theta^{C-TOP-TDPLSF-w}$. The total score of the optimal solution of C-TOP-TDPLSF-w is an upper bound on the score of the optimal solution of C-TOP-TDPLSF, i.e., $\Theta^{C-TOP-TDPLSF-nw} \leq \Theta^{C-TOP-TDPLSF-w}$.

5.3.3 Preprocessing techniques and valid inequalities

To strengthen the formulations presented in Section 5.3.1 and 5.3.2, we apply preprocessing steps to remove infeasible arcs as commonly done in the literature (Psaraftis, 1983; Savelsbergh, 1985; Schneider et al., 2014). An arc (i, j) can be removed from the set of possible arcs, i.e., x_{ij}^k can be fixed to zero, if one of the following conditions holds:

$$\begin{aligned}
i, j \in V_C: &\quad q_i + q_j > Q_k, && (5.19)\\
i \in V_C, j \in V_C: &\quad d_{0i} + s_i + d_{ij} + s_j + d_{j,|N|+1} > D_{max}, && (5.20)\\
i \in V_0, j \in V_{N+1}: &\quad a_i^{min} + s_i + d_{ij} > b_j^{max}, &&\\
&\quad \text{where } a_i^{min} = \min\{a_{iw} : w \in W_i\} \text{ and } b_j^{max} = \max\{b_{jw} : w \in W_j\}, && (5.21)\\
i \in V_0, j \in V_C: &\quad a_i^{min} + s_i + d_{ij} + s_j + d_{j,|N|+1} > T, \text{where } a_i^{min} = \min\{a_{iw} : w \in W_i\}. && (5.22)
\end{aligned}$$

Inequality (5.19) and (5.20) are well-known preprocessing steps based on capacity and tour duration violations. Inequalities (5.21) and (5.22) are based on time window and time horizon violations, respectively.

Moreover, to strengthen the formulation, we add the following well-known valid inequalities (Irnich and Villeneuve, 2006; Archetti et al., 2014; Lahyani et al., 2018; Darvish et al., 2020):

$$\begin{aligned}
x_{ij}^k + x_{ji}^k \leq 1 &\quad i \in V_C, j \in V_C : j \neq i, k \in K && (5.23)\\
x_{ij}^k + x_{jh}^k + x_{hi}^k \leq 2 &\quad i \in V_C, j \in V_C : j \neq i, h \in V_C : h \neq j \neq i, k \in K && (5.24)\\
\sum_{j \in V_{N+1}: j \neq i} x_{ij}^k \leq \sum_{l=1}^{i-1} \sum_{j \in V_{N+1}: j \neq l} x_{lj}^{k-1} &\quad i \in V_C, k \in K \setminus \{0\}. && (5.25)
\end{aligned}$$

Inequalities (5.23) and (5.24) are the two-cycle and the three-cycle elimination constraints, respectively. Inequalities (5.25) are symmetry breaking constraints.

In addition, we denote the list of the non-decreasingly-ordered demands by $\overline{q} = [\overline{q}^1, \overline{q}^2, \ldots, \overline{q}^{|V_C|}]$. The maximum number of demands with the lowest values that can fit into the vehicle capacity is an upper bound on the number of customers that can be served by each tour $k \in K$ and it can be computed as follows:

$$q^{UB} = \max\{h | \sum_{l=1}^{h} \overline{q}^l < Q_k\}.$$

Similarly, we denote the list of the increasingly-ordered service times by $\overline{s} = [\overline{s}^1, \overline{s}^2, \ldots, \overline{s}^l, \ldots, \overline{s}^{|V_C|}]$, with l denoting the position in this list. The maximum number of the lowest service times that can fit into the tour duration is an upper bound on the number of customers that can be served by each tour and it can be computed as follows:

$$s^{UB} = \max\{h | \sum_{l=1}^{h} \overline{s}^{l} < D_{max}\}.$$

The strongest upper bound among q^{UB} and s^{UB} on the number of customers that can be served by each tour corresponds to the minimum of the two. Hence, the following valid inequality is added:

$$\sum_{i \in V_0} \sum_{j \in V_{N+1}: j \neq i} x_{ij}^{k} \leq \min\{q^{UB}, s^{UB}\} \quad k \in K. \tag{5.26}$$

5.4 Iterated local search for the C-TOP-TDPLSF-nw and the C-TOP-TDPLSF-w

In this section, we introduce two ILS variants: ILS-noWait for solving the C-TOP-TDPLSF-nw, and ILS-cWait-fin for solving the C-TOP-TDPLSF-w. Because these two ILS variants differ only in whether and how a waiting decision is contemplated, they share a common structure. We refer to this structure as “ILS-algo”, and use the terms “ILS-noWait”, and “ILS-cWait-fin” to specify when the two variants differ. Our ILS implements the classical ILS framework originally proposed by Lourenço et al. (2003). The pseudocode of ILS-algo is given in Algorithm 8. ILS-algo starts with a feasible solution. Due to the nature of our problems, an empty solution S^e in which no customer is visited is feasible for the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w. Similar to the approach chosen for the ILS for the TOPTW by Vansteenwegen et al. (2009), our starting solution is set to S^e. Then, the ILS is executed. In each ILS iteration, a local search phase based on a variable neighborhood descent (VND) with first improvement (Section 5.4.1) is applied. Whenever a better solution than the incumbent best is found, the best-found solution is updated. The VND terminates when the total score of the solution stops improving. Then, the perturbation phase is applied to the best-found solution S^* (Section 5.4.2) to generate a new solution having different visited customers and routes. The ILS terminates after η iterations without improvement.

Algorithm 8: Pseudocode of the ILS-algo algorithm

```
$S^* \leftarrow S^e$, $S \leftarrow S^e$
while termination criterion not satisfied do
    $S \leftarrow VND(S)$
    if $c(S) > c(S^*)$ then
        $S^* \leftarrow S$
        $c(S^*) \leftarrow S$
    end
    $S \leftarrow Perturbation(S^*)$
end
return $S^*$
```

5.4.1 Variable neighborhood descent

In the VND, we iteratively evaluate neighboring solutions that are obtained by applying to a solution S a move uniquely defined by a neighborhood operator o from the ordered set of neighborhood operators O (described in Section 5.4.1.1) and a so-called generator arc (i, j). After the move is applied, the arc (i, j) is contained in the resulting solution. Because the pivoting rule of our VND is first improvement, the traversal order of the neighborhood operators and the generator arcs influences the search trajectory. Algorithm 9 shows the pseudocode of the VND applied to solution S. At the beginning of each ILS iteration, to increase the likelihood that different search trajectories are explored, the elements in the ordered neighborhood operator set O are randomly shuffled, resulting in the list $\tilde{O}$. Each generator arc is obtained by pairing a customer $i \in \tilde{V}_C$ to a vertex j belonging to L_i. Both lists $\tilde{V}_C$ and $\tilde{L}_i$ result from sorting strategies applied to the elements of the customer and vertex sets (V_C and L_i), respectively, and described in Section 5.4.1.2. Given a neighborhood operator and a generator arc, a move is applied to a solution S and a new solution S' is obtained. If the total collected score of the new solution S' is higher than the total collected score of S, then the solution S' becomes the new incumbent solution, and the search restarts from the first operator. If no improvement can be found, the VND terminates.

5.4.1.1 Neighborhood operators

The neighborhood operators contained in list O are defined using the generator arc principle introduced in Section 5.4.1. They are depicted in Figure 5.2 and are:

- relocate-1 moves one customer to a different position, and it is defined in both inter- and intra-route fashion.
- relocate-2 moves one customers and its predecessor to a different position, and it is defined in both inter- and intra-route fashion.
- exchange swaps the positions of two customers, and it is defined in both intra- and inter-route fashion.

Algorithm 9: Pseudocode of VND(S)

```
1  $improvement = false$
2  $\tilde{O} = shuffle(O)$
3  $\tilde{V}_C = sort(V_C)/shuffle(V_C)$
4  while $improvement$ do
5      $improvement = false$
6      for $o \in \tilde{O}$ do
7          for $i \in \tilde{V}_C$ do
8              for $j \in \tilde{L}_i$ do
9                  $S^{'} \leftarrow move(o, i, j, S)$
10                 if $c(S^{'}) > c(S)$ then
11                     $improvement = true$
12                     $S \leftarrow S^{'}$
13                     $c(S) \leftarrow c(S^{'})$
14                     go to line 4
15             end
16         end
17     end
18 end
19 return $S$
```

During the VND, non-visited customers are stored in a dummy route. Consequently, by applying any of the three aforementioned operators in an inter-route manner where at least one of the routes involved is the dummy route, customers can be added to or removed from the real routes. However, these operators are never applied in intra-route fashion for the dummy route.

Before a move is evaluated, we check whether the resulting solution is feasible with respect to vehicle capacity, tour duration, time horizon, and time windows. Regarding the latter aspect, we distinguish the following two cases depending on the problem and ILS variant.

For the ILS-noWait that solves the C-TOP-TDPLSF-nw (in which waiting is not allowed), if the arrival time at any customer in the resulting routes does not fall in any of its time windows, the move is considered infeasible.

For the ILS-cWait-fin that solves the C-TOP-TDPLSF-w, if the arrival time at a customer is outside any of its time windows, waiting is allowed, but only for a predefined period of time. The length of the waiting time depends on the type of the upcoming time window. If the upcoming time window is a preferred one or a less preferred one with a negative slope, i.e., $w \in W_i \cup LP_i^-$, then the vehicle is allowed to wait until its opening time. On the contrary, if the upcoming time window is less preferred with a positive slope, i.e., $w \in LP_i^+$, then the vehicle is allowed to wait until the middle of the time window. Once the visiting time at that customer is updated, we check the arrival time at the following customer. If the arrival time at this customer falls outside one of its time windows, we repeat the same procedure described above to determine the length of the waiting time and obtain the new visiting time. The procedure stops if at least one of the following three cases occurs: (i) the updated arrival times are feasible for all customers in that route, (ii) the limit on the tour duration gets violated, and (iii) the limit on the time horizon gets violated. In case (i), the move is feasible, while in case (ii) and (iii), the move is infeasible.

We have also tested the inclusion of the 2-Opt* neighborhood operator. This operator is used in inter-route fashion, removes one arc from each of two tours, and reconnects the first part of the first tour with the second part of the second tour and vice versa. However, preliminary results have shown that the ILS-noWait and ILS-cWait-fin without 2-Opt* dominate the ILS-noWait and ILS-cWait-fin with 2-Opt*, respectively, both regarding solution quality and runtime. Our preliminary results support the findings of Souffriau et al. (2013) in which they conclude that 2-Opt moves lead to increased runtime without improvements in solution quality because the shift in the arrival time at the customers following those that define the generator arc is likely to lead to infeasibility.

5.4.1.2 Generator arc set

To speed up the search, we consider only a small portion of the total number of arcs in A to be used as generator arcs. First, we discard all arcs that are infeasible according to our preprocessing techniques described in Section 5.3.3. Then, we apply sparsification methods, i.e., strategies designed to limit the number of arcs (among the feasible ones) considered as generator arcs. Sparsification methods have originally been proposed by Toth and Vigo (2003), and applied to multiple works on routing (e.g., Prins et al., 2007; Escobar et al., 2014; Schneider et al., 2017; Goeke, 2019; Cavagnini et al., 2024).

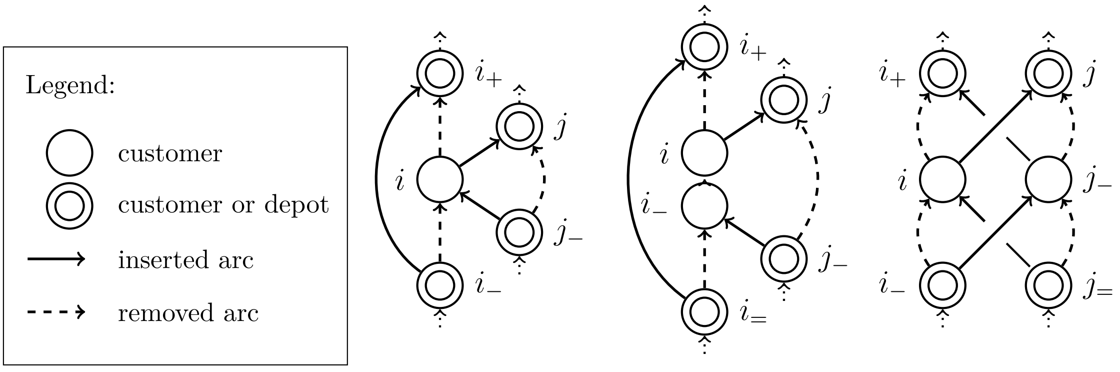
We investigate the use of different sparsification methods. All of these methods are customer-based, i.e., for each customer, only a subset of arcs is considered as generator arcs. In fact, sparsifying on a customer base guarantees that a given number of arcs incident to possibly isolated customers is always included.

In ILS-algo, each generator arc is obtained by pairing a vertex i from the list $\tilde{V}_C$ with a vertex j from the list L_i. List L_i contains only $\lceil \kappa \cdot |V_C| \rceil$ customers, with $0 < \kappa \leq 1$. In addition, L_i always includes the depot. In the lists $\tilde{V}_C$ and L_i, the customers and vertices, respectively, appear sorted according to one of the following different strategies:

- **sorted $\tilde{V}_C$-sorted** L_i: all customers in $\tilde{V}_C$ are sorted in decreasing order of their score. For each customer $i \in \tilde{V}_C$, list L_i contains the depot, and the closest $\lceil \kappa \cdot |V_C| \rceil$ customers to i, all sorted according to increasing distance.

- **sorted $\tilde{V}_C$-random** L_i: all customers in $\tilde{V}_C$ are sorted in decreasing order of their score. For each customer $i \in \tilde{V}_C$, list L_i contains, in random order, the depot, and $\lceil \kappa \cdot |V_C| \rceil$ random customers.

- **random $\tilde{V}_C$-sorted** L_i: all customers in $\tilde{V}_C$ are randomly sorted. For each customer $i \in \tilde{V}_C$, list L_i contains the depot, and the closest $\lceil \kappa \cdot |V_C| \rceil$ customers to i, all sorted according to increasing distance.



(a) relocate-1 (b) relocate-2 (c) exchange-1

Figure 5.2: Neighborhood operators of ILS-algo. The generator arc is denoted by (i, j). The predecessor and successor of i are denoted i_- and i_+, respectively.

- **random $\tilde{V}_C$-random** L_i all customers in $\tilde{V}_C$ are randomly sorted. For each customer $i \in \tilde{V}_C$, list L_i contains, in random order, the depot, and $\lceil \kappa \cdot |V_C| \rceil$ random customers.

5.4.2 Perturbation

The goal of the perturbation phase is to generate a new solution to reach new areas of the search space. We apply to the overall best-found solution S^* a given number of random feasible moves defined by the operators presented in Section 5.4.1.1. Because more moves are required to significantly perturb tours with many customers compared to tours with few customers, the number of applied moves is dependent on the number of customers in the instance. We set the number of applied moves to $\lceil \rho \cdot |V_C| \rceil$, with $0 < \rho < 1$.

5.4.3 Finalization phase

The finalization phase is applied after the ILS procedure terminates and is used only in ILS-cWait-fin. The goal of this phase is to introduce more flexibility in the determination of the waiting time at customers by adjusting their visiting times. In particular, during the execution of the ILS, the μ best-found solutions are saved in a solution pool. For each of these solutions, a linear program is solved by a commercial optimization solver. This model corresponds to the one of Section 5.3.2 in which the x_{ij}^k variables are fixed to the values obtained by the solution under consideration, and the only decision variables are t_i^k. This model provides the optimal waiting times for that solution.

5.5 Computational experiments

The goal of our computational study, performed on the instance sets described in Section 5.5.1, is threefold. First, in Section 5.5.2, we assess the improvement in the model formulations for the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w when preprocessing techniques and valid inequalities are included. Second, after tuning the parameters of ILS-algo in Section 5.5.3, we evaluate the performance of ILS-noWait and ILS-cWait-fin for solving the C-TOP-TDPLSF-nw and the C-TOP-TDPLSF-w, respectively, in Section 5.5.4. For ILS-cWait-fin, we also assess the impact of the finalization phase, and of our predefined waiting strategies. Finally, in Section 5.5.5, we derive managerial insights on, for example, the impact of the time window width, the number of vehicles, and the value of allowing waiting.

All experiments are performed on an Intel(R) Xeon(R) computer with a CPU E5-2430 v2 processor, at 2.50GHz with 64 GB RAM under Debian 12 (Bookworm) Slim. ILS-noWait, ILS-cWait-fin, and the models were implemented as single-thread code in C++ and compiled using gcc version 12.2. The models were solved with Gurobi solver version 10.0.0 by setting a time limit of two hours (i.e., 7200 seconds). In our preliminary experiments, a bug in the Gurobi presolve method was identified. Hence, we deactivated it. All other Gurobi parameters are set to their default values.

5.5.1 Description of the instances

Because C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w are new problems, no benchmark instance sets are available in the literature. Consequently, we generate our instance set by adapting the MC-TOP-MTW instances used in Souffriau et al. (2013) because their problem shares multiple features with ours. Souffriau et al. (2013) instances are available at `https://www.mech.kuleuven.be/en/cib/op`.

Section 5.5.1.1 describes how we adapted the Souffriau et al. (2013) instances for our problems. Moreover, in Section 5.5.1.2, we generate a set of small-scale instances that can be solved by means of the commercial optimization solver Gurobi and that can be used to assess the solution quality of ILS-noWait and ILS-cWait-fin.

5.5.1.1 Large-scale instances adapted from Souffriau et al. (2013)

The instances of Souffriau et al. (2013) are based on the vehicle routing problem with time windows (VRPTW) instances created by Solomon (1987) and Cordeau et al. (1997). From Solomon's instances, Souffriau et al. (2013) consider 29 instances (c101–c109, r101–r112, and rc101–rc108), all having 100 customers. We recall that the prefixes "c", "r", and "rc" represent instances in which customers are clustered, randomly distributed, and randomly clustered, respectively. From Cordeau's instances, Souffriau et al. (2013) consider eight instances (pr01–pr05, and pr07–pr09), with a number of customers between 48 and 240. In the rest of the paper, we refer to this reduced set of Solomon and Cordeau instances as “base” instances. For each instance, Souffriau et al. (2013) create one instance in which the number of tours (i.e., vehicles in our instances) is equal to one, two, three, and four, respectively. In total, 148 instances for the MC-TOP-MTW are obtained by Souffriau et al. (2013).

In our instances, customers’ service times s_i are kept the same as in Souffriau et al. (2013) instances, and the travel times d_{ij} are equal to the Euclidean distances, rounded to two decimals. Souffriau et al. (2013) set the customers’ scores equal to the customers’ demands of the VRPTW instances. For our instances, these scores correspond to the customers maximum scores p_i. Similar to Souffriau et al. (2013), the time horizon T is composed of 1000 units for all Cordeau-based instances, and for Solomon-based instances, it varies as follows: 1236 units for instances c101-c109, 230 for instances r101-112, and 240 for instances rc101-108.

Souffriau et al. (2013) study the MC-TOP-MTW motivated by an application in the tourism industry. As such, they have a budget that corresponds to the financial availability of a tourist to perform a tour that maximizes their collected score by visiting tourist attractions. Moreover, each tourist attraction has an entrance fee. In our instances, we consider the budget and the vertex entrance fees of Souffriau et al. (2013) as the vehicle capacity Q_k and the customer demands q_i, $i \in V_C$, respectively. However, while in Souffriau et al. (2013) instances, the budget increases as the number of tours increases, in our instances, the capacity of the vehicles always corresponds to the budget of Souffriau et al. (2013) instances with one tour.

In the VRPTW instances, every vertex has one time window. Souffriau et al. (2013) split this time window into four equal, consecutive, and non-overlapping time windows. In our instances, Souffriau et al. (2013)’s time windows uniquely translate into periods, each of which may or may not correspond to a time window in which a visit is allowed (i.e., it is preferred or less preferred). For each of the 148 Souffriau et al. (2013) instances, we create two instance sets called “4-period” and “8-period” instance sets. In the 4-period instance set, the four time windows of Souffriau et al. (2013) correspond to four periods. In the 8-period instance set, each of the four Souffriau et al. (2013)’s time windows is split into two equally large periods, obtaining a total of eight time periods. This allows us to analyze the impact of different time window lengths. In the 4-period instance set, we assign every customer one preferred, and two less preferred periods, leaving one period as closed. In the 8-period instance set, we assign every customer two preferred, and four less preferred periods, leaving two periods as closed. For both instance sets, we require that there must not be two consecutive periods or time windows of the same type. Considering this requirement, there are six feasible combinations in the 4-period instance set, and 24 feasible combinations in the 8-period instance set. A preferred period is translated to a preferred time window. If a less preferred period is between a preferred and closed period (closed and preferred) this corresponds to a less preferred time window with negative (positive) slope. Whenever there is a less preferred period between two preferred periods (two closed periods), we split that period in two less preferred time windows, the first one with a negative (positive) slope and the second one with a positive (negative) slope. Every customer is randomly assigned to one of these combinations. Due to the randomness of the assignment, a customer who may be open for visit at one point in time in a 4-period instance may be closed at the same time in the corresponding 8-period instance. Thus, the objective value of an optimal solution for the 4-period instance may be different to the one of the 8-period instance. In total, we obtain 296 instances: 148 in the 4-period instance set, and 148 in the 8-period instance set. We call this instance set “large-scale instance set”.

5.5.1.2 Small-scale instances

To generate our small-scale instance set, we pseudo-randomly select eleven out of the 37 “base” instances defined in Section 5.5.1.1. While selecting them, we ensured that the resulting eleven instances are such that the same proportion between the number of Cordeau et al. (1997) and Solomon (1987) instances is maintained as in the “base” instance set. Moreover, we guarantee that also the proportion between the number of Solomon (1987) instances in which customers are clustered, randomly distributed, and randomly clustered is kept as in the “base” instance set. For each of these eleven “base” instances, we have randomly drawn half the number of customers. For each such instance, we have considered one, two, and three vehicles, and four and eight periods, thus obtaining 66 instances.

Preliminary experiments have shown that Gurobi did not return a feasible solution with an acceptable optimality gap even after decreasing the number of customers. In fact, also the length of the time horizon has an influence on the complexity of the problem: the longer the time horizon, the more customers can be visited. However, reducing the length of the time horizon requires determining new time windows and recomputing all intervals for the piecewise linear function. Instead of reducing the length of the time horizon, we consider a more straightforward procedure that doubles all entries of the distance matrix and the service times.

5.5.2 Effectiveness of the preprocessing techniques and valid inequalities

In this section, we assess the effectiveness of the preprocessing techniques and valid inequalities when added to the model formulations for the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w. This allows us to determine the final configuration of the models to use for the comparison with ILS-algo.

These experiments are run by solving with Gurobi all the small-scale instances described in Section 5.5.1.2 by activating and deactivating the preprocessing techniques and valid inequalities. Table 5.1 shows the results by problem averaged over the 66 test instances. The columns of the table contain in order: the problem, the indication whether the preprocessing techniques and valid inequalities are activated, the average percentage gap obtained by Gurobi to the upper bound $\bar{\Delta}_{UB}(\%)$, the average runtime in seconds $\bar{t}(s)$, and the number of instances for which a feasible (# feasible) and an optimal (# optimal) solution is found within the time limit. The results show that, for both the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w, by including the preprocessing techniques, the average gaps to the upper bounds and the average runtimes improve. However, the inclusion of the valid inequalities does not result in improvements: the average gaps to the upper bounds and the average runtimes considerably increase. Moreover, when the valid inequalities are active, a smaller number of instances is solved to optimality. Consequently, for the final model configuration, we include only the preprocessing techniques.

Problem	Preprocessing techniques	Valid inequalities	$\overline{\Delta}_{UB}(\%)$	$\bar{t}(s)$	#feasible	#optimal
C-TOP-TDPLSF-nw	no	no	39.83	3951.02	66/66	30/66
	no	yes	55.74	4684.39	66/66	30/66
	yes	no	38.55	3924.77	66/66	30/66
	yes	yes	52.74	4548.54	66/66	29/66
C-TOP-TDPLSF-w	no	no	34.59	3935.40	66/66	31/66
	no	yes	47.44	4659.57	66/66	27/66
	yes	no	33.70	3905.72	66/66	31/66
	yes	yes	44.38	4458.64	66/66	28/66

Table 5.1: Comparison of the results (averaged over the 66 small-scale instances for each problem type) obtained with Gurobi with and without activating the preprocessing techniques and valid inequalities.

5.5.3 Parameter tuning

Because ILS-algo contains randomized elements, we performed ten runs of ILS-algo for each instance. To determine a decent parameter setting while avoiding to overfit the setting to the instances under consideration, we have conducted a parameter tuning on a subset of the large-scale instances. We have considered 88 instances obtained by selecting the same eleven "base" instances considered in the small-scale instance set but with the complete number of customers (i.e., not halved). For each "base" instance, we have considered one, two, three, and four vehicles, and four and eight periods, thus, obtaining 88 instances. We solved these 88 instances both for C-TOP-TDPLSF-nw and the C-TOP-TDPLSF-w.

For the ILS-noWait and the ILS-cWait-fin, the following parameters must be tuned: the multiplier ρ for determining the number $\lceil \rho |V_C| \rceil$ of perturbation moves to apply, the sorting strategy for the lists $\tilde{V_C}$ and L_i, the multiplier κ for determining the size $\lceil \kappa |V_C| \rceil$ of the generator arc set, and the number η of iterations without improvement. In addition, for the ILS-cWait-fin, also the number μ of best-found solutions stored in the pool must be tuned. An appropriate value for the multiplier ρ (i.e., $\rho = 0.1$) has been determined during the development of the algorithm. To avoid conducting full factorial parameter tuning experiments, we carry out the experiments with the aim of evaluating one parameter at a time, while keeping the other parameters constant. To ensure consistency, when setting the values of parameters that are common to both ILS-noWait and ILS-cWait-fin, we do not take into account the finalization phase for ILS-cWait-fin (i.e., $\mu = 0$).

We first set the parameters $\kappa = 0.1$ and $\eta = 50$, and we focus on assessing the most appropriate sorting strategy for the lists $\tilde{V_C}$ and L_i. Table 5.2 shows the results by ILS variant averaged over the 88 test instances. The columns of the table contain in order: the ILS variant, the value for the number μ of best-found solutions in the solution pool, the number η of iterations without improvement, the value of the multiplier κ for determining the generator arc set size, the sorting strategy for $\tilde{V_C}$ and L_i, the objective function values Θ^{best} and Θ^{avg} of the best and of the average run of our algorithm averaged over the 88 test instances, and the average runtime $\overline{t^a}(s)$ in seconds. The results show that, for ILS-noWait, the sorting strategy sorted $\tilde{V_C}$-sorted L_i leads to the highest collected score both in the best and average run, while runtimes are competitive to the ones obtained for the other sorting strategies. For ILS-cWait-fin, the sorting strategy sorted $\tilde{V_C}$-sorted L_i returns the highest collected score in the average run, while the sorting strategy random $\tilde{V_C}$-sorted L_i in the best run. However, considering that the sorting strategy random $\tilde{V_C}$-sorted L_i has the highest runtimes, we decide to fix the sorting strategy sorted $\tilde{V_C}$-sorted L_i for the following experiments.

ILS variant	μ	η	κ	$\tilde{V_C}$	L_i	$\overline{\Theta}^{best}$	$\overline{\Theta}^{avg}$	$\overline{t^a}(s)$
ILS-noWait	-	50	0.1	random	random	463.50	428.57	0.29
	-	50	0.1	sorted	random	468.52	434.13	0.24
	-	50	0.1	random	sorted	484.34	449.73	0.39
	-	50	0.1	sorted	sorted	486.21	460.18	0.29
ILS-cWait-fin	0	50	0.1	random	random	498.37	471.64	0.42
	0	50	0.1	sorted	random	499.69	473.54	0.33
	0	50	0.1	random	sorted	547.11	523.79	0.61
	0	50	0.1	sorted	sorted	546.61	530.68	0.49

Table 5.2: Results (averaged over the 88 test instances) of the parameter tuning experiments for determining the sorting strategy for the sets $\tilde{V_C}$ and L_i for the ILS-noWait and ILS-cWait-fin.

Next, we fix the sorting strategy sorted $\tilde{V_C}$-sorted L_i, $\eta = 50$, and we run ILS-noWait and ILS-cWait-fin for $\kappa = \{0.1, 0.3, 0.5, 1\}$ to determine the most appropriate value for the size of the generator arc set. Table 5.3 shows the results by problem type averaged over the 88 test instances and it is organized as Table 5.2. The results show that, for ILS-noWait, for increasing values of κ, the objective function values improve both in the best and in the average run, while the runtime increases. For

ILS-cWait-fin, the objective function value stops improving at $\kappa = 0.5$. To keep the runtimes limited while preserving solution quality, we fix $\kappa = 0.5$ for both problem types in the following experiments.

ILS variant	μ	η	κ	$\tilde{V_C}$	L_i	$\overline{\Theta}^{best}$	$\overline{\Theta}^{avg}$	$\overline{t^a}(s)$
ILS-noWait	-	50	0.1	sorted	sorted	486.21	460.18	0.29
	-	50	0.3	sorted	sorted	518.49	497.11	0.90
	-	50	0.5	sorted	sorted	525.33	505.26	1.45
	-	50	1.0	sorted	sorted	529.52	510.02	2.93
ILS-cWait-fin	0	50	0.1	sorted	sorted	546.61	530.68	0.49
	0	50	0.3	sorted	sorted	567.00	556.68	1.21
	0	50	0.5	sorted	sorted	569.12	559.63	1.81
	0	50	1.0	sorted	sorted	569.13	558.57	3.47

Table 5.3: Results (averaged over the 88 test instances for each problem type) of the parameter tuning experiments for determining the size of the generator arc set.

Then, we consider the sorting strategy sorted $\tilde{V_C}$-sorted L_i, $\kappa = 0.5$, and we run ILS-noWait and ILS-cWait-fin for increasing values of η until the objective function values stop to considerably improve. Table 5.4 shows the results by problem type averaged over the 88 test instances and it is organized as Table 5.2. The results show that, both for ILS-noWait and ILS-cWait-fin, by increasing the number η of iterations without improvement, the objective function values improve both in the best and in the average run. As expected, runtimes increase. However, by changing η from 1500 to 2000, the higher runtimes are not justified by the improvement in the objective function value. Hence, we fix $\eta = 1500$.

ILS variant	μ	η	κ	$\tilde{V_C}$	L_i	$\overline{\Theta}^{best}$	$\overline{\Theta}^{avg}$	$\overline{t^a}(s)$
ILS-noWait	-	50	0.5	sorted	sorted	525.33	505.26	1.45
	-	100	0.5	sorted	sorted	532.51	514.90	2.85
	-	500	0.5	sorted	sorted	544.24	532.20	13.87
	-	1000	0.5	sorted	sorted	548.52	536.68	27.19
	-	1500	0.5	sorted	sorted	551.17	540.07	40.75
	-	2000	0.5	sorted	sorted	551.98	541.75	52.33
ILS-cWait-fin	0	50	0.5	sorted	sorted	569.12	559.63	1.81
	0	100	0.5	sorted	sorted	572.81	564.74	3.56
	0	500	0.5	sorted	sorted	579.39	573.40	18.12
	0	1000	0.5	sorted	sorted	581.62	576.37	34.31
	0	1500	0.5	sorted	sorted	582.23	577.63	53.43
	0	2000	0.5	sorted	sorted	582.71	578.44	69.26

Table 5.4: Results (averaged over the 88 test instances for each problem type) of the parameter tuning experiments for determining the number of iterations η to consider.

Finally, we consider the sorting strategy sorted $\tilde{V_C}$-sorted L_i, $\kappa = 0.5$, $\eta = 1500$, and we run ILS-cWait-fin for increasing values of μ until the objective function values stop to considerably improve. Table 5.5 shows the results by problem type averaged over the 88 test instances and it is organized as Table 5.2. The results show that by increasing the number μ of best-found solutions stored in the solution pool, the objective function values improve both in the best and in the average run. As expected, runtimes increases. By changing μ from 200 to 300, the higher runtimes are not justified by the improvement in the objective function value. Hence, we fix $\mu = 200$.

ILS variant	μ	η	κ	$\tilde{V_C}$	L_i	$\overline{\Theta}^{best}$	$\overline{\Theta}^{avg}$	$\overline{t^a}(s)$
ILS-cWait-fin	0	1500	0.5	sorted	sorted	582.23	577.63	53.43
	1	1500	0.5	sorted	sorted	590.62	584.66	52.91
	5	1500	0.5	sorted	sorted	592.96	586.99	55.48
	10	1500	0.5	sorted	sorted	594.12	587.95	56.84
	50	1500	0.5	sorted	sorted	595.43	589.45	74.10
	100	1500	0.5	sorted	sorted	595.88	589.84	105.71
	200	1500	0.5	sorted	sorted	596.06	590.08	197.47
	300	1500	0.5	sorted	sorted	596.11	590.11	313.14

Table 5.5: Results (averaged over the 88 test instances for each problem type) of the parameter tuning experiments for determining the number of best-found solutions in the solution pool.

The values of the parameters for ILS-algo are summarized in Table 5.6.

Component	Parameter values
Multiplier for the number of perturbation moves	$\rho = 0.1$
Sorting strategy for $\tilde{V_C}$ and L_i	sorted $\tilde{V_C}$-sorted L_i
Multiplier for the size of the generator arc set	$\kappa = 0.5$
Number of iterations without improvement	$\eta = 1500$
Number of best-found solutions in solution pool (ILS-cWait-fin)	$\mu = 200$

Table 5.6: ILS-algo parameter values.

5.5.4 ILS-algo performance assessment

To assess the performance of ILS-algo, we compare the solutions of ILS-algo to the best solutions found by the commercial optimization solver Gurobi for the 66 small-scale instances described in Section 5.5.1.2.

Table 5.7 shows the comparison of the average results obtained with Gurobi and ILS-algo on the small-scale instances solved for C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w. The detailed results are reported in Tables 5.8 and 5.9. The columns of Table 5.7 contain: the problem type, the instance type, the Gurobi statistics related to the average percentage gap to the upper bound $\bar{\Delta}_{UB}(\%)$ and the average runtimes $\bar{t}(s)$ in seconds, and the statistics of the ILS-algo, i.e., the average percentage gap to the best solution found by Gurobi of the solutions obtained in the best and average run of ILS-algo ($\overline{\Delta}_{UB}^{best}(\%)$ and $\overline{\Delta}_{UB}^{avg}(\%)$), and the average runtimes ($\bar{t}^a(s)$) in seconds. The last three columns report the percentage of instances for which, in the best run, ILS-algo obtains a better, an equal, and a worse solution compared to the solution found by Gurobi.

Regarding Gurobi performance, Table 5.7 suggests that C-TOP-TDPLSF-nw is more difficult to solve for Gurobi than the C-TOP-TDPLSF-w as shown by the larger optimality gaps. Both for C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w, Cordeau instances are more difficult to solve for Gurobi as shown by larger optimality gaps and longer runtimes compared to those for Solomon instances.

For C-TOP-TDPLSF-nw, Table 5.7 shows that, in the best run, ILS-noWait obtains solutions of better quality than Gurobi for both the Cordeau and Solomon instances. In the average run, the solutions are of comparable quality. However, ILS-noWait terminates much faster than Gurobi. Moreover, compared to Gurobi, ILS-noWait finds solutions of better or equal quality for approximately 94% of instances.

For C-TOP-TDPLSF-w, Table 5.7 shows that ILS-cWait-fin returns high-quality solutions in fast runtimes both in the best and in the average run for the Cordeau instances. However, for the Solomon instances, despite ILS-cWait-fin finds solutions of better or equal quality than Gurobi for 39% of instances, there is potential for improvement in reducing the gaps.

		Gurobi		ILS-algo			Instances		
Problem type	Instance type	$\Delta_{UB}(\%)$	$\bar{t}(s)$	$\Delta^b_\Theta(\%)$	$\Delta^a_\Theta(\%)$	$\bar{t}^a(s)$	# better	# equal	# worse
	Cordeau	67.07	5466.17	-1.97	1.64	5.89	6/12	5/12	1/12
C-TOP-TDPLSF-nw	Solomon	32.21	3582.24	-0.14	0.04	0.73	6/54	45/54	3/54
	All	38.55	3924.77	-0.47	0.33	1.67	12/66	50/66	4/66
	Cordeau	54.05	5481.77	-0.86	-0.64	64.69	4/12	1/12	7/12
C-TOP-TDPLSF-w	Solomon	29.18	3555.49	2.53	2.60	38.59	4/54	19/54	31/54
	All	33.70	3905.72	1.91	2.01	43.34	8/66	20/66	38/66

Table 5.7: Comparison of results obtained with Gurobi and ILS-algo on the small-scale C-TOP-TDPLSF and C-TOP-TDPLSF-cw instances.

5.5.4.1 Effectiveness of ILS-cWait-fin with $\mu = 0$

We observed that ILS-cWait-fin does not find as many high-quality solutions for C-TOP-TDPLSF-w as ILS-noWait does for the C-TOP-TDPLSF-nw. The difference in results between Gurobi and ILS-cWait-fin for the C-TOP-TDPLSF-w may be due to: (i) the poorer performance of ILS-cWait-fin compared to ILS-noWait due to the introduction of the constrained waiting strategy, and (ii) the scarce effectiveness of the finalization phase in deciding waiting times compared to Gurobi's larger search space. To investigate the behavior of ILS-cWait-fin, we run ILS-cWait-fin with $\mu = 0$, i.e., without finalization phase, and compare its results to those obtained by Gurobi when solving a model for C-TOP-TDPLSF-w in which a constrained waiting strategy (corresponding to the one implemented in the ILS-cWait-fin) is allowed. We refer to this new problem “C-TOP-TDPLSF-cw”, and we provide its model in Appendix A. This analysis also determines whether the longer runtime of ILS-cWait-fin compared to ILS-noWait is due to the constrained waiting strategy or the finalization phase. Finally, ILS-cWait-fin without finalization phase serves as an independent heuristic that does not require the use of a commercial optimization solver.

Table 5.10 shows the comparison of the results obtained with Gurobi and ILS-cWait-fin without finalization phase (i.e., with $\mu = 0$) on the small-scale instances solved for C-TOP-TDPLSF-cw. The first five columns of the table contain the instance features. The following three columns contain the Gurobi statistics (objective function

					Gurobi			ILS-noWait		
Instance id	type	$\|K\|$	$\|N\|$	# periods	Θ	$\Delta_{UB}(\%)$	$t(s)$	$\Delta_{\Theta}^{best}(\%)$	$\Delta_{\Theta}^{avg}(\%)$	$t^a(s)$
pr04	Cordeau	1	96	4	157.95	116.63	7200.00	**-2.18**	**-2.15**	2.85
pr04	Cordeau	1	96	8	156.67	137.92	7200.00	**-8.00**	**-4.53**	3.77
pr04	Cordeau	2	96	4	299.58	120.98	7200.01	**-4.88**	**-3.67**	14.21
pr04	Cordeau	2	96	8	291.27	136.04	7200.01	**-0.92**	0.42	8.74
pr04	Cordeau	3	96	4	420.16	100.00	7200.00	**-6.73**	**-5.26**	18.65
pr04	Cordeau	3	96	8	418.00	105.96	7200.02	**-1.99**	2.02	15.28
pr07	Cordeau	1	36	4	122.52	0.00	16.00	**0.00**	7.39	0.47
pr07	Cordeau	1	36	8	117.76	0.00	74.09	**0.00**	3.86	0.47
pr07	Cordeau	2	36	4	198.15	0.00	703.92	**0.00**	10.92	1.24
pr07	Cordeau	2	36	8	201.12	20.87	7200.00	1.08	3.92	1.11
pr07	Cordeau	3	36	4	261.92	21.40	7200.00	**0.00**	5.23	1.72
pr07	Cordeau	3	36	8	275.52	44.97	7200.01	**0.00**	1.56	2.22
c101	Solomon	1	50	4	120.00	0.00	1.45	**0.00**	**0.00**	0.34
c101	Solomon	1	50	8	81.60	0.00	5.17	**0.00**	**0.00**	0.32
c101	Solomon	2	50	4	196.33	0.00	18.97	**0.00**	0.14	0.94
c101	Solomon	2	50	8	155.59	0.00	70.08	**0.00**	0.08	0.64
c101	Solomon	3	50	4	271.32	0.00	125.32	**0.00**	0.17	1.28
c101	Solomon	3	50	8	228.93	0.00	382.38	**0.00**	**0.00**	0.91
c107	Solomon	1	50	4	120.28	0.00	16.10	**0.00**	**0.00**	0.53
c107	Solomon	1	50	8	120.95	0.00	18.75	**0.00**	**0.00**	0.41
c107	Solomon	2	50	4	227.05	0.01	372.49	**0.00**	0.67	1.19
c107	Solomon	2	50	8	217.26	0.00	1719.29	**0.00**	**0.00**	0.83
c107	Solomon	3	50	4	318.49	9.09	7200.01	1.12	3.09	1.36
c107	Solomon	3	50	8	309.00	21.22	7200.00	0.33	0.43	1.49
c109	Solomon	1	50	4	146.98	0.00	149.66	**0.00**	**0.00**	0.48
c109	Solomon	1	50	8	135.57	0.00	713.21	**0.00**	**0.00**	0.50
c109	Solomon	2	50	4	277.30	11.18	7200.01	**0.00**	**0.00**	0.93
c109	Solomon	2	50	8	260.57	31.60	7200.00	**0.00**	0.17	1.03
c109	Solomon	3	50	4	363.14	20.59	7200.01	**-2.79**	**-2.79**	1.23
c109	Solomon	3	50	8	350.96	24.92	7200.01	**-1.12**	**-0.28**	1.42
r102	Solomon	1	50	4	59.44	0.00	76.01	**0.00**	**0.00**	0.37
r102	Solomon	1	50	8	69.46	0.00	415.69	**0.00**	**0.00**	0.40
r102	Solomon	2	50	4	112.30	0.00	2702.69	**0.00**	**0.00**	0.56
r102	Solomon	2	50	8	122.46	18.41	7200.00	**0.00**	**0.00**	0.64
r102	Solomon	3	50	4	163.93	77.58	7200.00	**0.00**	**0.00**	0.82
r102	Solomon	3	50	8	173.57	89.89	7200.01	**0.00**	0.59	1.22
r103	Solomon	1	50	4	119.52	0.00	206.93	**0.00**	**0.00**	0.36
r103	Solomon	1	50	8	122.33	0.00	598.13	**0.00**	**0.00**	0.41
r103	Solomon	2	50	4	192.92	36.88	7200.00	**0.00**	**0.00**	0.58
r103	Solomon	2	50	8	191.46	47.60	7200.00	**0.00**	**0.00**	0.96
r103	Solomon	3	50	4	253.02	62.57	7200.00	**0.00**	**0.00**	0.73
r103	Solomon	3	50	8	251.08	61.13	7200.00	**0.00**	2.32	1.23
r110	Solomon	1	50	4	90.10	0.00	539.77	**0.00**	**0.00**	0.40
r110	Solomon	1	50	8	84.47	0.00	1792.01	**0.00**	**0.00**	0.39
r110	Solomon	2	50	4	179.13	55.53	7200.01	**0.00**	**0.00**	0.86
r110	Solomon	2	50	8	158.42	143.09	7200.00	0.36	0.36	0.66
r110	Solomon	3	50	4	250.37	99.93	7200.00	**0.00**	0.68	0.93
r110	Solomon	3	50	8	218.87	147.99	7200.01	**-1.22**	**-0.65**	1.61
rc101	Solomon	1	50	4	69.31	0.00	2.25	**0.00**	**0.00**	0.30
rc101	Solomon	1	50	8	62.76	0.00	3.86	**0.00**	**0.00**	0.25
rc101	Solomon	2	50	4	121.31	0.00	10.40	**0.00**	**0.00**	0.54
rc101	Solomon	2	50	8	113.33	0.00	17.55	**0.00**	**0.00**	0.46
rc101	Solomon	3	50	4	167.31	0.00	50.35	**0.00**	**0.00**	0.70
rc101	Solomon	3	50	8	155.70	0.00	61.37	**0.00**	**0.00**	0.67
rc107	Solomon	1	50	4	86.19	0.00	149.97	**0.00**	**0.00**	0.34
rc107	Solomon	1	50	8	80.99	0.00	795.68	**0.00**	**0.00**	0.33
rc107	Solomon	2	50	4	158.02	19.13	7200.00	**0.00**	**0.00**	0.62
rc107	Solomon	2	50	8	159.99	85.91	7200.01	**0.00**	**0.00**	0.60
rc107	Solomon	3	50	4	212.86	76.06	7200.01	**-1.88**	**-1.05**	1.01
rc107	Solomon	3	50	8	219.99	89.40	7200.03	**-0.63**	**-0.56**	0.89
rc108	Solomon	1	50	4	90.00	0.01	426.11	**0.00**	**0.00**	0.32
rc108	Solomon	1	50	8	87.53	0.00	1999.26	**0.00**	**0.00**	0.32
rc108	Solomon	2	50	4	156.38	80.58	7200.00	**-1.54**	**-1.54**	0.65
rc108	Solomon	2	50	8	163.57	154.36	7200.00	**0.00**	**0.00**	0.58
rc108	Solomon	3	50	4	222.29	131.01	7200.00	**0.00**	0.11	0.81
rc108	Solomon	3	50	8	227.55	143.69	7200.01	**0.00**	**0.00**	0.91
Avg						38.55	3924.77	-0.47	0.33	1.67

Table 5.8: Comparison of results obtained with Gurobi and ILS-noWait on the small-scale C-TOP-TDPLSF-nw instances. Optimality has been proven for the underlined solutions. ILS-noWait solutions that are equal or improve Gurobi solutions are highlighted in boldface.

value Θ, the average percentage optimality gap $\Delta_{UB}(\%)$, and the runtime $t(s)$). The last three columns show the statistics of ILS-cWait-fin without finalization phase (percentage gap to Gurobi solution obtained in the best $\Delta_{\Theta}^{best}(\%)$ and average $\Delta_{\Theta}^{avg}(\%)$ run of ILS-cWait-fin without finalization, and the average runtime $t(s)$). The results show that ILS-cWait-fin without finalization phase achieves the same solution obtained with Gurobi for 45.45%, and finds new best-known solutions for 50.00% of instances.

					Gurobi			ILS-cWait-fin		
Instance id	type	$\lvert K\rvert$	$\lvert N\rvert$	# periods	Θ	$\Delta_{UB}(\%)$	$t(s)$	$\Delta_{\Theta}^{best}(\%)$	$\Delta_{\Theta}^{avg}(\%)$	$t^a(s)$
pr04	Cordeau	1	96	4	170.16	94.50	7200.00	**-1.05**	**-1.05**	76.49
pr04	Cordeau	1	96	8	165.86	140.15	7200.00	**-8.10**	**-7.87**	86.14
pr04	Cordeau	2	96	4	339.30	94.09	7200.00	1.65	1.80	105.77
pr04	Cordeau	2	96	8	320.02	116.54	7200.02	**-5.64**	**-5.64**	102.11
pr04	Cordeau	3	96	4	463.93	80.38	7200.02	**-5.23**	**-4.19**	127.64
pr04	Cordeau	3	96	8	499.08	72.51	7200.02	2.27	3.04	135.00
pr07	Cordeau	1	36	4	122.94	$\underline{0.00}$	15.48	**0.00**	**0.00**	15.68
pr07	Cordeau	1	36	8	137.31	$\underline{0.00}$	77.63	0.66	0.66	16.21
pr07	Cordeau	2	36	4	215.98	0.01	888.00	1.42	1.69	24.46
pr07	Cordeau	2	36	8	236.30	5.78	7200.01	1.06	1.06	23.95
pr07	Cordeau	3	36	4	287.40	19.54	7200.00	2.61	2.77	36.18
pr07	Cordeau	3	36	8	323.56	25.14	7200.00	0.02	0.02	26.64
c101	Solomon	1	50	4	130.02	$\underline{0.00}$	0.57	0.03	0.03	27.15
c101	Solomon	1	50	8	130.08	$\underline{0.00}$	1.11	15.40	15.41	27.52
c101	Solomon	2	50	4	250.02	$\underline{0.00}$	2.18	**0.00**	**0.00**	48.11
c101	Solomon	2	50	8	239.95	$\underline{0.00}$	9.26	8.29	8.29	36.38
c101	Solomon	3	50	4	350.08	$\underline{0.00}$	15.78	3.37	4.14	54.79
c101	Solomon	3	50	8	340.76	$\underline{0.00}$	46.49	9.01	9.01	42.77
c107	Solomon	1	50	4	138.71	$\underline{0.00}$	10.26	**0.00**	**0.00**	27.96
c107	Solomon	1	50	8	131.53	$\underline{0.00}$	31.35	1.57	1.57	28.50
c107	Solomon	2	50	4	259.61	$\underline{0.00}$	267.81	3.69	3.69	40.45
c107	Solomon	2	50	8	261.57	$\underline{0.00}$	1357.14	3.83	4.83	37.87
c107	Solomon	3	50	4	369.61	0.01	5713.40	4.66	5.50	44.35
c107	Solomon	3	50	8	368.23	13.25	7200.00	4.49	4.49	44.67
c109	Solomon	1	50	4	172.63	$\underline{0.00}$	102.46	11.59	11.59	28.97
c109	Solomon	1	50	8	175.95	$\underline{0.00}$	476.27	6.38	6.38	34.48
c109	Solomon	2	50	4	292.60	14.46	7200.01	3.41	3.41	44.22
c109	Solomon	2	50	8	296.01	27.58	7200.00	**0.00**	**0.00**	40.16
c109	Solomon	3	50	4	388.39	22.41	7200.01	**-0.94**	**-0.94**	42.60
c109	Solomon	3	50	8	405.86	17.45	7200.01	0.89	0.89	42.39
r102	Solomon	1	50	4	72.00	$\underline{0.00}$	154.59	3.31	3.31	26.40
r102	Solomon	1	50	8	72.00	$\underline{0.00}$	444.80	**0.00**	**0.00**	26.35
r102	Solomon	2	50	4	140.00	$\underline{0.00}$	3691.15	6.70	6.70	39.20
r102	Solomon	2	50	8	141.46	34.39	7200.01	**0.00**	**0.00**	38.10
r102	Solomon	3	50	4	201.00	58.86	7200.00	3.27	3.27	43.84
r102	Solomon	3	50	8	200.47	72.44	7200.00	2.55	2.55	45.65
r103	Solomon	1	50	4	122.00	$\underline{0.00}$	286.90	**0.00**	**0.00**	26.90
r103	Solomon	1	50	8	122.00	$\underline{0.00}$	623.59	**0.00**	**0.00**	26.83
r103	Solomon	2	50	4	201.01	37.60	7200.00	**0.00**	**0.00**	38.73
r103	Solomon	2	50	8	193.24	48.60	7200.00	**-0.14**	**-0.14**	41.16
r103	Solomon	3	50	4	274.06	50.69	7200.01	**0.00**	**0.00**	46.21
r103	Solomon	3	50	8	259.34	60.72	7200.00	**-0.07**	0.90	46.11
r110	Solomon	1	50	4	94.00	$\underline{0.00}$	573.18	4.15	4.18	37.46
r110	Solomon	1	50	8	90.68	$\underline{0.00}$	1879.28	**0.00**	**0.00**	34.39
r110	Solomon	2	50	4	184.10	60.28	7200.00	2.30	2.30	44.65
r110	Solomon	2	50	8	175.32	112.39	7200.00	1.15	1.15	43.29
r110	Solomon	3	50	4	257.83	92.26	7200.00	2.23	2.41	52.35
r110	Solomon	3	50	8	247.32	123.05	7200.00	4.36	4.36	46.94
rc101	Solomon	1	50	4	90.00	$\underline{0.00}$	1.17	**0.00**	**0.00**	26.89
rc101	Solomon	1	50	8	90.00	$\underline{0.00}$	1.07	**0.00**	**0.00**	26.38
rc101	Solomon	2	50	4	160.01	$\underline{0.00}$	7.49	**0.00**	**0.00**	37.45
rc101	Solomon	2	50	8	160.00	$\underline{0.00}$	8.83	**0.00**	**0.00**	37.52
rc101	Solomon	3	50	4	219.35	$\underline{0.00}$	18.67	**0.00**	**0.00**	43.69
rc101	Solomon	3	50	8	220.00	$\underline{0.00}$	43.47	3.03	3.03	43.81
rc107	Solomon	1	50	4	93.31	$\underline{0.00}$	155.08	**0.00**	**0.00**	27.35
rc107	Solomon	1	50	8	97.01	$\underline{0.00}$	649.91	9.83	9.83	27.84
rc107	Solomon	2	50	4	171.71	16.94	7200.00	3.83	3.83	40.85
rc107	Solomon	2	50	8	167.01	83.61	7200.00	2.23	2.23	42.52
rc107	Solomon	3	50	4	232.01	68.91	7200.00	3.43	3.43	50.75
rc107	Solomon	3	50	8	240.01	76.51	7200.00	2.80	2.80	46.34
rc108	Solomon	1	50	4	90.00	$\underline{0.00}$	495.07	**0.00**	**0.00**	27.11
rc108	Solomon	1	50	8	91.71	$\underline{0.00}$	2127.83	4.06	4.06	27.63
rc108	Solomon	2	50	4	160.19	98.97	7200.00	**0.00**	**0.00**	43.68
rc108	Solomon	2	50	8	179.69	127.41	7200.00	2.62	2.62	41.55
rc108	Solomon	3	50	4	229.53	132.86	7200.00	**0.00**	**0.00**	49.20
rc108	Solomon	3	50	8	249.70	123.82	7200.00	**-0.38**	**-0.38**	45.62
Avg						33.70	3905.72	1.91	2.01	43.34

Table 5.9: Comparison of results obtained with Gurobi and ILS-cWait-fin on the small-scale C-TOP-TDPLSF-w instances. Optimality has been proven for the underlined solutions. ILS-cWait-fin solutions that are equal or improve Gurobi solutions are highlighted in boldface.

Moreover, ILS-cWait-fin without finalization terminates much faster than Gurobi. Hence, ILS-cWait-fin without finalization phase returns a good solution quality in fast runtimes for C-TOP-TDPLSF-cw instances.

We make the following two observations regarding the performance of ILS-cWait-fin for solving the C-TOP-TDPLSF-w. First, the difference in results between Gurobi and ILS-cWait-fin for the C-TOP-TDPLSF-w (see Table 5.9) is due to the lack of flexibility

					Gurobi			ILS-cWait-fin $\mu = 0$		
Instance id	type	$\|K\|$	$\|N\|$	# periods	Θ	$\Delta_{UB}(\%)$	$t(s)$	$\Delta_{\Theta}^{best}(\%)$	$\Delta_{\Theta}^{avg}(\%)$	$t^a(s)$
pr04	Cordeau	1	96	4	153.01	188.24	7200.04	**-11.52**	**-11.52**	3.18
pr04	Cordeau	1	96	8	162.00	189.87	7200.01	**-8.59**	**-8.59**	4.93
pr04	Cordeau	2	96	4	317.93	127.01	7200.03	**-4.17**	**-4.17**	9.07
pr04	Cordeau	2	96	8	301.01	139.78	7200.04	**-10.36**	**-10.36**	11.74
pr04	Cordeau	3	96	4	471.74	82.48	7200.01	**-2.74**	**-1.69**	16.19
pr04	Cordeau	3	96	8	400.26	115.06	7200.01	**-18.83**	**-17.10**	24.43
pr07	Cordeau	1	36	4	122.52	$\underline{0.00}$	86.47	**0.00**	**0.00**	0.44
pr07	Cordeau	1	36	8	121.62	$\underline{0.00}$	1033.71	**0.00**	**0.00**	0.68
pr07	Cordeau	2	36	4	211.00	37.61	7200.00	**0.00**	0.42	1.06
pr07	Cordeau	2	36	8	210.27	48.93	7200.00	**-0.65**	**-0.65**	1.19
pr07	Cordeau	3	36	4	280.35	50.59	7200.02	1.78	1.90	2.02
pr07	Cordeau	3	36	8	291.53	44.21	7200.00	**-1.19**	**-1.19**	1.50
c101	Solomon	1	50	4	130.00	$\underline{0.00}$	2.21	0.02	0.02	0.48
c101	Solomon	1	50	8	109.94	$\underline{0.00}$	7.41	**0.00**	**0.00**	0.57
c101	Solomon	2	50	4	224.84	$\underline{0.00}$	29.15	**0.00**	**0.00**	1.11
c101	Solomon	2	50	8	199.97	$\underline{0.00}$	372.21	**0.00**	**0.00**	1.04
c101	Solomon	3	50	4	308.24	$\underline{0.00}$	429.46	**0.00**	**0.00**	1.85
c101	Solomon	3	50	8	269.28	18.72	7200.06	**0.00**	**0.00**	1.56
c107	Solomon	1	50	4	125.30	$\underline{0.00}$	48.48	**0.00**	**0.00**	0.57
c107	Solomon	1	50	8	128.91	$\underline{0.00}$	89.47	**0.00**	**0.00**	0.74
c107	Solomon	2	50	4	233.53	6.74	7200.01	**-0.75**	**-0.75**	1.24
c107	Solomon	2	50	8	230.53	12.53	7200.00	**-0.73**	**-0.73**	1.53
c107	Solomon	3	50	4	329.42	35.93	7200.03	**-0.31**	**-0.31**	2.02
c107	Solomon	3	50	8	325.36	33.34	7200.00	**-1.68**	**-1.68**	2.11
c109	Solomon	1	50	4	146.98	$\underline{0.00}$	3006.77	**0.00**	**0.00**	0.59
c109	Solomon	1	50	8	138.19	36.16	7200.01	**0.00**	**0.00**	0.76
c109	Solomon	2	50	4	277.30	33.35	7200.00	**0.00**	**0.00**	1.23
c109	Solomon	2	50	8	251.13	51.68	7200.00	**-6.51**	**-6.51**	1.38
c109	Solomon	3	50	4	360.85	32.22	7200.01	**-3.44**	**-3.44**	1.84
c109	Solomon	3	50	8	361.85	31.86	7200.00	**-1.39**	**-1.39**	1.99
r102	Solomon	1	50	4	67.00	$\underline{0.00}$	882.70	**0.00**	**0.00**	0.53
r102	Solomon	1	50	8	69.46	$\underline{0.00}$	1548.35	**0.00**	**0.00**	0.52
r102	Solomon	2	50	4	128.00	103.20	7200.00	**0.00**	**0.00**	1.07
r102	Solomon	2	50	8	129.96	105.20	7200.05	**0.00**	**0.00**	1.06
r102	Solomon	3	50	4	184.26	98.00	7200.00	**-0.11**	**-0.11**	1.50
r102	Solomon	3	50	8	181.50	101.60	7200.00	**-1.29**	**-1.29**	1.70
r103	Solomon	1	50	4	119.52	$\underline{0.00}$	2010.53	**0.00**	**0.00**	0.49
r103	Solomon	1	50	8	122.00	20.54	7200.01	**-0.27**	**-0.27**	0.59
r103	Solomon	2	50	4	185.28	74.35	7200.01	**-4.12**	**-4.12**	0.86
r103	Solomon	2	50	8	191.13	77.56	7200.00	**-0.17**	**-0.17**	1.20
r103	Solomon	3	50	4	253.02	71.40	7200.00	**-0.74**	**-0.74**	1.21
r103	Solomon	3	50	8	246.13	76.19	7200.02	**-2.01**	**-2.01**	1.76
r110	Solomon	1	50	4	90.10	$\underline{0.00}$	6976.73	**0.00**	0.03	0.71
r110	Solomon	1	50	8	84.47	161.57	7200.01	**0.00**	**0.00**	0.72
r110	Solomon	2	50	4	174.14	145.75	7200.03	**-2.87**	**-2.87**	1.28
r110	Solomon	2	50	8	157.85	191.60	7200.00	**-0.46**	**-0.46**	1.21
r110	Solomon	3	50	4	246.27	129.27	7200.01	**-1.66**	**-1.45**	1.97
r110	Solomon	3	50	8	213.94	165.36	7200.00	**-3.26**	**-3.26**	1.86
rc101	Solomon	1	50	4	90.00	$\underline{0.00}$	4.02	**0.00**	**0.00**	0.36
rc101	Solomon	1	50	8	70.00	$\underline{0.00}$	7.63	**0.00**	**0.00**	0.45
rc101	Solomon	2	50	4	160.01	$\underline{0.00}$	13.68	**0.00**	**0.00**	0.72
rc101	Solomon	2	50	8	132.96	$\underline{0.00}$	106.63	**0.00**	**0.00**	0.90
rc101	Solomon	3	50	4	214.69	$\underline{0.00}$	113.48	**0.00**	**0.00**	1.11
rc101	Solomon	3	50	8	186.29	0.01	2900.28	2.68	2.68	1.51
rc107	Solomon	1	50	4	86.19	$\underline{0.00}$	1148.27	**0.00**	**0.00**	0.46
rc107	Solomon	1	50	8	87.00	$\underline{0.00}$	4751.50	**0.00**	**0.00**	0.46
rc107	Solomon	2	50	4	158.02	92.87	7200.01	**0.00**	**0.00**	0.94
rc107	Solomon	2	50	8	156.38	116.51	7200.00	**-2.31**	**-2.31**	0.99
rc107	Solomon	3	50	4	214.04	103.69	7200.01	**-1.32**	**-1.32**	1.76
rc107	Solomon	3	50	8	220.86	108.39	7200.00	**-0.23**	**-0.23**	1.78
rc108	Solomon	1	50	4	90.00	50.97	7200.00	**0.00**	**0.00**	0.42
rc108	Solomon	1	50	8	87.53	158.70	7200.01	**0.00**	**0.00**	0.51
rc108	Solomon	2	50	4	153.50	186.23	7200.00	**-3.45**	**-3.45**	0.89
rc108	Solomon	2	50	8	139.81	231.35	7200.00	**-16.99**	**-16.99**	1.02
rc108	Solomon	3	50	4	219.88	167.47	7200.00	**-1.10**	**-1.10**	1.32
rc108	Solomon	3	50	8	227.34	153.22	7200.00	**-0.59**	**-0.59**	1.60
Avg						63.75	5187.42	-1.69	-1.63	2.07

Table 5.10: Comparison of results obtained with Gurobi and ILS-cWait-fin without finalization phase (i.e., $\mu = 0$) on the small-scale C-TOP-TDPLSF-cw instances. Optimality has been proven for the underlined solutions. ILS-cWait-fin without finalization solutions that are equal or improve Gurobi solutions are highlighted in boldface.

in determining more appropriate visiting times at customer locations. Second, because the runtimes of ILS-cWait-fin without finalization phase are comparable to those of ILS-noWait (see Table 5.8), the longer runtimes of ILS-cWait-fin reported in Table 5.9 are primarily due to the execution of the finalization phase.

5.5.4.2 Assessing the effectiveness of the finalization phase

To evaluate the effectiveness of the finalization phase, we solve C-TOP-TDPLSF-w using both ILS-cWait-fin with $\mu = 0$ and ILS-cWait-fin with the finalization phase. Then, we compare these results to those obtained by Gurobi for the C-TOP-TDPLSF-w.

Table 5.11 shows the results obtained with ILS-cWait-fin without finalization phase ($\mu = 0$) and with finalization phase when solving the small-scale C-TOP-TDPLSF-w instances. The detailed results are presented in Table 5.18 in Appendix B. For each algorithmic variant, Table 5.11 displays the average percentage gap obtained in the best (Δ_{Θ}^{best}) and average (Δ_{Θ}^{avg}) run of the algorithm to the upper bound provided by Gurobi, and the average runtime ($t^a(s)$) in seconds. The results show that the finalization phase is very effective: the average percentage gaps obtained in the best and average run of the algorithm are decreased by approximately five percentage points when ILS-cWait-fin includes the finalization phase. However, this comes at the expense of higher runtimes.

	ILS-cWait-fin $\mu = 0$			ILS-cWait-fin		
	$\Delta_{\Theta}^{best}(\%)$	$\Delta_{\Theta}^{avg}(\%)$	$t^a(s)$	$\Delta_{\Theta}^{best}(\%)$	$\Delta_{\Theta}^{avg}(\%)$	$t^a(s)$
C-TOP-TDPLSF-w	6.81	6.86	2.07	1.91	2.01	43.34

Table 5.11: Comparison of the average results obtained with ILS-cWait-fin without finalization phase ($\mu = 0$) and with finalization phase when solving the small-scale C-TOP-TDPLSF-w instances.

5.5.4.3 Assessing the effectiveness of our constrained waiting strategy

In the literature of TOPTW with allowed waiting, heuristics employ the waiting strategy in which a vehicle delays its visit to a customer until the opening time of the customer's time window if it arrives early. This approach is based on the assumption that waiting beyond the opening time does not increase the collected score and only consumes route duration. However, in our problem, the scores within less preferred time windows are determined by a piecewise linear function with slope different from zero. Consequently, simply waiting for the next time window's opening may yield a lower score compared to visiting at a better point within that time window.

In this section, we evaluate the effectiveness of using a more sophisticated constrained waiting strategy (i.e., the one included in ILS-cWait-fin) rather than merely waiting for the upcoming time window's opening. We solve the C-TOP-TDPLSF-w small-scale instances using both ILS-cWait-fin and a modified version called ILS-cWait-fin-simple, in which waiting is only allowed until the opening of the upcoming time window.

The detailed results of this comparison are presented in Table 5.19 in Appendix B, while Table 5.12 shows the averaged results. For each method, Table 5.12 displays the average percentage gap obtained in the best (Δ_{Θ}^{best}) and average (Δ_{Θ}^{avg}) run of the algorithm to the best-found solution by Gurobi. The results show that considering a more sophisticated waiting strategy is fundamental for obtaining high-quality solutions for the C-TOP-TDPLSF-w. In fact, with ILS-cWait-fin, the average percentage gaps obtained in the best and average run of the algorithm decrease by approximately 1.4 percentage points compared to those achieved with ILS-cWait-fin-simple. Even though a more sophisticated waiting strategy may lead to longer route durations because of longer waiting times and hence, to fewer visited customers, it allows to collect higher scores. This suggests that algorithms from the extant literature will likely fail to deliver high-quality solutions for the instances of C-TOP-TDPLSF-w due to their reliance on a too simple waiting strategy.

	ILS-cWait-fin-simple		ILS-cWait-fin	
	$\Delta_{\Theta}^{best}(\%)$	$\Delta_{\Theta}^{avg}(\%)$	$\Delta_{\Theta}^{best}(\%)$	$\Delta_{\Theta}^{avg}(\%)$
C-TOP-TDPLSF-w	3.22	3.47	1.91	2.01

Table 5.12: Comparison of the average results obtained with ILS-cWait-fin in which waiting is only allowed until the opening of the upcoming time window (ILS-cWait-fin-simple) and ILS-algo (i.e., in which a more elaborated waiting strategy is implemented) when solving the small-scale C-TOP-TDPLSF-w instances.

5.5.5 Managerial insights

In this section, we derive managerial insights on the impact of the time window width, and the number of vehicles on the solutions for the C-TOP-TDPLSF-nw (Section 5.5.5.1) and the C-TOP-TDPLSF-w (Section 5.5.5.2). Moreover, in Section 5.5.5.3, we evaluate the value of waiting by comparing the solutions obtained for the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w. For our analysis, we solve the large-scale instances described in Section 5.5.1.1 by using ILS-noWait for the C-TOP-TDPLSF-nw, and the ILS-cWait-fin for the C-TOP-TDPLSF-w. We consider the solutions returned by the best and the average run of the ILS-noWait and ILS-cWait-fin to compute the following metrics:

- Θ^{best}: objective function value of the best run.
- Θ^{avg}: objective function value of the average run.
- $t^a(s)$: average runtime in seconds.
- cust visited(%): number of visited customers computed as a percentage of the total number of customers.

- cust visited P(%): number of customers visited in a preferred time window computed as a percentage of the number of visited customers.
- cust visited LP(%): number of customers visited in a less preferred time window computed as a percentage of the number of visited customers.
- avg load(%): vehicle load when leaving the depot computed as an average over the number of vehicles and expressed as a percentage of the vehicle capacity.
- avg duration(%): tour duration computed as an average over the number of vehicles and expressed as a percentage of the maximum tour duration.
- waiting cust(%) (computed only for the C-TOP-TDPLSF-w): number of customers at which vehicles wait computed as a percentage of the number of visited customers.
- waiting time(%) (computed only for the C-TOP-TDPLSF-w): percentage of the overall tour duration that is spent on waiting.

In the following sections, we report the value of these metrics averaged by instance type. The detailed values of these metrics by instance for the C-TOP-TDPLSF-nw and the C-TOP-TDPLSF-w are reported in Appendix C.

5.5.5.1 Managerial insights for the solutions of C-TOP-TDPLSF-nw

Table 5.13 shows the results of the above metrics for instances with four and eight periods. We recall that the larger the number of periods is, the tighter the time windows are. The results show that when time windows are tight, the average collected score is lower than when time windows are wide due to a limitation of the flexibility of the routing operations. The reduction of the collected score is due to a lower percentage of visited customers, resulting in a higher inefficiency caused by lower vehicle loads and shorter tour durations that leave part of the drivers' working hours unused. Despite of the fewer visited customers, a larger number of them is visited in a preferred time window. Hence, tighter time windows allow for better adherence to preferred time windows.

periods	Θ^{best}	Θ^{avg}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)
4	568.60	558.00	56.57	25.11	64.23	35.77	79.97	88.39
8	567.58	553.12	61.07	24.75	66.81	33.19	79.29	88.26

Table 5.13: Metrics averaged by 4-period and 8-period instances for the C-TOP-TDPLSF-nw solved with ILS-noWait.

Table 5.14 shows the results of the above metrics for instances with increasing number of vehicles. The results show that the collected score increases when more vehicles are available due to more customers being visited. Hence, the customer service level also increases. Despite of visiting more customers, the number of those served within a preferred time window decreases. More vehicles also correspond to a lower vehicle load utilization and shorter tour duration.

$\|K\|$	Θ^{best}	Θ^{avg}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)
1	265.03	260.29	9.03	10.70	66.10	33.90	80.50	88.99
2	487.48	477.46	37.64	20.49	65.48	34.52	79.68	88.58
3	677.27	662.67	78.15	29.74	65.57	34.43	78.62	88.06
4	842.57	821.84	110.47	38.77	64.93	35.07	79.72	87.68

Table 5.14: Metrics averaged by vehicle number for the C-TOP-TDPLSF-nw solved with ILS-noWait.

For the C-TOP-TDPLSF-nw in which waiting is not allowed, our results suggest the following:

- When deciding the width of the time windows to offer to their customers, practitioners should consider the tradeoff between: (i) visiting fewer customers and being more inefficient but with a better adherence to their preferred time windows, or (ii) visiting more customers and being more efficient but with a worse adherence to their preferred time windows.

- Enlarging the vehicle fleet increases the percentage of served customers by approximately ten percentage points when an additional vehicle is added, although the increase tends to diminish as more vehicles are used.

- With larger vehicle fleets, the driver's working hours that remain unused increase by one percentage point. Practitioners should consider assigning drivers alternative tasks or modifying working contracts.

- Offering wider time windows or increasing the number of vehicles leads to a worse adherence to the preferred time windows which, in the long term, can compromise the perceived service quality.

5.5.5.2 Managerial insights for the solutions of C-TOP-TDPLSF-w

Table 5.15 shows the results of the above metrics for instances with four and eight periods when waiting is allowed. The results show that, when time windows are tight, the average collected score is higher than when time windows are wide. The increase in the collected score is not due to more visited customers but to a higher percentage of customers served in their preferred time windows. With tighter time windows, vehicle loads and tour durations also increase. Tighter time windows imply higher waiting times and more customers at which drivers have to wait.

Table 5.16 shows the results of the above metrics for instances with increasing number of vehicles when waiting is allowed. The results show that as the number of

periods	Θ^{best}	Θ^{avg}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)	waiting cust(%)	waiting time(%)
4	603.41	598.11	208.82	26.45	60.88	39.12	83.14	89.87	6.57	5.10
8	613.89	606.99	220.17	26.40	61.70	38.30	83.80	90.08	8.01	5.24

Table 5.15: Metrics averaged by 4-period and 8-period instances for the C-TOP-TDPLSF-w solved with ILS-cWait-fin.

available vehicles increases, the collected score also increases due to both more visited customers and a higher percentage of them served in a preferred time window. With more vehicles, each driver completes their tour faster. By increasing the number of vehicles, the total waiting time first increases and then becomes stable. Moreover, with a higher number of vehicles, waiting is needed at more customer locations.

$\lvert K\rvert$	Θ^{best}	Θ^{avg}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)	waiting cust(%)	waiting time(%)
1	283.33	281.93	104.14	11.38	60.34	39.66	84.90	90.56	3.07	4.65
2	521.23	516.77	176.35	21.57	60.66	39.34	82.69	90.33	6.17	4.93
3	726.82	718.58	243.62	31.56	61.27	38.73	82.92	89.79	8.66	5.56
4	903.22	892.92	333.88	41.18	62.89	37.11	83.36	89.22	11.27	5.54

Table 5.16: Metrics averaged by vehicle number for the C-TOP-TDPLSF-w solved with ILS-cWait-fin.

For the C-TOP-TDPLSF-w in which waiting is allowed, our results suggest the following:

- Practitioners should offer tight time windows to their customers because: (i) Higher scores can be achieved with less visited customers, (ii) A better service level is provided to those visited customers, and (iii) A better utilization of vehicle capacity and drivers' working hours is achieved.
- For every additional vehicle, the percentage of served customers increases by approximately ten percentage points, although the increase tends to diminish as more vehicles are used.
- Offering tight time windows or increasing the number of vehicles leads to increased waiting times. Practitioners should consider assigning drivers alternative tasks during waiting periods or training drivers to handle waiting times productively. With more vehicles, drivers also complete their routes faster. Practitioners should consider assigning drivers additional deliveries within the same work shift.

5.5.5.3 The value of waiting

Table 5.17 shows the results of the metrics for the C-TOP-TDPLSF-nw solved by ILS-noWait, and C-TOP-TDPLSF-w solved by ILS-cWait-fin. The results show that, when waiting is allowed, the collected score increases due to more visited customers, which imply increased capacity utilization and tour duration. Despite of more visited customers, the number of those served in a preferred time window decreases.

Problem	Θ^{best}	Θ^{avg}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)
C-TOP-TDPLSF-nw	568.09	555.56	58.82	24.93	65.52	34.48	79.63	88.33
C-TOP-TDPLSF-w	608.65	602.55	214.50	26.42	61.29	38.71	83.47	89.97

Table 5.17: Metrics averaged by problem type for the C-TOP-TDPLSF-nw solved with ILS-noWait, and the C-TOP-TDPLSF-w solved with ILS-cWait-fin.

The comparison of the results obtained for C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w suggests the following:

- Allowing waiting at customer locations increases scores due to more visited customers.
- Practitioners should be aware of a worse adherence to the preferred time windows which, in the long term, can compromise the perceived service quality.

5.6 Conclusion

In this paper, we introduce the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w, two practically relevant problems that represent customers' preferences accurately. We propose their mathematical formulations and effective preprocessing techniques. To solve them in reasonable runtimes, we design two ILS variants, namely ILS-noWait and ILS-cWait-fin, that differ in how the waiting decisions are evaluated in the local search phase. Both algorithms show a good performance, even though ILS-cWait-fin struggles to consistently determine good visiting times. Moreover, including in the algorithm a more complex waiting strategy than the one usually implemented in the literature is vital for obtaining C-TOP-TDPLSF-w high-quality solutions. Our results show that wider time windows or a large number of vehicles lead to worse adherence to the preferred time windows in urban contexts in which waiting is not allowed. When waiting is allowed, the same effect is observed but when time windows are tight.

References

Y. Amarouche, R. N. Guibadj, E. Chaalal, and A. Moukrim (2020). "Effective neighborhood search with optimal splitting and adaptive memory for the team orienteering problem with time windows". In: *Computers & Operations Research* 123. Article ID 105039. DOI: 10.1016/j.cor.2020.105039.

C. Archetti, N. Bianchessi, and M. G. Speranza (2013). "Optimal solutions for routing problems with profits". In: *Discrete Applied Mathematics* 161.4-5, pp. 547–557. DOI: 10.1016/j.dam.2011.12.021.

C. Archetti, M. G. Speranza, and D. Vigo (2014). “Chapter 10: Vehicle Routing Problems with Profits”. In: *Vehicle Routing.* Ed. by P. Toth and D. Vigo. Philadelphia, PA: Society for Industrial and Applied Mathematics, pp. 273–297. ISBN: 978-1-61197-358-7. DOI: 10.1137/1.9781611973594.ch10.

C. Archetti, D. Feillet, A. Hertz, and M. G. Speranza (2009). “The capacitated team orienteering and profitable tour problems”. In: *The Journal of the Operational Research Society* 60.6, pp. 831–842. DOI: 10.1057/palgrave.jors.2602603.

C. Archetti, N. Bianchessi, S. Irnich, and M. G. Speranza (2014). “Formulations for an inventory routing problem”. In: *International Transactions in Operational Research* 21.3, pp. 353–374. DOI: 10.1111/itor.12076.

E. Barrena, D. Canca, L. C. Coelho, and G. Laporte (2023). “Analysis of the selective traveling salesman problem with time-dependent profits”. In: *TOP* 31, pp. 165–193. DOI: 10.1007/s11750-022-00632-6.

R. Cavagnini, M. Schneider, and A. Theiß (2024). “A granular iterated local search for the asymmetric single truck and trailer routing problem with satellite depots at DHL Group”. In: *Networks* 83.1, pp. 3–29. DOI: 10.1002/net.22178.

I.-M. Chao, B. L. Golden, and E. A. Wasil (1996). “The team orienteering problem”. In: *European Journal of Operational Research* 88.3, pp. 464–474. DOI: 10.1016/0377-2217(94)00289-4.

J.-F. Cordeau, M. Gendreau, and G. Laporte (1997). “A tabu search heuristic for periodic and multi-depot vehicle routing problems”. In: *Networks* 30.2, pp. 105–119. DOI: 10.1002/(SICI)1097-0037(199709)30:2%3C105::AID-NET5%3E3.0.CO;2-G.

M. Darvish, L. C. Coelho, and R. Jans (2020). *Comparison of symmetry breaking and input ordering techniques for routing problems.* FSA-2020-008. CIRRELT.

A. Ekici and A. Retharekar (2013). “Multiple agents maximum collection problem with time dependent rewards”. In: *Computers & Industrial Engineering* 64.4, pp. 1009–1018. DOI: 10.1016/j.cie.2013.01.010.

G. Erdoğan and G. Laporte (2013). “The orienteering problem with variable profits”. In: *Networks* 61.2, pp. 104–116. DOI: 10.1002/net.21496.

E. Erkut and J. Zhang (1996). “The maximum collection problem with time-dependent rewards”. In: *Naval Research Logistics* 43.5, pp. 749–763. DOI: 10.1002/(SICI)1520-6750(199608)43:5%3C749::AID-NAV10%3E3.0.CO;2-J.

J. W. Escobar, R. Linfati, and P. Toth (2014). “A hybrid granular tabu search algorithm for the multi-depot vehicle routing problem”. In: *Journal of the Operational Research Society* 65.1, pp. 37–48. DOI: 10.1057/jors.2013.102.

D. Feillet, P. Dejax, and M. Gendreau (2005). “Traveling Salesman Problems with Profits”. In: *Transportation Science* 39.2, pp. 188–205. DOI: 10.1287/trsc.1030.0079.

D. Gavalas, C. Konstantopoulos, K. Mastakas, and G. Pantziou (2014). “A survey on algorithmic approaches for solving tourist trip design problems”. In: *Journal of Heuristics* 20.3, pp. 291–328. DOI: 10.1007/s10732-014-9242-5.

D. Gavalas, C. Konstantopoulos, K. Mastakas, and G. Pantziou (2019). “Efficient Cluster-Based Heuristics for the Team Orienteering Problem with Time Windows”. In: *Asia-Pacific Journal of Operational Research* 36.01. DOI: 10.1142/S0217595919500015.

D. Goeke (2019). “Granular tabu search for the pickup and delivery problem with time windows and electric vehicles”. In: *European Journal of Operational Research* 278.3, pp. 821–836. DOI: 10.1016/j.ejor.2019.05.010.

R. N. Guibadj and A. Moukrim (2014). “Memetic algorithm with an efficient split procedure for the Team Orienteering Problem with Time Windows”. In: *Artificial Evolution.* Ed. by P. Legrand, M.-M. Corsini, J.-K. Hao, N. Monmarché, E. Lutton, and M. Schoenauer, pp. 183–194. DOI: 10.1007/978-3-319-11683-9_15.

A. Gunawan, H. C. Lau, P. Vansteenwegen, and K. Lu (2017). “Well-tuned algorithms for the team orienteering problem with time windows”. In: *Journal of the Operational Research Society* 68, pp. 861–876.

A. Gunawan, K. M. Ng, G. Kendall, and J. Lai (2018). “An iterated local search algorithm for the team orienteering problem with variable profits”. In: *Engineering Optimization* 50.7, pp. 1148–1163. DOI: 10.1080/0305215X.2017.1417398.

Q. Hu and A. Lim (2014). “An iterative three-component heuristic for the team orienteering problem with time windows”. In: *European Journal of Operational Research* 232.2, pp. 276–286. DOI: 10.1016/j.ejor.2013.06.011.

S. Irnich and D. Villeneuve (2006). “The shortest-path problem with resource constraints and k-cycle elimination for $k \geq 3$”. In: *INFORMS Journal on Computing* 18.3, pp. 391–406. DOI: 10.1287/ijoc.1040.0117.

M. G. Kantor and M. B. Rosenwein (1992). “The Orienteering Problem with Time Windows”. In: *The Journal of the Operational Research Society* 43.6, pp. 629–635. DOI: 10.2307/2583018.

M. Khodadadian, A. Divsalar, C. Verbeeck, A. Gunawan, and P. Vansteenwegen (2022). “Time dependent orienteering problem with time windows and service time dependent profits”. In: *Computers & Operations Research* 143. Article ID 105794. DOI: 10.1016/j.cor.2022.105794.

H. Kim, B.-I. Kim, and D.-j. Noh (2020). “The multi-profit orienteering problem”. In: *Computers & Industrial Engineering* 149. Article ID 106808. DOI: 10.1016/j.cie.2020.106808.

N. Labadie, J. Melechovskỳ, and R. W. Calvo (2010). “An effective hybrid evolutionary local search for orienteering and team orienteering problems with time windows”. In: *Parallel Problem Solving from Nature, PPSN XI: 11th International Conference,* *Kraków, Poland, September 11-15, 2010, Proceedings, Part II 11*. Springer, pp. 219–228. DOI: 10.1007/978-3-642-15871-1_23.

N. Labadie, J. Melechovskỳ, and R. Wolfler Calvo (2011). "Hybridized evolutionary local search algorithm for the team orienteering problem with time windows". In: *Journal of Heuristics* 17, pp. 729–753. DOI: 10.1007/s10732-010-9153-z.

N. Labadie, R. Mansini, J. Melechovskỳ, and R. W. Calvo (2012). "The team orienteering problem with time windows: An LP-based granular variable neighborhood search". In: *European Journal of Operational Research* 220.1, pp. 15–27. DOI: 10.1016/j.ejor.2012.01.030.

R. Lahyani, L. C. Coelho, and J. Renaud (2018). "Alternative formulations and improved bounds for the multi-depot fleet size and mix vehicle routing problem". In: *OR Spectrum* 40.1, pp. 125–157. DOI: 10.1007/s00291-017-0494-y.

S.-W. Lin and F. Y. Vincent (2012). "A simulated annealing heuristic for the team orienteering problem with time windows". In: *European Journal of Operational Research* 217.1, pp. 94–107. DOI: 10.1016/j.ejor.2011.08.024.

H. R. Lourenço, O. C. Martin, and T. Stützle (2003). "Iterated Local Search". In: *Handbook of Metaheuristics*. Springer US, pp. 320–353. DOI: 10.1007/0-306-48056-5_11.

G. Peng, R. Dewil, C. Verbeeck, A. Gunawan, L. Xing, and P. Vansteenwegen (2019). "Agile earth observation satellite scheduling: An orienteering problem with time-dependent profits and travel times". In: *Computers & Operations Research* 111, pp. 84–98. DOI: 10.1016/j.cor.2019.05.030.

C. Prins, C. Prodhon, A. Ruiz, P. Soriano, and R. Wolfler Calvo (2007). "Solving the Capacitated Location-Routing Problem by a Cooperative Lagrangean Relaxation-Granular Tabu Search Heuristic". In: *Transportation Science* 41.4, pp. 470–483. DOI: 10.1287/trsc.1060.0187.

H. N. Psaraftis (1983). "k-Interchange procedures for local search in a precedence-constrained routing problem". In: *European Journal of Operational Research* 13.4, pp. 391–402. DOI: 10.1016/0377-2217(83)90099-1.

J. Ruiz-Meza and J. R. Montoya-Torres (2022). "A systematic literature review for the tourist trip design problem: Extensions, solution techniques and future research lines". In: *Operations Research Perspectives* 9. Article ID 100228. DOI: 10.1016/j.orp.2022.100228.

M. W. Savelsbergh (1985). "Local search in routing problems with time windows". In: *Annals of Operations research* 4, pp. 285–305. DOI: 10.1007/BF02022044.

V. Schmid and J. F. Ehmke (2017). "An effective large neighborhood search for the team orienteering problem with time windows". In: *Computational Logistics: 8th International Conference, ICCL 2017, Southampton, UK, October 18-20, 2017, Proceedings 8*. Ed. by T. Bektaş, S. Coniglio, A. Martinez-Sykora, and S. Voß. Springer, pp. 3–18. DOI: 10.1007/978-3-319-68496-3_1.

M. Schneider, F. Schwahn, and D. Vigo (2017). “Designing granular solution methods for routing problems with time windows”. In: *European Journal of Operational Research* 263.2, pp. 493–509. DOI: 10.1016/j.ejor.2017.04.059.

M. Schneider, A. Stenger, and D. Goeke (2014). “The electric vehicle-routing problem with time windows and recharging stations”. In: *Transportation science* 48.4, pp. 500–520. DOI: 10.1287/trsc.2013.0490.

M. M. Solomon (1987). “Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints”. In: *Operations Research* 35.2, pp. 254–265. DOI: 10.1287/opre.35.2.254.

W. Souffriau, P. Vansteenwegen, G. Vanden Berghe, and D. van Oudheusden (2013). “The Multiconstraint Team Orienteering Problem with Multiple Time Windows”. In: *Transportation Science* 47.1, pp. 53–63. DOI: 10.1287/trsc.1110.0377.

H. Tang, E. Miller-Hooks, and R. Tomastik (2007). “Scheduling technicians for planned maintenance of geographically distributed equipment”. In: *Transportation Research Part E: Logistics and Transportation Review* 43.5, pp. 591–609. DOI: 10.1016/j.tre.2006.03.004.

P. Toth and D. Vigo (2003). “The granular tabu search and its application to the vehicle-routing problem”. In: *INFORMS Journal on Computing* 15.4, pp. 333–346. DOI: 10.1287/ijoc.15.4.333.24890.

F. Tricoire, M. Romauch, K. F. Doerner, and R. F. Hartl (2010). “Heuristics for the multi-period orienteering problem with multiple time windows”. In: *Computers & Operations Research* 37.2, pp. 351–367. DOI: 10.1016/j.cor.2009.05.012.

P. Vansteenwegen, W. Souffriau, G. Vanden Berghe, and D. van Oudheusden (2009). “Iterated local search for the team orienteering problem with time windows”. In: *Computers & Operations Research* 36.12, pp. 3281–3290. DOI: 10.1016/j.cor.2009.03.008.

Q. Yu, Y. Adulyasak, L.-M. Rousseau, N. Zhu, and S. Ma (2022). “Team Orienteering with Time-Varying Profit”. In: *INFORMS Journal on Computing* 34.1, pp. 262–280. DOI: 10.1287/ijoc.2020.1026.

V. F. Yu, P. Jewpanya, S.-W. Lin, and A. Redi (2019). “Team orienteering problem with time windows and time-dependent scores”. In: *Computers & Industrial Engineering* 127, pp. 213–224. DOI: 10.1016/j.cie.2018.11.044.

5.7 Appendix

A C-TOP-TDPLSF with constrained waiting strategy (C-TOP-TDPLSF-cw)

In this section, we describe the C-TOP-TDPLSF-cw as a variant of the C-TOP-TDPLSF, and we show how to modify Model (5.1)–(5.17) for this new problem.

In the C-TOP-TDPLSF-cw, if the arrival time at a customer falls outside customer's time windows, waiting at the customer is allowed but only for a predefined period of time. The length of the waiting time depends on the type of the upcoming time window. If the upcoming time window is a preferred one or a less preferred one with a negative slope, i.e., $w \in W_i \cup LP_i^-$, then the vehicle is allowed to wait until its opening time. On the contrary, if the upcoming time window is less preferred with a positive slope, i.e., $w \in LP_i^+$, then the vehicle is allowed to wait until the middle of the time window.

Model (5.1)–(5.17) is modified as follows. We introduce the continuous variables $z_i^k \in \mathbb{R}_+$ to represent the waiting time of vehicle $k \in K$ at customer $i \in V_C$. The objective function is changed as follows due to the dependency of the score on the start time of the visiting time defined as the arrival time plus the waiting time:

$$\Theta^{C-TOP-TDPLSF-cw} = \max \sum_{i \in V_C} f_i(t_i^k + z_i^k) \tag{5.27}$$

Constraints (5.6), and (5.10)–(5.13) are changed, respectively, as follows:

$$D_{max} - d_{0i}x_{0i}^k - (d_{jN+1} + s_j)x_{jN+1}^k - (t_j^k + z_j^k - t_i^k) \geq 0 \quad i \in V_C, j \in V_0, k \in K \tag{5.28}$$

$$t_i^k + z_i^k + s_i + d_{ij} \leq t_j^k + M(1 - x_{ij}^k) \quad i \in V_C, j \in V_{N+1} : j \neq i, k \in K \tag{5.29}$$

$$t_i^k + z_i^k + s_i + d_{ij} \geq t_j^k - M(1 - x_{ij}^k) \quad i \in V_C, j \in V_{N+1} : j \neq i, k \in K \tag{5.30}$$

$$a_{iw}y_{iw}^k \leq t_i^k + z_i^k \leq M(1 - y_{iw}^k) + b_{iw} \quad w \in W_i, i \in V_C, k \in K \tag{5.31}$$

$$t_i^k + z_i^k \leq M \sum_{j \in V_{N+1}} x_{ij}^k \quad i \in V_{N+1}, k \in K. \tag{5.32}$$

Similarly to the C-TOP-TDPLSF formulation, the M in constraints (5.29) and (5.30) can be substituted by $T + s_i + d_{ij}$ and the M in constraints (5.31) and (5.32) can be replaced by $\min\{T - d_{i0}, b_i^{max}\}$, where $b_i^{max} = \max\{b_{iw} : w \in W_i\}$. Moreover, we introduce the following decision variables:

- $\gamma_i^k \in \{0, 1\}, i \in V_C, k \in K$: binary variable equal to one if the arrival time of vehicle k at customer i falls in a less preferred or preferred time window, equal to zero otherwise.

- $\phi_{iw}^k \in \{0, 1\}, i \in V_C, w \in W_i, k \in K$: binary variable equal to one if the arrival time of vehicle k at customer i is smaller than the opening time of time window

$w \in W_i$, equal to zero otherwise.

- $\lambda_{iw}^k \in \{0,1\}, i \in V_C, w \in W_i, k \in K$: binary variable equal to one if the arrival time of vehicle k at customer i is bigger than the opening time of the previous time window $w-1$, equal to zero otherwise.

Figure 5.3 shows an example of how the upcoming time window with respect to the arrival time t_i^k that falls in a closed time period is identified through the variables λ_{iw}^k and ϕ_{iw}^k. In the figure, the horizontal axis represents a section of the time horizon with five time periods and four time windows (w_1, w_2, w_3, and w_4). The arrival time t_i^k falls in a closed time period. Variable ϕ_{iw}^k is equal to one for time windows w_3 and w_4 because t_i^k is smaller than the opening time of these two time windows (i.e., $t_i^k < a_3$ and $t_i^k < a_4$). For w_1 and w_2, ϕ_{iw}^k takes value zero. Variable λ_{iw}^k is equal to one for time windows w_1, w_2, and w_3 because t_i^k is bigger than the opening time of their previous time windows (i.e., $t_i^k > 0$ for w_1, $t_i^k > a_1$ for w_2, and $t_i^k > a_2$ for w_3). The time window for which both variable ϕ_{iw}^k and λ_{iw}^k take value one is the upcoming time window with respect to the arrival time t_i^k.

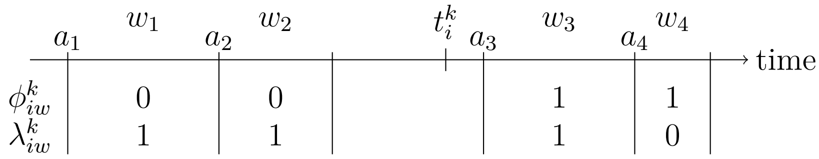


Figure 5.3: Example of how the upcoming time window with respect to the arrival time t_i^k is identified through the variables λ_{iw}^k and ϕ_{iw}^k.

In addition, we add the following constraints:

$$f_i(t_i^k) \leq M\gamma_i^k \qquad i \in V_C, k \in K \tag{5.33}$$

$$z_i^k \leq M(1-\gamma_i^k) \qquad i \in V_C, k \in K \tag{5.34}$$

$$t_i^k \leq a_{iw-1} + M\lambda_{iw}^k \qquad w \in P_i \cup LP_i, i \in V_C, k \in K \tag{5.35}$$

$$t_i^k \geq a_{iw}(1-\phi_{iw}^k) \qquad w \in P_i \cup LP_i^-, i \in V_C, k \in K \tag{5.36}$$

$$z_i^k \leq (a_{iw} - t_i^k) + M(1-\phi_{iw}^k) \qquad w \in P_i \cup LP_i^-, i \in V_C, k \in K \tag{5.37}$$

$$z_i^k \geq (a_{iw} - t_{ik}) - M(2-(1-\gamma_i^k) - \lambda_{iw}^k) \qquad w \in P_i \cup LP_i^-, i \in V_C, k \in K \tag{5.38}$$

$$z_i^k \leq \left(\frac{a_{iw}+b_{iw}}{2} - t_i^k\right) + M(1-\phi_{iw}^k) \qquad w \in LP_i^+, i \in V_C, k \in K \tag{5.39}$$

$$z_i^k \geq \left(\frac{a_{iw}+b_{iw}}{2} - t_i^k\right) - M(2-(1-\gamma_i^k) - \lambda_{iw}^k) \qquad w \in LP_i^+, i \in V_C, k \in K \tag{5.40}$$

$$z_i^k \geq 0 \qquad i \in V_C, k \in K \tag{5.41}$$

$$\gamma_i^k \in \{0,1\} \qquad i \in V_C, k \in K \tag{5.42}$$

$$\phi_{iw}^k \in \{0,1\} \qquad w \in W_i, i \in V_C, k \in K \tag{5.43}$$

$$\lambda_{iw}^k \in \{0,1\} \qquad w \in W_i, i \in V_C, k \in K \tag{5.44}$$

Constraints (5.33) guarantee that, if the arrival time falls in a less preferred or preferred time window, i.e., the score is positive, γ_i is equal to one. Because the score collected at a customer cannot exceed its full score p_i, the M in constraints (5.33) can be replaced by p_i. Constraints (5.34) forces the waiting time to be zero if the arrival time falls in a less preferred or preferred time window. To strengthen the formulation, the M in constraints (5.34) is substituted by $T - d_{iN+1}$.

Constraints (5.35) guarantee that the binary variable λ_{iw} is equal to one if the arrival time t_i^k is bigger than the opening time of the previous time window $w - 1$.

Constraints (5.36)–(5.38) define the waiting time if the arrival time precedes the opening time of a preferred time window $w \in P_i$ or of a less preferred time window with negative slope. Considered with constraints (5.35), constraints (5.36) identify the upcoming time window, i.e., the time window of which opening time is considered for determining the length of the waiting time. Because ϕ_{iw} is equal to one when the arrival time precedes the opening time of a preferred time window $w \in P_i$ or of a less preferred time window with negative slope, and λ_{iw} is equal to one when the arrival time is bigger than the opening time of the previous time window $w - 1$, the time window w for which both variables are equal to one is the time window to consider. In this case, the waiting time corresponds to the difference between the opening of this time window and the arrival time.

In a similar way, constraints (5.39)–(5.40) define the waiting time if the arrival time precedes the opening time of a less preferred time window with positive slope. In this case, the waiting time corresponds to the difference between the middle point of the time window and the arrival time.

The big-M in constraints (5.37) and (5.39) can be substituted with the tighter value D_{max} because the maximum waiting time cannot be greater than the maximum tour duration. Finally, constraints (5.41)–(5.44) define the domain of the additional variables.

The score of the optimal solution of C-TOP-TDPLSF-cw is an upper bound on the total collected score of the optimal solution of C-TOP-TDPLSF-nw, i.e., $\Theta^{C-TOP-TDPLSF-nw} \leq \Theta^{C-TOP-TDPLSF-cw}$, and a lower bound on the total collected score of C-TOP-TDPLSF-w, i.e., $\Theta^{C-TOP-TDPLSF-cw} \leq \Theta^{C-TOP-TDPLSF-w}$.

B Detailed results for small-scale instances

In this section, we report the results detailed by instance for the small-scale instances. Table 5.18 shows the comparison of the results for the C-TOP-TDPLSF-w instances solved by ILS-cWait-fin with $\mu = 0$ and ILS-cWait-fin. Table 5.19 shows the comparison of the results obtained with ILS-cWait-fin in which waiting is only allowed until the opening of the upcoming time window (ILS-cWait-fin-simple) and ILS-algo.

					Gurobi			ILS-cWait-fin $\mu = 0$			ILS-cWait-fin		
Instance id	type	$\|K\|$	$\|N\|$	# periods	Θ	$\Delta_{UB}(\%)$	$t(s)$	$\Delta_{\Theta}^{best}(\%)$	$\Delta_{\Theta}^{avg}(\%)$	$t^a(s)$	$\Delta_{\Theta}^{best}(\%)$	$\Delta_{\Theta}^{avg}(\%)$	$t^a(s)$
pr04	Cordeau	1	96	4	170.16	94.50	7200.00	**-0.28**	**-0.28**	3.18	**-1.05**	**-1.05**	76.49
pr04	Cordeau	1	96	8	165.86	140.15	7200.00	**-6.06**	**-6.06**	4.93	**-8.10**	**-7.87**	86.14
pr04	Cordeau	2	96	4	339.30	94.09	7200.00	2.39	2.39	9.07	1.65	1.80	105.77
pr04	Cordeau	2	96	8	320.02	116.54	7200.02	**-3.81**	**-3.81**	11.74	**-5.64**	**-5.64**	102.11
pr04	Cordeau	3	96	4	463.93	80.38	7200.02	**-4.47**	**-3.41**	16.19	**-5.23**	**-4.19**	127.64
pr04	Cordeau	3	96	8	499.08	72.51	7200.02	4.70	6.08	24.43	2.27	3.04	135.00
pr07	Cordeau	1	36	4	122.94	$\underline{0.00}$	15.48	0.34	0.34	0.44	**0.00**	**0.00**	15.68
pr07	Cordeau	1	36	8	137.31	$\underline{0.00}$	77.63	11.43	11.43	0.68	0.66	0.66	16.21
pr07	Cordeau	2	36	4	215.98	0.01	888.00	2.31	2.71	1.06	1.42	1.69	24.46
pr07	Cordeau	2	36	8	236.30	5.78	7200.01	10.44	10.44	1.19	1.06	1.06	23.95
pr07	Cordeau	3	36	4	287.40	19.54	7200.00	4.19	4.31	2.02	2.61	2.77	36.18
pr07	Cordeau	3	36	8	323.56	25.14	7200.00	8.83	8.83	1.50	0.02	0.02	26.64
c101	Solomon	1	50	4	130.02	$\underline{0.00}$	0.57	0.03	0.03	0.48	0.03	0.03	27.15
c101	Solomon	1	50	8	130.08	$\underline{0.00}$	1.11	15.48	15.48	0.57	15.40	15.41	27.52
c101	Solomon	2	50	4	250.02	$\underline{0.00}$	2.18	10.07	10.07	1.11	**0.00**	**0.00**	48.11
c101	Solomon	2	50	8	239.95	$\underline{0.00}$	9.26	16.66	16.66	1.04	8.29	8.29	36.38
c101	Solomon	3	50	4	350.08	$\underline{0.00}$	15.78	11.95	11.95	1.85	3.37	4.14	54.79
c101	Solomon	3	50	8	340.76	$\underline{0.00}$	46.49	20.98	20.98	1.56	9.01	9.01	42.77
c107	Solomon	1	50	4	138.71	$\underline{0.00}$	10.26	9.67	9.67	0.57	**0.00**	**0.00**	27.96
c107	Solomon	1	50	8	131.53	$\underline{0.00}$	31.35	1.99	1.99	0.74	1.57	1.57	28.50
c107	Solomon	2	50	4	259.61	$\underline{0.00}$	267.81	9.37	9.37	1.24	3.69	3.69	40.45
c107	Solomon	2	50	8	261.57	$\underline{0.00}$	1357.14	11.22	11.22	1.53	3.83	4.83	37.87
c107	Solomon	3	50	4	369.61	0.01	5713.40	10.59	10.60	2.02	4.66	5.50	44.35
c107	Solomon	3	50	8	368.23	13.25	7200.00	10.16	10.16	2.11	4.49	4.49	44.67
c109	Solomon	1	50	4	172.63	$\underline{0.00}$	102.46	14.86	14.86	0.59	11.59	11.59	28.97
c109	Solomon	1	50	8	175.95	$\underline{0.00}$	476.27	21.46	21.46	0.76	6.38	6.38	34.48
c109	Solomon	2	50	4	292.60	14.46	7200.01	5.23	5.23	1.23	3.41	3.41	44.22
c109	Solomon	2	50	8	296.01	27.58	7200.00	9.64	9.64	1.38	**0.00**	**0.00**	40.16
c109	Solomon	3	50	4	388.39	22.41	7200.01	3.89	3.89	1.84	**-0.94**	**-0.94**	42.60
c109	Solomon	3	50	8	405.86	17.45	7200.01	9.60	9.60	1.99	0.89	0.89	42.39
r102	Solomon	1	50	4	72.00	$\underline{0.00}$	154.59	6.94	6.94	0.53	3.31	3.31	26.40
r102	Solomon	1	50	8	72.00	$\underline{0.00}$	444.80	3.53	3.53	0.52	**0.00**	**0.00**	26.35
r102	Solomon	2	50	4	140.00	$\underline{0.00}$	3691.15	8.57	8.57	1.07	6.70	6.70	39.20
r102	Solomon	2	50	8	141.46	34.39	7200.01	8.13	8.13	1.06	**0.00**	**0.00**	38.10
r102	Solomon	3	50	4	201.00	58.86	7200.00	8.23	8.23	1.50	3.27	3.27	43.84
r102	Solomon	3	50	8	200.47	72.44	7200.00	8.29	8.29	1.70	2.55	2.55	45.65
r103	Solomon	1	50	4	122.00	$\underline{0.00}$	286.90	2.03	2.03	0.49	**0.00**	**0.00**	26.90
r103	Solomon	1	50	8	122.00	$\underline{0.00}$	623.59	**-0.27**	**-0.27**	0.59	**-0.27**	**-0.27**	26.83
r103	Solomon	2	50	4	201.01	37.60	7200.00	4.02	4.02	0.86	**0.00**	**0.00**	38.73
r103	Solomon	2	50	8	193.24	48.60	7200.00	0.92	0.92	1.20	**-0.14**	**-0.14**	41.16
r103	Solomon	3	50	4	274.06	50.69	7200.01	7.00	7.00	1.21	**0.00**	**0.00**	46.21
r103	Solomon	3	50	8	259.34	60.72	7200.00	3.19	3.19	1.76	**-0.07**	0.90	46.11
r110	Solomon	1	50	4	94.00	$\underline{0.00}$	573.18	4.15	4.18	0.71	4.15	4.18	37.46
r110	Solomon	1	50	8	90.68	$\underline{0.00}$	1879.28	6.85	6.85	0.72	**0.00**	**0.00**	34.39
r110	Solomon	2	50	4	184.10	60.28	7200.00	2.70	2.70	1.28	2.30	2.30	44.65
r110	Solomon	2	50	8	175.32	112.39	7200.00	9.55	9.55	1.21	1.15	1.15	43.29
r110	Solomon	3	50	4	257.83	92.26	7200.00	2.89	3.10	1.97	2.23	2.41	52.35
r110	Solomon	3	50	8	247.32	123.05	7200.00	10.67	10.67	1.86	4.36	4.36	46.94
rc101	Solomon	1	50	4	90.00	$\underline{0.00}$	1.17	**0.00**	**0.00**	0.36	**0.00**	**0.00**	26.89
rc101	Solomon	1	50	8	90.00	$\underline{0.00}$	1.07	22.22	22.22	0.45	**0.00**	**0.00**	26.38
rc101	Solomon	2	50	4	160.01	$\underline{0.00}$	7.49	**0.00**	**0.00**	0.72	**0.00**	**0.00**	37.45
rc101	Solomon	2	50	8	160.00	$\underline{0.00}$	8.83	16.90	16.90	0.90	**0.00**	**0.00**	37.52
rc101	Solomon	3	50	4	219.35	$\underline{0.00}$	18.67	2.12	2.12	1.11	**0.00**	**0.00**	43.69
rc101	Solomon	3	50	8	220.00	$\underline{0.00}$	43.47	17.60	17.60	1.51	3.03	3.03	43.81
rc107	Solomon	1	50	4	93.31	$\underline{0.00}$	155.08	7.63	7.63	0.46	**0.00**	**0.00**	27.35
rc107	Solomon	1	50	8	97.01	$\underline{0.00}$	649.91	10.32	10.32	0.46	9.83	9.83	27.84
rc107	Solomon	2	50	4	171.71	16.94	7200.00	7.97	7.97	0.94	3.83	3.83	40.85
rc107	Solomon	2	50	8	167.01	83.61	7200.00	4.20	4.20	0.99	2.23	2.23	42.52
rc107	Solomon	3	50	4	232.01	68.91	7200.00	6.53	6.53	1.76	3.43	3.43	50.75
rc107	Solomon	3	50	8	240.01	76.51	7200.00	7.77	7.77	1.78	2.80	2.80	46.34
rc108	Solomon	1	50	4	90.00	$\underline{0.00}$	495.07	**0.00**	**0.00**	0.42	**0.00**	**0.00**	27.11
rc108	Solomon	1	50	8	91.71	$\underline{0.00}$	2127.83	4.56	4.56	0.51	4.06	4.06	27.63
rc108	Solomon	2	50	4	160.19	98.97	7200.00	0.87	0.87	0.89	**0.00**	**0.00**	43.68
rc108	Solomon	2	50	8	179.69	127.41	7200.00	8.97	8.97	1.02	2.62	2.62	41.55
rc108	Solomon	3	50	4	229.53	132.86	7200.00	3.15	3.15	1.32	**0.00**	**0.00**	49.20
rc108	Solomon	3	50	8	249.70	123.82	7200.00	8.41	8.41	1.60	**-0.38**	**-0.38**	45.62
Avg						33.70	3905.72	6.81	6.86	2.07	1.91	2.01	43.34

Table 5.18: Comparison of results obtained with Gurobi and ILS-cWait-fin without finalization phase (i.e. $\mu = 0$) on the small-scale C-TOP-TDPLSF-w instances.

					Gurobi	ILS-cWait-fin-simple		ILS-cWait-fin	
Instance id	type	$\|K\|$	$\|N\|$	# periods	Θ	$\Delta_\Theta^{best}(\%)$	$\Delta_\Theta^{avg}(\%)$	$\Delta_\Theta^{best}(\%)$	$\Delta_\Theta^{avg}(\%)$
pr04	Cordeau	1	96	4	170.16	-0.68	-0.68	-1.05	-1.05
pr04	Cordeau	1	96	8	165.86	-12.18	-9.46	-8.10	-7.87
pr04	Cordeau	2	96	4	339.30	0.57	1.63	1.65	1.80
pr04	Cordeau	2	96	8	320.02	-4.94	-4.69	-5.64	-5.64
pr04	Cordeau	3	96	4	463.93	-3.40	-3.34	-5.23	-4.19
pr04	Cordeau	3	96	8	499.08	2.23	3.21	2.27	3.04
pr07	Cordeau	1	36	4	122.94	0.00	0.00	0.00	0.00
pr07	Cordeau	1	36	8	137.31	6.25	6.25	0.66	0.66
pr07	Cordeau	2	36	4	215.98	1.42	1.60	1.42	1.69
pr07	Cordeau	2	36	8	236.30	6.56	7.86	1.06	1.06
pr07	Cordeau	3	36	4	287.40	2.76	2.76	2.61	2.77
pr07	Cordeau	3	36	8	323.56	4.80	6.74	0.02	0.02
c101	Solomon	1	50	4	130.02	0.02	0.02	0.03	0.03
c101	Solomon	1	50	8	130.08	15.48	15.48	15.40	15.41
c101	Solomon	2	50	4	250.02	8.02	8.02	0.00	0.00
c101	Solomon	2	50	8	239.95	12.19	12.19	8.29	8.29
c101	Solomon	3	50	4	350.08	10.07	10.07	3.37	4.14
c101	Solomon	3	50	8	340.76	11.75	11.75	9.01	9.01
c107	Solomon	1	50	4	138.71	4.56	4.56	0.00	0.00
c107	Solomon	1	50	8	131.53	2.39	2.39	1.57	1.57
c107	Solomon	2	50	4	259.61	3.69	3.69	3.69	3.69
c107	Solomon	2	50	8	261.57	5.25	5.25	3.83	4.83
c107	Solomon	3	50	4	369.61	3.31	4.45	4.66	5.50
c107	Solomon	3	50	8	368.23	5.76	6.20	4.49	4.49
c109	Solomon	1	50	4	172.63	11.59	11.59	11.59	11.59
c109	Solomon	1	50	8	175.95	15.77	15.77	6.38	6.38
c109	Solomon	2	50	4	292.60	3.41	3.41	3.41	3.41
c109	Solomon	2	50	8	296.01	7.86	8.63	0.00	0.00
c109	Solomon	3	50	4	388.39	-0.94	-0.94	-0.94	-0.94
c109	Solomon	3	50	8	405.86	5.70	7.88	0.89	0.89
r102	Solomon	1	50	4	72.00	3.31	3.31	3.31	3.31
r102	Solomon	1	50	8	72.00	3.53	3.53	0.00	0.00
r102	Solomon	2	50	4	140.00	6.70	6.70	6.70	6.70
r102	Solomon	2	50	8	141.46	9.08	9.08	0.00	0.00
r102	Solomon	3	50	4	201.00	3.27	3.27	3.27	3.27
r102	Solomon	3	50	8	200.47	6.91	7.39	2.55	2.55
r103	Solomon	1	50	4	122.00	0.00	0.00	0.00	0.00
r103	Solomon	1	50	8	122.00	-0.27	-0.27	-0.27	-0.27
r103	Solomon	2	50	4	201.01	0.00	0.00	0.00	0.00
r103	Solomon	2	50	8	193.24	-0.14	-0.14	-0.14	-0.14
r103	Solomon	3	50	4	274.06	4.73	4.73	0.00	0.00
r103	Solomon	3	50	8	259.34	-0.07	2.80	-0.07	0.90
r110	Solomon	1	50	4	94.00	4.15	4.21	4.15	4.18
r110	Solomon	1	50	8	90.68	0.00	0.00	0.00	0.00
r110	Solomon	2	50	4	184.10	2.30	2.30	2.30	2.30
r110	Solomon	2	50	8	175.32	1.15	1.15	1.15	1.15
r110	Solomon	3	50	4	257.83	2.23	2.23	2.23	2.41
r110	Solomon	3	50	8	247.32	4.36	4.36	4.36	4.36
rc101	Solomon	1	50	4	90.00	0.00	0.00	0.00	0.00
rc101	Solomon	1	50	8	90.00	0.00	0.00	0.00	0.00
rc101	Solomon	2	50	4	160.01	0.00	0.00	0.00	0.00
rc101	Solomon	2	50	8	160.00	0.00	0.00	0.00	0.00
rc101	Solomon	3	50	4	219.35	0.00	0.00	0.00	0.00
rc101	Solomon	3	50	8	220.00	0.00	0.00	3.03	3.03
rc107	Solomon	1	50	4	93.31	0.00	0.00	0.00	0.00
rc107	Solomon	1	50	8	97.01	6.85	6.85	9.83	9.83
rc107	Solomon	2	50	4	171.71	3.83	3.83	3.83	3.83
rc107	Solomon	2	50	8	167.01	2.23	2.23	2.23	2.23
rc107	Solomon	3	50	4	232.01	3.43	3.43	3.43	3.43
rc107	Solomon	3	50	8	240.01	5.15	5.47	2.80	2.80
rc108	Solomon	1	50	4	90.00	0.00	0.00	0.00	0.00
rc108	Solomon	1	50	8	91.71	4.06	4.06	4.06	4.06
rc108	Solomon	2	50	4	160.19	0.00	0.00	0.00	0.00
rc108	Solomon	2	50	8	179.69	2.62	2.62	2.62	2.62
rc108	Solomon	3	50	4	229.53	0.00	0.00	0.00	0.00
rc108	Solomon	3	50	8	249.70	3.84	3.84	-0.38	-0.38
Avg						3.22	3.47	1.91	2.01

Table 5.19: Comparison of the results obtained with ILS-cWait-fin in which waiting is only allowed until the opening of the upcoming time window (ILS-cWait-fin-simple) and ILS-algo (i.e., in which a more elaborated waiting strategy is implemented) when solving the small-scale C-TOP-TDPLSF-w instances.

C Detailed results for the large-scale instances

In this section, we report the detailed results by instance for the large-scale instances. Table 5.20 and Table 5.21 show the results obtained by ILS-noWait for the Solomon and Cordeau C-TOP-TDPLSF-nw instances, respectively. Table 5.22 and Table 5.23 show the results obtained by ILS-cWait-fin for the Solomon and Cordeau C-TOP-TDPLSF-w instances.

Instance id	type	$\|K\|$	$\|N\|$	periods	Θ^{best}	Θ^{avg}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)
c101	Solomon	1	100	4	229.48	221.79	2.70	9.00	22.22	77.78	87.05	89.17
c101	Solomon	1	100	8	241.20	232.52	3.79	10.00	50.00	50.00	93.42	92.19
c101	Solomon	2	100	4	422.88	414.26	11.51	20.00	40.00	60.00	91.61	94.32
c101	Solomon	2	100	8	444.05	438.74	11.92	19.00	68.42	31.58	89.17	95.79
c101	Solomon	3	100	4	595.82	585.47	18.56	26.00	38.46	61.54	90.87	86.54
c101	Solomon	3	100	8	612.50	608.21	19.18	28.00	60.71	39.29	94.27	93.64
c101	Solomon	4	100	4	776.64	754.54	23.58	36.00	38.89	61.11	97.93	88.16
c101	Solomon	4	100	8	793.00	766.55	35.77	37.00	56.76	43.24	106.32	89.57
c102	Solomon	1	100	4	277.08	277.01	5.20	10.00	70.00	30.00	97.50	98.57
c102	Solomon	1	100	8	314.50	308.89	7.15	11.00	81.82	18.18	96.75	99.25
c102	Solomon	2	100	4	504.45	497.30	12.87	21.00	47.62	52.38	88.12	98.92
c102	Solomon	2	100	8	564.35	559.34	19.69	21.00	80.95	19.05	96.50	99.77
c102	Solomon	3	100	4	718.78	698.49	42.33	29.00	51.72	48.28	94.00	93.14
c102	Solomon	3	100	8	778.44	766.20	45.23	30.00	83.33	16.67	94.50	99.12
c102	Solomon	4	100	4	872.28	864.06	91.73	40.00	52.50	47.50	94.94	95.89
c102	Solomon	4	100	8	963.48	934.41	55.18	40.00	75.00	25.00	93.75	99.09
c103	Solomon	1	100	4	361.01	361.01	4.90	10.00	50.00	50.00	73.61	95.44
c103	Solomon	1	100	8	363.40	363.40	5.40	10.00	50.00	50.00	82.03	99.13
c103	Solomon	2	100	4	655.37	651.08	19.52	21.00	57.14	42.86	85.77	97.67
c103	Solomon	2	100	8	646.67	645.57	16.80	20.00	60.00	40.00	87.20	99.32
c103	Solomon	3	100	4	894.41	882.78	33.96	32.00	65.62	34.38	89.30	99.76
c103	Solomon	3	100	8	880.45	859.63	48.13	31.00	67.74	32.26	90.51	97.56
c103	Solomon	4	100	4	1082.54	1070.24	69.15	41.00	63.41	36.59	86.09	97.88
c103	Solomon	4	100	8	1053.26	1045.07	82.16	39.00	76.92	23.08	84.70	97.41
c104	Solomon	1	100	4	375.98	375.98	4.56	10.00	80.00	20.00	63.55	99.26
c104	Solomon	1	100	8	377.41	377.41	6.98	10.00	80.00	20.00	87.46	98.93
c104	Solomon	2	100	4	685.01	677.80	21.44	22.00	72.73	27.27	84.78	99.95
c104	Solomon	2	100	8	687.46	680.06	21.58	21.00	85.71	14.29	77.42	99.29
c104	Solomon	3	100	4	919.17	908.36	41.31	32.00	62.50	37.50	81.16	99.03
c104	Solomon	3	100	8	950.01	940.76	45.42	32.00	81.25	18.75	85.84	99.31
c104	Solomon	4	100	4	1106.37	1097.10	64.53	43.00	72.09	27.91	87.17	99.55
c104	Solomon	4	100	8	1152.32	1133.59	76.33	42.00	73.81	26.19	86.87	96.99
c105	Solomon	1	100	4	269.09	269.09	3.52	10.00	60.00	40.00	91.60	96.81
c105	Solomon	1	100	8	287.49	287.49	3.54	10.00	60.00	40.00	77.29	93.35
c105	Solomon	2	100	4	511.20	506.65	12.89	19.00	68.42	31.58	98.76	92.64
c105	Solomon	2	100	8	534.90	534.90	9.16	19.00	63.16	36.84	93.80	93.91
c105	Solomon	3	100	4	723.86	695.61	16.81	28.00	57.14	42.86	89.12	93.14
c105	Solomon	3	100	8	753.80	747.47	25.38	27.00	81.48	18.52	89.50	87.97

Instance id	type	$\|K\|$	$\|N\|$	periods	Θ^{best}	Θ^{avg}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)
c105	Solomon	4	100	4	885.78	880.70	36.75	36.00	63.89	36.11	90.55	91.43
c105	Solomon	4	100	8	923.31	902.11	36.90	36.00	83.33	16.67	90.60	88.13
c106	Solomon	1	100	4	289.40	289.04	6.02	10.00	30.00	70.00	98.69	97.93
c106	Solomon	1	100	8	281.14	275.49	5.28	10.00	80.00	20.00	98.37	92.97
c106	Solomon	2	100	4	561.49	554.49	22.52	20.00	55.00	45.00	83.99	98.00
c106	Solomon	2	100	8	521.00	518.08	15.94	19.00	73.68	26.32	83.91	94.04
c106	Solomon	3	100	4	756.09	740.73	29.81	29.00	51.72	48.28	94.39	96.85
c106	Solomon	3	100	8	715.18	697.27	22.35	27.00	66.67	33.33	82.46	90.35
c106	Solomon	4	100	4	920.10	901.64	38.53	39.00	56.41	43.59	84.44	94.21
c106	Solomon	4	100	8	896.63	864.08	43.32	38.00	65.79	34.21	82.80	91.26
c107	Solomon	1	100	4	306.25	306.25	5.34	10.00	70.00	30.00	95.50	97.74
c107	Solomon	1	100	8	314.54	310.48	6.07	11.00	54.55	45.45	72.50	98.06
c107	Solomon	2	100	4	568.20	560.04	15.01	21.00	71.43	28.57	88.42	98.66
c107	Solomon	2	100	8	566.87	563.98	17.83	21.00	76.19	23.81	83.58	96.97
c107	Solomon	3	100	4	763.00	761.20	26.73	29.00	75.86	24.14	72.83	95.13
c107	Solomon	3	100	8	793.97	780.19	37.39	30.00	73.33	26.67	78.61	93.98
c107	Solomon	4	100	4	955.50	945.17	38.07	40.00	70.00	30.00	84.83	95.37
c107	Solomon	4	100	8	967.06	947.19	57.86	39.00	87.18	12.82	82.42	94.51
c108	Solomon	1	100	4	286.14	284.99	5.87	11.00	72.73	27.27	97.99	98.47
c108	Solomon	1	100	8	301.75	301.75	4.37	10.00	60.00	40.00	91.96	96.56
c108	Solomon	2	100	4	558.72	557.29	15.04	20.00	55.00	45.00	94.08	90.86
c108	Solomon	2	100	8	563.58	563.58	17.11	19.00	42.11	57.89	90.40	90.64
c108	Solomon	3	100	4	751.20	744.66	23.71	30.00	63.33	36.67	98.59	92.16
c108	Solomon	3	100	8	758.51	754.76	31.66	28.00	71.43	28.57	94.57	87.98
c108	Solomon	4	100	4	932.87	924.39	57.89	38.00	60.53	39.47	98.72	89.86
c108	Solomon	4	100	8	949.08	936.29	47.63	36.00	77.78	22.22	97.15	88.75
c109	Solomon	1	100	4	343.40	343.40	4.19	10.00	60.00	40.00	85.80	92.30
c109	Solomon	1	100	8	342.17	342.17	5.14	10.00	70.00	30.00	81.36	93.10
c109	Solomon	2	100	4	615.85	610.84	15.14	20.00	75.00	25.00	97.63	92.21
c109	Solomon	2	100	8	623.47	622.51	17.55	20.00	65.00	35.00	96.89	96.92
c109	Solomon	3	100	4	809.16	802.81	21.93	28.00	67.86	32.14	97.63	89.49
c109	Solomon	3	100	8	821.07	817.75	26.64	29.00	68.97	31.03	97.04	94.54
c109	Solomon	4	100	4	1078.27	1075.38	53.82	42.00	59.52	40.48	154.66	97.48
c109	Solomon	4	100	8	1069.78	1056.56	74.68	41.00	75.61	24.39	146.89	97.02
r101	Solomon	1	100	4	109.98	108.73	1.18	5.00	60.00	40.00	60.25	87.83
r101	Solomon	1	100	8	161.63	138.95	1.75	8.00	37.50	62.50	73.33	98.79
r101	Solomon	2	100	4	215.90	212.86	2.01	11.00	27.27	72.73	69.01	91.38
r101	Solomon	2	100	8	228.97	224.85	3.45	12.00	75.00	25.00	74.07	94.44
r101	Solomon	3	100	4	316.44	310.55	5.20	16.00	43.75	56.25	65.93	88.14
r101	Solomon	3	100	8	349.91	343.54	8.26	18.00	72.22	27.78	76.63	92.50
r101	Solomon	4	100	4	406.62	396.14	7.21	21.00	23.81	76.19	68.58	89.80
r101	Solomon	4	100	8	431.90	419.37	9.62	22.00	68.18	31.82	70.68	84.29
r102	Solomon	1	100	4	240.03	235.79	2.17	10.00	80.00	20.00	85.26	99.77
r102	Solomon	1	100	8	235.48	232.84	4.08	10.00	70.00	30.00	94.44	99.84
r102	Solomon	2	100	4	411.67	408.63	7.43	19.00	73.68	26.32	90.98	97.47
r102	Solomon	2	100	8	419.30	397.76	8.75	18.00	72.22	27.78	88.65	97.51
r102	Solomon	3	100	4	538.14	527.05	11.87	25.00	76.00	24.00	70.63	99.47
r102	Solomon	3	100	8	538.26	529.29	20.06	26.00	65.38	34.62	76.54	97.67
r102	Solomon	4	100	4	651.64	633.41	21.38	31.00	67.74	32.26	65.83	98.07

Instance id	type	$\|K\|$	$\|N\|$	periods	Θ^{best}	Θ^{avg}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)
r102	Solomon	4	100	8	642.41	627.68	16.56	32.00	68.75	31.25	63.80	90.98
r103	Solomon	1	100	4	260.69	260.69	2.47	10.00	50.00	50.00	99.26	98.96
r103	Solomon	1	100	8	258.70	258.70	2.67	9.00	88.89	11.11	82.90	98.29
r103	Solomon	2	100	4	433.53	427.91	10.10	19.00	63.16	36.84	98.35	99.17
r103	Solomon	2	100	8	448.53	436.41	12.59	18.00	83.33	16.67	76.93	97.58
r103	Solomon	3	100	4	597.38	591.99	15.49	25.00	68.00	32.00	75.06	97.54
r103	Solomon	3	100	8	608.19	595.36	22.68	30.00	80.00	20.00	88.54	98.33
r103	Solomon	4	100	4	732.96	722.16	21.40	35.00	77.14	22.86	83.96	97.42
r103	Solomon	4	100	8	752.79	718.70	37.04	38.00	68.42	31.58	88.28	98.08
r104	Solomon	1	100	4	264.07	263.23	6.71	12.00	75.00	25.00	98.78	99.05
r104	Solomon	1	100	8	274.32	263.95	3.04	11.00	81.82	18.18	84.22	97.76
r104	Solomon	2	100	4	474.72	472.79	11.98	22.00	72.73	27.27	85.92	99.04
r104	Solomon	2	100	8	493.73	484.43	11.77	20.00	70.00	30.00	80.34	98.61
r104	Solomon	3	100	4	658.60	653.11	17.99	30.00	60.00	40.00	76.73	99.29
r104	Solomon	3	100	8	669.28	660.67	21.99	31.00	80.65	19.35	80.23	98.57
r104	Solomon	4	100	4	796.96	786.92	23.36	39.00	64.10	35.90	72.79	98.73
r104	Solomon	4	100	8	806.11	800.09	25.37	38.00	65.79	34.21	73.03	98.92
r105	Solomon	1	100	4	175.53	171.57	1.84	8.00	87.50	12.50	67.45	89.17
r105	Solomon	1	100	8	176.91	161.61	2.74	7.00	28.57	71.43	56.86	85.56
r105	Solomon	2	100	4	321.46	318.00	7.10	15.00	66.67	33.33	69.61	93.01
r105	Solomon	2	100	8	356.64	312.69	6.91	17.00	35.29	64.71	90.69	88.42
r105	Solomon	3	100	4	482.98	455.75	14.24	24.00	58.33	41.67	77.97	87.88
r105	Solomon	3	100	8	478.80	442.92	11.70	22.00	36.36	63.64	84.58	89.35
r105	Solomon	4	100	4	593.43	568.54	14.30	30.00	70.00	30.00	81.47	91.47
r105	Solomon	4	100	8	610.69	558.04	17.59	30.00	40.00	60.00	78.63	88.17
r106	Solomon	1	100	4	239.93	239.93	2.24	9.00	66.67	33.33	72.91	97.25
r106	Solomon	1	100	8	234.26	233.05	2.96	10.00	70.00	30.00	95.55	99.43
r106	Solomon	2	100	4	431.44	422.22	9.59	19.00	63.16	36.84	65.96	97.39
r106	Solomon	2	100	8	436.84	431.83	9.04	19.00	63.16	36.84	85.62	99.15
r106	Solomon	3	100	4	577.77	567.61	12.77	27.00	62.96	37.04	70.07	98.88
r106	Solomon	3	100	8	597.93	584.51	20.85	28.00	67.86	32.14	80.46	99.02
r106	Solomon	4	100	4	741.12	713.51	18.33	39.00	56.41	43.59	78.48	95.85
r106	Solomon	4	100	8	725.17	707.79	27.88	36.00	72.22	27.78	79.04	95.10
r107	Solomon	1	100	4	259.00	256.89	3.29	11.00	100.00	0.00	81.78	98.92
r107	Solomon	1	100	8	261.15	250.14	4.00	11.00	63.64	36.36	91.08	99.85
r107	Solomon	2	100	4	451.39	448.01	7.30	20.00	80.00	20.00	87.55	98.45
r107	Solomon	2	100	8	451.51	438.31	11.48	20.00	65.00	35.00	91.45	99.10
r107	Solomon	3	100	4	638.88	632.34	20.46	30.00	80.00	20.00	85.13	97.82
r107	Solomon	3	100	8	616.48	607.10	20.75	26.00	73.08	26.92	89.84	99.00
r107	Solomon	4	100	4	816.75	791.03	25.91	39.00	79.49	20.51	92.10	97.39
r107	Solomon	4	100	8	778.49	765.83	27.84	36.00	58.33	41.67	89.13	98.70
r108	Solomon	1	100	4	269.10	266.45	5.14	11.00	90.91	9.09	90.04	98.65
r108	Solomon	1	100	8	268.52	264.73	4.24	11.00	90.91	9.09	98.01	99.77
r108	Solomon	2	100	4	472.90	466.57	11.61	20.00	70.00	30.00	91.37	98.84
r108	Solomon	2	100	8	488.79	473.16	13.55	20.00	70.00	30.00	79.42	99.56
r108	Solomon	3	100	4	666.31	654.29	14.63	29.00	82.76	17.24	90.04	99.08
r108	Solomon	3	100	8	672.01	654.93	23.06	29.00	82.76	17.24	94.25	99.65
r108	Solomon	4	100	4	813.67	805.07	19.78	37.00	75.68	24.32	90.49	99.46
r108	Solomon	4	100	8	840.40	814.40	46.20	40.00	77.50	22.50	95.58	97.88

Instance id	type	$\lvert K\rvert$	$\lvert N\rvert$	periods	Θ^{best}	Θ^{avg}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)
r109	Solomon	1	100	4	227.18	217.34	3.37	11.00	54.55	45.45	95.46	96.29
r109	Solomon	1	100	8	237.10	228.28	3.03	10.00	70.00	30.00	73.11	98.23
r109	Solomon	2	100	4	406.03	398.95	6.41	17.00	64.71	35.29	73.03	97.27
r109	Solomon	2	100	8	423.96	416.73	10.35	19.00	68.42	31.58	86.81	96.79
r109	Solomon	3	100	4	558.46	546.95	12.76	25.00	72.00	28.00	77.82	95.98
r109	Solomon	3	100	8	592.40	576.39	17.19	26.00	65.38	34.62	85.99	97.67
r109	Solomon	4	100	4	723.25	706.48	19.67	35.00	60.00	40.00	87.44	93.43
r109	Solomon	4	100	8	726.36	701.36	25.28	34.00	61.76	38.24	80.00	94.19
r110	Solomon	1	100	4	238.77	237.66	2.86	11.00	72.73	27.27	87.34	97.22
r110	Solomon	1	100	8	227.47	222.65	2.89	10.00	80.00	20.00	75.67	99.68
r110	Solomon	2	100	4	441.34	436.20	10.38	19.00	78.95	21.05	61.39	97.59
r110	Solomon	2	100	8	434.75	430.20	9.22	19.00	78.95	21.05	63.29	98.34
r110	Solomon	3	100	4	621.95	613.36	14.67	28.00	67.86	32.14	66.53	97.33
r110	Solomon	3	100	8	617.32	587.35	16.84	29.00	65.52	34.48	70.60	96.97
r110	Solomon	4	100	4	762.71	754.80	41.66	38.00	76.32	23.68	73.03	98.10
r110	Solomon	4	100	8	766.58	746.93	29.43	37.00	62.16	37.84	62.52	97.98
r111	Solomon	1	100	4	253.00	251.29	3.65	11.00	63.64	36.36	81.16	98.58
r111	Solomon	1	100	8	258.45	254.58	3.87	10.00	90.00	10.00	73.55	99.74
r111	Solomon	2	100	4	436.90	432.55	8.13	18.00	77.78	22.22	61.36	99.10
r111	Solomon	2	100	8	454.01	443.82	11.74	20.00	70.00	30.00	72.02	99.00
r111	Solomon	3	100	4	623.84	617.99	18.04	30.00	53.33	46.67	71.56	97.06
r111	Solomon	3	100	8	641.31	621.51	19.72	29.00	72.41	27.59	65.47	98.32
r111	Solomon	4	100	4	791.72	751.51	23.26	38.00	63.16	36.84	67.97	99.38
r111	Solomon	4	100	8	779.40	772.32	34.76	40.00	67.50	32.50	82.03	96.95
r112	Solomon	1	100	4	256.63	256.63	4.05	11.00	72.73	27.27	95.85	98.36
r112	Solomon	1	100	8	249.57	249.36	3.32	10.00	50.00	50.00	89.78	96.60
r112	Solomon	2	100	4	462.67	458.20	16.24	21.00	66.67	33.33	80.44	97.67
r112	Solomon	2	100	8	456.80	451.20	15.17	20.00	70.00	30.00	81.78	97.22
r112	Solomon	3	100	4	662.31	660.42	14.24	29.00	75.86	24.14	69.48	98.27
r112	Solomon	3	100	8	650.13	632.48	24.28	30.00	70.00	30.00	81.88	98.91
r112	Solomon	4	100	4	839.21	822.68	29.23	41.00	68.29	31.71	71.81	96.87
r112	Solomon	4	100	8	819.63	793.62	26.99	39.00	66.67	33.33	72.56	99.08
rc101	Solomon	1	100	4	168.19	166.50	1.51	8.00	50.00	50.00	88.67	95.46
rc101	Solomon	1	100	8	177.06	177.06	1.65	8.00	75.00	25.00	90.06	91.88
rc101	Solomon	2	100	4	309.86	309.86	3.15	12.00	66.67	33.33	63.22	83.45
rc101	Solomon	2	100	8	327.85	321.95	5.77	14.00	71.43	28.57	70.18	90.27
rc101	Solomon	3	100	4	445.77	427.73	7.55	19.00	57.89	42.11	58.65	89.09
rc101	Solomon	3	100	8	455.42	435.50	9.04	21.00	66.67	33.33	60.83	88.93
rc101	Solomon	4	100	4	584.70	558.27	13.43	27.00	51.85	48.15	65.36	89.34
rc101	Solomon	4	100	8	574.75	557.86	12.90	28.00	64.29	35.71	74.35	93.49
rc102	Solomon	1	100	4	211.15	199.57	2.67	9.00	55.56	44.44	74.29	99.48
rc102	Solomon	1	100	8	203.94	199.17	3.27	9.00	66.67	33.33	86.36	96.31
rc102	Solomon	2	100	4	386.85	381.25	6.92	19.00	52.63	47.37	86.91	97.63
rc102	Solomon	2	100	8	398.73	385.16	6.34	17.00	52.94	47.06	74.69	97.66
rc102	Solomon	3	100	4	567.27	555.89	11.98	26.00	57.69	42.31	75.60	97.87
rc102	Solomon	3	100	8	554.79	542.36	20.76	25.00	48.00	52.00	76.49	95.97
rc102	Solomon	4	100	4	746.98	721.36	19.03	36.00	66.67	33.33	76.76	96.00
rc102	Solomon	4	100	8	702.80	681.86	29.44	31.00	48.39	51.61	70.77	97.78
rc103	Solomon	1	100	4	219.58	216.77	3.35	9.00	66.67	33.33	79.14	96.80

Instance id	type	$\lvert K\rvert$	$\lvert N\rvert$	periods	Θ^{best}	Θ^{avg}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)
rc103	Solomon	1	100	8	216.59	215.88	2.46	8.00	87.50	12.50	98.86	94.98
rc103	Solomon	2	100	4	430.95	428.23	5.79	17.00	82.35	17.65	87.86	93.55
rc103	Solomon	2	100	8	420.23	418.02	7.33	17.00	82.35	17.65	97.29	95.60
rc103	Solomon	3	100	4	607.04	602.13	13.54	26.00	73.08	26.92	89.14	95.34
rc103	Solomon	3	100	8	598.81	595.31	20.45	26.00	57.69	42.31	95.90	98.48
rc103	Solomon	4	100	4	788.60	778.35	19.90	36.00	66.67	33.33	90.43	96.21
rc103	Solomon	4	100	8	773.96	759.42	29.74	35.00	68.57	31.43	94.29	97.28
rc104	Solomon	1	100	4	248.77	243.08	3.89	9.00	66.67	33.33	86.64	98.80
rc104	Solomon	1	100	8	243.99	239.28	3.69	10.00	60.00	40.00	92.32	98.88
rc104	Solomon	2	100	4	463.65	453.06	7.26	18.00	61.11	38.89	84.97	99.17
rc104	Solomon	2	100	8	471.36	468.82	12.81	19.00	63.16	36.84	88.23	97.81
rc104	Solomon	3	100	4	682.61	652.60	12.60	27.00	74.07	25.93	80.36	99.33
rc104	Solomon	3	100	8	693.53	673.31	19.02	30.00	66.67	33.33	92.93	98.50
rc104	Solomon	4	100	4	887.16	828.34	14.95	38.00	71.05	28.95	79.88	98.86
rc104	Solomon	4	100	8	870.63	853.28	28.22	37.00	72.97	27.03	84.27	96.88
rc105	Solomon	1	100	4	186.95	182.11	2.44	8.00	62.50	37.50	52.32	96.55
rc105	Solomon	1	100	8	189.28	187.69	2.32	9.00	66.67	33.33	73.31	97.74
rc105	Solomon	2	100	4	371.00	358.01	6.08	16.00	62.50	37.50	63.94	96.99
rc105	Solomon	2	100	8	361.12	343.01	7.05	18.00	61.11	38.89	68.29	94.82
rc105	Solomon	3	100	4	540.32	527.98	10.08	26.00	73.08	26.92	78.16	95.34
rc105	Solomon	3	100	8	489.93	479.87	11.21	22.00	72.73	27.27	57.97	88.29
rc105	Solomon	4	100	4	690.76	664.41	14.48	34.00	73.53	26.47	76.91	96.01
rc105	Solomon	4	100	8	643.50	627.37	18.14	32.00	56.25	43.75	66.00	94.58
rc106	Solomon	1	100	4	199.29	196.07	2.31	9.00	55.56	44.44	70.74	97.09
rc106	Solomon	1	100	8	212.60	203.28	2.75	8.00	75.00	25.00	58.37	96.22
rc106	Solomon	2	100	4	392.39	387.58	6.03	17.00	58.82	41.18	63.46	96.08
rc106	Solomon	2	100	8	405.55	392.92	7.11	16.00	75.00	25.00	61.97	96.58
rc106	Solomon	3	100	4	571.49	567.34	10.30	25.00	64.00	36.00	69.74	95.37
rc106	Solomon	3	100	8	599.19	581.37	14.00	27.00	55.56	44.44	73.81	97.11
rc106	Solomon	4	100	4	745.18	710.00	18.81	36.00	58.33	41.67	75.31	94.95
rc106	Solomon	4	100	8	725.17	717.43	19.50	32.00	65.62	34.38	63.81	93.20
rc107	Solomon	1	100	4	209.71	209.28	2.32	8.00	62.50	37.50	80.81	98.70
rc107	Solomon	1	100	8	225.36	225.02	2.86	9.00	88.89	11.11	97.98	92.70
rc107	Solomon	2	100	4	418.80	416.12	8.14	16.00	68.75	31.25	89.60	93.69
rc107	Solomon	2	100	8	432.64	430.13	8.92	18.00	55.56	44.44	93.13	95.82
rc107	Solomon	3	100	4	614.94	611.64	16.63	25.00	60.00	40.00	91.99	95.06
rc107	Solomon	3	100	8	623.19	622.70	14.21	27.00	48.15	51.85	94.21	95.78
rc107	Solomon	4	100	4	789.15	781.75	26.26	40.00	55.00	45.00	110.25	96.52
rc107	Solomon	4	100	8	813.60	798.18	22.17	36.00	63.89	36.11	95.15	95.42
rc108	Solomon	1	100	4	223.49	222.67	3.13	10.00	40.00	60.00	94.36	97.50
rc108	Solomon	1	100	8	220.45	219.14	2.32	8.00	75.00	25.00	59.59	98.14
rc108	Solomon	2	100	4	437.02	432.17	6.60	19.00	52.63	47.37	86.75	96.15
rc108	Solomon	2	100	8	459.06	439.54	11.76	20.00	75.00	25.00	93.61	98.28
rc108	Solomon	3	100	4	632.80	627.65	15.29	29.00	62.07	37.93	92.54	94.24
rc108	Solomon	3	100	8	666.19	634.37	19.51	28.00	57.14	42.86	88.47	98.66
rc108	Solomon	4	100	4	826.37	822.66	24.42	37.00	51.35	48.65	84.02	97.20
rc108	Solomon	4	100	8	839.93	811.91	34.45	37.00	59.46	40.54	88.44	98.18
Avg					540.42	530.04	17.05	22.90	65.75	34.25	83.33	95.91

Instance id	type	$\|K\|$	$\|N\|$	periods	Θ^{best}	Θ^{avg}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)

Table 5.20: Detailed results for the Solomon large-scale instances solved for the C-TOP-TDPLSF-nw by ILS-noWait.

Instance id	type	$\lvert K\rvert$	$\lvert N\rvert$	periods	Θ^{best}	Θ^{avg}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)
pr01	Cordeau	1	48	4	225.85	224.30	4.10	31.25	73.33	26.67	65.69	54.36
pr01	Cordeau	1	48	8	232.26	220.61	4.08	29.17	57.14	42.86	62.88	54.43
pr01	Cordeau	2	48	4	377.60	361.21	9.48	52.08	64.00	36.00	63.73	59.21
pr01	Cordeau	2	48	8	366.13	329.87	9.27	52.08	52.00	48.00	56.79	50.84
pr01	Cordeau	3	48	4	490.93	482.85	15.13	72.92	57.14	42.86	61.00	56.22
pr01	Cordeau	3	48	8	485.12	465.81	20.78	70.83	47.06	52.94	52.01	53.64
pr01	Cordeau	4	48	4	555.86	548.84	15.10	87.50	59.52	40.48	52.31	55.00
pr01	Cordeau	4	48	8	543.35	526.42	19.12	87.50	50.00	50.00	47.99	51.19
pr02	Cordeau	1	96	4	327.79	326.94	17.92	19.79	84.21	15.79	81.55	66.30
pr02	Cordeau	1	96	8	305.60	294.88	25.37	17.71	58.82	41.18	61.31	51.47
pr02	Cordeau	2	96	4	585.92	572.22	48.86	35.42	67.65	32.35	63.37	60.24
pr02	Cordeau	2	96	8	553.32	542.57	71.36	33.33	62.50	37.50	62.17	57.62
pr02	Cordeau	3	96	4	769.11	750.70	83.53	47.92	73.91	26.09	61.66	60.93
pr02	Cordeau	3	96	8	726.76	705.38	88.95	46.88	75.56	24.44	52.40	56.04
pr02	Cordeau	4	96	4	903.98	884.06	140.94	59.38	70.18	29.82	54.13	59.94
pr02	Cordeau	4	96	8	853.64	841.83	177.38	55.21	73.58	26.42	51.18	56.16
pr03	Cordeau	1	144	4	305.35	299.58	22.33	11.81	70.59	29.41	78.36	66.24
pr03	Cordeau	1	144	8	310.99	296.12	23.52	11.81	58.82	41.18	66.60	61.90
pr03	Cordeau	2	144	4	573.12	557.80	80.69	20.83	63.33	36.67	65.05	62.90
pr03	Cordeau	2	144	8	564.61	532.10	81.71	20.14	79.31	20.69	60.35	65.83
pr03	Cordeau	3	144	4	800.78	780.18	226.25	34.03	63.27	36.73	75.82	60.68
pr03	Cordeau	3	144	8	771.79	749.53	225.09	34.03	63.27	36.73	67.98	63.17
pr03	Cordeau	4	144	4	959.38	929.57	319.11	41.67	70.00	30.00	65.26	60.25
pr03	Cordeau	4	144	8	924.09	898.36	307.62	40.28	58.62	41.38	63.17	63.68
pr04	Cordeau	1	192	4	367.38	359.60	47.92	9.90	63.16	36.84	84.41	64.81
pr04	Cordeau	1	192	8	344.18	331.36	40.13	9.38	61.11	38.89	80.18	59.20
pr04	Cordeau	2	192	4	679.99	664.70	172.66	19.27	67.57	32.43	90.92	61.73
pr04	Cordeau	2	192	8	643.71	599.40	163.40	19.27	59.46	40.54	75.33	63.84
pr04	Cordeau	3	192	4	942.11	921.62	277.81	27.60	66.04	33.96	85.08	63.14
pr04	Cordeau	3	192	8	894.04	878.81	360.13	24.48	68.09	31.91	69.08	65.63
pr04	Cordeau	4	192	4	1174.32	1156.67	552.73	34.38	62.12	37.88	74.08	63.17
pr04	Cordeau	4	192	8	1132.12	1089.77	574.92	31.77	65.57	34.43	69.79	65.45
pr05	Cordeau	1	240	4	429.39	418.21	81.88	10.00	50.00	50.00	76.02	59.44
pr05	Cordeau	1	240	8	388.95	377.03	61.67	8.33	55.00	45.00	57.61	63.87
pr05	Cordeau	2	240	4	834.87	811.42	476.81	20.00	66.67	33.33	76.66	60.53
pr05	Cordeau	2	240	8	750.43	731.76	403.25	17.08	58.54	41.46	66.34	63.84
pr05	Cordeau	3	240	4	1125.23	1104.67	980.38	25.83	67.74	32.26	59.19	62.37
pr05	Cordeau	3	240	8	1055.44	1031.09	876.85	24.17	63.79	36.21	58.67	62.69
pr05	Cordeau	4	240	4	1386.91	1360.95	848.83	32.08	66.23	33.77	57.29	61.64
pr05	Cordeau	4	240	8	1351.85	1300.27	1222.15	30.83	67.57	32.43	55.81	59.81
pr07	Cordeau	1	72	4	230.45	230.45	3.13	16.67	75.00	25.00	71.93	57.03
pr07	Cordeau	1	72	8	229.30	225.58	6.46	18.06	76.92	23.08	66.49	61.74
pr07	Cordeau	2	72	4	446.12	434.10	15.27	36.11	61.54	38.46	81.74	60.28
pr07	Cordeau	2	72	8	425.67	407.76	16.38	34.72	64.00	36.00	85.42	60.16
pr07	Cordeau	3	72	4	599.33	577.06	41.88	51.39	70.27	29.73	80.15	58.76
pr07	Cordeau	3	72	8	602.45	551.58	33.56	50.00	61.11	38.89	76.70	62.77
pr07	Cordeau	4	72	4	685.39	673.10	47.13	61.11	65.91	34.09	70.81	60.14
pr07	Cordeau	4	72	8	695.34	663.80	63.68	62.50	62.22	37.78	76.43	60.87
pr08	Cordeau	1	144	4	352.46	341.70	26.77	11.81	58.82	41.18	67.42	64.15
pr08	Cordeau	1	144	8	339.91	317.07	25.21	11.81	52.94	47.06	67.04	61.34
pr08	Cordeau	2	144	4	639.94	629.54	103.34	22.22	56.25	43.75	68.16	61.83
pr08	Cordeau	2	144	8	595.02	571.93	107.22	22.22	56.25	43.75	69.99	59.96
pr08	Cordeau	3	144	4	881.68	844.22	187.43	33.33	72.92	27.08	73.10	64.80
pr08	Cordeau	3	144	8	841.71	812.09	229.43	33.33	64.58	35.42	66.45	59.76
pr08	Cordeau	4	144	4	1080.58	1054.52	341.35	41.67	71.67	28.33	68.87	62.03
pr08	Cordeau	4	144	8	1058.06	1008.29	420.43	40.97	64.41	35.59	64.93	61.25
pr09	Cordeau	1	216	4	328.67	320.63	36.50	8.80	68.42	31.58	55.22	58.93
pr09	Cordeau	1	216	8	338.67	321.56	30.64	7.87	76.47	23.53	53.71	69.95
pr09	Cordeau	2	216	4	660.75	636.49	181.45	18.06	76.92	23.08	66.11	61.69
pr09	Cordeau	2	216	8	600.33	582.57	201.20	15.28	63.64	36.36	51.75	59.04
pr09	Cordeau	3	216	4	935.72	912.91	533.09	25.00	61.11	38.89	59.16	63.40
pr09	Cordeau	3	216	8	882.38	871.35	420.29	23.15	60.00	40.00	55.54	66.58
pr09	Cordeau	4	216	4	1167.09	1129.02	609.48	32.41	68.57	31.43	59.12	62.94
pr09	Cordeau	4	216	8	1146.87	1100.80	595.35	30.09	66.15	33.85	52.32	63.96
Avg					668.41	648.08	210.25	32.29	64.67	35.33	66.22	60.82

Table 5.21: Detailed results for the Cordeau large-scale instances solved for the C-TOP-TDPLSF-nw by ILS-noWait.

Instance id	type	$\|K\|$	$\|N\|$	periods	Θ^{best}	Θ^{best}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)	waiting cust(%)	waiting time(%)
c101	Solomon	1	100	4	270.43	270.43	84.61	9.00	44.44	55.56	82.80	90.64	4.00	9.17
c101	Solomon	1	100	8	280.47	280.47	84.49	9.00	33.33	66.67	93.84	92.67	5.00	8.70
c101	Solomon	2	100	4	492.91	490.55	148.55	18.00	55.56	44.44	86.62	94.29	10.00	10.19
c101	Solomon	2	100	8	528.35	526.79	126.23	20.00	50.00	50.00	82.27	96.81	13.00	7.30
c101	Solomon	3	100	4	710.54	701.97	151.84	27.00	59.26	40.74	87.40	94.51	14.00	11.65
c101	Solomon	3	100	8	753.46	744.68	166.31	30.00	56.67	43.33	94.55	94.12	15.00	4.25
c101	Solomon	4	100	4	875.97	862.25	219.30	36.00	58.33	41.67	94.43	90.72	18.00	9.44
c101	Solomon	4	100	8	920.69	909.96	203.00	37.00	64.86	35.14	100.42	90.41	18.00	6.07
c102	Solomon	1	100	4	301.19	300.94	80.63	11.00	45.45	54.55	93.00	99.97	3.00	1.54
c102	Solomon	1	100	8	330.03	327.35	87.75	10.00	70.00	30.00	95.50	99.97	5.00	6.01
c102	Solomon	2	100	4	571.05	559.26	137.53	20.00	55.00	45.00	96.00	99.37	7.00	3.22
c102	Solomon	2	100	8	607.34	599.82	144.76	21.00	76.19	23.81	98.00	97.06	9.00	3.17
c102	Solomon	3	100	4	769.54	766.64	185.45	31.00	54.84	45.16	94.33	99.74	7.00	3.54
c102	Solomon	3	100	8	850.03	845.17	188.56	31.00	74.19	25.81	91.92	99.18	9.00	2.54
c102	Solomon	4	100	4	954.19	947.64	261.65	41.00	51.22	48.78	93.00	99.05	14.00	4.26
c102	Solomon	4	100	8	1044.04	1036.58	264.61	42.00	76.19	23.81	92.25	99.07	14.00	3.58
c103	Solomon	1	100	4	365.59	365.60	104.36	10.00	70.00	30.00	72.34	99.44	2.00	0.56
c103	Solomon	1	100	8	368.45	368.26	77.64	10.00	50.00	50.00	86.33	99.99	2.00	1.23
c103	Solomon	2	100	4	661.71	659.55	143.14	20.00	65.00	35.00	84.66	97.81	6.00	3.90
c103	Solomon	2	100	8	650.13	645.98	156.60	22.00	68.18	31.82	92.21	99.93	4.00	0.82
c103	Solomon	3	100	4	904.42	896.38	193.81	32.00	65.62	34.38	90.30	98.52	7.00	2.30
c103	Solomon	3	100	8	885.14	873.08	246.63	30.00	70.00	30.00	87.12	98.06	10.00	3.80
c103	Solomon	4	100	4	1091.36	1083.49	270.34	42.00	71.43	28.57	87.00	97.89	8.00	3.21
c103	Solomon	4	100	8	1080.92	1074.34	317.20	41.00	60.98	39.02	87.52	98.63	9.00	2.77
c104	Solomon	1	100	4	379.43	378.93	89.72	10.00	50.00	50.00	75.42	99.02	2.00	0.93
c104	Solomon	1	100	8	386.42	381.28	77.08	11.00	63.64	36.36	67.39	99.94	3.00	0.56
c104	Solomon	2	100	4	691.66	686.87	132.89	21.00	57.14	42.86	77.34	99.50	3.00	0.59
c104	Solomon	2	100	8	698.43	694.93	149.56	21.00	76.19	23.81	78.85	99.83	2.00	0.26
c104	Solomon	3	100	4	930.08	921.69	206.41	33.00	66.67	33.33	88.29	99.75	5.00	0.62
c104	Solomon	3	100	8	947.39	939.31	247.86	33.00	75.76	24.24	89.19	99.78	5.00	1.00
c104	Solomon	4	100	4	1121.35	1118.36	347.07	43.00	58.14	41.86	87.37	98.65	7.00	1.73
c104	Solomon	4	100	8	1157.07	1148.04	306.06	42.00	83.33	16.67	84.41	98.88	8.00	2.06
c105	Solomon	1	100	4	290.07	290.07	92.86	9.00	22.22	77.78	74.62	97.80	7.00	9.33
c105	Solomon	1	100	8	296.33	296.33	97.74	10.00	40.00	60.00	98.66	94.95	4.00	1.21
c105	Solomon	2	100	4	548.20	547.97	116.92	19.00	52.63	47.37	88.93	95.60	9.00	7.85
c105	Solomon	2	100	8	554.14	551.39	113.88	20.00	45.00	55.00	93.23	96.20	8.00	4.98
c105	Solomon	3	100	4	757.42	753.86	143.24	28.00	60.71	39.29	86.26	94.43	11.00	7.48
c105	Solomon	3	100	8	786.73	785.81	182.76	29.00	65.52	34.48	97.96	96.52	10.00	2.51
c105	Solomon	4	100	4	937.42	928.21	200.66	37.00	56.76	43.24	86.83	93.97	11.00	3.86
c105	Solomon	4	100	8	975.88	960.64	229.87	38.00	60.53	39.47	93.37	93.90	12.00	3.40
c106	Solomon	1	100	4	307.29	307.29	81.84	10.00	70.00	30.00	82.52	98.00	4.00	7.57
c106	Solomon	1	100	8	299.08	299.08	86.60	10.00	80.00	20.00	95.92	98.69	3.00	6.59
c106	Solomon	2	100	4	573.40	573.40	131.42	20.00	40.00	60.00	90.85	98.97	6.00	2.71
c106	Solomon	2	100	8	561.33	549.52	123.34	20.00	50.00	50.00	87.09	96.72	9.00	5.20
c106	Solomon	3	100	4	796.82	786.31	182.09	30.00	46.67	53.33	91.45	99.29	11.00	4.25
c106	Solomon	3	100	8	776.26	760.05	188.40	30.00	56.67	43.33	93.25	98.18	10.00	5.55
c106	Solomon	4	100	4	979.06	972.48	236.16	38.00	50.00	50.00	84.68	92.72	13.00	5.30
c106	Solomon	4	100	8	946.36	939.23	248.60	38.00	68.42	31.58	85.58	95.26	7.00	3.17
c107	Solomon	1	100	4	325.57	325.57	87.35	10.00	50.00	50.00	75.50	99.77	5.00	4.25

Instance id	type	$\lvert K\rvert$	$\lvert N\rvert$	periods	Θ^{best}	Θ^{best}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)	waiting cust(%)	waiting time(%)
c107	Solomon	1	100	8	330.02	330.04	84.43	11.00	63.64	36.36	73.67	99.77	3.00	1.92
c107	Solomon	2	100	4	593.54	592.90	135.08	20.00	70.00	30.00	85.58	95.47	6.00	2.36
c107	Solomon	2	100	8	596.21	588.50	125.51	21.00	57.14	42.86	88.00	96.94	7.00	1.94
c107	Solomon	3	100	4	819.30	810.27	168.26	31.00	77.42	22.58	81.67	96.32	5.00	2.33
c107	Solomon	3	100	8	825.80	819.18	177.22	31.00	54.84	45.16	87.50	97.61	10.00	3.68
c107	Solomon	4	100	4	1007.43	1002.62	254.68	41.00	60.98	39.02	84.92	94.13	9.00	2.46
c107	Solomon	4	100	8	1009.78	1003.84	243.54	41.00	63.41	36.59	83.67	96.43	11.00	5.47
c108	Solomon	1	100	4	299.84	299.84	83.50	10.00	30.00	70.00	98.21	92.49	2.00	5.85
c108	Solomon	1	100	8	310.00	309.47	78.99	10.00	90.00	10.00	98.66	94.69	2.00	2.70
c108	Solomon	2	100	4	571.69	569.41	117.69	19.00	63.16	36.84	94.64	95.93	5.00	3.58
c108	Solomon	2	100	8	574.84	573.39	126.08	19.00	42.11	57.89	90.85	95.27	7.00	9.17
c108	Solomon	3	100	4	782.93	779.47	169.08	29.00	58.62	41.38	97.10	93.72	11.00	4.76
c108	Solomon	3	100	8	789.00	777.12	175.28	29.00	75.86	24.14	99.26	94.05	5.00	3.81
c108	Solomon	4	100	4	969.63	962.98	238.76	39.00	64.10	35.90	97.82	92.16	9.00	1.64
c108	Solomon	4	100	8	968.95	956.76	223.83	37.00	70.27	29.73	97.27	91.19	11.00	6.06
c109	Solomon	1	100	4	345.73	345.73	80.82	10.00	70.00	30.00	72.49	93.86	1.00	1.61
c109	Solomon	1	100	8	350.05	350.05	78.80	10.00	70.00	30.00	94.67	94.93	2.00	2.10
c109	Solomon	2	100	4	629.76	621.90	113.91	19.00	78.95	21.05	97.78	95.42	6.00	6.39
c109	Solomon	2	100	8	639.04	634.42	137.85	19.00	68.42	31.58	90.38	95.61	5.00	2.49
c109	Solomon	3	100	4	838.52	820.78	173.20	30.00	56.67	43.33	98.32	96.93	7.00	2.13
c109	Solomon	3	100	8	857.21	844.02	204.95	29.00	55.17	44.83	96.84	94.22	8.00	5.58
c109	Solomon	4	100	4	1099.19	1088.05	318.13	44.00	63.64	36.36	165.38	98.54	8.00	0.77
c109	Solomon	4	100	8	1073.97	1070.58	296.42	41.00	65.85	34.15	141.79	97.14	12.00	0.96
r101	Solomon	1	100	4	173.00	173.00	72.85	8.00	50.00	50.00	98.52	98.42	6.00	11.40
r101	Solomon	1	100	8	187.00	187.00	75.45	8.00	62.50	37.50	71.60	98.71	4.00	1.22
r101	Solomon	2	100	4	313.00	309.50	114.27	14.00	50.00	50.00	92.35	91.87	10.00	18.02
r101	Solomon	2	100	8	323.00	323.00	126.10	15.00	53.33	46.67	88.52	98.77	10.00	5.14
r101	Solomon	3	100	4	440.00	436.69	145.03	18.00	55.56	44.44	78.19	90.04	15.00	26.13
r101	Solomon	3	100	8	446.80	443.06	138.38	21.00	52.38	47.62	74.90	90.11	15.00	14.67
r101	Solomon	4	100	4	548.00	543.00	159.71	25.00	56.00	44.00	78.52	91.88	21.00	19.21
r101	Solomon	4	100	8	558.10	549.89	174.18	28.00	57.14	42.86	76.48	91.62	21.00	9.39
r102	Solomon	1	100	4	248.35	247.89	79.47	10.00	80.00	20.00	83.61	99.91	1.00	0.60
r102	Solomon	1	100	8	244.78	242.75	78.70	11.00	36.36	63.64	95.04	99.58	5.00	6.69
r102	Solomon	2	100	4	444.89	442.15	129.42	19.00	63.16	36.84	85.71	99.86	6.00	2.24
r102	Solomon	2	100	8	455.44	453.13	123.11	20.00	50.00	50.00	90.08	97.96	9.00	5.34
r102	Solomon	3	100	4	605.95	603.31	145.22	28.00	60.71	39.29	85.86	98.30	9.00	3.77
r102	Solomon	3	100	8	629.99	629.82	152.09	28.00	50.00	50.00	83.41	98.53	13.00	5.90
r102	Solomon	4	100	4	742.48	734.41	223.78	35.00	68.57	31.43	77.74	98.70	11.00	3.86
r102	Solomon	4	100	8	762.62	760.48	196.43	35.00	54.29	45.71	77.71	99.12	16.00	7.11
r103	Solomon	1	100	4	262.00	262.00	73.23	10.00	50.00	50.00	99.26	99.58	1.00	0.63
r103	Solomon	1	100	8	265.01	263.61	79.59	10.00	80.00	20.00	94.85	99.58	3.00	3.92
r103	Solomon	2	100	4	450.34	448.84	121.86	19.00	57.89	42.11	92.74	99.78	4.00	2.58
r103	Solomon	2	100	8	467.01	462.80	146.99	19.00	68.42	31.58	88.33	98.66	7.00	2.76
r103	Solomon	3	100	4	627.64	622.40	167.04	29.00	68.97	31.03	90.81	99.33	6.00	1.89
r103	Solomon	3	100	8	650.71	641.24	178.76	30.00	63.33	36.67	92.34	99.05	7.00	1.92
r103	Solomon	4	100	4	784.91	774.94	209.51	39.00	71.79	28.21	92.14	98.84	8.00	1.69
r103	Solomon	4	100	8	828.32	813.34	252.74	39.00	51.28	48.72	90.03	99.33	14.00	3.79
r104	Solomon	1	100	4	270.67	270.67	78.51	11.00	72.73	27.27	91.29	99.59	4.00	3.77
r104	Solomon	1	100	8	277.99	275.31	72.86	11.00	81.82	18.18	84.22	99.47	2.00	1.84

Instance id	type	$\|K\|$	$\|N\|$	periods	Θ^{best}	Θ^{best}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)	waiting cust(%)	waiting time(%)
r104	Solomon	2	100	4	481.81	480.47	141.04	21.00	71.43	28.57	89.73	99.62	5.00	2.64
r104	Solomon	2	100	8	498.16	495.76	122.32	20.00	70.00	30.00	80.34	99.62	4.00	1.07
r104	Solomon	3	100	4	674.29	669.72	192.80	30.00	53.33	46.67	77.46	99.67	8.00	3.06
r104	Solomon	3	100	8	680.47	668.80	211.68	30.00	73.33	26.67	78.59	99.61	7.00	3.79
r104	Solomon	4	100	4	837.89	834.19	232.35	41.00	63.41	36.59	88.30	98.96	11.00	3.01
r104	Solomon	4	100	8	837.75	823.06	299.27	41.00	65.85	34.15	77.59	99.24	11.00	4.55
r105	Solomon	1	100	4	216.69	216.69	99.05	9.00	33.33	66.67	80.98	99.59	2.00	4.15
r105	Solomon	1	100	8	207.58	207.58	80.09	9.00	55.56	44.44	82.75	96.68	7.00	6.31
r105	Solomon	2	100	4	386.62	386.62	131.78	18.00	50.00	50.00	97.16	97.44	6.00	10.63
r105	Solomon	2	100	8	386.74	378.95	118.47	17.00	41.18	58.82	90.69	93.28	10.00	10.87
r105	Solomon	3	100	4	530.05	527.42	144.97	24.00	58.33	41.67	88.50	93.96	12.00	13.60
r105	Solomon	3	100	8	539.35	537.37	155.87	24.00	37.50	62.50	90.52	90.38	12.00	14.77
r105	Solomon	4	100	4	674.97	669.99	180.79	34.00	58.82	41.18	80.78	94.13	10.00	6.94
r105	Solomon	4	100	8	679.71	677.31	194.17	34.00	44.12	55.88	80.44	92.33	17.00	9.96
r106	Solomon	1	100	4	253.96	250.06	81.63	12.00	50.00	50.00	98.52	99.45	2.00	0.17
r106	Solomon	1	100	8	246.45	243.88	76.80	10.00	60.00	40.00	93.51	100.00	1.00	0.74
r106	Solomon	2	100	4	437.95	434.65	127.11	18.00	72.22	27.78	66.70	99.65	3.00	2.08
r106	Solomon	2	100	8	453.89	453.04	132.16	20.00	60.00	40.00	83.86	99.80	5.00	1.22
r106	Solomon	3	100	4	634.17	627.47	183.01	30.00	56.67	43.33	78.05	99.66	5.00	1.63
r106	Solomon	3	100	8	632.39	620.38	180.92	30.00	53.33	46.67	82.25	99.72	8.00	1.38
r106	Solomon	4	100	4	799.23	795.90	221.07	39.00	51.28	48.72	77.09	96.96	9.00	4.74
r106	Solomon	4	100	8	789.40	781.20	225.12	38.00	73.68	26.32	78.71	96.64	10.00	6.86
r107	Solomon	1	100	4	259.00	257.98	76.35	11.00	100.00	0.00	81.78	98.92	0.00	0.00
r107	Solomon	1	100	8	267.01	267.01	73.13	10.00	70.00	30.00	85.87	99.67	3.00	2.43
r107	Solomon	2	100	4	455.45	453.87	128.73	20.00	85.00	15.00	91.64	99.25	2.00	1.19
r107	Solomon	2	100	8	465.69	460.82	119.69	19.00	68.42	31.58	89.50	99.65	5.00	2.36
r107	Solomon	3	100	4	649.76	647.59	177.78	30.00	83.33	16.67	90.71	97.17	3.00	1.65
r107	Solomon	3	100	8	652.61	649.27	175.40	27.00	62.96	37.04	89.78	97.71	8.00	3.49
r107	Solomon	4	100	4	832.33	823.69	235.96	39.00	79.49	20.51	90.71	95.84	4.00	3.66
r107	Solomon	4	100	8	801.52	798.96	222.21	35.00	74.29	25.71	83.97	97.77	9.00	2.80
r108	Solomon	1	100	4	270.00	270.00	86.43	11.00	90.91	9.09	90.04	99.63	1.00	0.98
r108	Solomon	1	100	8	273.68	271.13	79.79	11.00	72.73	27.27	93.14	99.70	2.00	0.37
r108	Solomon	2	100	4	476.10	472.18	138.30	21.00	85.71	14.29	94.25	98.69	2.00	0.86
r108	Solomon	2	100	8	494.64	491.39	141.61	21.00	76.19	23.81	87.94	98.65	3.00	1.23
r108	Solomon	3	100	4	673.67	657.56	174.20	29.00	79.31	20.69	93.58	99.66	4.00	0.53
r108	Solomon	3	100	8	695.44	693.63	188.98	31.00	83.87	16.13	91.89	97.00	4.00	1.19
r108	Solomon	4	100	4	841.19	827.61	235.64	38.00	73.68	26.32	93.58	98.84	5.00	1.26
r108	Solomon	4	100	8	867.96	859.39	254.84	39.00	79.49	20.51	94.25	98.05	8.00	2.41
r109	Solomon	1	100	4	248.90	248.05	97.88	12.00	58.33	41.67	92.94	99.59	2.00	1.20
r109	Solomon	1	100	8	251.18	248.68	100.83	11.00	36.36	63.64	95.63	100.00	4.00	3.07
r109	Solomon	2	100	4	430.65	429.54	121.77	19.00	63.16	36.84	84.54	98.33	6.00	3.31
r109	Solomon	2	100	8	453.62	448.10	121.86	20.00	40.00	60.00	89.50	99.14	6.00	5.28
r109	Solomon	3	100	4	600.43	598.65	169.37	27.00	70.37	29.63	82.97	95.82	6.00	2.77
r109	Solomon	3	100	8	625.54	617.43	166.07	29.00	44.83	55.17	88.63	96.64	8.00	4.76
r109	Solomon	4	100	4	738.42	730.98	189.25	37.00	64.86	35.14	90.63	96.22	5.00	3.28
r109	Solomon	4	100	8	770.09	759.17	201.98	38.00	50.00	50.00	83.78	95.75	11.00	3.18
r110	Solomon	1	100	4	240.00	238.23	81.44	11.00	72.73	27.27	87.34	100.00	2.00	2.78
r110	Solomon	1	100	8	251.16	248.49	94.40	11.00	54.55	45.45	89.45	99.59	2.00	1.32
r110	Solomon	2	100	4	440.93	440.57	119.09	18.00	83.33	16.67	63.01	97.81	0.00	0.00

Instance id	type	$\|K\|$	$\|N\|$	periods	Θ^{best}	Θ^{best}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)	waiting cust(%)	waiting time(%)
r110	Solomon	2	100	8	466.83	459.01	140.68	20.00	65.00	35.00	71.45	99.82	3.00	1.43
r110	Solomon	3	100	4	625.34	621.76	163.25	30.00	73.33	26.67	70.28	98.88	3.00	0.22
r110	Solomon	3	100	8	645.73	636.73	177.72	30.00	56.67	43.33	74.50	99.73	6.00	2.38
r110	Solomon	4	100	4	784.73	780.52	225.36	38.00	68.42	31.58	69.59	98.88	7.00	3.18
r110	Solomon	4	100	8	794.73	789.73	229.31	37.00	59.46	40.54	65.82	98.08	8.00	2.78
r111	Solomon	1	100	4	262.73	262.33	81.04	11.00	63.64	36.36	76.04	99.59	2.00	3.06
r111	Solomon	1	100	8	261.70	260.47	73.20	11.00	45.45	54.55	80.75	99.64	4.00	3.63
r111	Solomon	2	100	4	459.46	458.73	121.66	20.00	50.00	50.00	69.81	99.62	5.00	5.37
r111	Solomon	2	100	8	471.05	460.67	133.01	20.00	65.00	35.00	69.32	99.62	3.00	1.84
r111	Solomon	3	100	4	649.20	645.80	178.33	31.00	58.06	41.94	75.99	99.83	6.00	2.10
r111	Solomon	3	100	8	664.01	658.36	178.40	30.00	56.67	43.33	76.45	98.08	5.00	1.81
r111	Solomon	4	100	4	817.79	802.88	231.05	42.00	45.24	54.76	78.46	99.79	7.00	1.58
r111	Solomon	4	100	8	812.21	799.36	250.64	40.00	57.50	42.50	75.87	99.31	10.00	3.56
r112	Solomon	1	100	4	258.31	257.10	77.62	11.00	63.64	36.36	99.41	97.43	1.00	0.72
r112	Solomon	1	100	8	262.78	260.23	83.31	11.00	72.73	27.27	96.74	97.72	3.00	2.95
r112	Solomon	2	100	4	462.91	461.65	145.80	20.00	55.00	45.00	80.96	99.09	3.00	0.44
r112	Solomon	2	100	8	467.52	462.02	123.75	20.00	65.00	35.00	81.48	99.21	5.00	1.74
r112	Solomon	3	100	4	664.39	655.49	183.88	30.00	66.67	33.33	71.80	96.93	3.00	0.31
r112	Solomon	3	100	8	665.84	654.97	185.42	30.00	63.33	36.67	79.16	99.22	9.00	2.32
r112	Solomon	4	100	4	844.96	829.98	227.59	41.00	73.17	26.83	68.44	98.17	6.00	1.30
r112	Solomon	4	100	8	831.96	823.65	277.83	41.00	68.29	31.71	77.48	96.94	8.00	0.90
rc101	Solomon	1	100	4	192.01	192.01	78.67	8.00	75.00	25.00	91.25	93.51	4.00	11.09
rc101	Solomon	1	100	8	201.83	201.83	82.41	9.00	44.44	55.56	99.80	95.24	4.00	12.92
rc101	Solomon	2	100	4	363.13	361.56	126.78	16.00	56.25	43.75	70.08	96.19	7.00	6.45
rc101	Solomon	2	100	8	381.98	381.98	122.33	18.00	55.56	44.44	97.12	91.67	5.00	8.23
rc101	Solomon	3	100	4	520.06	518.45	146.07	23.00	52.17	47.83	77.07	93.25	9.00	7.08
rc101	Solomon	3	100	8	537.33	532.80	159.69	25.00	60.00	40.00	91.72	95.49	11.00	5.19
rc101	Solomon	4	100	4	667.76	661.32	202.86	33.00	51.52	48.48	80.52	93.50	14.00	7.30
rc101	Solomon	4	100	8	681.97	670.14	193.67	32.00	56.25	43.75	90.31	93.61	13.00	4.87
rc102	Solomon	1	100	4	219.12	217.72	95.50	10.00	60.00	40.00	97.65	99.85	2.00	2.50
rc102	Solomon	1	100	8	222.94	222.94	80.89	8.00	37.50	62.50	78.68	99.85	3.00	5.00
rc102	Solomon	2	100	4	414.75	413.58	123.95	20.00	50.00	50.00	87.46	98.25	3.00	0.50
rc102	Solomon	2	100	8	434.56	434.56	106.80	16.00	50.00	50.00	77.51	98.11	8.00	6.86
rc102	Solomon	3	100	4	603.02	598.05	163.43	29.00	62.07	37.93	92.79	98.80	5.00	1.22
rc102	Solomon	3	100	8	625.61	625.61	135.61	25.00	52.00	48.00	82.71	98.45	12.00	6.51
rc102	Solomon	4	100	4	776.23	766.64	208.37	36.00	61.11	38.89	79.94	98.59	11.00	5.25
rc102	Solomon	4	100	8	792.47	774.24	193.44	35.00	51.43	48.57	82.17	98.75	15.00	5.07
rc103	Solomon	1	100	4	226.00	223.42	89.09	9.00	88.89	11.11	82.29	95.62	1.00	3.06
rc103	Solomon	1	100	8	224.00	222.14	76.36	9.00	66.67	33.33	74.00	99.85	2.00	3.13
rc103	Solomon	2	100	4	436.92	434.34	129.90	17.00	88.24	11.76	83.86	97.00	1.00	1.51
rc103	Solomon	2	100	8	437.74	437.74	110.80	16.00	68.75	31.25	87.14	98.89	3.00	2.25
rc103	Solomon	3	100	4	619.55	614.40	148.52	27.00	66.67	33.33	87.62	98.59	6.00	2.04
rc103	Solomon	3	100	8	635.06	630.90	149.44	24.00	62.50	37.50	90.38	99.16	6.00	2.92
rc103	Solomon	4	100	4	798.26	785.45	200.47	37.00	62.16	37.84	92.64	97.50	6.00	2.11
rc103	Solomon	4	100	8	817.46	805.81	206.10	35.00	65.71	34.29	90.93	98.18	8.00	4.31
rc104	Solomon	1	100	4	254.20	252.80	76.68	9.00	66.67	33.33	86.64	100.00	1.00	1.20
rc104	Solomon	1	100	8	258.23	256.56	80.10	10.00	50.00	50.00	99.50	99.92	2.00	1.56
rc104	Solomon	2	100	4	479.61	478.68	121.39	18.00	55.56	44.44	88.23	99.93	3.00	1.27
rc104	Solomon	2	100	8	495.24	483.85	133.66	20.00	70.00	30.00	98.41	98.75	3.00	1.81

Instance id	type	$\|K\|$	$\|N\|$	periods	Θ^{best}	Θ^{best}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)	waiting cust(%)	waiting time(%)
rc104	Solomon	3	100	4	700.82	691.27	154.67	28.00	67.86	32.14	78.24	99.88	5.00	0.99
rc104	Solomon	3	100	8	713.23	704.38	181.93	29.00	75.86	24.14	92.54	98.55	5.00	2.28
rc104	Solomon	4	100	4	898.13	894.63	213.81	39.00	69.23	30.77	79.42	99.73	7.00	1.01
rc104	Solomon	4	100	8	892.98	884.07	212.23	36.00	72.22	27.78	81.18	99.43	7.00	2.59
rc105	Solomon	1	100	4	204.83	199.73	82.50	11.00	63.64	36.36	94.60	97.80	1.00	2.19
rc105	Solomon	1	100	8	218.00	218.00	85.22	10.00	70.00	30.00	83.21	98.94	4.00	3.72
rc105	Solomon	2	100	4	395.09	388.80	118.44	18.00	55.56	44.44	63.42	96.64	4.00	2.86
rc105	Solomon	2	100	8	414.16	406.89	121.32	20.00	60.00	40.00	87.18	98.00	5.00	2.01
rc105	Solomon	3	100	4	576.90	569.95	151.75	27.00	55.56	44.44	67.77	96.72	6.00	2.46
rc105	Solomon	3	100	8	597.52	597.32	155.87	28.00	67.86	32.14	74.86	97.53	10.00	3.62
rc105	Solomon	4	100	4	750.02	739.78	202.71	36.00	61.11	38.89	76.84	96.57	8.00	2.69
rc105	Solomon	4	100	8	776.30	775.27	178.92	36.00	66.67	33.33	74.81	97.76	12.00	3.02
rc106	Solomon	1	100	4	203.93	203.93	101.23	8.00	25.00	75.00	83.26	97.44	3.00	4.93
rc106	Solomon	1	100	8	215.96	215.96	91.35	9.00	66.67	33.33	59.15	96.02	2.00	0.62
rc106	Solomon	2	100	4	405.87	405.19	112.34	17.00	41.18	58.82	77.00	97.48	4.00	2.68
rc106	Solomon	2	100	8	419.02	414.18	123.77	17.00	70.59	29.41	62.36	96.96	5.00	0.96
rc106	Solomon	3	100	4	597.99	592.00	153.12	27.00	44.44	55.56	76.26	98.31	9.00	3.68
rc106	Solomon	3	100	8	611.70	610.58	162.71	27.00	51.85	48.15	74.75	97.49	9.00	2.04
rc106	Solomon	4	100	4	783.84	770.76	210.30	37.00	59.46	40.54	76.21	94.70	11.00	2.86
rc106	Solomon	4	100	8	777.00	770.73	212.85	34.00	79.41	20.59	69.80	95.71	9.00	3.99
rc107	Solomon	1	100	4	213.64	213.36	84.65	8.00	50.00	50.00	65.25	97.12	3.00	8.67
rc107	Solomon	1	100	8	239.23	236.99	73.05	10.00	60.00	40.00	99.19	98.54	3.00	3.08
rc107	Solomon	2	100	4	422.73	422.41	117.99	16.00	62.50	37.50	81.82	92.91	3.00	4.53
rc107	Solomon	2	100	8	449.97	445.89	124.05	19.00	42.11	57.89	95.56	97.47	6.00	3.70
rc107	Solomon	3	100	4	628.22	622.58	153.69	26.00	61.54	38.46	92.53	94.74	3.00	0.90
rc107	Solomon	3	100	8	653.68	644.98	162.31	28.00	67.86	32.14	92.59	97.86	5.00	3.42
rc107	Solomon	4	100	4	801.40	793.60	213.53	39.00	56.41	43.59	106.57	95.27	6.00	2.18
rc107	Solomon	4	100	8	848.13	840.64	223.70	35.00	62.86	37.14	94.34	97.07	7.00	2.94
rc108	Solomon	1	100	4	235.92	234.96	79.27	10.00	30.00	70.00	83.46	98.33	2.00	2.66
rc108	Solomon	1	100	8	246.53	246.53	73.75	9.00	66.67	33.33	69.55	99.85	2.00	1.41
rc108	Solomon	2	100	4	451.14	444.48	116.44	20.00	30.00	70.00	83.27	98.97	4.00	2.98
rc108	Solomon	2	100	8	469.45	464.59	125.13	18.00	66.67	33.33	83.83	99.09	6.00	2.51
rc108	Solomon	3	100	4	648.44	642.60	160.39	28.00	46.43	53.57	80.20	97.97	6.00	2.89
rc108	Solomon	3	100	8	696.50	684.73	209.64	30.00	60.00	40.00	93.55	99.34	9.00	2.06
rc108	Solomon	4	100	4	851.27	847.34	236.09	39.00	46.15	53.85	85.95	98.28	7.00	2.46
rc108	Solomon	4	100	8	875.41	851.34	247.31	40.00	65.00	35.00	95.54	98.73	8.00	1.60
Avg					572.69	567.76	153.59	23.80	61.05	38.95	86.24	97.46	6.59	3.83

Table 5.22: Detailed results for the Solomon large-scale instances solved for the C-TOP-TDPLSF-w by ILS-cWait-fin

Instance id	type	$\|K\|$	$\|N\|$	periods	Θ^{best}	Θ^{best}	$t^a(s)$	cust visited(%)	cust visited P(%)	cust visited LP(%)	avg load(%)	avg duration(%)	waiting cust(%)	waiting time(%)
pr01	Cordeau	1	48	4	246.21	246.21	35.32	33.33	62.50	37.50	78.87	65.57	8.33	13.16
pr01	Cordeau	1	48	8	261.33	261.33	42.40	35.42	64.71	35.29	81.79	59.68	12.50	12.63
pr01	Cordeau	2	48	4	406.75	405.06	43.42	56.25	55.56	44.44	64.08	61.55	20.83	13.45
pr01	Cordeau	2	48	8	430.62	423.84	61.00	62.50	46.67	53.33	71.23	60.16	25.00	14.81
pr01	Cordeau	3	48	4	538.50	534.09	58.21	79.17	50.00	50.00	66.06	57.10	25.00	17.52
pr01	Cordeau	3	48	8	545.60	534.79	69.58	79.17	76.32	23.68	59.36	59.59	29.17	22.61
pr01	Cordeau	4	48	4	628.44	623.05	64.69	93.75	62.22	37.78	55.26	57.51	31.25	20.11
pr01	Cordeau	4	48	8	630.85	623.99	86.94	91.67	61.36	38.64	52.19	61.55	39.58	16.28
pr02	Cordeau	1	96	4	346.76	344.59	89.40	21.88	61.90	38.10	82.52	66.48	4.17	4.56
pr02	Cordeau	1	96	8	322.60	321.62	82.57	18.75	66.67	33.33	71.62	51.95	2.08	15.79
pr02	Cordeau	2	96	4	603.35	602.19	189.96	37.50	75.00	25.00	65.94	60.97	8.33	10.22
pr02	Cordeau	2	96	8	606.41	596.16	165.43	35.42	58.82	41.18	64.26	61.24	11.46	9.37
pr02	Cordeau	3	96	4	810.73	807.14	222.40	48.96	74.47	25.53	59.67	61.64	12.50	10.91
pr02	Cordeau	3	96	8	800.57	781.26	224.42	48.96	63.83	36.17	63.60	59.79	8.33	10.10
pr02	Cordeau	4	96	4	963.01	950.42	296.52	63.54	68.85	31.15	57.95	61.12	17.71	12.12
pr02	Cordeau	4	96	8	929.79	918.84	308.57	59.38	70.18	29.82	55.60	58.38	12.50	12.33
pr03	Cordeau	1	144	4	312.40	311.41	153.44	12.50	55.56	44.44	73.85	64.51	2.08	4.33
pr03	Cordeau	1	144	8	350.23	350.23	149.15	13.89	65.00	35.00	81.28	62.10	3.47	9.92
pr03	Cordeau	2	144	4	603.76	597.80	236.53	22.92	63.64	36.36	67.97	63.85	5.56	8.90
pr03	Cordeau	2	144	8	637.14	627.75	282.27	26.39	71.05	28.95	76.15	62.80	6.25	7.57
pr03	Cordeau	3	144	4	832.78	825.88	346.38	34.72	56.00	44.00	68.96	60.22	9.72	13.87
pr03	Cordeau	3	144	8	875.73	866.90	439.88	36.11	63.46	36.54	70.65	64.07	9.72	12.85
pr03	Cordeau	4	144	4	1039.18	1021.69	511.07	45.83	63.64	36.36	69.24	59.53	11.81	15.67
pr03	Cordeau	4	144	8	1047.96	1026.02	596.29	47.22	60.29	39.71	71.68	63.76	12.50	10.30
pr04	Cordeau	1	192	4	411.86	410.17	236.62	11.46	68.18	31.82	85.41	64.85	2.08	5.52
pr04	Cordeau	1	192	8	394.54	388.54	265.81	10.94	71.43	28.57	77.62	67.24	3.65	4.13
pr04	Cordeau	2	192	4	751.97	744.38	531.70	21.88	61.90	38.10	74.89	65.66	5.21	6.21
pr04	Cordeau	2	192	8	751.43	741.54	697.16	21.35	56.10	43.90	82.80	65.27	5.73	3.43
pr04	Cordeau	3	192	4	1038.30	1012.87	637.32	31.77	65.57	34.43	78.84	64.13	6.25	5.21
pr04	Cordeau	3	192	8	1040.74	1024.82	809.08	32.29	53.23	46.77	93.95	67.81	7.81	5.87
pr04	Cordeau	4	192	4	1304.92	1275.87	1084.56	41.15	63.29	36.71	88.72	64.68	7.81	5.02
pr04	Cordeau	4	192	8	1317.72	1279.19	1373.94	40.62	64.10	35.90	88.22	64.88	11.98	9.76
pr05	Cordeau	1	240	4	450.20	447.37	453.85	10.42	68.00	32.00	78.19	57.93	1.25	12.75
pr05	Cordeau	1	240	8	456.07	450.14	383.92	11.25	59.26	40.74	89.51	64.27	2.08	8.10
pr05	Cordeau	2	240	4	863.50	850.69	742.36	20.00	68.75	31.25	74.48	62.22	3.75	11.26
pr05	Cordeau	2	240	8	860.32	853.75	837.33	18.75	68.89	31.11	67.26	64.60	5.83	13.32
pr05	Cordeau	3	240	4	1207.21	1192.08	1170.84	28.33	58.82	41.18	66.00	64.71	5.83	10.69
pr05	Cordeau	3	240	8	1173.66	1156.25	1242.57	27.50	57.58	42.42	72.80	62.10	5.83	11.30
pr05	Cordeau	4	240	4	1481.13	1463.84	1458.43	36.67	55.68	44.32	70.09	64.23	6.67	11.45
pr05	Cordeau	4	240	8	1446.64	1427.49	1549.20	34.58	62.65	37.35	61.28	64.71	8.33	11.65
pr07	Cordeau	1	72	4	242.25	242.25	53.35	18.06	76.92	23.08	99.73	64.10	4.17	8.86
pr07	Cordeau	1	72	8	259.65	259.65	61.43	19.44	42.86	57.14	93.87	66.26	6.94	3.94
pr07	Cordeau	2	72	4	469.14	466.72	79.39	38.89	60.71	39.29	91.21	63.38	6.94	5.37
pr07	Cordeau	2	72	8	485.12	478.62	79.91	38.89	60.71	39.29	86.38	64.18	15.28	12.06
pr07	Cordeau	3	72	4	623.40	622.03	92.51	54.17	58.97	41.03	88.83	61.88	12.50	7.50
pr07	Cordeau	3	72	8	663.20	656.00	109.99	52.78	68.42	31.58	83.79	63.89	15.28	10.61
pr07	Cordeau	4	72	4	746.04	739.97	141.30	69.44	68.00	32.00	84.57	61.35	13.89	8.22
pr07	Cordeau	4	72	8	781.82	773.75	156.13	68.06	65.31	34.69	85.73	62.35	25.00	11.06
pr08	Cordeau	1	144	4	381.08	373.35	157.18	14.58	52.38	47.62	88.11	66.46	4.17	6.23
pr08	Cordeau	1	144	8	381.32	379.26	154.62	12.50	61.11	38.89	50.47	62.04	4.17	14.14
pr08	Cordeau	2	144	4	698.23	691.90	241.24	27.08	56.41	43.59	80.20	64.59	6.94	8.01
pr08	Cordeau	2	144	8	701.73	684.47	317.55	26.39	57.89	42.11	75.09	61.18	8.33	12.26
pr08	Cordeau	3	144	4	943.55	934.40	427.27	36.81	56.60	43.40	71.25	63.43	10.42	7.72
pr08	Cordeau	3	144	8	955.13	929.91	424.99	38.19	50.91	49.09	74.34	63.08	10.42	9.94
pr08	Cordeau	4	144	4	1177.99	1171.93	651.59	47.92	57.97	42.03	69.80	65.00	11.11	7.86
pr08	Cordeau	4	144	8	1142.23	1125.42	627.74	46.53	55.22	44.78	76.47	62.11	12.50	8.72
pr09	Cordeau	1	216	4	383.20	377.47	288.35	9.26	60.00	40.00	60.99	60.06	2.31	8.95
pr09	Cordeau	1	216	8	394.57	383.24	275.41	10.19	81.82	18.18	66.62	61.24	1.39	7.72
pr09	Cordeau	2	216	4	725.16	708.50	560.03	21.30	60.87	39.13	72.32	64.60	3.24	6.41
pr09	Cordeau	2	216	8	717.55	714.91	578.10	17.59	63.16	36.84	57.80	69.79	3.70	5.42
pr09	Cordeau	3	216	4	1028.31	1016.66	903.51	28.24	54.10	45.90	70.54	62.30	5.56	9.31
pr09	Cordeau	3	216	8	1057.54	1008.42	834.00	28.24	57.38	42.62	62.89	66.70	8.80	8.04
pr09	Cordeau	4	216	4	1315.21	1287.40	1166.10	35.65	63.64	36.36	65.95	62.47	6.94	8.11
pr09	Cordeau	4	216	8	1302.47	1285.74	1247.36	35.65	66.23	33.77	63.10	65.14	9.26	9.78
Avg					739.01	728.64	435.31	35.94	62.17	37.83	73.44	62.84	9.83	10.02

Table 5.23: Detailed results for the Cordeau large-scale instances solved for the C-TOP-TDPLSF-w by ILS-cWait-fin.

Chapter 6

Conclusion and Outlook

This thesis contributes with the development of metaheuristics for four routing problems, each with unique complexities and practical applications. We show how tailored algorithmic design helps solving large-scale real-world instances of these routing problems, significantly improving solution quality and computational efficiency compared to state-of the art methods.

Motivated by the problem of routing mail carriers, we propose a metaheuristic to solve large-scale ASTTRPSD instances, called ILS-ASTTRPSD, making use of the natural decomposition of the problem into first-level-tour and second-level-tours. On real-world instances provided by DHL, our approach consistently outperforms existing solutions. This improvement can be explained by a difference in the solution structure where mail carriers spend less time driving and walking and stop at fewer parking spots, thereby also reducing stress originating from parking activities for mail carriers. In our computational experiments we also evaluate the impact of not considering parking and loading times. While ignoring both these time components results in higher travel times, the drawbacks of ignoring loading times are more limited than those of ignoring parking times. Future research could focus on the introduction of fluctuations in travel times or availability of parking spots during different times of the day.

Because besides to delivering letters, mail carriers also have to pick them up at the depot at so-called preparation tables, we introduce the VRPDOC, which can be used to model the problem of planning the routes of several mail carriers within a delivery area, including optimizing the household-to-mail carrier assignment. Because this problem can be solved both at the tactical and operational level, we propose two metaheuristics ILS-VRPDOC$_{quality}$ and ILS-VRPDOC$_{speed}$ to focus more on solution quality or runtime. On large-scale real-world instances, our approach returns solutions of better quality with respect to both, the total travel time of mail carriers and the complexity of the letter collections at the depot compared to the DHL solutions. For mail carriers, this means that letter collection is resulting in less confusion and mistakes and is leading to faster learning processes. Our numerical studies show that travel times could be reduced by 11% on average if depot operation constraints are ignored, however, ignoring them extremely complicates the letter collection at the tables. An interesting future research direction could be to solve the stochastic problem that aims on finding a robust assignment of the street segments to the tables considering the different demand scenarios, defined by the different number of mail carriers needed.

For the AngleTSP and the AngleDistanceTSP, we propose a heuristic based on a GTS framework that considers the geometric features of the problem into our construction heuristic and the sparsification methods, called GTS-angular. The computational experiments demonstrate that the performance of GTS-angular lies on the Pareto frontier of heuristic AngleTSP and AngleDistanceTSP methods. With GTS-angular, we find new best-known solutions on around 80% of AngleTSP and AngleDistanceTSP instances for which an optimal solution was not yet found.

Finally, we introduce the C-TOP-TDPLSF-nw and C-TOP-TDPLSF-w. To solve them in reasonable runtimes, we design two ILS variants, namely ILS-noWait and ILS-cWait-fin, that differ in how the waiting decisions are evaluated in the local search phase. ILS-noWait shows a good performance, but on ILS-cWait-fin there is still room for improvements. To improve the performance on ILS-cWait-fin we tried to solve the mathematical model that determines the optimal waiting times in the evaluation of every move in the ILS. However, this increases the runtime drastically. To overcome this problem, future research could focus on finding a better performing algorithm to determine optimal or near optimal waiting times in the evaluation of moves.

Collectively, all these studies demonstrate the importance of metaheuristic approaches to solve complex real-world routing problems.