KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Technology

Solving Vehicle Routing Problems

Jan Christiaens

Supervisors:
Prof. dr. ir. Greet Vanden Berghe
Prof. dr. Patrick De Causmaecker

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor of Engineering Technology (PhD)

May 2021

Solving Vehicle Routing Problems

Jan CHRISTIAENS

Examination committee:
Prof. dr. ir. Pieter Rauwoens, chair
Prof. dr. ir. Greet Vanden Berghe, supervisor
Prof. dr. Patrick De Causmaecker, supervisor
Prof. dr. ir. Giovanni Samaey
Prof. dr. Joost Vennekens
dr. Geir Hasle
(Sintef, Norway)
dr. Gilbert Laporte
(HEC Montréal, Canada)
dr. An De Wispelaere
(Conundra, Belgium)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor of Engineering Technology (PhD)

May 2021

© 2021 KU Leuven – Faculty of Engineering Technology
Uitgegeven in eigen beheer, Jan Christiaens, Celestijnenlaan 200A box 2402, B-3001 Leuven (Belgium)

Alle rechten voorbehouden. Niets uit deze uitgave mag worden vermenigvuldigd en/of openbaar gemaakt worden door middel van druk, fotokopie, microfilm, elektronisch of op welke andere wijze ook zonder voorafgaande schriftelijke toestemming van de uitgever.

All rights reserved. No part of the publication may be reproduced in any form by print, photoprint, microfilm, electronic or any other means without written permission from the publisher.

Preface

The motivation for this research originated from my strong interest in developing algorithms for solving combinatorial optimisation problems. Prior to obtaining my master's degree, I was offered a research position at the CODeS research group. While working on vehicle routing problems, I came in contact with the software company Conundra who required an algorithm to optimize the transportation of prisoners. Soon I was given the opportunity to commence a PhD in cooperation with Conundra. During my time as a researcher I came across many interesting vehicle routing problems. Modeling and solving these problems inspired me to develop a simple and flexible method for solving them. In this PhD thesis I introduce an example of one such inspirational problem in Chapter 4: the Prisoner Transportation Problem. The core contribution of this thesis is a state-of-the-art heuristic for solving a wide range of vehicle routing problems: Slack Induction by String Removals (SISRs). Chapters 4 and 5 have been published in the European Journal of Operational Research and Transportation Science, respectively. While the development of SISRs was a personal endeavour, the publication of the prisoner transportation problem was a joint effort in collaboration with Hatice Çalik, Tony Wauters, Reshma Chirayil Chandrasekharan and Greet Vanden Berghe.

Abstract

Vehicle Routing Problems (VRPs) are one of the most common logistics problems faced by a wide range of organisations. Parcel delivery organisations are confronted with challenging transportation puzzles due to the increasing number of parcels which must be delivered on a daily basis as a result of ever-increasing levels of online shopping. Even companies whose core business is not transporting goods may also find themselves confronted with having to efficiently solve transportation problems. For example, a made-to-order kitchen manufacturer must not only produce kitchens but also ensure that those kitchens are delivered to customers in a timely and efficient manner.

The first attempt to solve the VRP was made by Dantzig et al. (1954). In the years following, many academics have intensively studied the problem. Nevertheless, solving large-scale VRP instances sufficiently fast for real-world scenarios remains challenging. During the last decade, state-of-the-art solution methods have evolved towards genetic algorithms where multiple solutions evolve simultaneously. Other successful methods rely on exploring multiple neighborhoods within a local search algorithm.

This PhD introduces a simple yet powerful solution method for solving VRPs: Slack Induction by String Removals (SISRs). SISRs consists of three components: a fleet minimization technique, a ruin method and a recreate method guided by Simulated Annealing (Kirkpatrick et al. 1983). This approach is shown to be remarkable effective across a wide range of VRP variants, while its implementation remains straightforward and flexible enough to accommodate novel routing constraints. Simplified and flexible solution approaches such as SISRs are essential in order to solve real-world routing problems, as such problems may become very computationally complex when special constraints are involved. The Prisoner Transportation Problem, introduced in this PhD thesis, is one such example of a setting in which routing complexity increases considerably due to the restrictions involved when it comes to combining multiple prisoners within a single vehicle. Inspired by real-world VRPs like the Prisoner Transportation Problem (PTP), SISRs successfully contributes to both academic theory as well as vehicle routing optimisation in practice.

Beknopte samenvatting

Rittenplanning is een van de meest voorkomende logistieke problemen voor een grote verscheidenheid aan organisaties. Bezorgingsdiensten worden geconfronteerd met uitdagende transportpuzzels vanwege een groot aantal pakketten die dagelijks moeten worden bezorgd als gevolg van de snel groeiende onlineverkoop. Zelfs bedrijven waarvan transport niet de core business is, worden geconfronteerd met complexe rittenplanningen. Een keukenfabrikant moet bijvoorbeeld niet alleen keukens produceren maar er ook voor zorgen dat de keukens tijdig en efficiënt aan de klanten geleverd worden.

Rittenplanning, waarvan de eerste oplossingspogingen te danken zijn aan Dantzig et al. (1954) is reeds intensief bestudeerd door de academische wereld. Toch blijft het oplossen van grootschalige VRP-instanties in aanvaardbare tijd een grote uitdaging. In het afgelopen decennium zijn veelbelovende genetische algoritmen ontwikkeld waarbij meerdere oplossingen gelijktijdig evolueren. Andere succesvolle methoden zijn gebaseerd op het lokaal zoeken in meerdere omgevingen binnen de volledige oplossingsruimte.

Deze doctoraatsthesis introduceert een eenvoudige maar krachtige oplossingsmethode voor rittenplanningen: Slack Induction by String Removals (SISRs). SISRs bestaat uit drie componenten: een techniek om het aantal voertuigen te minimaliseren, een methode om een oplossing af te breken en een methode om de oplossing terug op te bouwen. De aanpak is opmerkelijk doeltreffend voor een brede waaier aan rittenplanningsvarianten, terwijl de implementatie eenvoudig en flexibel is zodat nieuwe voorwaarden voor uitvoerbare planningen gemakkelijk toe te voegen zijn. Vereenvoudigde en flexibele oplossingsmethoden zoals SISRs zijn essentieel om reële rittenplanningen op te lossen wegens de snel oplopende rekenkundige complexiteit van het probleem. Het gevangenentransport, geïntroduceerd in deze doctoraatsthesis, is een voorbeeldprobleem waarvan de complexiteit aanzienlijk toeneemt vanwege de voorwaarden bij het combineren van meerdere gevangenen in hetzelfde voertuig. Geïnspireerd door reële transportpuzzels zoals het gevangenentransport, draagt SISRs met succes bij op zowel het academische vlak als de optimalisatie van rittenplanning in de praktijk.

List of Abbreviations

ALNS	Adaptive Large Neighborhood Search
ASB-RR	Ruin and Recreate by Adjacent String removal and Blinks
ASRS	Automated Storage and Retrieval Systems
BVCP	Bounded Vertex Coloring Problem
CCVRP	Cumulative CVRP
CM	Constructive Method
CODeS	Combinatorial Optimization and Decision Support
CSSPP	Container Ship Stowage Planning Problem
CVRP	Capacitated Vehicle Routing Problem
DARP	Dial-a-Ride Problem
FAB	Fabrication Plant
FOUP	Front Opening Unified Pod
FPTP	Mixed Integer Programming Formulation for the PTP
HBSS	Heuristic-Biased Stochastic Sampling
HGSADC	Hybrid Genetic Search with Advanced Diversity Control
ILS-SP	Iterated Local Search with Set Partitioning
LNS	Large Neighborhood Search
LS	Local Search-based

MCVRP	Multi-Compartment VRP
MDVRP	Multi-Depot VRP
MDVRPMDP	Multi-Depot VRP with Mixed Delivery & Pickup
NA	Neighbor Acceptance
NS	Neighbor Selection
ODM	Origin Destination Matrix
OSM	Open Street Map
OVRP	Open VRP
PAH	Prisoner Allocation Heuristic
PAP	Prisoner Allocation Problem
PDP	Pickup & Delivery Problem
PDPT	Pickup & Delivery Problem with Transfers
PDPTW	Pickup & Delivery Problem with Time Windows
PTP	Prisoner Transportation Problem
R&R	Ruin & Recreate
$\mathcal{R}^-$	Ruin method
$\mathcal{R}^+$	Recreate method
RPDPTW	Rich Pickup and Delivery Problem with Time Windows
SA	Simulated Annealing
SC	Stop Criterion
SDVRPTW	Split Delivery VRP with Time Windows
SISRs	Slack Induction by String Removals
string	a sequence of consecutive locations in a tour
TSP	Traveling Salesman Problem
UHGS	Unified Hybrid Genetic Search
VRP	Vehicle Routing Problem
VRPB	VRP with Backhauls
VRPMDP	VRP with Mixed Delivery & Pickup
VRPSDP	VRP with Simultaneous Delivery & Pickup
VRPTW	VRP with Time Windows

Contents

Abstract iii

Beknopte samenvatting v

List of Abbreviations viii

Contents ix

List of Figures xi

List of Tables xiii

1 Introduction 1

2 The Vehicle Routing Problem (VRP) 7

2.1 The Capacitated Vehicle Routing Problem (CVRP) 8

2.2 CVRP Variants 8

3 Heuristics 11

3.1 Local Search-based Metaheuristic 12

3.2 Simulated Annealing 13

3.3 Classical VRP Neighborhoods 15

3.3.1 TSP neighborhoods 16
3.3.2 CVRP neighborhoods 18
3.3.3 VRPTW neighborhoods 20
3.4 Ruin & Recreate Strategies 21

4 The Prisoner Transportation Problem (PTP) 25

4.1 Introduction 26
4.2 Related problems 27
4.3 Notation and problem formulation 29
4.3.1 A mixed integer programming formulation (FPTP) . . . 31
4.3.2 Symmetry breaking constraints and additional valid inequalities 34
4.4 Modeling and testing the variants and special cases of the PTP 35
4.4.1 Testing the formulations 36
4.5 PTP Algorithm 41
4.5.1 Constructive heuristic 41
4.5.2 Local search improvement 41
4.6 Prisoner allocation heuristic 42
4.6.1 Greedy Prisoner Allocation Heuristic 44
4.7 Accommodating time constraints 45
4.7.1 Time windows 46
4.7.2 Maximum trip duration 47
4.7.3 Simultaneous servicing (merged service times) 50
4.7.4 Maximum ride times 51
4.8 PTP instances and computational results 52
4.9 Comparison of the formulations and iterative algorithms for PTP 57
4.9.1 Formulation with single-index time variables (FPTP-2) 57
4.9.2 Formulation without δ variables (FPTP-3) 58

4.10 Conclusion 60

5 Slack Induction by String Removals (SISRs) 61

5.1 Spatial Slack 62

5.2 Rationale 62

5.3 Adjacent String Removal ($\mathcal{R}^-$) 64

5.4 Greedy Insertion with Blinks ($\mathcal{R}^+$) 68

5.5 Simulated Annealing for SISRs 73

5.6 Fleet Minimisation 74

5.7 Parameters 75

5.7.1 Parameters for Distance Minimisation 77

5.7.2 Parameters for Fleet Minimisation 79

5.8 Computational Results 80

5.8.1 CVRP 81

5.8.2 CVRP Variants 85

5.9 Conclusions 89

6 Conclusions and Afterthoughts 91

A Detailed Results for CVRP 95

B Detailed Results for Classical CVRP benchmarks 99

C Detailed Results for CVRP Variants 105

Bibliography 123

List of Publications 132

List of Figures

3.1 Local Search Metaheuristic 12

3.2 Search trajectory through the solution space 13

3.3 Simulated Annealing 13

3.4 Envelope of the acceptance probability h 14

3.5 String Definitions 15

3.6 Three 2-opt moves modifying a solution from (a) to (d) 16

3.7 A 2-optimal solution (a) without any improving neighbors (b, c and d) 17

3.8 Three 3-opt neighbors (b, c and d) for the 2-optimal solution (a) 17

3.9 Inter-route 2-opt move 19

3.10 Exchange (a-b), cross (c-d) and relocate (e-f) 20

4.1 Underlying network representation for the PTP formulations . 30

4.2 The notation for vehicle compartments. 31

4.3 A trip transporting four prisoners (a) and their presence in the truck (b). 43

4.4 Prisoner allocation and conflict graphs. 43

4.5 Solution for allocating prisoners. 44

4.6 Time window and service time notations. 46

4.7 Feasibility time windows, schedules without (a) and with wait times (b). 47

4.8 Fixing the schedules, without (a) and with (b) wait times. . . . 47

4.9 Visual representation of Eq. 4.52 (a) and Eq. 4.54 (b). 49

4.10 Four examples of merging services a and b into m. 51

5.1 Examples of Ruin Methods which Introduce Insufficient Slack . 63

5.2 An Example of Adjacent String Removals 64

5.3 Example of 'String' Removal 66

5.4 Example of 'Split String' Removal 67

5.5 Examples of 'String' and 'Split String' Removals 67

5.6 An Example of SISRs' Ruin Method 68

5.7 Gaps of the Average Results 83

List of Tables

3.1 Cooling schedules 14
3.2 Overview of R&R Methodologies 23

4.1 Comparison of PTP and NSS methods 38
4.2 Results for NC and DARP methods. 40
4.3 Common properties of the PTP instances. 53
4.4 PTP vehicle types. 53
4.5 PTP instance and results for vehicle types v1 to v5. 55
4.6 PTP instance and results for vehicle types v6 to v10. 56
4.7 Comparison of PTP methods. 59

5.1 HBSS rank selection probabilities 71
5.2 Comparison of rank selection probabilities for HBSS and Blinks 72
5.3 Parameter Settings for SISRs $\mathcal{R}^-$ and $\mathcal{R}^+$ methods 76
5.4 Parameter Settings for SISRs' Fleet Minimisation and SA . . . 77
5.5 Maximum String Cardinality and Average Number of Removed Locations 78
5.6 Split String Parameters α and β 78
5.7 Blink Rate Parameter γ 79
5.8 Fleet Minimisation with Different L^{max} Settings 79

5.9 Maximum String Cardinality and Average Number of Removed Locations 80
5.10 Split String Parameters α and β 80
5.11 Blink Rate Parameter γ 80
5.12 Benchmark Subsets 81
5.13 Summarized Computational Results 82
5.14 p-values of the Wilcoxon Rank-Sum Test 84
5.15 CVRP Benchmark Sets 84
5.16 Results for the Classical Benchmark Sets 85
5.17 Adjustments Accommodating VRP Variants 87
5.18 Adjustments Accommodating VRP Variants (continued) 88

A1 Results for the CVRP, benchmark set of Uchoa et al. (2017) - Small 96
A2 Results for the CVRP, benchmark set of Uchoa et al. (2017) - Medium 97
A3 Results for the CVRP, benchmark set of Uchoa et al. (2017) - Large 98

B1 Results for the CVRP, benchmark set of Christofides and Eilon (1969) 99
B2 Results for the CVRP, benchmark set of Christofides et al. (1979) 100
B3 Results for the CVRP, benchmark set of Christofides et al. (1979) 100
B4 Results for the CVRP, benchmark set of Fisher (1994) 100
B5 Results for the CVRP, benchmark set of Augerat et al. (1995) . . 101
B6 Results for the CVRP, benchmark set of Augerat et al. (1995) . 102
B7 Results for the CVRP, benchmark set of Augerat et al. (1995) . 102
B8 Results for the CVRP, benchmark set of Rochat and Taillard (1995) 103
B9 Results for the CVRP, benchmark set of Golden et al. (1998) . 103

B10 Results for the CVRP, benchmark set of Li et al. (2005) 104

C1 Results for the MDVRP, benchmark set of Cordeau et al. (1997) 107

C2 Results for the VRPB, benchmark set of Goetschalckx and Jacobs-Blecha (1989) - part 1 108

C3 Results for the VRPB, benchmark set of Goetschalckx and Jacobs-Blecha (1989) - part 2 109

C4 Results for the VRPMDP, benchmark set of Salhi and Nagy (1999) 110

C5 Results for the VRPSDP, benchmark set of Salhi and Nagy (1999) 111

C6 Results for the MDVRPMDP, benchmark set of Salhi and Nagy (1999) 112

C7 Results for the OVRP, benchmark set of Christofides et al. (1979), Fisher (1994), Golden et al. (1998) 113

C8 Results for the CCVRP, benchmark set of Christofides et al. (1979), Golden et al. (1998) 114

C9 Results for the VRPTW, benchmark set of Gehring and Homberger (2001) 115

C10 Results for the PDPTW, benchmark set of Li and Lim (2001) . 116

C11 Best results for the PDPTW, benchmark set of Li and Lim (2001) 117

C12 Results for the DARP, benchmark set of Cordeau and Laporte (2003) 118

Chapter 1

Introduction

The Vehicle Routing Problem (VRP) in practice

When scheduling the routes of vehicles was first approached as an optimisation problem, the size or complexity of the problem was no more than the construction of a single tour for one vehicle. The problem originated in the sales sector, where salesmen traveled from city to city in order to sell their products. The optimisation problem of minimising the cost associated with this traveling was introduced in the early 1950s and formulated as the Traveling Salesman Problem (TSP). Usually these 'simple' optimisation problems contain no more than 20 locations, as visiting more locations on a single day was practically impossible. Human schedulers might not be able to design the optimal tour, but it is usually pretty easy for them to come up with a very good one by hand. However, as the number of stop locations increased, possibly spread over multiple days and vehicles (VRP), the complexity of planning such trips explodes and becomes unmanageable for human schedulers. The potential for an organisation to reduce the number of vehicles and working hours of drivers can result in significant operational savings when dedicated algorithms are deployed for optimizing schedules.

In modern transportation organisations and postal services such as DHL and UPS, a far greater number of locations must be visited in order to serve all customers. Up to 20,000 locations or more in a single day while employing a fleet of hundreds of vehicles has become a reality. While it may be clear that scheduling such logistical operations efficiently is beyond human capability, smaller problems continue to be handled by human schedulers. Based on vast amounts of experience and carefully developed rules of thumb, these schedulers manage to plan the organisation's tasks. Yet despite their incredible skills, they are no match for dedicated optimisation algorithms, whose use can result in significant transportation cost reductions. On the other hand, the flexibility required to handle unusual or unexpected events are difficult to hand over to automated systems, never mind incorporating new conditions (constraints) solutions must satisfy.

The first task in tackling optimization problems is to define the problem. What exactly should be optimized? What resources are used when fulfilling requests and tasks? Which constraints are present, or what defines a solution to be (in)feasible? Answering these questions can itself be a very difficult step. Usually the human planners are trained in rules of thumb, but it can be challenging to translate these into well-defined constraints: "We don't exceed available capacity, except sometimes when we have to." Human planners may impose constraints which may be occasionally ignored in practice or which are developed during their career for reasons irrelevant to the actual problem. When a proper problem definition is finally obtained, computer models can be implemented and their performance tested. Modern VRPs may include time windows, driver

schedules, driving regulations, driver skills, vehicles properties and so on, which may result in a very complex task to design an efficient implementation for.

Finally, it may be very difficult to accommodate special objectives corresponding to business models which are neither necessary nor directly linked to obvious transportation costs, such as fuel or drivers. In fact, desired objectives may even conflict with one another. For example, it is not unusual for transportation companies to prefer schedules of higher transportation cost but which do not require any outsourcing, than cost-effective schedules which require some outsourcing. Therefore, dedicated solution approaches may be necessary for minimizing simply the total travel distance versus minimizing some resource usage (e.g. the number of deployed vehicles).

The evolution towards overcomplicated schemes

Academically speaking, this thesis is motivated by a desire to improve state-of-the-art algorithms. The ultimate goal is to design a simple algorithm capable of obtaining high quality solutions within reasonable amounts of time. One such popular example would be the two-opt heuristic for solving the TSP: a very simple algorithm which is capable of improving inferior solutions very quickly. The solution should not be more complex than the problem.

Often, solution approaches which improve upon state-of-the-art methods result in methods of incremental complexity and decremental improvement. The major drawback of this is that it risks state-of-the-art methods becoming less flexible and unlikely to be deployed in practice. Although all improvements are important from academic perspective, as they provide insight into the nature of problems, in practice there is also a pressing need for simplicity, flexibility and efficiency.

In this work we try to answer to the need for simple, flexible, high performant heuristics for use in practice. The trajectory of this research began by developing a framework for solving a wide range of VRPs in practice. While the added value in solving real-world problems was clearly there, comparing the performance to academic state-of-the-art approaches shifted this research towards developing a new state-of-the-art approach which was easy to implement in practice. We show that improving the state of the art does not necessarily require a heightening of algorithmic complexity. Even better, the heuristic introduced in this thesis outperforms previous state-of-the-art methods while enabling the solving of a wide range of VRPs with different objectives and problem structures, such as the Capacitated Vehicle Routing Problem (CVRP) and the Pickup & Delivery Problem (PDP).

Having a background in practical and real-world problems is extremely useful for developing a solution approach well-suited for deployment in practice. This was, perhaps, the key ingredient behind the successful results presented in this thesis. The background knowledge and experience acquired in optimization heuristics is demonstrated by way of listing the most important research assignments undertaken throughout the past several years.

Trajectory of research activities

During my project for the master Artificial Intelligence course, a Distance Oracle was developed. Based on Open Street Map (OSM) data of a certain area, the oracle was able to estimate the shortest path distance between two arbitrary locations. In order to do so, the OSM nodes were clustered based on the likelihood that origin nodes share a large proportion of their shortest paths to many destination nodes and vice versa. Shortest paths were sampled by Dijkstra's algorithm (Dijkstra et al. 1959) during the clustering process and while building an Origin Destination Matrix (ODM) between all clusters' entrance and exit nodes. In addition, a Graphical User Interface for visualising the OSM network was implemented.

A competitive algorithm was designed in my master thesis for the Container Ship Stowage Planning Problem (CSSPP) which is a nested problem for inland container transportation by barges (EUROPORTS). Containers are stacked such that container relocations during voyages should be minimized. Substantial cost and time reductions are made possible by optimizing loading and unloading the barges.

The first project realised at the Combinatorial Optimization and Decision Support (CODeS) research group was commissioned by the inland terminal operator BCTN and concerned the optimization of container transportation performed by inland barges, a problem variant of the Pickup & Delivery Problem with Time Windows (PDPTW). The algorithm designed minimizes total travel distance for a set of barges. This research project established the datastructures for PDPTW and a basic VRP heuristic. Additionally, the visualisation code developed in the master year has since been compiled into a general and user friendly visualisation library: LorgFX (Library of Reusable GUIs for JavaFX).

Next, a new PDPTW was addressed: the Prisoner Transportation Problem (PTP), commissioned first by Möbius and later Conundra. Due to the reusibility of the PDPTW data structure and visualisation library implemented in previous research, together with very clean and error free data provided, quick progress was made in this project. In addition, similar to the container transportation research, a PDPTW solution is only valid if a nested optimisation problem is solved: the assignment of prisoners to compartments in secured trucks. In

this research significant cost reductions for the transportation company were successfully demonstrated.

At the request of Conundra, a tool for building an ODM compatible with their current matrix files was implemented. The tool runs Dijkstra's algorithm n times for an $n \times n$ matrix. Rather than building a routable graph from OSM data, a graph was first generated by the Open Source Routing Machine tool. Employing Dijkstra's single source shortest path algorithm is far less efficient than all-pairs shortest paths algorithms for generating matrices, requiring more than 16 hours of calculation time for $30,000$ locations. Yet the long computation times for such matrices were acceptable given that these large ODMs were only computed once and used for building smaller ODMs for a subset of their locations.

Following this, a solution approach for optimizing Automated Storage and Retrieval Systems (ASRS) was developed for a large machine manufacturer. The problem was modeled as a PDP with a single vehicle for which the delivery location could be selected from a given set of locations. A superior data structure for a variety of VRP models was developed. The algorithm was successfully implemented in the organisation's software infrastructure.

A Baekeland-mandate was initiated in cooperation with Conundra after the previous projects. To begin with, the company was solving regular VRPs by re-employing the code for solving the PTP. Since a PDP implementation is fairly inefficient for solving VRPs, the code was stripped back and improved by transferring the superior data structure developed during the ASRS project. These improvements proved very successful and were immediately deployed in practice for solving Conundra's customer vehicle routing. The first year of the Baekeland-mandate consisted mainly of extending the code into a framework for accommodating each of their customer's individual requirements, such as drivers' lunch break policies or multi-tour capability.

Academic research was also initiated by participating in the VeRoLog challenge (2014) with a team of colleagues: Tony Wauters, Túlio Toffolo and Sam Van Malderen. After obtaining the best results among all participating teams and winning this competition, the fundamental concepts developed were transferred into Conundra's framework and proved successful in practice as well.

These concepts were afterwards applied to academic problems, starting with the most basic VRP: the CVRP. While being able to improve the vast majority of state-of-the-art CVRP results with the algorithm introduced as Ruin and Recreate by Adjacent String removal and Blinks (ASB-RR), the academic world required the new solution approach to be evaluated on other VRP variants as well. Such variants includes: Multi-Depot VRP (MDVRP), Open VRP (OVRP), Cumulative CVRP (CCVRP) and more. After the period of the

Baekeland mandate, ASB-RR was extended and improved for solving the VRP with Time Windows (VRPTW), PDPTW and Dial-a-Ride Problem (DARP), demonstrating the wide applicability of the rebranded algorithm: SISRs.

The final research project was imec's Fabrication Plant (FAB) scheduling, optimized during a two year project. In brief, imec develops innovative processes for semiconductor manufacturing while scaling down the size in order to fit more semiconductors on a single chip. The raw material used is silicone in the form of disks with a diameter of 200 or 300mm, called wafers. Process operations on such wafers are scheduled by employing two resources: operators and tools. An operation is initiated by an operator which brings a Front Opening Unified Pod (FOUP) (a closed box holding up to 25 silicon wafers) to the specific tool and starts the process operation. When the tool finishes the operation, an operator removes the FOUP from the tool. Although the problem's data structures were implemented from scratch, the core data structure is very similar to the one used for the VRPTW. The fundamental insight in efficient data structures for optimization problems enabled fast development of an efficient FAB scheduling approach.

Research results reported in this PhD thesis

Although each of these research projects was very interesting, only a careful selection closely related to the VRP are documented in this PhD thesis. The core contribution of this thesis is a state-of-the-art heuristic for solving a wide range of VRPs: Slack Induction by String Removals (SISRs). The development of this heuristic was possible thanks to the extensive experience in solving real world optimization problems, resulting in SISRs. The thesis begins by first defining and situating the CVRP in relation to the academic literature (Chapter 2). An in-depth review of the classical solution approaches is given in Chapter 3, which provides the reader with background knowledge related to dedicated heuristics for solving VRPs as well as heuristics more generally. How such heuristics may be employed for solving a real-world VRP is demonstrated in Chapter 4, where the solution approach for the PTP is explained in depth. The thesis ends with a full description of SISRs provided in Chapter 5 with a detailed computational analysis which demonstrates its high performance in general.

Chapter 2

The Vehicle Routing Problem (VRP)

2.1 The Capacitated Vehicle Routing Problem (CVRP)

The CVRP, primarily considered by the current thesis, is defined as follows. Let $G = \{N, A\}$ be a complete undirected graph in which N is the set of nodes and A is the set of arcs connecting the nodes. The nodes $n_i \in N$ for $i \in \{0, ..., n\}$ represent locations in a 2-dimensional space where n_0 corresponds to the *depot* and the other n nodes to *customers* having a demand q_i. Each arc (i, j) in $A = \{(i, j) : i, j \in N, i \neq j\}$ is associated with a *cost* c_{ij}. An unlimited homogeneous fleet of vehicles with capacity Q is situated at the depot. The CVRP consists of designing vehicle *tours* at minimum cost such that each customer is served exactly once by a single vehicle while each tour starts at the depot v_0, serves customers without exceeding the vehicle's capacity Q and finally ends at the same depot v_0.

2.2 CVRP Variants

While the CVRP constitutes the primary problem addressed by the present thesis, the general applicability of SISRs is demonstrated by way of solving several other associated problems within the domain of vehicle routing. Key references are supplied for papers which provide a description for each individual problem.

Multi-Depot VRP (MDVRP)
Vehicles are available at a fixed number of depots. Vehicles are assumed to return to the depot where their tour starts. This problem was introduced by Gillett and Johnson (1976). The algorithms by Pisinger and Røpke (2007), Subramanian et al. (2013) and Vidal et al. (2012) are considered as the current best approaches.

VRP with Backhauls (VRPB)
The VRP with Backhauls, introduced by Goetschalckx and Jacobs-Blecha (1989), considers serving two sets of customers: delivery (linehaul) and pickup (backhaul) customers. Each tour must serve at least one delivery customer and all delivery customers must be served before any pickup customers are served. A tour's total delivery and pickup demand should not exceed its vehicle's capacity. The algorithms by Gajpal and Abad (2009), Zachariadis and Kiranoudis (2012) and Vidal et al. (2014) represent the best solution approaches for this problem.

VRP with Mixed Delivery & Pickup (VRPMDP)
This problem, introduced by Min (1989), relaxes the VRP with Backhauls

(VRPB) by enabling any order of delivery and pickup customers within a tour. This implies that a vehicle's used capacity may increase and decrease during the tour, but should never exceed the vehicle's capacity. Best solutions are obtained by Subramanian et al. (2013) and Vidal et al. (2014).

VRP with Simultaneous Delivery & Pickup (VRPSDP)
Unlike the VRP with Mixed Delivery & Pickup (VRPMDP) this problem, also introduced by Min (1989), does not require customers to be associated exclusively with either delivery or pickup demand. This implies that a single customer can have both delivery and pickup requests. Best solutions are obtained by the same authors as for the VRPMDP.

Multi-Depot VRP with Mixed Delivery & Pickup (MDVRPMDP)
Min (1989) generalizes the VRP with Simultaneous Delivery & Pickup (VRPSDP) by addition of multiple depots into the Multi-Depot VRP with Mixed Delivery & Pickup. Best approaches are those of Røpke and Pisinger (2006b) and Subramanian et al. (2013).

Open VRP (OVRP)
In contrast to the CVRP, vehicles must not return to the depot after serving the last customer in the Open VRP, introduced by Sariklis and Powell (2000). This reflects situations in which customers are assigned to external carriers. Current best approaches are those proposed by Subramanian et al. (2013) and Vidal et al. (2014).

Cumulative CVRP (CCVRP)
The Cumulative CVRP (Ngueveu et al. 2010) requires, similar to 'minavg routing' (Campbell et al. 2008), the sum of arrival times at customers to be minimized rather than the total traveled distance. For example, the CCVRP occurs when the quality of the solution depends on the waiting time of each customer. The approach of Vidal et al. (2014) obtained the current best known solutions for this problem.

VRP with Time Windows (VRPTW)
The VRP with Time Windows extends the CVRP with temporal constraints. Customers are associated with individual time windows within which they prefer to be served. Vehicles start their tour in the depot and are assumed to arrive back at the depot without exceeding the maximum tour duration as defined by Gehring and Homberger (1999). The current best approaches are those proposed by Repoussis et al. (2009), Nagata et al. (2010) and Vidal et al. (2013).

Pickup & Delivery Problem with Time Windows (PDPTW)
The PDPTW requires transportation of goods between pickup and delivery locations (Dumas et al. 1991). Similar to the VRPTW, temporal constraints are present. The PDPTW should not be confused with Delivery & Pickup problems

where goods are transported to or from the depot, rather than between the customers themselves. Røpke and Pisinger (2006a), Nagata and Kobayashi (2010) and Curtois et al. (2018) have contributed the current best solution approaches.

Dial-a-Ride Problem (DARP)

In the Dial-a-Ride Problem, which was first examined by Wilson et al. (1971), transportation of customers between specified pickup and delivery locations are considered. The most recent definition of the DARP (Cordeau and Laporte 2003) is in line with the work of Jaw et al. (1986). The DARP restricts the PDPTW by imposing a maximum ride time in order to manage user inconvenience. Best approaches are those of Chassaing et al. (2016), Masmoudi et al. (2017) and Gschwind and Drexl (2019).

Chapter 3

Heuristics

3.1 Local Search-based Metaheuristic

While exact mathematical optimization methods attempt to find the optimal solution for a given problem, heuristics search for reasonably good solutions within a small amount of computation time. Local Search-based (LS) heuristics begin with an initial solution s_0 for the problem provided by a Constructive Method (CM). LS seeks better solutions by iteratively replacing the current solution s with an improving or equal neighbor solution $s \leftarrow s'$, a substitution referred to as a *move*. The neighborhood function $\mathcal{N}$ is a mapping of solution s to the set of its neighbors $s \mapsto \mathcal{N}(s)$ or simply its *neighborhood*. LS terminates in a local optimum $\bar{s}$ when $\mathcal{N}(\bar{s})$ contains no improving neighbor. By contrast, *metaheuristics* are capable of 'escaping' from such local optima by their ability of moving to lower quality neighbors. The subset $\mathcal{C}(s) \subseteq \mathcal{N}(s)$ represents a set of candidate solutions, from which s' is selected (Neighbor Selection (NS)) by a local search method and possibly accepted by the Neighbor Acceptance (NA) criterion. The metaheuristic terminates when a certain Stop Criterion (SC) is satisfied, for example computation time limit or maximum number of iterations. Fig. 3.1 introduces a schematic representation of a metaheuristic.

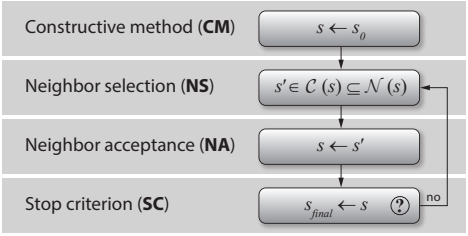

Figure 3.1 Local Search Metaheuristic

A sequence of moves denotes a walk, or a search trajectory, through the solution space by iteratively moving from the current solution to one of its neighbors (Crainic and Toulouse 2003). The search trajectory in Fig. 3.2, for example, begins at the initial solution ($s \leftarrow s_0$). The neighborhood $\mathcal{N}(s_0)$ contains nine neighbors of which four are present in the set of candidates $\mathcal{C}(s_0)$. Solution s_1 is accepted from $\mathcal{C}(s_0)$ by the first move m_1 ($s \leftarrow s_1$) and neighborhood $\mathcal{N}(s_2)$ is reached after performing the second move m_2 ($s \leftarrow s_2$).

Notice that neighborhood $\mathcal{N}(s_2)$ overlaps with initial neighborhood $\mathcal{N}(s_0)$, potentially resulting in metaheuristic 'cycling'. The *Tabu Search* metaheuristic (Glover 1986), for example, maintains characteristics of previously visited solutions ~~in a set T~~ to avoid such cycling behavior. *Simulated Annealing* is an often-employed meta-heuristic whose NA criterion accepts non-improving

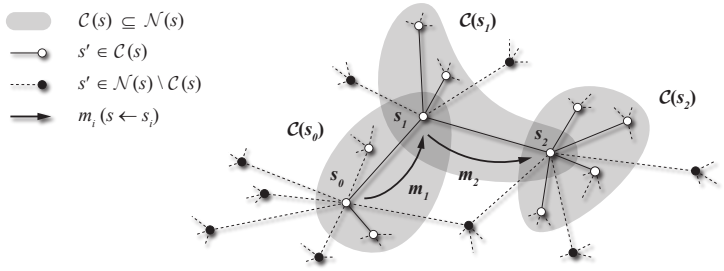

Figure 3.2 **Search trajectory through the solution space**

neighbors based upon the annealing process of metals. A more detailed explanation of Simulated Annealing is located within the following section. We refer the reader to Talbi (2009) for an in-depth overview of many other metaheuristics.

3.2 Simulated Annealing

The field of metallurgy defines annealing as the process by which the physical properties of materials are modified through controlled heating and cooling. A statistical model concerning the energy changes in such annealing systems was developed by Metropolis et al. (1953). Based on this work, Kirkpatrick et al. (1983) introduced Simulated Annealing (SA) as a metaheuristic capable of solving combinatorial optimisation problems. The transition to a new state $s \leftarrow s'$ is dependent upon the energy change of the system $\Delta E = E(s') - E(s)$ and the current temperature $\mathcal{T}$. While improving or equal quality neighbors ($\Delta E \leq 0$) are always accepted, the transition to a non-improving neighbor occurs with an acceptance probability function $h(\Delta E, \mathcal{T}) = exp(-\Delta E/\mathcal{T})$. Non-improving solutions are therefore more likely to be accepted at high values of $\mathcal{T}$ or low values of ΔE Fig. 3.3(a). This probability distribution is employed by SA as the probabilistic NA criterion.

SA begins at an initial temperature, $\mathcal{T}_0$, high enough such that all non-improving neighbors are likely accepted. Assuming the system is cooled to the theoretical final temperature $\mathcal{T}_1 = 0$ over an infinite length of time, SA will converge to the problem's optimum solution. In practice, the system is cooled to a final temperature $\mathcal{T}_1 > 0$ (the stop criterion). The cooling rate is specified by a *cooling schedule* $\mathcal{T}(i)$ as a function of the normalized annealing time i ($0 \leq i \leq 1$). Fig. 3.3(b) illustrates the envelopes of the *Linear*, *Exponential*, *Hyperbolic* and *Logarithmic* cooling schedules when the system is cooled from

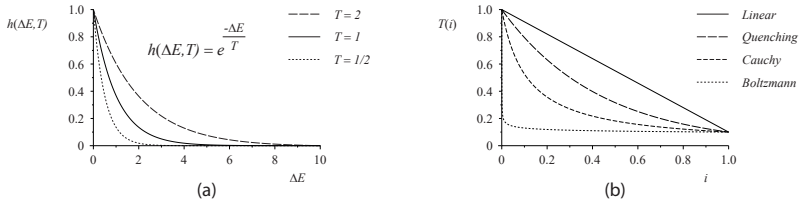

Figure 3.3 Simulated Annealing
Note. Acceptance distribution (a) and cooling schedule (b)

$\mathcal{T}_0 = 1$ to $\mathcal{T}_1 = 0.1$. The corresponding equations are provided by Table 3.1. The influence on the acceptance distribution is visualized in Fig. 3.4 for the Linear (a), Quenching (b) and Cauchy (c) cooling schedules.

Table 3.1 Cooling schedules

	Temperature	Cooling constant
Linear	$\mathcal{T}(i) = \mathcal{T}_0 + c.i$	$c = \mathcal{T}_0 - \mathcal{T}_1$
Exponential	$\mathcal{T}(i) = \mathcal{T}_0.c^i$	$c = \mathcal{T}_1/\mathcal{T}_0$
Hyperbolic	$\mathcal{T}(i) = \mathcal{T}_0/(1 + c.i)$	$c = (\mathcal{T}_0 - \mathcal{T}_1)/\mathcal{T}_0$
Logarithmic	$\mathcal{T}(i) = \mathcal{T}_0/(1 + ln(c.i))$	$c = exp((\mathcal{T}_0 - \mathcal{T}_1)/\mathcal{T}_1)$

When the linear schedule is applied (Fig. 3.4(a)), non-improving solutions are potentially accepted quite far into the search, thus implying a rather diversified search. The hyperbolic schedule (c) only accepts non-improving solutions at the very beginning of the search. An exponential schedule represents a good compromise for many researchers, wherein a diversified search is obtained during the first half of annealing-time, while the search becomes strongly intensified during the second half (b).

Most metaheuristics require *parameter tuning* when applied to a given problem as there are no parameter settings ensuring good performance on all problem types. For example, larger problems generally require more iterations to obtain gradual convergence. SA requires $\mathcal{T}_0$ and $\mathcal{T}_1$ to be 'tuned' which further impacts upon the cooling schedule shape, and therefore finding good values for these parameters proves a difficult task.

3.3 Classical VRP Neighborhoods

Before explaining the incentive behind the algorithm introduced by this thesis, the present section first analyzes the local search improvement operators

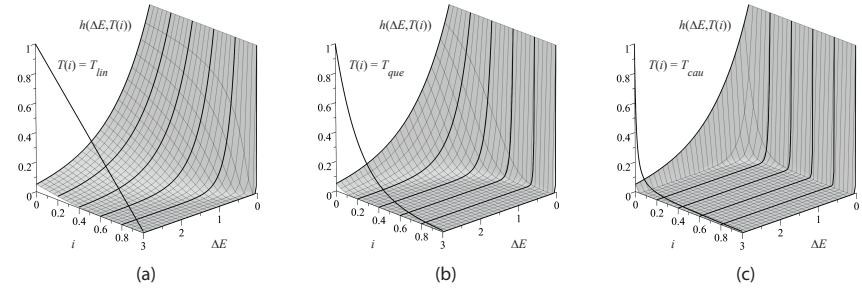

Figure 3.4 Envelope of the acceptance probability h

Note. Envelope of the acceptance probability h based on three cooling schedules: Linear (a), Exponential (b) and Hyperbolic (c).

employed by TSP, CVRP and VRPTW heuristics. Such an analysis will indicate the common mechanisms of the corresponding neighborhoods. While readers already familiar with such details are free to skip over this section, it does in fact contribute a comprehensive comparative overview of local search developments in a vehicle routing context.

The following subsections refer to a sequence of consecutive nodes in a tour as a string. Strings may contain zero customers to, at most, all customers served by the tour. The number of customers included in a string is referred to as the *cardinality* of the string and denoted by $|string|$. A string within tour T is referred to as T-*string*. The string's direction is defined relative to its origin tour A which may be preserved (A-$string^{+}$), reversed (A-$string^{-}$) or arbitrary (A-*string*) (see Fig. 3.5(a)). The term string may be substituted by the term *head* (B-*head*) or *tail* (B-*tail*) when the string includes, respectively, the first or last customer served by the tour B (Fig. 3.5(b)).

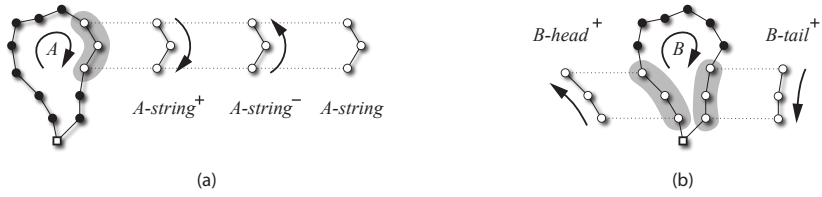

Figure 3.5 String Definitions

3.3.1 TSP neighborhoods

Optimising the TSP essentially implies reaching an optimum sequence for connecting an entire set of spatially-distributed nodes. Such a sequence cannot be optimum whenever an edge intersects another, as confirmed by Flood:

> *"There is one useful general theorem, which is quickly discovered by each one who considers the traveling-salesman problem. In the euclidean plane it states simply that the minimal tour does not intersect itself." (Flood 1956)*

This theorem represents the foundation for the most powerful TSP heuristics, detailed in the following sections.

2-opt (Croes 1958)

One possible suggestion for remedying the optimality violation detailed by Flood is to locate such intersecting edges and reverse the direction of the intermediate nodes between these edges. The most basic implementation of such a suggestion corresponds to the well-known *2-opt* move (Croes 1958), which considers two non-consecutive edges. Fig. 3.6(a-b) illustrates a 2-opt move which removes two intersecting edges before reconnecting the nodes by two non-intersecting edges. Using the string terminology introduced earlier, we could interpret such a move as simply reversing the string's direction ($string^{+} \rightarrow string^{-}$).

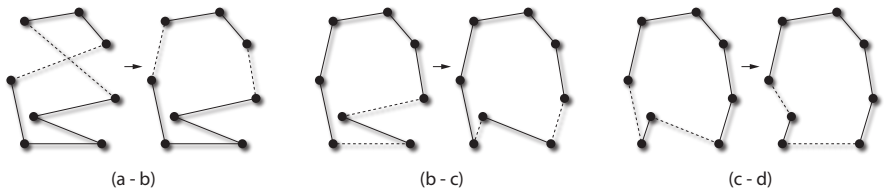

Figure 3.6 Three 2-opt moves modifying a solution from (a) to (d)

Although no pair of edges intersect in solution (b), solution (c) represents an improved neighbor in its 2-opt neighborhood. As with the $(b-c)$ move, (c) has no intersecting edges. The optimal solution (d) is obtained by a 2-opt move applied to solution (c). Nevertheless, 2-opt does not guarantee generating the global optimum, as demonstrated by Fig. 3.7. Solution (a) has five neighbors in its 2-opt neighborhood while only three distinct solutions exist: (b), (c) and (d). Solution (a) has a tour length of 26 whereas neighbors (b), (c) and (d) have

tour lengths of 26.06, 35.88 and 29.12 respectively. Therefore no improving solutions within the 2-opt neighborhood are available and solution (a) is said to be *2-optimal*. Metaheuristics may be employed to escape from this local optimum. Flood's theorem served as the foundation for the creation of the 2-opt move which is known to be reasonably effective with neighborhood sizes of $O(n^2)$, where n denotes the number of edges.

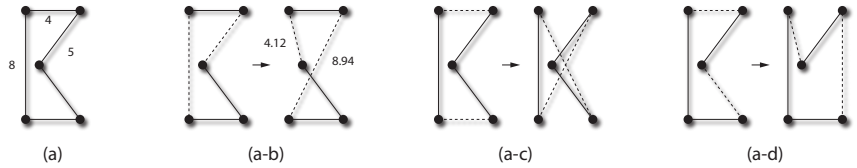

Figure 3.7 A 2-optimal solution (a) without any improving neighbors (b, c and d)

3-opt (Lin 1965)

The 2-opt move provides the basic principle for the *3-opt* move (Lin 1965), which considers three rather than two edges, thus enabling the exploration of a larger neighorhood. Fig. 3.8(a) reproduces the same solutions as Fig. 3.7(a), proven 2-optimal by enumeration, and is consequently employed to illustrate 3-opt neighborhoods. Three edges are removed and replaced with others from the 3-opt neighborhood. If one of the removed edges were replaced by its original edge, it would result in a 2-opt neighbor. Due to this property, removing three consecutive edges in the current solution (a) always results in a 2-opt neighbor, given one of the three edges will always be replaced by its original edge. There are only five possibilities to remove three non-consecutive edges, of which only three - a_b, a_c and a_d - are symmetrically distinct. There exist three distinct possibilities for replacing the selected edges in (a_b). Only one option, (b), replaces all three edges with new ones, thus resulting in a neighbor not part of the 2-opt neighborhood. This situation also occurs in solutions (c) and (d).

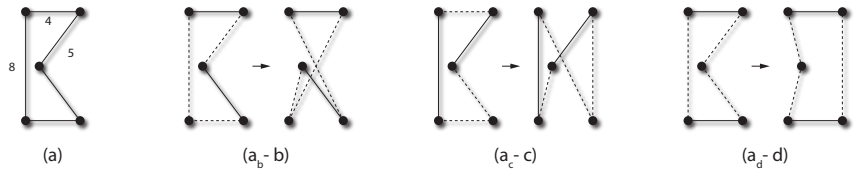

Figure 3.8 Three 3-opt neighbors (b, c and d) for the 2-optimal solution (a)

Evidently, it is possible to reach optimal solution d, which was not one of the 2-opt neighbors, using only improving moves. One should be aware, however,

that exploring the 3-opt neighborhood increases the neighborhood size from $O(n^2)$ to $O(n^3)$.

Lin-Kernighan (Lin and Kernighan 1973)

The algorithm developed by Lin and Kernighan (1973) is considered to be one of the best methods for solving TSPs. Lin-Kernighan applies *k-opt*, the most general *-opt*, where the value of k is dynamically varied. The algorithm starts with $k = 2$ in each iteration and increases this value when it determines that a $(k + 1)$-*opt* move may improve the solution until one of the stop criteria is satisfied. The best-known implementation of Lin-Kernighan was developed by Helsgaun (2000), who also provides a comprehensive description of the algorithm.

Or-opt (Or 1976)

Or-opt exchanges (Or 1976) consider three edges and thus belong to the class of 3-opt heuristics. However, Or-opt iteratively relocates a 3-*string* to another location until no further improvements are possible. The procedure is subsequently executed with a 2-*string* and, finally, a 1-*string* (single node). Multiple implementation variants such as whether to apply the first- or best-improving string location are compared by Babin et al. (2007). While some variants may be superior to the original algorithm, Or-opt exchanges are not among the best performing algorithms for solving TSPs.

3.3.2 CVRP neighborhoods

Given the CVRP represents a generalisation of the TSP, all TSP heuristics may be applied to a single tour in CVRP solutions as *intra-route* neighborhoods which are employed as either independent neighborhoods, parts of compound moves or post-processing procedures.

2-opt as an inter-route operator

In the 2-opt neighborhood, as previously detailed, two edges are selected and the direction of the enclosed string reversed ($string^+ \rightarrow string^-$). In a CVRP solution, two edges from different tours A and B are considered, enabling 2-opt to operate as an inter-route neighborhood. The edge selected in tour A splits the tour into A-$head^+$ and A-$tail^+$ strings, the edge selected in tour B splits

it into $B\text{-}head^{+}$ and $B\text{-}tail^{+}$ strings. There are now two possibilities insofar as reconnecting the resulting strings. In Fig. 3.9(a-b) $A\text{-}tail^{+}$ and $B\text{-}tail^{+}$ are swapped by connecting them to $B\text{-}head^{+}$ and $A\text{-}head^{+}$, respectively. In this case, no strings are reversed. The second option is shown in Fig. 3.9(a-c). B's head string is reversed ($B\text{-}head^{-}$) and connected to A's head-string ($A\text{-}head^{+}$). The same process occurs for the tail-strings, with a-tail being connected to $B\text{-}tail^{-}$.

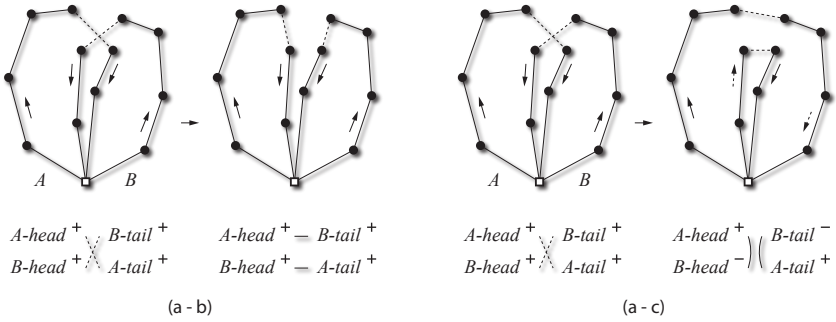

Figure 3.9 Inter-route 2-opt move
Note. String directions preserved (a-b) and reversed (a-c).

Or-opt as an inter-route operator

The Or-opt neighborhood may be easily employed as a CVRP inter-route neighborhood by inserting the removed string into a different tour with sufficient capacity. However, when all vehicle capacities are completely utilized or all remaining capacities are less than the smallest demand, Or-opt functions only as an inter-route neighborhood, provided customers are relocated to a new empty tour.

Single, double, pair+single and double pair procedures (Waters 1987)

Greater capacity slack may be introduced into the solution by exploring other neighborhoods. Waters (1987) introduced single, double, pair+single and double pair procedures. These procedures remove one or two strings from the solution before reinserting them into optimal positions. The *single* procedure removes only one node (1-*string*) from the solution. The *double* procedure removes two 1-*string*s. The *pair+single* procedure removes both a 2-*string* (pair) and 1-*string* (single). Finally, two 2-*string*s are removed by the *double pair* procedure. The overall improvement method explores the two-opt neighborhood

first, followed by the double pair, pair+single, double and finally the single neighborhood.

Chain-exchange (Fahrion and Wrede 1990)

While Waters (1987) restricted the operation to string cardinalities of one and two, Fahrion and Wrede (1990) permitted two larger string cardinalities (*M-string* and *P-string*) through their *chain-exchange* procedure. String cardinalities M and P are bounded to be no more than one half of the average tour size as their research indicates that larger strings do not generally result in further improvements. Finally, as with Waters (1987), strings are reinserted into optimal positions.

3.3.3 VRPTW neighborhoods

The neighborhoods described in the previous section enable the reversal of the removed strings, a process which often results in time window infeasibilities for VRPTW solutions. Therefore, string order is usually preserved by the neighborhoods applied to VRPTWs.

Exchange, cross and relocate (Savelsbergh 1988)

The *exchange*, *cross* and *relocate* neighborhoods are introduced in Savelsbergh's Ph.D. dissertation (Savelsbergh 1988) and related research paper (Savelsbergh 1992). These neighborhoods relocate strings between two tours A and B. *Exchange* simply swaps two strings: an A-*string*$^+$ is inserted at the B-*string*'s original location and vice-versa (Fig. 3.10(a,b)). *Cross* swaps the tails of two tours, equivalent to the inter-route extension of the two-opt neighborhood which preserves the string orders (Fig. 3.10(c,d)). Finally, *relocate* inserts an A-*string*$^+$ into another tour B (Fig. 3.10(e,f)). *Relocate* represents a generalization of the Or-opt neighborhood given that the A-*string*'s cardinality is not limited to three.

Neighborhoods are explored simultaneously, with the best neighbor in each iteration selected until no improvements are possible. A computational analysis of these neighborhoods is detailed by both Prosser and Shaw (1996) and Van Breedam (1994). These studies conclude that *relocate* improves solutions the most, and *exchange* the least.

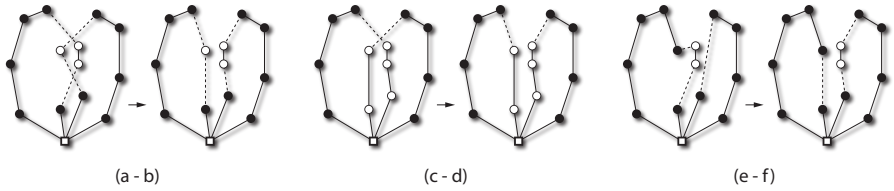

Figure 3.10 Exchange (a-b), cross (c-d) and relocate (e-f)

2-opt* (Potvin and Rousseau 1995)

Equivalent to Savelsbergh's cross move, *2-opt** (Potvin and Rousseau 1995) generalized the 2-opt neighborhood into an inter-route neighborhood for the VRPTW, with the aim of preserving node order. Their first implementation applies 2-opt* until a 2-opt* optimal solution is found. Next, Or-opt is applied to this solution until an Or-opt optimal solution is found. The procedure is repeated until the optimum solution is achieved for both neighborhoods. The second implementation explores neighborhoods simultaneously. Results indicate their approaches perform well on VRPTW problems with tight time windows.

CROSS-exchange (Taillard et al. 1997)

Taillard et al. (1997) combine Savelsbergh's inter-route *exchange*, *cross* and *relocate* neighborhoods in the *CROSS-exchange* neighborhood. They also enable *relocate* to operate within a single tour. NS is performed using a neighborhood reduction technique. An approximation matrix is employed which stores previous calculations, thus reducing computation time.

The following section will introduce Ruin & Recreate (R&R) which enables one to interpret each of these classical neighborhoods in a more general manner.

3.4 Ruin & Recreate Strategies

Classical neighborhoods for VRP heuristics - such as, 2-opt (Croes 1958) 2-opt* (Potvin and Rousseau 1995), k-opt (Lin 1965), Or-opt (Or 1976), exchange, cross and relocate (Savelsbergh 1988) and CROSS-exchange (Taillard et al. 1997) - are all defined via slight modifications to the current solution. They therefore exhibit limited neighborhood size complexity, implying a full neighborhood $\mathcal{N}(s)$ of a solution s may be evaluated within a reasonable amount of computational time.

Very large neighborhoods often occur when a significant part of the solution is modified, making a full neighborhood exploration impractical. Therefore, neighbors may be selected from a reduced candidate set $\mathcal{C}(s) \subseteq \mathcal{N}(s)$ or a single neighbor may be generated by first ruining the current solution before recreating it into a feasible one. Such a method may be defined by a Ruin method ($\mathcal{R}^-$) and Recreate method ($\mathcal{R}^+$). This underlying concept is present in a variety of research papers. To our knowledge, it was established for the first time by Dees and Smith (1981) in their *Rip-Up and Reroute* strategies for wiring point-to-point connections in electronic design automation. Shaw (1998) introduced Large Neighborhood Search (LNS) wherein the ruin phase is implemented as the removal of related customers (related in terms of time, distance and being served by the same vehicle). Their branch-and-bound technique includes Limited Discrepancy Search (LDS) (Harvey and Ginsberg 1995) which optimally inserts the removed customers in the recreate phase. Shaw (1998) applied LNS to both the CVRP and VRPTW. The term R&R was introduced by Schrimpf et al. (2000) who applied the technique to a number of prominent problems including the TSP and VRPTW. They ruined solutions by removing either randomly selected customers, customers within a certain radius or one single string. The solution is recreated by greedily reinserting the removed customers in a random order at minimum cost. Røpke and Pisinger (2006a) defined several $\mathcal{R}^-$ and several $\mathcal{R}^+$ methods which compete to modify the solution in their Adaptive Large Neighborhood Search (ALNS) framework and applied their framework to the PDPTW. They implemented seven different strategies for selecting customers for removal: random, worst, related, cluster, time oriented, historical node-pair and historical request-pair. The solution is recreated by employing either greedy or regret insertion (Potvin and Rousseau 1993) with or without noise function. Pisinger and Røpke (2007) applied ALNS to the Rich Pickup and Delivery Problem with Time Windows (RPDPTW).

Recent heuristic development based on the R&R principle builds upon the general ALNS framework by enlarging the set of $\mathcal{R}^-$ and $\mathcal{R}^+$ methods. Examples of such studies include, but are not limited to the papers by Laporte et al. (2010) and Ribeiro and Laporte (2012). In distinct contrast to the recent trend of introducing more and more $\mathcal{R}^-$ and $\mathcal{R}^+$ methods, the present thesis introduces a streamlined R&R implementation: SISRs (SISRs, phonetically "scissors"). This proposed approach employs a single $\mathcal{R}^-$ and a single $\mathcal{R}^+$ method: *adjacent string removal* and *greedy insertion with blinks*, respectively, with the significant additional benefit of being highly reproducible. Both methods are detailed in Sections 5.3 and 5.4. An overview of these R&R methodological implementations is provided in Table 3.2.

Table 3.2 Overview of R&R Methodologies

Shaw (1998) (LNS)	
$\mathcal{R}^-$ → related removal	$\mathcal{R}^+$ → optimal insertion (branch-and-bound with LDS)
Schrimpf et al. (2000)	
$\mathcal{R}^-$ → random, radial or string removal	$\mathcal{R}^+$ → greedy insertion
Pisinger and Røpke (2007) (ALNS)	
$\mathcal{R}^-$ → random, worst, related, cluster, time-oriented or historical removal	$\mathcal{R}^+$ → greedy or regret insertion with or without noise function
This study's approach (SISRs)	
$\mathcal{R}^-$ → adjacent string removal	$\mathcal{R}^+$ → greedy insertion with blinks

Chapter 4

The Prisoner Transportation Problem (PTP)

4.1 Introduction

This thesis introduces the Prisoner Transportation Problem (PTP) to the academic community. The PTP, while being formally introduced by the present chapter for the first time, has previously been alluded to within operational research literature on occasion. In most cases, however, the problem constitutes a mere academic footnote without any meaningful or rigorous elaboration, for example by Partyka and Hall (2000). In essence, the problem concerns the transportation of prisoners to and from various destinations - such as hospitals and courts - in special prisoner transportation vehicles which are subdivided into smaller individual compartments.

The PTP constitutes a variant of the commonplace VRP, but more specifically the DARP. Two unique components make it a particularly interesting problem from a research point of view. The first one is simultaneous servicing, which enables multiple prisoners to embark or disembark simultaneously rather than sequentially. This is unlike the standard practice throughout most vehicle routing and delivery problems. The second component concerns inter-passenger conflicts, which may occur either at compartment or vehicle level. A compartment conflict indicates two prisoners who are not permitted to simultaneously occupy the same compartment, whereas a vehicle conflict denotes that they may not be transported in the same vehicle.

The PTP is a complicated problem and one which is currently solved manually by groups of planners who formulate schedules for the following day. These planners are expected to incorporate many safety measures while constructing such schedules. The real world case which inspired this research required up to three employees with certain skill combinations to transport a single prisoner. There are seven vehicle types, each of which has a different compartment layout and safety level. The vast majority of vehicles, located at one of thirteen depots, are of two types. The first is equipped with one compartment containing three seats and two compartments containing two seats, while the second type is equipped with only one three-seat and one four-seat compartment. After delivery of a prisoner within their time window, the vehicle may either travel onwards or wait on-site for its next service to begin. When certain prisoners are on board, it may be forbidden to wait on-site. Finally, maximum trip lengths should be respected for each individual prisoner. Given that many hundreds of prisoners require transportation on a daily basis, these planners employ technically unnecessary constraints, such as requiring vehicles to only carry prisoners of the same category for the entire day. These constraints make the problem more manageable for human planners, but are not necessary from a computational perspective and may unnecessarily avoid solutions which would be feasible in reality.

While the PTP brings together a myriad of features and constraints, many of these are commonplace throughout other VRPs or PDPs. Therefore, in the interest of stimulating further research concerning the PTP, only those constraints which are the most characterizing for the PTP are maintained.

The following section first reviews related literature and indicates similarities and differences with respect to the PTP in Section 4.2. Section 4.3 formally describes the PTP with a mathematical programming formulation, while Section 4.4 adapts this formulation to several variants of the PTP. It also includes a computational analysis using these models and an iterative procedure to solve small size problem instances. Section 4.5 then provides a global overview of the contributing heuristic algorithm, while Sections 4.6 and 4.7 describe how inter-passenger conflicts and time-related restrictions are handled. Section 4.8 describes instances and discusses experimental results and, finally, Section 4.10 provides concluding remarks and outline some possible future research directions.

4.2 Related problems

Given that the PTP concerns the transportation of people, as opposed to goods or commodities, it is associated with some uncommon constraints. Despite the vast number of published VRP models and algorithms, the PTP is unsolvable with existing approaches. Indeed, due to the novel constraints outlined in the introduction, the PTP generalises both the VRP and the DARP. When considering passenger transportation, DARP and the handicapped persons transportation problem (Toth and Vigo 1997) provide inspiring foundations, albeit inter-passenger conflicts are still absent in these studies.

The PTP differs from the DARP in that inter-passenger conflicts must be respected by preventing 'conflicting' prisoners from sharing a vehicle or compartment. Incorporating this constraint dramatically increases computational complexity compared to existing passenger transportation problems. The DARP variant studied by Beaudry et al. (2010) considers the transportation of patients who need to be transported in isolation as well as those who can be safely transported with others. Isolated patients can be considered as a special case of the conflicts present in the PTP where certain requests have vehicle conflicts with all other requests. This dedication of vehicles to a single patient can also be found in the study by Parragh et al. (2012). While Molenbruch et al. (2017) introduce a DARP with restrictions similar to the PTP's vehicle conflicts, they do not consider compartments, and hence no compartment conflicts, in their study. Product incompatibility restrictions

within compartments are considered in the Multi-Compartment VRP (MCVRP) (Fallahi et al. 2008). The three-index mixed integer programming formulation provided by Derigs et al. (2011) can be adapted to several variants of the MCVRP. However, it is not straightforward to modify this formulation to solve the PTP, primarily due to the pickup and delivery aspect of the PTP, which allows for the transportation of conflicting prisoners in the same compartment at different legs of a vehicle trip. This possibly necessitates decision variables with four or more indices. Such a formulation is provided by Lahyani et al. (2015) for a multi-product, multi-period MCVRP to restrict the number of different product types in a compartment to one at any period. From a modeling point of view, a vehicle trip at a period in this MCVRP variant can be considered as one leg of a vehicle trip in the PTP. Note that this type of compartment restriction remains a special case of the one in the PTP where every prisoner (or a group of prisoners) conflicts with every other prisoner (or every other group of prisoners) at the compartment level. Another VRP variant studied by both Oppen and Løkketangen (2008) and Oppen et al. (2010) requires certain animal types to be held in different compartments. However, since the animal types considered in these papers have dedicated compartments, no conflict can occur within compartments. The same characteristics appears in other MCVRP variants which dedicate separate compartments to conflicting goods.

A second significant difference between the PTP and other related problems concerns the way in which service times must be handled. Prisoner transportation enables the simultaneous execution of multiple services at a single location, for example (dis)embarking multiple prisoners at once. In such a case, the total service time at a location must be greater than or equal to the maximum of those individual service times. The combined service times encountered in batching machines scheduling (Potts and Kovalyov 2000) are similar. Indeed, a batching machine's service time equals the maximum service time of all items in the batch. However, Potts and Kovalyov (2000) do not consider any time windows in their schedules.

The DARP, meanwhile, assumes no more than one service at a time. On-site service times may be categorized as deterministic, stochastic, unknown or dependent according to Eksioglu et al. (2009). Their taxonomic review and recent classification of VRPs never consider multiple services per location. Nevertheless, many situations of this nature do occur in practice. For example, services may be conducted sequentially, and thus their execution order becomes important. As such, one on-site execution sequence may prove infeasible due to time window violations whereas another may introduce unnecessary waiting times. Parragh et al. (2015) introduce the DARP with split requests and profits, where group requests may be split across vehicles while service times are assumed independent of the number of people being picked-up or delivered. Although the

problem is somewhat similar to the PTP, the model proposed by Parragh et al. (2015) does not accommodate the simultaneous servicing of passengers whose time windows are possibly different. Desaulniers (2010) assumes service times to be quantity-independent in the Split Delivery VRP with Time Windows (SDVRPTW). Salani and Vacca (2011) introduce the discrete SDVRPTW which assumes delivery-dependent service times. Although uncommon in academic papers, this model is very appropriate in real-world logistics. Service times may, for example, equal the sum of a delivery setup time and a duration proportional to the number of items delivered. In the context of prisoner transportation, the critical period during which prisoners (dis)embark must be minimized. The transportation organization consequently assigns sufficient staff and resources to accommodate the handling of each prisoner's service independently. Multiple on-site services are therefore executed at once, provided the services' time windows are respected.

From a fleet perspective, the PTP also generalizes the MCVRP. The MCVRP deals with specific product types, each having a corresponding dedicated compartment, possibly of flexible sizes (Henke et al. 2015). Waste collection, which exhibits conflicts at the product type level, represents an important application domain for MCVRP. The PTP, by contrast, expresses conflicts at an individual level. Additionally, prisoners may occupy any compartment provided there are no conflicts with other prisoners occupying that compartment at the same time. Paraskevopoulos et al. (2017) indicate the existence of multiple-compartment vehicles for transporting non-mixable products. They explicitly mention that research regarding heterogeneous vehicle types and multiple products is largely absent.

When comparing the PTP to modeling knowledge available in the literature, both inter-passenger conflicts and simultaneous servicing are new and provide interesting challenges to academia.

4.3 Notation and problem formulation

Let $G = (N, A)$ be a network such as that depicted in Figure 4.1, with arc set A and node set $N = U \cup I$ where U represents the set of depot nodes and $I = I^P \cup I^D = \{1, \ldots, 2n\}$ represents the set of service nodes with $I^P = \{1, \ldots, n\}$ corresponding to pickup nodes and $I^D = \{n+1, \ldots, 2n\}$ denoting delivery nodes. The set of depots U is a union of two partitions of identical size: $U^1 = \{u_1, \ldots, u_k\}$ and $U^2 = \{u_1^e, \ldots, u_k^e\}$ where U^1 denotes the set of original depots as the starting depots of vehicles and where U^2 contains their duplicates as the ending depots. Thus, for any node $u \in U^1$, U^2 contains one and only one

duplicate node u^e. Moreover, $A = A^1 \cup A^2 \cup A^3$ where $A^1 = \{(i,j) : i \in U^1, j \in I\}$, $A^2 = \{(j,i) : i \in U^2, j \in I\}$ and $A^3 = \{(i,j) : i,j \in I : i \neq j, i \neq j+n\}$.

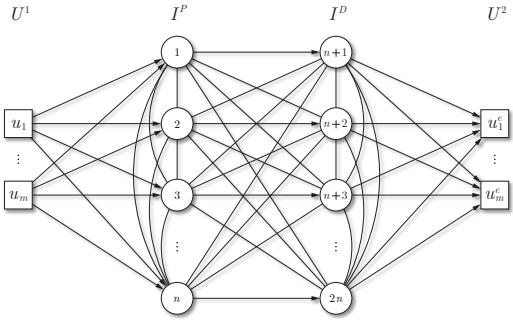

Figure 4.1 Underlying network representation for the PTP formulations

Define P as the set of prisoners and R as the set of all transportation requests. Each request corresponds to a node tuple $(i, n+i)$ such that $1 \leq i \leq n$. Each node $i \in I$ associated with request $r(i) = r(n+i) \in R$ of prisoner $p(i) = p(n+i) \in P$ has a service time s_i and a time window $[t_i^-, t_i^+]$. Each service must be completed within its time window. Note that it is possible for an individual prisoner to have multiple requests and multiple services (nodes) may take place at the same physical location. Let L be the set of all distinct physical locations of the nodes in N and $l(i) \in L$ denote the location of $i \in I$, then the length e_{ij} of arc $(i,j) \in A$ is equal to the travel time from $l(i) \in L$ to $l(j) \in L$, with $e_{ij} = 0$ if $l(i) = l(j)$. We further define $t^+_{max} = \max_{i \in I^D} t_i^+$ and $t_{max} = t^+_{max} + \max_{(i,j) \in A^2} e_{ij}$ to be used in the formulations.

Multiple services occurring at the same location may be executed simultaneously. However, partial, or even the complete, simultaneous execution of services depends on the degree to which their respective time windows overlap. In a general setting, we denote $M = \{(i,j) : i,j \in R\}$ as the set of request pairs which can be served simultaneously.

Let $V = \{1, \ldots, |V|\}$ denote the set of vehicles and $u_v, u^e_v \in U$ denote the starting and ending depot nodes of $v \in V$. All vehicles are of the same type and each vehicle v is defined by a subset C_v of compartments C. To be more explicit, $C = \{c_1, c_2, \ldots, c_k, c_{k+1}, \ldots, c_{2k}, \ldots, c_{(|V|-1)k+1}, \ldots, c_{|V|k}\}$ and $C_v = \{c_{(v-1)k+1}, \ldots, c_{(v-1)k+k}\}$. Each compartment $c \in C$ has a capacity of q_c and belongs to vehicle $v(c) \in V$. Figure 4.2 illustrates the notation for vehicle compartments. Let H^c and H^v be the sets of all pairs of prisoners who may not simultaneously occupy the same compartment and who may not travel in the same vehicle, respectively. Analogous to H^c and H^v, we further introduce

$R^c = \{(r(i), r(j)) : (p(i), p(j)) \in H^c\}$ and $R^v = \{(r(i), r(j)) : (p(i), p(j)) \in H^v\}$ as the conflict sets at the request level.

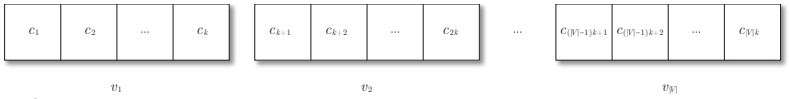

Figure 4.2 The notation for vehicle compartments.

The trip length for any request r, which begins when the prisoner is embarked and ends immediately before they disembark, should not exceed a certain time threshold L_r. Finally, the duration of a vehicle's trip may be no longer than D_{max}. The objective of the problem is to serve all prisoners while minimizing the total time spent by the vehicles.

We introduce three formulations which are slightly different from each other. The computational performances of these models are quite similar and none of them is clearly dominant over the others. In this section we present only the one that gives the best average solution values and keep the other formulations in Section 4.9 since the best solution values are obtained from the others for some of the instances.

4.3.1 A mixed integer programming formulation (FPTP)

We define the following decision variables: $z_{ij}^v = 1$ if vehicle v traverses arc $(i, j) \in A$, 0 otherwise; $x_{ij}^{rc} = 1$ if the prisoner associated with request r is carried in compartment c on arc $(i, j) \in A^3$, 0 otherwise; $y_{ij}^v = 1$ if node i precedes node j (not necessarily immediately) on the trip of vehicle v, 0 otherwise; τ_i^v is the departure time at node $i \in N$ by vehicle $v \in V$; and δ_i^v is the merged service time at node $i \in I$ by vehicle $v \in V$. The following is a Mixed Integer Programming Formulation for the PTP (FPTP).

$$\min \sum_{v \in V} (\tau_{u_v^e}^v - \tau_{u_v}^v) \tag{4.1}$$

$$\text{s.t.} \sum_{j \in I^P} z_{u_v j}^v \leq 1 \qquad \forall v \in V \tag{4.2}$$

$$\sum_{j \in I^P} z_{u_v j}^v = \sum_{j \in I^D} z_{j u_v^e}^v \qquad \forall v \in V \tag{4.3}$$

$$\sum_{j\in N} z_{ij}^{v} = \sum_{j\in N} z_{ji}^{v} \qquad \forall v \in V, i \in I \tag{4.4}$$

$$\sum_{v\in V}\sum_{j\in I} z_{ij}^{v} = 1 \qquad \forall i \in I^{P} \tag{4.5}$$

$$\sum_{c\in C_v} x_{ij}^{rc} \leq z_{ij}^{v} \qquad \forall v \in V, (i,j) \in A^{3}, r \in R \tag{4.6}$$

$$\sum_{j\in I}\sum_{c\in C} x_{ij}^{r(i)c} = 1 \qquad \forall i \in I^{P} \tag{4.7}$$

$$\sum_{j\in I} x_{ij}^{r(i)c} = \sum_{j\in I} x_{j(i+n)}^{r(i)c} \qquad \forall i \in I^{P}, c \in C \tag{4.8}$$

$$\sum_{j\in I} x_{ij}^{rc} = \sum_{j\in I} x_{ji}^{rc} \qquad \forall r \in R, i \in I : r \neq r(i), c \in C \tag{4.9}$$

$$x_{ij}^{rc} + x_{ij}^{r(i)c} \leq 1 \qquad \forall (r, r(i)) \in R^{c}, (i,j) \in A^{3}, c \in C \tag{4.10}$$

$$\sum_{c\in C_v} x_{ij}^{rc} + \sum_{c\in C_v} x_{ij}^{r(i)c} \leq 1 \qquad \forall (r, r(i)) \in R^{v}, (i,j) \in A^{3}, v \in V \tag{4.11}$$

$$\sum_{r\in R} x_{ij}^{rc} \leq q_c z_{ij}^{v(c)} \qquad \forall (i,j) \in A^{3}, c \in C \tag{4.12}$$

$$\tau_i^{v} + e_{ij} + \delta_j^{v} \leq \tau_j^{v} + t_{max}(1 - z_{ij}^{v}) \qquad \forall v \in V, (i,j) \in A : l(i) \neq l(j) \tag{4.13}$$

$$\tau_i^{v} \leq \tau_j^{v} + t_i^{+}(1 - z_{ij}^{v}) \qquad \forall v \in V, (i,j) \in A^{3} : l(i) = l(j) \tag{4.14}$$

$$\delta_i^{v} \geq \sum_{j\in I : l(i)=l(j)} s_j z_{ij}^{v} \qquad \forall v \in V, i \in I \tag{4.15}$$

$$\delta_j^{v} \geq \sum_{i\in I : l(i)=l(j)} s_i z_{ij}^{v} \qquad \forall v \in V, j \in I \tag{4.16}$$

$$\tau_{n+i}^{v} - \tau_i^{v} - s_{i+n} \leq L_{r(i)} \qquad \forall v \in V, i \in I^{P} \tag{4.17}$$

$$\tau^v_{u^e_v} - \tau^v_{u_v} \leq D_{max} \qquad \forall v \in V \tag{4.18}$$

$$\tau^v_{u^e_v} - \tau^v_{u_v} \geq \sum_{i \in I^P} (e_{u_v i} + s_i) z^v_{u_v i} + \sum_{i \in I^D} (e_{i u^e_v} + s_i) z^v_{i u^e_v} + \sum_{(i,j) \in A^3} e_{ij} z^v_{ij} \qquad \forall v \in V \tag{4.19}$$

$$t^-_i + s_i \leq \tau^v_i \leq t^+_i \qquad \forall v \in V, i \in I \tag{4.20}$$

$$s_i \leq \delta^v_i \leq L_{r(i)} \qquad \forall i \in I, v \in V \tag{4.21}$$

$$z^v_{ij} \leq y^v_{ij}, \qquad \forall (i,j) \in A^3, v \in V \tag{4.22}$$

$$y^v_{ij} + y^v_{ji} = 1, \qquad \forall (i,j) \in A^3, v \in V \tag{4.23}$$

$$y^v_{ij} + y^v_{jl} + y^v_{li} \leq 2 \qquad \forall (i,j),(j,l),(l,i) \in A^3, v \in V \tag{4.24}$$

$$y^v_{ij} \in \{0,1\} \qquad \forall v \in V, (i,j) \in A \tag{4.25}$$

$$z^v_{ij} \in \{0,1\} \qquad \forall v \in V, (i,j) \in A \tag{4.26}$$

$$x^{rc}_{ij} \in \{0,1\} \qquad \forall r \in R, c \in C, (i,j) \in A^3 \tag{4.27}$$

The objective function (4.1) minimizes the total time spent by the vehicles. Constraints (4.2) limit the number of outgoing arcs from the depot of any vehicle to one, whereas Constraints (4.3) equate this number to the number of incoming arcs to its duplicate depot node. For any intermediate node, Constraints (4.4) ensure that every vehicle which enters also leaves that node. Constraints (4.5) ensure that each pickup is served by a vehicle. Constraints (4.6) prevent requests to traverse an arc in a compartment if the vehicle associated with this compartment is not traversing that arc. While Constraints (4.7) assign a compartment to each request when leaving its pickup node, Constraints (4.8) and (4.9) make sure that the request is in the same compartment when entering its delivery node as well as any intermediate nodes, respectively. Constraints (4.10) and (4.11) prevent request assignments causing compartment and vehicle conflicts, respectively. The capacity restrictions of compartments are enforced by Constraints (4.12). Since simultaneously servicing distinct requests at the same location is possible, Constraints (4.13) update the time labels of consecutively visited nodes of distinct locations. When the locations of two consecutively visited nodes are identical, Constraints (4.14) ensure that the departure time at the preceding node is less than or equal to the following node's departure

time. Constraints (4.15) and (4.16) ensure that the merged service time of the nodes which are being simultaneously serviced is no less than the maximum of their individual service times. Constraints (4.17) and (4.18) impose the maximum travel time for requests and for vehicles, respectively. Constraints (4.19) require the total time spent by a vehicle to be greater than or equal to the total traveling time plus the total service time of the nodes visited by it. By Constraints (4.20), the departure time at each node is limited to be no earlier than the earliest departure time plus the service time and no later than the latest departure time. Constraints (4.21) restrict merged service time at a node to be greater than or equal to its service time and less than or equal to the maximum traveling time of the associated passenger. We eliminate sub-tours via Constraints (4.22)-(4.24), which are known to provide tight bounds (Öncan et al. 2009). Finally, Constraints (4.25)-(4.27) are binary restrictions.

4.3.2 Symmetry breaking constraints and additional valid inequalities

When vehicles are identical, Constraints (4.28) break the symmetry among the vehicles of any depot. Let v^{1*} denote the smallest index vehicle of the closest depot to the smallest index pickup node, Constraint (4.29) ensures that this pickup node is served by v^{1*}. For any identical pair of compartments in the same vehicle, Constraints (4.30) assign compartments to new requests in lexicographic order of compartment indices if both are empty.

$$\sum_{j \in I^P} z_{uj}^{v_2} \leq \sum_{j \in I^P} z_{uj}^{v_1} \qquad \forall v_1 < v_2 \in V : u = u_{v_1} = u_{v_2} \tag{4.28}$$

$$\sum_{(1,j) \in A^3} z_{1j}^{v^{1*}} \geq 1 \tag{4.29}$$

$$\sum_{(i,j) \in A^3} x_{ij}^{r(i)c_2} \leq \sum_{(i,j) \in A^3} \sum_{r \neq r(i)} (x_{ij}^{rc_1} + x_{ij}^{rc_2}) \quad \forall i \in I^P,\ c_1, c_2 \in C : c_1 < c_2,\ v(c_1) = v(c_2),\ q_{c_1} = q_{c_2} \tag{4.30}$$

Although Constraints (4.31)-(4.33) are implied by Constraints (4.2)-(4.5), their inclusion improves the performance of the formulation when solving with CPLEX.

$$\sum_{v \in V} \sum_{j \in U^1 \cup I} z_{ji}^{v} = 1 \qquad \forall i \in I^P \tag{4.31}$$

$$\sum_{v \in V} \sum_{j \in I} z_{ji}^{v} = 1 \qquad \forall i \in I^{D} \tag{4.32}$$

$$\sum_{v \in V} \sum_{j \in I \cup U^2} z_{ij}^{v} = 1 \qquad \forall i \in I^{D} \tag{4.33}$$

Constraints (4.34) and (4.35) are tighter than Constraints (4.13) for restricting the departure times at the depot nodes when the triangular inequality holds for $e_{ij}, (i,j) \in A$.

$$\tau_{u_v}^{v} + e_{u_v i} + \delta_i \leq \tau_i^{v} + t_{max}^{+}(1 - \sum_{j \in I} z_{ij}^{v}) \qquad \forall v \in V, i \in I \tag{4.34}$$

$$\tau_i^{v} + e_{i u_v^e} \leq \tau_{u_v^e}^{v} + t_{max}(1 - \sum_{j \in I} z_{ji}^{v}) \qquad \forall v \in V, i \in I \tag{4.35}$$

4.4 Modeling and testing the variants and special cases of the PTP

Heterogeneous fleets and passenger-vehicle compatibility restrictions: The formulations presented in Section 4.3, 4.9.1 and 4.9.2 solve the generalization of the PTP with a heterogeneous fleet of vehicles. When certain passengers can only be transported by a subset of vehicles, one can easily adapt these formulations by defining $V_i \subset V$ as the set of vehicles which can transport the passenger associated with node $i \in I$.

Minimizing the total routing cost: A common objective function used in the vehicle routing literature is to minimize the total routing cost, which is usually proportional to the total distance traveled. Let f_{ij}^{v} be the cost of traversing arc $(i,j) \in A$ with vehicle $v \in V$. The PTP to minimize the total routing cost can be solved by replacing the objective function with (4.36):

$$\min \sum_{v \in V} \sum_{(i,j) \in A} f_{ij}^{v} z_{ij}^{v}. \tag{4.36}$$

No simultaneous servicing (NSS): When $M = \emptyset$ (no passenger pair may be served simultaneously), the problem becomes a generalization of the DARP with passenger conflicts. To model this general case, we exclude δ variables, the constraints including those variables as well as Constraints (4.14) from FPTP. We then add (4.37) to update the departure times of consecutively visited nodes.

$$\tau_i^{v} + e_{ij} + s_j \leq \tau_j^{v} + t_{max}(1 - z_{ij}^{v}) \qquad \forall v \in V, (i,j) \in A \tag{4.37}$$

No passenger conflicts (NC): In order to formulate the special case of the PTP where there are no passenger conflicts ($R^C = \emptyset = H$), the compartment index of the x variables can be replaced with a single vehicle index. In this case, $x_{ij}^{rv} = 1$ if the passenger associated with request r is carried in vehicle v on arc $(i,j) \in A^3$, and 0 otherwise. While Constraints (4.10) and (4.11) are no more relevant and can therefore be omitted, Constraints (4.6)-(4.12) must be replaced with (4.38)-(4.42).

$$x_{ij}^{rv} \leq z_{ij}^{v} \qquad \forall v \in V, (i,j) \in A^3, r \in R \tag{4.38}$$

$$\sum_{j \in I} \sum_{v \in V} x_{ij}^{r(i)v} = 1 \qquad \forall i \in I^P \tag{4.39}$$

$$\sum_{j \in I} x_{ij}^{r(i)v} = \sum_{j \in I} x_{j(i+n)}^{r(i)v} \qquad \forall i \in I^P, v \in V \tag{4.40}$$

$$\sum_{j \in I} x_{ij}^{rv} = \sum_{j \in I} x_{ji}^{rv} \qquad \forall r \in R, i \in I : r \neq r(i), v \in V \tag{4.41}$$

$$\sum_{r \in R} x_{ij}^{rv} \leq \sum_{c \in C : v(c) = v} q_c z_{ij}^{v} \qquad \forall (i,j) \in A^3, v \in V \tag{4.42}$$

The dial-a-ride problem: When there are no passenger conflicts ($R^C = \emptyset = H$) and simultaneous servicing is not possible for any pair of prisoners ($M = \emptyset$), the PTP reduces to the DARP with the objective of minimizing the total travel time of the vehicles. For this case, FPTP can be utilized to solve the DARP by employing the aforementioned modifications for the NSS and the NC cases.

The PTP with transfers: An interesting variant of pickup-and-delivery problems involves transferring the load from one vehicle to another at certain transshipment points. This variant is referred to as Pickup & Delivery Problem with Transfers (PDPT) in the literature. Although transfers are not considered in this particular PTP application that we study, such a variant might be worthwhile investigating. For modeling the PTP with transfers, we refer the reader to a very elegant study by Rais et al. (2014) that formulates several variants of the PDPT which can easily be combined with the formulations we provide for the PTP.

4.4.1 Testing the formulations

Although the PTP considers there to be an infinite number of vehicles available, the number of vehicles utilized in any feasible solution cannot be greater than

the number of requests. Preliminary experiments indicate that models perform relatively better when $|V| \leq 3$ but suffer from slow convergence when $|V| \geq 4$. In order to overcome this difficulty, we implemented an iterative algorithm which utilizes any of these formulations to solve the problem for a fixed number $1 \leq k \leq |V|$ of vehicles at each iteration k.

Let $F(UB, k)$ be any of the PTP models where $|V| = k$, $\sum_{j \in I^P} z^v_{u_v j} = 1, \forall v \in V$ and $UB - \epsilon$ is an upper bound on the objective function value with $\epsilon > 0$ being a sufficiently small number. Let Θ be an optimal solution obtained from $F(UB, k)$ and $f(\Theta)$ be its value, Algorithm 1 details an iterative procedure for solving the PTP to proven optimality.

Algorithm 1 Iterative algorithm (IT)

Input: n, $UB \leftarrow \infty$, $k \leftarrow 1$, $\epsilon > 0$
Output: *bestSolution*

```
while k ≤ n do
    Solve F(UB, k)
    if feasible with the optimal solution Θ then
        UB ← f(Θ) − ε
        bestSolution ← Θ
        k ← k + 1
    else if infeasible then
        if UB = ∞ then
            k ← k + 1
        else
            k ← n + 1
return bestSolution
```

In order to investigate the impact of simultaneous servicing and conflicts on the performance of the models and the solutions they provide, we conduct tests removing these components one at a time. Let PF be one of the aforementioned formulations for some variant of the PTP, we denote the iterative procedure for solving PF as IT-PF in this section.

Table 4.1 compares the performance of the PTP and NSS formulations and their iterative procedures on the instances with up to 10 requests (20 service nodes). The 'id' of an instance consists of three numbers in a format $n_g_n^C$ where n is the number of requests to serve, g is the group code and n^C is the number of compartments in each vehicle. Among these instances, the first group ($g = 0$) contains a large number of services that can be served simultaneously, none of which conflict with one another. The second group of instances ($g = 1$) has either no or only few services that can be served simultaneously and it contains

many conflicting requests. The third group ($g = 2$) has more balanced instances with both mergeable services and conflicting requests. For each instance, the compartment configuration is provided in the second column: 1-1-1-1 indicates four compartments where each has a capacity of a single unit, 2-2 denotes two compartments where each has a capacity of 2, while the final configuration is simply 4 and denotes a single compartment whose capacity is 4.

Table 4.1 Comparison of PTP and NSS methods

			PTP		FPTP			IT-PTP		NSS		FNSS			IT-NSS		PTP (4.36)	
id	Comp.	n	Best	k	Obj	$t(s)$	$g\%$	Obj	$t(s)$	Best	k	Obj	$t(s)$	$g\%$	Obj	$t(s)$	Best	k
4_0_4	1-1-1-1	4	**54**	1	**54**	9.73	0.00	**54**	3.21	**129**	1	**129**	17.22		**129**	5.97	**9**	1
4_1_4	1-1-1-1		**271**	2	**271**	0.22	0.00	**271**	0.65	**271**	2	**271**	2.32		**271**	0.82	**42**	1
4_1_2	2-2		**271**	2	**271**	0.24	0.00	**271**	17.12	**271**	2	**271**	0.24		**271**	11.38	**42**	1
4_1_1	4		**271**	2	**271**	0.19	0.00	**271**	15.43	**271**	2	**271**	0.16		**271**	8.31	**46**	1
4_2_4	1-1-1-1		**181**	2	**181**	0.62	0.00	**181**	2.04	**226**	2	**226**	3.52		**226**	3.09	**102**	1
4_2_2	2-2		**181**	2	**181**	0.98	0.00	**181**	2.10	**226**	2	**226**	4.07		**226**	1.69	**102**	1
4_2_1	4		**181**	2	**181**	1.25	0.00	**181**	12.54	**226**	2	**226**	1.21		**226**	3.78	**102**	1
6_0_4	1-1-1-1	6	**90**	2	90	TL	93.89	**90**	405.92	**193**	1	**193**	TL	94.82	**193**	3198.74	**13**	1
6_1_4	1-1-1-1		**399**	1	**399**	93.68	0.00	**399**	10.29	**399**	1	**399**	78.55		**399**	8.25	**69**	1
6_1_2	2-2		**399**	1	**399**	5.96	0.00	**399**	11.78	**399**	1	**399**	8.18		**399**	3.49	**69**	1
6_1_1	4		**403**	1	**403**	2.35	0.00	**403**	7.88	**403**	1	**403**	1.82		**403**	1.20	**73**	1
6_2_4	1-1-1-1		**282**	3	**282**	353.12	0.00	**282**	34.49	**342**	3	**342**	1137.91		**342**	135.57	**106**	1
6_2_2	2-2		**282**	3	**282**	130.94	0.00	**282**	13.61	**342**	3	**342**	380.57		**342**	39.69	**106**	1
6_2_1	4		**282**	3	**282**	6.27	0.00	**282**	114.64	**342**	3	**342**	47.34		**342**	38.67	**106**	1
8_0_4	1-1-1-1	8	106	2	126	TL	97.88	106	TL	256	1	256	TL	98.44	256	TL	**15**	1
8_1_4	1-1-1-1		**516**	2	519	TL	60.67	516	TL	**529**	2	536	TL	74.82	529	TL	**83**	2
8_1_2	2-2		**516**	2	516	TL	37.44	**516**	1659.33	**529**	2	529	TL	61.06	**529**	564.73	**83**	2
8_1_1	4		**521**	2	**521**	169.04	0.00	**521**	23.22	**536**	2	**536**	273.47		**536**	17.12	**83**	2
8_2_4	1-1-1-1		**356**	3	393	TL	59.45	356	TL	423	2	423	TL	69.98	423	TL	**177**	1
8_2_2	2-2		**356**	3	356	TL	16.57	**356**	2175.07	**423**	2	423	TL	39.95	**423**	3672.67	**177**	1
8_2_1	4		**356**	3	**356**	2757.98	0.00	**356**	823.36	**423**	2	423	TL	27.23	**423**	1251.62	**177**	1
10_0_4	1-1-1-1	10	106	2	273	TL	100.00	106	TL	331	3	366	TL	100.00	331	TL	20	2
10_1_4	1-1-1-1		599	2	607	TL	75.86	599	TL	629	2	642	TL	75.62	629	TL	**134**	2
10_1_2	2-2		599	2	682	TL	84.90	599	TL	629	2	674	TL	84.72	629	TL	**134**	2
10_1_1	4		**601**	2	603	TL	31.82	**601**	488.14	**631**	2	633	TL	37.60	**631**	436.35	**134**	2
10_2_4	1-1-1-1		451	2	706	TL	76.00	451	TL	541	2	631	TL	73.22	541	TL	227	2
10_2_2	2-2		450	2	531	TL	67.93	450	TL	540	2	581	TL	70.42	540	TL	**227**	2
10_2_1	4		450	2	467	TL	63.09	450	TL	540	2	540	TL	67.12	540	TL	**227**	2

Obj: best solution value; k: number of vehicles used; $t(s)$: solving time in seconds; $g\%$: CPLEX gap; TL: time limit (3600 seconds) is reached.

Table 4.1 indicates the optimal values in boldface and the best solution values obtained if none of the methods reach proven optimality. Some of these optimal values are obtained by the iterative procedures of other formulations (see 4.9). Columns '$g\%$' report the CPLEX gap for the compact formulations when they reach the time limit of one hour. If the time limit is reached at any iteration of the algorithms, this is indicated with 'TL' under the solving time column '$t(s)$' and the objective value of the best solution obtained is reported under column

'Obj'. We do not report CPLEX gaps for the iterative algorithms as they are only relevant for the restrictive individual problems requiring exactly k vehicles, but not for the original problem with at most n vehicles. The last two columns of this table provide the results obtained from the PTP where the objective function is replaced with Equation (4.36) to minimize the total routing cost.

When there are many conflicting passengers, we observe that the compartment configuration matters: the best solution value increases for those instances where vehicles have a single compartment rather than multiple. The optimal values mostly increase as the number of requests gets larger, but the magnitude of the increase is less for $g = 0$ instances. Note that these instances are high in number of requests that can be served simultaneously, with none of them conflicting. This enables serving more requests without incurring any additional travel or service time if the new requests can be combined with the existing ones, as in the case for instances 8_0_4 and 10_0_4.

Comparing the performance of individual PTP methods, we observe that although FPTP exhibits a satisfactory performance on small instances, particularly the ones with 4 requests, it performs poorly on larger instances. On the other hand, using the iterative algorithms we are able to obtain the optimal solutions for all the instances with up to 8 requests, except for 8_0_4. The iterative algorithms only terminate with proven optimality for 10_1_1 among the instances where $n = 10$.

The performance of the NSS methods follows a pattern similar to that of the PTP methods in the sense that the PTP instances which reach the time limit also remain unsolved. In general, the $g = 0$ instances appear the most difficult for all methods. The compact formulations are in particular unable to obtain high quality dual bounds quickly and they need to explore a large number of branch-and-bound nodes. We suspect this is due to a large number of symmetric solutions given that the time windows of most requests in this group largely overlap and there are no conflicts to, at least partially, break the symmetry.

The most interesting observation when comparing the best solution values of the PTP and the NSS is that one can save a considerable amount of time by simultaneously serving requests whenever possible. The gain is particularly significant for those instances which contain a large number of requests convenient for simultaneous servicing.

Table 4.2 provides the results for the NC and the DARP. Testing different compartment configurations is irrelevant for these problems since no passenger conflicts are present. Therefore, all the instances in this table have vehicles with a single compartment whose capacity is 4. The last two columns of this table report the results obtained from the DARP, where the objective function

is replaced with (4.36) to minimize the total routing cost. When comparing the best solution values of the NC and the PTP, we observe that it is higher for the PTP in all instances that contain conflicting customers, except for 6_2_1 which has a relatively sparse conflict matrix. The best solution values of the PTP are generally lower than the best solution values of the DARP. However, we observe that these values might be higher for the PTP for those instances where there is little to gain by simultaneous servicing and for which passenger conflicts are in effect.

Table 4.2 Results for NC and DARP methods.

		PTP	NC		FNC		IT-NC		DARP		FDARP		IT-DARP		DARP (4.36)	
id	n	Best	Best	k	Obj	$t(s)$ $g\%$	Obj	$t(s)$	Best	k	Obj	$t(s)$ $g\%$	Obj	$t(s)$	Best	k
4_0_1	4	54	**54**	1	**54**	2.35	**54**	11.42	**129**	1	**129**	10.28	**129**	2.85	**9**	1
4_1_1		271	**271**	2	**271**	0.27	**271**	4.43	**271**	2	**271**	0.49	**271**	1.26	**42**	1
4_2_1		181	**181**	2	**181**	1.09	**181**	8.50	**226**	2	**226**	0.9	**226**	1.34	**102**	**1**
6_0_1	6	90	**90**	2	**90**	279.78	**90**	119.19	**193**	1	**193**	1947.75	**193**	151.21	**13**	1
6_1_1		403	**399**	1	**399**	14.14	**399**	9.35	**399**	1	**399**	20.12	**399**	2.04	**69**	1
6_2_1		282	**282**	3	**282**	14.77	**282**	66.89	**342**	3	**342**	155.15	**342**	12.54	**106**	1
8_0_1	8	106	106	2	106	TL 96.23	106	TL	256	1	256	TL 98.34	256	TL	**15**	2
8_1_1		521	**514**	**1**	516	TL 31.36	**514**	378.64	**529**	2	536	TL 35.06	**529**	748.25	**83**	2
8_2_1		356	**350**	**3**	352	TL 44.55	**350**	5385.99	414	2	414	TL 71.56	414	TL	**137**	2
10_0_1	10	106	106	1	110	TL 96.36	106	TL	320	3	326	TL 99.39	320	TL	16	2
10_1_1		601	576	3	609	TL 91.20	601	TL	610	2	639	TL 92.04	610	TL	104	2
10_2_1		450	444	2	507	TL 80.19	444	TL	535	2	576	TL 79.35	535	TL	216	2

When comparing the best solution values of the PTP and its variants, we observe that among all the problems included in these tables, the best solution values are always the highest for the NSS. For those which consider inter-passenger conflicts, the least number of instances are solved to optimality for the NSS. The results indicate that it was computationally easier to solve the PTP with Objective (4.36) than the original PTP on these instances. This is reflected in both the number of instances solved to optimality and the average solving time. A similar pattern is observed when comparing the DARP and the DARP with Objective (4.36) as well.

The mathematical models introduced in this chapter provide formal descriptions for the PTP variants and insights for solving them to optimality. However, the challenging combination of constraints present in these formulations make it difficult to solve real-world instances of the PTP in reasonable amounts of time. As the practitioners need to update their trips on a daily basis, we also introduce a heuristic to solve realistically generated large-scale instances and provide an additional computational analysis on these instances.

4.5 PTP Algorithm

The heuristic approach introduced for addressing the PTP in this thesis utilizes the ruin & recreate approach proposed by Schrimpf et al. (2000), guided by a stochastic Local Search-based metaheuristic (LS). LS is employed given that it is both computationally quick and effective for the capacitated vehicle routing problem and similar other problems (Laporte 2009). The PTP is accordingly decomposed into a master and a subproblem. The master problem is modeled as a multi depot dial-a-ride problem with simultaneous servicing (Section 4.7), while the prevention of the inter-passenger conflicts (Section 4.6) corresponds to the subproblem.

4.5.1 Constructive heuristic

The initial solution is created by a greedy constructive method. All prisoners are first stored in a list while the solution is completely empty. The resulting unscheduled prisoner list is shuffled before employing *best insertion*. The procedure iterates over all feasible combinations for both pickup and delivery insertions inside the current partial solution and eventually inserts the prisoner's request into the solution at the minimal additional cost. If no such feasible combination exists, such as when none of the vehicles inside the incumbent solution are capable of transporting the prisoner within D_{max}, a new trip is initiated at the depot closest to the relevant prisoner. This best insertion procedure is repeated until all prisoner transports are scheduled.

4.5.2 Local search improvement

Simulated Annealing (Kirkpatrick et al. 1983) is utilized to guide the local search towards better solutions. The initial temperature is set to 100 and is cooled down to 1 (the final temperature). Both temperatures were empirically determined. The cooling rate α is computed such that the final temperature is reached after one million iterations. Classical local search neighbourhoods such as 2-opt* would prove impractical given the PTP's constraints since, for example, timing and allocation feasibilities are both affected when adjusting the order of visits in a trip. Therefore, a more flexible local search heuristic is applied during the improvement iterations. A current solution's neighbour is generated iteratively by a ruin & recreate procedure. This procedure begins from a copy of the current solution and removes its first trip. The removed prisoners are subsequently appended to the unscheduled prisoner list. The ruined solution is afterwards recreated by the same best insertion procedure

employed for the initial solution's construction. The resulting neighbour solution is passed to the metaheuristic's acceptance procedure for either rejecting or accepting it as the incumbent solution. Furthermore, a neighbour may only be accepted if it contains either an equal or greater number of prisoners than the current solution. If the neighbour is rejected, the current solution's first trip is moved to the end of the solution's list of trips.

4.6 Prisoner allocation heuristic

Given a vehicle's trip and a set of prisoners, one should check whether or not all prisoners can be allocated to truck compartments without violating inter-passenger conflicts. Formally, this feasibility problem can be defined as follows: Let $T = \{T_0, \ldots, T_\eta\}$ be the set of transit points of a vehicle's trip. Define P_t as the set of prisoners to be allocated to the compartments of the vehicle at transit point $t \in T$, $P^T = \bigcup_{t \in T} P_t$, and F as the set of conflicting prisoner pairs. Prisoner Allocation Problem (PAP) checks if a set P^T of prisoners can be feasibly allocated to the compartments of the vehicle such that no conflicting prisoners are in the same compartment at the same time.

The PAP is a generalization of the Bounded Vertex Coloring Problem (BVCP). Consider the special case of the PAP where all compartments have identical capacities, say q, and all prisoners of a vehicle trip T are picked up at the transit point T_1 and delivered to transit point T_2. The latter implies that all prisoners will be traveling in the vehicle between points T_1 and T_2. This special case of the PAP reduces to the BVCP which is known to be NP-Complete on general graphs with $q \geq 3$ (Hansen et al. 1993).

In order to formulate the PAP, define a binary decision variable W_{pc} which equals 1 if prisoner $p \in P^T$ is assigned to compartment $c \in C$, and 0 otherwise. Equations (4.43)-(4.46) provide an integer programming model for the PAP. Constraints (4.43) guarantee that each prisoner in P^T is assigned to exactly one compartment in C. Constraints (4.44) ensure that the number of assigned prisoners to a compartment c never exceeds its capacity q_c, while Constraints (4.45) ensure no conflicting prisoners are assigned to the same compartment. Note that set F only contains those conflicting pairs of prisoners whose presence in the vehicle overlaps in time.

$$(FPAP) \qquad \sum_{c \in C} W_{p,c} = 1 \qquad \forall\, p \in P^T \tag{4.43}$$

$$\sum_{p \in P_t} W_{p,c} \leq q_c \qquad \forall\, c \in C, t \in T \tag{4.44}$$

$$W_{p1,c} + W_{p2,c} \leq 1 \qquad \forall\ (p1, p2) \in F, c \in C \tag{4.45}$$

$$W_{p,c} \in \{0, 1\} \qquad \forall\ p \in P^T, c \in C \tag{4.46}$$

This feasibility check must be conducted very frequently within the PTP heuristic. The fact that FPAP requires a relatively lengthy amount of time encourages us to employ a fast greedy approach for solving the PAP. Consider an iterative approach where prisoners are added to a trip one by one. The feasibility check implies that, for each prisoner insertion evaluation, a feasible prisoner-truck allocation must be achieved during the complete trip.

Figure 4.3 illustrates an example trip which stops at six transit points while transporting four prisoners. Figure 4.3(a) shows the sequence of the trip's transit points and prisoner transports a, b, c and d via dashed arrows, whereas Figure 4.3(b) indicates the vehicle's occupation throughout the trip.

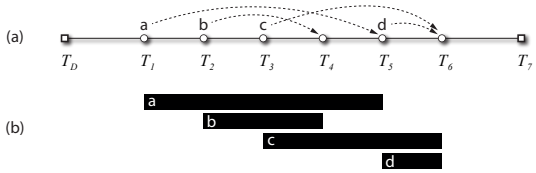

Figure 4.3 A trip transporting four prisoners (a) and their presence in the truck (b).

Figure 4.4(a) illustrates a vehicle with two compartments with (b) corresponding to a graph H^c wherein an edge indicates an inter-passenger conflict. It is noteworthy that while prisoners b and d may not share the same compartment, this edge is irrelevant given that the presence of these two prisoners in the vehicle does not overlap in time. Therefore this edge may be removed from the graph, resulting in (c). In order to find a feasible prisoner allocation, one may consider a graph coloring problem wherein the number of an individual color's assignments is bound to a compartment's capacity at each transit point.

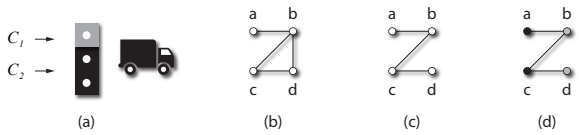

Figure 4.4 Prisoner allocation and conflict graphs.

Figure 4.4(d) provides a feasible allocation: Prisoners a and c are assigned to compartment 2 and prisoners b and d to compartment 1. Notice that while the color for compartment 1 is employed more than its corresponding capacity, it

never exceeds this capacity at any single transit point as illustrated in Figure 4.5.

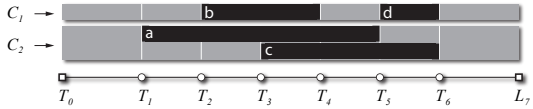

Figure 4.5 **Solution for allocating prisoners.**

4.6.1 Greedy Prisoner Allocation Heuristic

This section introduces a greedy Prisoner Allocation Heuristic (PAH) to assign prisoners to compartments. The algorithm is based on a graph coloring heuristic proposed by Al-Omari and Sabri (2006). The pseudocode is provided by Algorithm 2. All prisoners which are transported by trip T are included in set P^T. These prisoners are sequentially assigned to one of the vehicle's compartments. To begin, the prisoner p with the fewest feasible compartment allocations is selected from P^T (lines 3-11). If this number of compartments is the same for two prisoners, then the prisoner with the greatest number of conflicts (the number of incident edges in Figure 4.4 (c)) is selected. Next, the compartment with the greatest number of free seats is selected from the set of all compartments with no conflicting prisoners for p (lines 12-21). If prisoner p cannot be assigned to any compartment, the heuristic fails to find a feasible assignment for the trip.

Although Algorithm 2 does not guarantee finding a feasible allocation (should one exist), the extensive experimental evaluations conducted indicate its effectiveness. In these experiments, the PAH is tested on a set of randomly generated prisoner allocation instances, which are proven to be feasible using the FPAP. Vehicles contain up to 10 compartments, with the number of seats within each individual compartment between 1 and 10. The number of transported prisoners is 50-100% of the total number of seats in the vehicle. The fraction of conflicting pairs is selected from between 30% and 100% of all pairs. Finally, the number of transit points is fixed to 10 while the number of transit points at which a prisoner can be present in the vehicle is at least 1 and at most 7.

Among 19581 PAP instances mathematically proven to be feasible, the PAH was able to generate a feasible solution for 18949, corresponding to a success rate of 96.7%. Given its efficiency, the greedy PAH was taken to be sufficiently effective for employment within the PTP heuristic.

Algorithm 2 Prisoner Allocation Heuristic

Input: Trip T and vehicle V

```
1:  procedure PAH(T,V)
2:      P^T ← prisoners transported by T
3:      while P^T ≠ ∅ do
4:          p ← null                          ▷ select the next prisoner to allocate
5:          for p' ∈ P^T do
6:              if p = null then
7:                  p ← p'
8:              else if #nonConfComps(p') < #nonConfComps(p) then
9:                  p ← p'
10:             else if #nonConfComps(p') = #nonConfComps(p) then
11:                 if #conflicts(p') > #conflicts(p) then
12:                     p ← p'
13:         c ← null                          ▷ select a compartment for p
14:         for c' ∈ nonConflictingComps(p) do
15:             if c = null then
16:                 c ← c'
17:             else if #freeSeats(c') < #freeSeats(c) then
18:                 c ← c'
19:         if c ≠ null then
20:             assign p to c
21:         else
22:             return false
23:     return true
24: end procedure
```

4.7 Accommodating time constraints

When composing or evaluating a trip, intelligent time constraint handling is indispensable. This section details how the PTP's time constraints are employed for constructing feasible trips. The discussion is presented for a single vehicle trip and, for simplicity reasons, vehicle indices are dropped from the notations and equations.

4.7.1 Time windows

Let k be a prisoner's pickup or delivery service. The service time s_k to embark or disembark the prisoner from the vehicle must be completed within its time window $t_k = [t_k^-, t_k^+]$. Note that this time window definition deviates from the standard definition, where the service time must start but not necessarily be completed within the time window. Figure 4.6 represents the time window by way of a gray rectangle, with the earliest possible execution of the service illustrated by a black rectangle above this time window while the latest possible execution is illustrated with a black box below this same time window. Figure 4.6(a) illustrates a lengthy time window for servicing prisoner a. A service which must take place at a specific time is represented by way of the time window's length equaling the service time ($s_k = |t_k|$), as indicated in Figure 4.6(b). The third possible case occurs when a truck departs or arrives at the depot with no prisoners on board, here the service is only restricted by a time window without any service time for (dis)embarking a prisoner ($s_k = 0$), as illustrated in Figure 4.6(c).

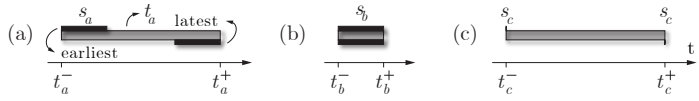

Figure 4.6 **Time window and service time notations.**

Inserting a single service into a trip's sequence of η transit points requires evaluating the feasibility of completing all services within their time windows. Although it is possible to evaluate this in $O(\eta)$ time by iterating over the complete trip, an alternative approach based upon auxiliary variables enables $O(1)$ evaluation. First, earliest arrival times a_k^- and earliest departure times d_k^- are precalculated at every location by iterating from the earliest transit over the entire trip (Eq. 4.47). Recall that e_{ij} denotes the travel time from service i to j. Second, latest departure times d_k^+ and latest arrival times a_k^+ are precalculated by iterating over the trip from the latest to earliest transit (Eq. 4.48).

$$a_k^- = d_{k-1}^- + e_{k-1,k} \qquad d_k^- = \max\{a_k^-, t_k^-\} + s_k \tag{4.47}$$

$$d_k^+ = a_{k+1}^+ - e_{k,k+1} \qquad a_k^+ = \min\{d_k^+, t_k^+\} - s_k \tag{4.48}$$

Arrival and departure times may therefore only occur within their associated *arrival time window* $a_k = [a_k^-, a_k^+]$ and *departure time window* $d_k = [d_k^-, d_k^+]$. Such *feasibility time windows* are illustrated via two examples in Figure 4.7. A trip's transit points are denoted by the set $\{T_0, T_1, T_2, T_3\}$, where the first and last points correspond to the depot. The travel time from T_i to T_j (e_{ij}) is represented by an arrow. The arrival time window a and departure time

window d are illustrated by gray rectangles above and below the service time window t, denoted by $(a, t, d)_i$ for transit point T_i.

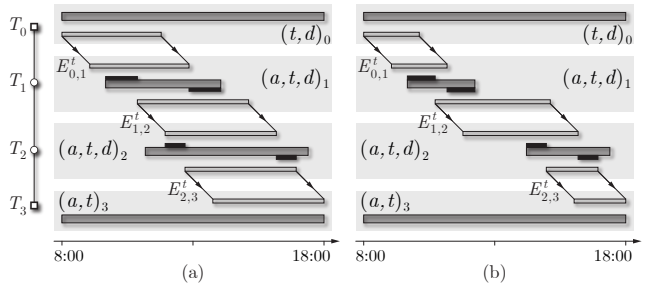

Figure 4.7 **Feasibility time windows, schedules without (a) and with wait times (b).**

Consider the insertion of a service k between services i and j of a trip's service sequence. It is sufficient to simply ensure feasibility of earliest arrival and earliest departure at k: $d_k^- \leq \min\{t_k^+, d_k^+\}$. Equivalently, one could check for feasible latest departure and latest arrival times at k: $a_k^+ \geq \max\{t_k^-, a_k^-\}$. In essence, the length of each arrival window $|a_k|$ is equal to the precomputed maximum delays, as in the *push forward* method proposed by Savelsbergh (1992, 1990) and Kindervater and Savelsbergh (2018).

4.7.2 Maximum trip duration

In addition to time window constraints, a trip is only feasible if its minimum duration D is no longer than D_{max}. Given feasibility time windows a_k and d_k for all trip transit points, one may determine a fixed time schedule by either earliest or latest trip execution of minimum duration. Figure 4.8 provides sample fixed schedules for the time windows provided by Figure 4.7.

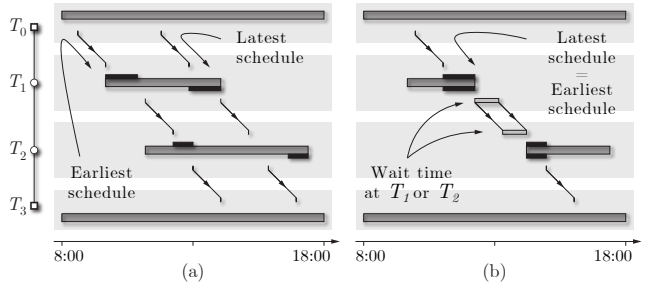

Figure 4.8 **Fixing the schedules, without (a) and with (b) wait times.**

The earliest schedule is obtained when all a_k^+ values are updated by pulling back the latest arrival time at the final transit point as much as possible ($a_\eta^+ = a_\eta^-$) and enabling the trip's schedule to begin at a_0^+. Similarly, if the earliest departure time at the first transit point were pushed forward as much as possible ($d_0^- = d_0^+$), before updating all following d_k^+, the latest schedule may be obtained, starting at d_0^-. The duration of a trip may now simply be obtained by $D = a_\eta^- - d_0^+$ of either the earliest or latest schedule. While this approach is inefficient as regards determining D due to its $O(\eta)$ runtime, it is employed given that it helps us introduce the following $O(1)$ approach. First, notice that if no inevitable wait times are imposed, see Figure 4.8(a), a trip's duration D is equal to the sum of all travel and service times. This simply corresponds to the amount of time that the trip's vehicle is active, which we refer to as A, given as in Eq. 4.49.

$$A = \sum_{k=0}^{\eta-1} e_{k,k+1} + \sum_{k=0}^{\eta} s_k \tag{4.49}$$

When inevitable wait times are imposed, as illustrated in Figure 4.8(b), the earliest and latest fixed schedules are equal. Let d_0^+ be the latest start time of the trip, a_η^- be the earliest end time of the trip, and $S = a_\eta^- - d_0^+$ denote the time elapsed in between, a trip's duration can then be obtained using S. It is therefore possible to determine D without first fixing the schedule.

When no wait times occur, S may be smaller than D, and even negative. When wait times are imposed, active duration A is shorter than D. Therefore, knowledge of any wait times is not required to calculate a trip's duration via Eq. 4.50.

$$D = \max\{A, S\} = \max\left\{\sum_{k=0}^{\eta-1} e_{k,k+1} + \sum_{k=0}^{\eta} s_k,\ a_\eta^- - d_0^+\right\} \tag{4.50}$$

It is possible to determine the effect of inserting a service i upon a trip's duration as follows. First, as an alternative to the recurrence formulations for earliest arrival time a_k^- (Eq. 4.47) and latest departure time d_k^+ (Eq. 4.48), explicit formulations can be derived intuitively. Consider the example in Figure 4.9, which consists of six transit points, with each point's time window t_i denoted by a gray rectangle and its associated service time by a black square. At every T_i in Figure 4.9 (a), service i is executed as soon as possible, with its initiation dependent on the start of its time window t_i^-. As such, the earliest arrival time at T_k depends exclusively on the start of the time window at T_i. Let $a_k^-(i)$ be

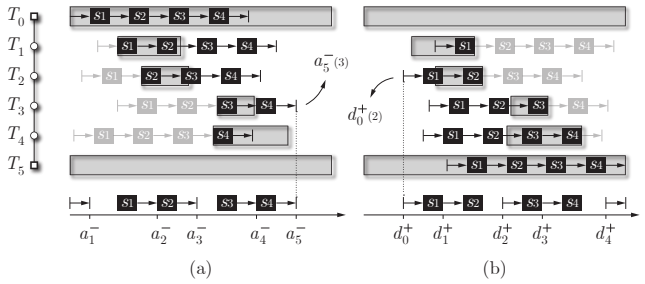

Figure 4.9 **Visual representation of Eq. 4.52 (a) and Eq. 4.54 (b).**

defined as in Eq. 4.51 for $0 \leq i < k$. Then, Eq. 4.52 gives a_k^-.

$$a_k^-(i) = t_i^- + \sum_{j=i}^{k-1} \Big(s_j + e_{j,j+1} \Big) \tag{4.51}$$

$$a_k^- = \min_{0 \leq i < k} a_k^-(i) = \min_{0 \leq i < k} \Big[t_i^- + \sum_{j=i}^{k-1} \Big(s_j + e_{j,j+1} \Big) \Big] \tag{4.52}$$

For the example in Figure 4.9 (a), earliest arrival times $a_5^-(0...4)$ depend on $T_{2...4}$ whereas the time window of T_3 delays the arrival time at T_5 the most and therefore $a_5^- = a_5^-(3)$. The earliest arrival times at all other transit points may be obtained via Eq. 4.52, with the results indicated on the timeline of Figure 4.9 (a).

A similar approach is employed to derive all latest departure times d_k^+. At every T_i in Figure 4.9 (b), service i is executed as late as possible, depending exclusively upon the end of its time window t_i^+ and preceded by all prior travel and service times. Define $d_k^+(i)$ as in Eq. 4.53 for $k < i \leq \eta$, Eq. 4.54 then gives the value of d_k^+. Among all latest departure times $d_0^+(1...5)$ in Figure 4.9 (b), $d_0^+(2)$ is the earliest and thus $d_0^+ = d_0^+(2)$.

$$d_k^+(i) = t_i^+ - \sum_{j=k+1}^{i} \Big(s_j + e_{j-1,j} \Big) \tag{4.53}$$

$$d_k^+ = \min_{k < i \leq \eta} d_k^+(i) = \min_{k < i \leq \eta} \Big[t_i^+ - \sum_{j=k+1}^{i} \Big(s_j + e_{j-1,j} \Big) \Big] \tag{4.54}$$

When a service j is inserted between transit points T_i and T_k, the resulting trip duration D^* is affected as follows. First, A^* is obtained by simply adjusting the

traversed edges and adding the necessary service time (Eq. 4.55). The earliest arrival time a_η^- at the last transit point may consequently be delayed due to the additional travel and service times. Let a_k^{-*} be the new earliest arrival time at T_k and A_k^+ be the sum of all service and travel times from T_k to T_η. The earliest arrival time at T_η is only delayed if $a_k^{-*} + A_k^+ > a_\eta^-$ (Eq. 4.56). Similarly, a_i^{+*} is defined as the new latest departure time at T_i and A_i^- as the sum of all service and travel times from T_0 to T_i. The new latest departure time d_0^+ can be obtained via Eq. 4.57.

$$A^* = A - e_{i,k} + e_{i,j} + e_{j,k} + s_j \tag{4.55}$$

$$a_\eta^{-*} = \max\{a_\eta^-, a_k^{-*} + A_k^+\} \tag{4.56}$$

$$d_0^{+*} = \min\{d_\eta^+, d_i^{+*} - A_i^-\} \tag{4.57}$$

Assuming feasibility time windows a_i and d_i are precalculated together with the sum of preceding A_i^- and succeeding A_i^+ activity at each transit point T_i, it is possible to validate time window feasibility and maximum trip duration in $O(1)$.

4.7.3 Simultaneous servicing (merged service times)

Multiple services conducted by a vehicle at a single transit point may be executed simultaneously provided that each individual service is executed within its time window. Therefore, the total time spent at the transit point may be longer than the longest service time of those services. Execution of services $a(t_a, s_a)$ and $b(t_b, s_b)$ at a single transit point may therefore be merged into the service $m(t_m, s_m)$ which represents the shortest feasible execution of both. Earliest departure d_m^- and latest arrival a_m^+ times are derived simply by Eqs. 4.58 and 4.59.

$$d_m^- = \min\{d_a^-, d_b^-\} \tag{4.58}$$

$$a_m^+ = \max\{a_a^+, a_b^+\} \tag{4.59}$$

Merged service time s_m and its associated time window t_m are derived from d_m^- and a_m^+ by Eqs. 4.60, 4.61 and 4.62.

$$s_m = \max\{s_a, s_b, d_m^- - a_m^+\} \tag{4.60}$$

$$t_m^- = d_m^- - s_m \tag{4.61}$$

$$t_m^+ = a_m^+ + s_m \tag{4.62}$$

Figure 4.10 illustrates four examples of two services $(t, s)_a$ and $(t, s)_b$ merged into single service $(t, s)_m$. This difference between the earliest departure d_m^- and latest arrival time a_m^+ is progressively increased in each example from (a) to (d). The difference in example (a) is negative, thus much shorter than the longest service time, which ultimately results in a service with a lengthy time window. Meanwhile, for (d) the difference is greater than the longest service time and thus equals merged service time s_m with time window $t_m = s_m$, thereby representing a fixed service in time.

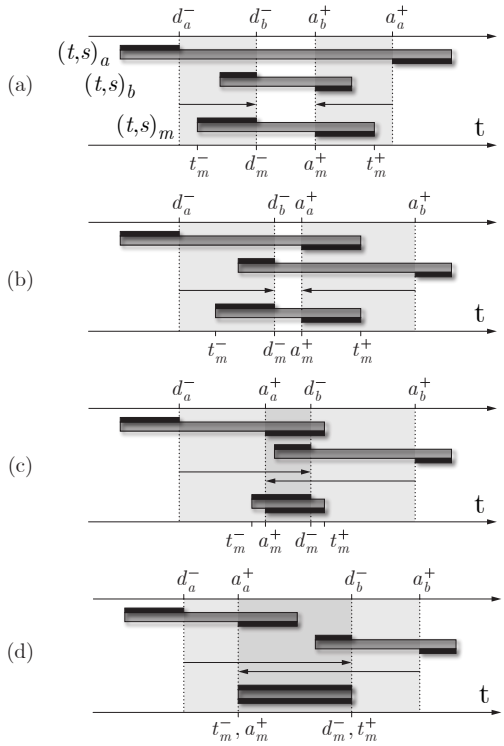

Figure 4.10 **Four examples of merging services a and b into m.**

4.7.4 Maximum ride times

In order to guarantee maximum ride time feasibility, we apply the following preprocessing step. Let t_r^{p-} and s_r^p be the beginning of pickup time window and pickup service time for request r and let t_r^{d+} and s_r^d be the end of delivery time window and delivery service time. Feasibility for the request's maximum ride

time L_r is ensured by restricting the earliest pickup time $\hat{t}_r^{p-}$, as in Eq. 4.63.

$$\hat{t}_r^{p-} = \max\left\{t_r^{p-}, t_r^{d+} - \left(s_r^p + L_r + s_r^d\right)\right\} \tag{4.63}$$

4.8 PTP instances and computational results

The proposed instances have been made publicly available at the PTP benchmark site[1] in order to stimulate further research regarding the problem. Real world PTP data and the accompanying schedules for a single day of prisoner transports were obtained, which enabled a fair comparison between the heuristically generated and the manually constructed solutions. The location coordinates of PTP instances that were generated using real world data have been altered in the interest of preserving confidentiality. As such, it is impossible to retrieve the original coordinates from any of the instances that have been made publicly available. This ensures that private user information cannot be unambiguously extrapolated from these instances.

Table 4.3 summarizes instance properties as a function of the number of prisoner transportation requests. The smallest number of requests r begins at 50, immediately challenging exact approaches. Instances of sizes 100, 200, 400 and, in order to reflect the real world problem size, 700 were also generated. The number of prisoners per instance p varies from 38 to 476, with the number of unique locations associated with their trips ranging from 28 to 169, of which between 1 and 13 are depot locations. Cartesian coordinates for each location are obtained by way of classical multidimensional scaling of real travel times. Therefore, the cartesian distance between the obtained locations' coordinates provide realistic travel times. A single depot is assumed for the smallest instances, all the way up to a total of 13 for the largest instances. This number of depots is based on the ratio of trips per depot present in the real world data. The maximum trip duration is fixed to 9 hours (540 minutes) for all instances. For each transportation request, separate time windows and service times for pickup and delivery services are defined. Time windows occur in a timespan of 24 hours, and although service times are typically 15 minutes, they may be as long as 225 minutes.

Ten different vehicle types are considered for each of these instances (Table 4.4), resulting in a total of 50 PTP instances. Each vehicle type's capacity Q totals four seats for $v1$ - $v5$ and seven seats for $v6$ - $v10$, which are grouped in various compartments. The vehicle types contain compartments of various sizes, a few of which are worth detailing. Vehicle types $v1$ and $v6$ have compartments

[1] http://www.vrp-rep.org/datasets/item/2020-0001.html

Table 4.3 Common properties of the PTP instances.

Name	r	p	l	u	D_{max}
(x)-r050-(v)	50	38	28	1	540
(x)-r100-(v)	100	71	45	2	540
(x)-r200-(v)	200	140	70	4	540
(x)-r400-(v)	400	264	101	8	540
(x)-r700-(v)	700	476	169	13	540

of one single seat. Such vehicles always avoid inter-passenger conflicts at compartment level, however they are less common in practice due to their high investment costs. Vehicle types $v2$ and $v7$ represent vehicle configurations with two (almost) equal sized compartments. Vehicles of $v5$ and $v10$ type contain one single compartment and are, therefore, incapable of combining conflicting prisoners.

Table 4.4 PTP vehicle types.

Name	Q	C	Name	Q	C
v1	4	$\{1,1,1,1\}$	v6	7	$\{1,1,1,1,1,1,1\}$
v2	4	$\{2,2\}$	v7	7	$\{3,4\}$
v3	4	$\{1,1,2\}$	v8	7	$\{2,2,3\}$
v4	4	$\{1,3\}$	v9	7	$\{1,6\}$
v5	4	$\{4\}$	v10	7	$\{7\}$

In order to assess the impact of inter-passenger conflicts and maximum ride time restrictions, this set of instances - referred to as Set A - is duplicated to solve the PTP by relaxing these constraints one at a time. This results in three additional instance sets: Set B with $H \neq \emptyset$ and $L_r = D_{max}$ $\forall r \in R$, Set C with $H = \emptyset$ and $L_r = 150$ $\forall r \in R$, and Set D with $H = \emptyset$ and $L_r = D_{max}$ $\forall r \in R$.

Relaxation of these conflicts essentially constitutes instances for the multi depot DARP with simultaneous servicing. Additionally, when relaxing both conflicts and maximum ride durations, the problem becomes a multi-depot pickup-and-delivery problem with time windows and simultaneous servicing. Notice that only one vehicle type is employed for the problem sets where conflicts are relaxed, given that the vehicle's total capacity then becomes the only relevant constraint.

In order to provide initial solutions and verify *error-free* instance files for these four benchmark sets, ten runs are conducted for which the best ($Best$) and average ($Avg.$) results are accompanied by average run times T_a in minutes.

These results are detailed in Tables 4.5 and 4.6. Considering only the real world scale of the problem instances, which contain 700 requests, we observe the following. The presence of inter-passenger conflicts implies an overall cost increase of up to 4.4% when employing small vehicles ($v1$ - $v5$) and up to 5.3% when employing large vehicles ($v6$ - $v10$). Meanwhile, the effect of inter-passenger conflicts at compartment level can be determined by comparing results for $v1$, which is always capable of combining conflicting prisoners, and $v5$, which is always incapable of combing them. Employing $v1$ rather than $v5$ implies a cost reduction of 1.8%. Similarly, employing $v6$ rather than $v10$ implies a cost reduction of 1.6%. Increasing the vehicle's capacity from 4 ($v1$ - $v5$) to 7 seats ($v6$ - $v10$) may reduce the total cost by up to 2.1%. Finally, we observe that the presence of the maximum ride time constraint may triple the cost. These results are not without practical significance given that the maximum ride time constraints were not initially considered as critical constraints by the prisoner transportation company involved in this research. However, after modeling the problem and evaluating the solution approach through these experiments, it became apparent that maximum ride times are indeed crucial for prisoner transportation. On the other hand, it is very likely that this observation is not only related to the constraint, but also to how it is accommodated by preprocessing the time windows. Therefore, the computational results of maximum ride time must not be regarded as an independent observation.

Table 4.5 PTP instance and results for vehicle types v1 to v5.

Set A(v1-v5) ($H \neq \emptyset, L_r = 150$)				Set B(v1-v5) ($H \neq \emptyset, L_r = D_{max}$)			
Instance	Best	Avg.	T_a	Instance	Best	Avg.	T_a
a-r050-v1	(18) 7385	7385.7	1.1	b-r050-v1	(6) 2208	2216.0	3.0
a-r050-v2	(18) 7382	7384.4	1.0	b-r050-v2	(6) 2211	2226.4	2.6
a-r050-v3	(18) 7382	7387.2	1.1	b-r050-v3	(6) 2211	2217.6	2.9
a-r050-v4	(18) 7413	7413.2	1.0	b-r050-v4	(6) 2220	2229.9	2.8
a-r050-v5	(18) 7434	7441.1	1.0	b-r050-v5	(6) 2392	2403.3	2.3
a-r100-v1	(37) 16514	16524.5	1.9	b-r100-v1	(13) 5317	5348.9	4.4
a-r100-v2	(38) 16511	16524.9	1.7	b-r100-v2	(13) 5275	5322.1	3.9
a-r100-v3	(37) 16509	16530.2	1.8	b-r100-v3	(13) 5294	5338.6	4.2
a-r100-v4	(38) 16546	16559.9	1.8	b-r100-v4	(13) 5315	5364.0	4.0
a-r100-v5	(38) 16604	16615.2	1.7	b-r100-v5	(14) 5458	5489.6	3.3
a-r200-v1	(67) 30308	30376.3	3.6	b-r200-v1	(21) 9778	9922.6	9.5
a-r200-v2	(67) 30337	30370.4	3.3	b-r200-v2	(22) 9878	9990.4	8.0
a-r200-v3	(67) 30311	30359.8	3.5	b-r200-v3	(22) 9814	9967.3	9.0
a-r200-v4	(67) 30371	30427.1	3.4	b-r200-v4	(22) 9872	9990.6	8.3
a-r200-v5	(68) 30671	30751.5	3.2	b-r200-v5	(22) 9896	10075.8	6.9
a-r400-v1	(128) 59030	59164.4	7.3	b-r400-v1	(41) 19562	19809.8	18.3
a-r400-v2	(128) 59134	59212.2	6.8	b-r400-v2	(42) 19715	19924.5	15.7
a-r400-v3	(127) 59030	59152.6	7.2	b-r400-v3	(42) 19765	19971.0	17.3
a-r400-v4	(127) 59123	59207.2	7.0	b-r400-v4	(41) 19576	19896.5	16.3
a-r400-v5	(131) 60128	60307.6	6.5	b-r400-v5	(42) 20014	20305.2	13.4
a-r700-v1	(274) 120836	121019.7	12.0	b-r700-v1	(84) 41270	41570.8	27.7
a-r700-v2	(272) 121292	121503.2	11.5	b-r700-v2	(84) 41511	41930.5	24.5
a-r700-v3	(272) 120820	120990.8	12.0	b-r700-v3	(85) 41116	41466.1	26.7
a-r700-v4	(275) 120865	121143.1	11.6	b-r700-v4	(83) 41218	41593.0	25.5
a-r700-v5	(276) 123044	123185.8	10.9	b-r700-v5	(87) 42527	42804.5	20.8

Set C(v1) ($H = \emptyset, L_r = 150$)				Set D(v1) ($H = \emptyset, L_r = D_{max}$)			
Instance	Best	Avg.	T_a	Instance	Best	Avg.	T_a
c-r050-v1	(18) 7343	7352.5	1.1	d-r050-v1	(6) 2118	2133.3	3.0
c-r100-v1	(37) 16346	16374.9	2.0	d-r100-v1	(12) 4991	5070.6	6.8
c-r200-v1	(64) 29510	29572.6	3.7	d-r200-v1	(21) 9316	9453.4	14.5
c-r400-v1	(123) 57411	57583.1	7.4	d-r400-v1	(40) 18821	19094.2	24.9
c-r700-v1	(261) 117399	117947.9	12.1	d-r700-v1	(78) 38125	38604.9	36.0

Table 4.6 PTP instance and results for vehicle types v6 to v10.

Set A(v6-v10) ($H \neq \emptyset, L_r = 150$)				Set B(v6-v10) ($H \neq \emptyset, L_r = D_{max}$)			
Instance	Best	Avg.	T_a	Instance	Best	Avg.	T_a
a-r050-v6	(18) 7343	7343.0	1.1	b-r050-v6	(6) 2102	2126.1	3.0
a-r050-v7	(18) 7343	7343.0	1.0	b-r050-v7	(6) 2113	2127.7	3.0
a-r050-v8	(18) 7343	7343.0	1.1	b-r050-v8	(6) 2102	2124.9	3.0
a-r050-v9	(18) 7371	7373.4	1.0	b-r050-v9	(6) 2131	2141.4	3.0
a-r050-v10	(18) 7395	7399.5	1.0	b-r050-v10	(7) 2318	2327.7	2.5
a-r100-v6	(36) 16112	16135.3	2.0	b-r100-v6	(13) 4970	5014.5	5.8
a-r100-v7	(37) 16175	16195.3	1.7	b-r100-v7	(14) 4982	5022.9	4.4
a-r100-v8	(37) 16185	16201.4	1.8	b-r100-v8	(13) 4987	5015.0	4.8
a-r100-v9	(36) 16113	16124.6	1.8	b-r100-v9	(14) 4945	5013.9	4.6
a-r100-v10	(37) 16206	16224.5	1.7	b-r100-v10	(13) 5107	5137.6	3.8
a-r200-v6	(65) 29377	29431.2	3.8	b-r200-v6	(20) 8970	9067.7	12.9
a-r200-v7	(65) 29375	29431.7	3.3	b-r200-v7	(20) 9010	9096.7	9.2
a-r200-v8	(65) 29404	29448.8	3.5	b-r200-v8	(20) 8939	9067.4	10.6
a-r200-v9	(66) 29421	29452.8	3.4	b-r200-v9	(21) 9043	9131.8	9.6
a-r200-v10	(66) 29694	29727.6	3.2	b-r200-v10	(20) 9160	9298.7	7.9
a-r400-v6	(124) 57248	57330.0	7.6	b-r400-v6	(37) 17625	17802.8	25.3
a-r400-v7	(125) 57236	57380.2	6.8	b-r400-v7	(39) 17812	17927.8	18.1
a-r400-v8	(123) 57272	57404.6	7.0	b-r400-v8	(38) 17692	17872.6	20.0
a-r400-v9	(124) 57274	57406.5	7.0	b-r400-v9	(40) 17837	17961.8	18.8
a-r400-v10	(128) 58636	58734.6	6.5	b-r400-v10	(40) 18204	18434.4	14.9
a-r700-v6	(264) 118372	118748.7	12.6	b-r700-v6	(78) 37566	37937.0	35.3
a-r700-v7	(267) 118722	119041.6	11.3	b-r700-v7	(77) 37756	38060.7	26.2
a-r700-v8	(263) 118282	118720.6	11.8	b-r700-v8	(79) 37734	38101.8	29.1
a-r700-v9	(263) 118301	118642.3	11.6	b-r700-v9	(77) 37947	38218.2	27.9
a-r700-v10	(273) 120523	120690.7	10.8	b-r700-v10	(81) 39263	39652.5	22.0

Set C(v6) ($H = \emptyset, L_r = 150$)				Set D(v6) ($H = \emptyset, L_r = D_{max}$)			
Instance	Best	Avg.	T_a	Instance	Best	Avg.	T_a
c-r050-v6	(17) 7303	7303.0	1.2	d-r050-v6	(5) 1964	1973.5	3.0
c-r100-v6	(36) 15948	15969.6	2.1	d-r100-v6	(12) 4620	4660.9	10.2
c-r200-v6	(61) 28427	28502.2	3.9	d-r200-v6	(19) 8305	8460.8	24.2
c-r400-v6	(116) 55359	55606.1	7.7	d-r400-v6	(35) 16379	16552.8	41.0
c-r700-v6	(250) 114166	114657.3	12.5	d-r700-v6	(70) 33554	33767.2	51.2

4.9 Comparison of the formulations and iterative algorithms for PTP

4.9.1 Formulation with single-index time variables (FPTP-2)

In this formulation, $\boldsymbol{x}$, $\boldsymbol{y}$ and $\boldsymbol{z}$ variables are defined as before. Variable τ_i denotes the departure time from node $i \in I$ and δ_i is the merged service time of node $i \in I$. Moreover, ϕ^v is the departure time from the starting depot and Φ^v is the arrival time to the ending depot of vehicle $v \in V$. The following is a mixed integer programming formulation for the PTP with single-index time variables.

$$\min \sum_{v \in V} (\Phi^v - \phi^v) \tag{4.64}$$

s.t. (4.2) – (4.12), (4.22) – (4.27)

$$\tau_i + e_{ij} + \delta_j \leq \tau_j + t^+_{max}(1 - \sum_{v \in V} z^v_{ij}) \quad \forall (i,j) \in A^3 : l(i) \neq l(j) \tag{4.65}$$

$$\tau_i \leq \tau_j + t^+_i(1 - \sum_{v \in V} z^v_{ij}) \quad \forall (i,j) \in A^3 : l(i) = l(j) \tag{4.66}$$

$$\delta_i \geq \sum_{v \in V} \sum_{j \in I} s_j z^v_{ij} \quad \forall i \in I \tag{4.67}$$

$$\delta_j \geq \sum_{v \in V} \sum_{i \in I} s_i z^v_{ij} \quad \forall j \in I \tag{4.68}$$

$$\tau_{n+i} - \tau_i - s_{i+n} \leq L_{r(i)} \quad \forall i \in I^P \tag{4.69}$$

$$\Phi^v - \phi^v \leq D_{max} \quad \forall v \in V \tag{4.70}$$

$$\Phi^v - \phi^v \geq \sum_{i \in I^P} (e_{u_v i} + s_i) z^v_{u_v i} + \sum_{i \in I^D} (e_{i u^e_v} + s_i) z^v_{i u^e_v} + \sum_{(i,j) \in A^3} e_{ij} z^v_{ij} \quad \forall v \in V \tag{4.71}$$

$$\Phi^v \geq \tau_i + e_{i u^e_v} + t_{max}(z^v_{i u^e_v} - 1) \quad \forall v \in V, i \in I \tag{4.72}$$

$$\phi^v \leq \tau_i - e_{u_v i} - \delta_i + t^+_{max}(1 - z^v_{u_v i}) \quad \forall v \in V, i \in I \tag{4.73}$$

$$t_i^- + s_i \leq \tau_i \leq t_i^+ \qquad \forall i \in I \tag{4.74}$$

$$s_i \leq \delta_i \leq L_{r(i)} \qquad \forall i \in I \tag{4.75}$$

The objective function (4.64) of FPTP-2 is expressed as a linear function of single-index Φ^v and ϕ^v variables for $v \in V$. Constraints (4.65), (4.73) and (4.72) replace (4.13) of the original model. The remaining constraints are analogous to those omitted from FPTP.

4.9.2 Formulation without δ variables (FPTP-3)

In addition to FPTP and FPTP-2, we provide a third formulation which does not contain $\boldsymbol{\delta}$ variables. Model FPTP-3 then becomes:

min (4.1)

s.t. (4.2) – (4.12), (4.22) – (4.27), (4.14), (4.17) – (4.20)

$$\tau_i^v + e_{ij} + s_j \leq \tau_j^v + t_{max}(1 - z_{ij}^v) \qquad \forall v \in V, (i,j) \in A : l(i) \neq l(j) \tag{4.76}$$

As in the case of FPTP, we utilize all the valid inequalities introduced in Section 4.3 for PTP-2 and PTP-3 whenever applicable. Additionally for PTP-2, we replace (4.72) and (4.73) with stronger constraints (4.77) and (4.78).

$$\Phi^v \geq \tau_i + e_{iu_v^e} + t_{max}(\sum_{j \in I} z_{ij}^v - 1) \qquad \forall v \in V, i \in I \tag{4.77}$$

$$\phi^v \leq \tau_i - e_{u_v i} - \delta_i + t_{max}^+(1 - \sum_{j \in I} z_{ji}^v) \qquad \forall v \in V, i \in I \tag{4.78}$$

Table 4.7 Comparison of PTP methods.

					FPTP			FPTP-2			FPTP-3			IT-FPTP		IT-PTP-2		IT-PTP-3	
id	Comp.	n	Best	k	Obj	$t(s)$	$g\%$	Obj	$t(s)$	$g\%$	Obj	$t(s)$	$g\%$	Obj	$t(s)$	Obj	$t(s)$	Obj	$t(s)$
4_0_4	1-1-1-1	4	**54**	1	**54**	9.73	0.00	**54**	48.04	0.00	**54**	5.2	0.00	**54**	3.21	**54**	9.44	**54**	4.30
4_1_4	1-1-1-1		**271**	2	**271**	0.22	0.00	**271**	9.62	0.00	**271**	0.38	0.00	**271**	0.65	**271**	7.55	**271**	0.92
4_1_2	2-2		**271**	2	**271**	0.24	0.00	**271**	3.50	0.00	**271**	0.18	0.00	**271**	17.12	**271**	6.22	**271**	0.64
4_1_1	4		**271**	2	**271**	0.19	0.00	**271**	3.72	0.00	**271**	0.22	0.00	**271**	15.43	**271**	5.78	**271**	0.36
4_2_4	1-1-1-1		**181**	2	**181**	0.62	0.00	**181**	10.47	0.00	**181**	1.43	0.00	**181**	2.04	**181**	8.08	**181**	2.57
4_2_2	2-2		**181**	2	**181**	0.98	0.00	**181**	5.53	0.00	**181**	1.16	0.00	**181**	2.10	**181**	11.91	**181**	1.52
4_2_1	4		**181**	2	**181**	1.25	0.00	**181**	4.76	0.00	**181**	0.37	0.00	**181**	12.54	**181**	6.86	**181**	1.49
6_0_4	1-1-1-1	6	**90**	2	90	TL	93.89	90	TL	93.33	**90**	TL	92.33	**90**	405.92	**90**	399.15	**90**	506.80
6_1_4	1-1-1-1		**399**	1	**399**	93.68	0.00	**399**	72.13	0.00	**399**	35.17	0.00	**399**	10.29	**399**	8.74	**399**	6.69
6_1_2	2-2		**399**	1	**399**	5.96	0.00	**399**	12.93	0.00	**399**	7.47	0.00	**399**	11.78	**399**	7.77	**399**	3.85
6_1_1	4		**403**	1	**403**	2.35	0.00	**403**	9.03	0.00	**403**	1.83	0.00	**403**	7.88	**403**	21.27	**403**	0.98
6_2_4	1-1-1-1		**282**	3	**282**	353.12	0.00	**282**	331.20	0.00	**282**	89.94	0.00	**282**	34.49	**282**	38.02	**282**	75.83
6_2_2	2-2		**282**	3	**282**	130.94	0.00	**282**	192.35	0.00	**282**	110.53	0.00	**282**	13.61	**282**	15.19	**282**	16.63
6_2_1	4		**282**	3	**282**	6.27	0.00	**282**	36.82	0.00	**282**	11.82	0.00	**282**	114.64	**282**	74.58	**282**	6.44
8_0_4	1-1-1-1	8	106	2	126	TL	97.88	294	TL	99.46	128	TL	100.00	106	TL	106	TL	106	TL
8_1_4	1-1-1-1		**516**	2	519	TL	60.67	516	TL	66.38	516	TL	56.77	516	TL	**516**	4281.62	**516**	3165.77
8_1_2	2-2		**516**	2	516	TL	37.44	516	TL	38.87	516	TL	39.15	**516**	1659.33	**516**	849.50	**516**	1114.13
8_1_1	4		**521**	2	**521**	169.04	0.00	**521**	215.38	0.00	**521**	223.97	0.00	**521**	23.22	**521**	56.44	**521**	24.25
8_2_4	1-1-1-1		**356**	3	393	TL	59.45	356	TL	49.89	356	TL	54.19	356	TL	**356**	5061.98	356	TL
8_2_2	2-2		**356**	3	356	TL	16.57	356	TL	18.03	356	TL	18.57	**356**	2175.07	**356**	911.33	**356**	2160.09
8_2_1	4		**356**	3	**356**	2757.98	0.00	356	TL	16.29	356	2543.25	0.00	**356**	823.36	**356**	1037.22	**356**	486.27
10_0_4	1-1-1-1	10	106	2	273	TL	100.00	423	TL	100.00	366	TL	100.00	106	TL	106	TL	131	TL
10_1_4	1-1-1-1		599	2	607	TL	75.86	601	TL	75.39	607	TL	73.63	599	TL	599	TL	599	TL
10_1_2	2-2		599	2	682	TL	84.90	620	TL	70.91	656	TL	84.36	599	TL	599	TL	599	TL
10_1_1	4		**601**	2	603	TL	31.82	603	TL	34.30	603	TL	32.43	**601**	488.14	**601**	410.27	**601**	334.30
10_2_4	1-1-1-1		451	2	706	TL	76.00	841	TL	79.84	770	TL	78.01	451	TL	453	TL	453	TL
10_2_2	2-2		450	2	531	TL	67.93	579	TL	69.01	519	TL	66.90	450	TL	450	TL	450	TL
10_2_1	4		450	2	467	TL	63.09	453	TL	57.24	451	TL	60.53	450	TL	450	TL	450	TL

4.10 Conclusion

This chapter introduced the Prisoner Transportation Problem (PTP), which constitutes a generalization of common combinatorial optimization problems: vehicle routing with time windows, dial-a-ride and multi-compartment vehicle routing. The PTP is interesting from a practical viewpoint because its daily operations inevitably represent a huge cost for society. Its modeling difficulty is primarily related to the presence of inter-passenger conflicts and simultaneous servicing, two intricate constraints for which models were previously absent. The PTP therefore posed a significant academic challenge.

Together with several mathematical formulations this chapter contributed a set of instances and provides initial benchmarks obtained by applying a local search based approach in order to stimulate further research on the PTP. While constructing and optimizing the PTP's trips, feasible prisoner-to-compartment allocations are essential. An efficient and effective heuristic has been introduced for solving this subproblem.

The mathematical formulations introduced in this chapter set in place the strong foundations necessary for subsequent research on the PTP. Possible future research directions include establishing further theoretical insights as well as practical algorithms for improving solutions. Several variants being already discussed in this chapter, it is also worthwhile to study the impact of additional real world constraints which have, for the sake of clarity, been omitted in this study.

Chapter 5

Slack Induction by String Removals (SISRs)

This chapter provides a detailed description of the SISRs method for solving VRPs. The concepts of capacity slack, spatial slack and three premises are introduced in Section 5.2 which lay out the rationale for the proposed method. A detailed description of SISRs' $\mathcal{R}^-$ and $\mathcal{R}^+$ methods may be found in Section 5.3 and Section 5.4 respectively.

5.1 Spatial Slack

When customers are removed from a tour's sequence, several types of slack may be introduced. First, the most apparent type of slack may be *capacity slack* referring to the vehicle's capacity which is set free. Second, *spatial slack* may be identified if one imagines that the vehicle's travel distance is limited. Given this limited distance, the vehicle's reachable area may be defined. This area appears as an elliptical region around each edge, where each edge's end points are the focus points for its corresponding ellipse. This area increases when the locations of the removed customers are no longer visited and we refer to this increase as spatial slack. While the reduced travel distance contributes the most for gaining spatial slack, it is not a necessary condition. For example, removing the intermediate of three collinear locations does not reduce the travel distance while the reachable area - spatial slack - does increase given the larger focal distance. Although the vehicle's travel distance is not limited within the CVRP definition, one may express spatial slack relative to the vehicle's original tour as the additional reachable area delimited by its original travel distance.

5.2 Rationale

An effective $\mathcal{R}^-$ method should introduce both types of slack into the solution such that the possibilities for improving the solution are greatly enhanced. The following premises serve as the basis for such $\mathcal{R}^-$ method. Each premise addresses the potential failures of alternative $\mathcal{R}^-$ methods, within the context of slack induction.

Premise 1 ***Remove a 'sufficient' number of customers***
Removing a small number of customers (such as in Fig. 5.2(a_1)), may fail to introduce sufficient capacity slack in the vehicles for serving customers of higher demand. While it is visually clear (a_2) by which vehicles the removed customers should be served, it may be impossible if the vehicles gained insufficient capacity

slack.

Premise 2 ***Remove 'adjacent' customers***
Removing customers who are not geographically close (not adjacent) to each other (such as randomly-selected customers in Fig. 5.2(b_1)), generally introduces capacity and spatial slack scattered across many tours (b_2). Such slack is unlikely to be beneficial when recreating the solution. The recreated solution may be comparable to the result of multiple ruin and recreate iterations wherein a small number of customers is removed each time.

Premise 3 ***Remove 'strings' of customers***
Removing customers of which their selection is only based on adjacency (as in the case of radial removal, Fig. 5.2(c_1)) may often fail to capture complete detours which remain present in the ruined state (c_2). This may be remedied by increasing the radius, however, much more customers than strictly necessary may be removed as well. Therefore, removing strings may be much more effective.

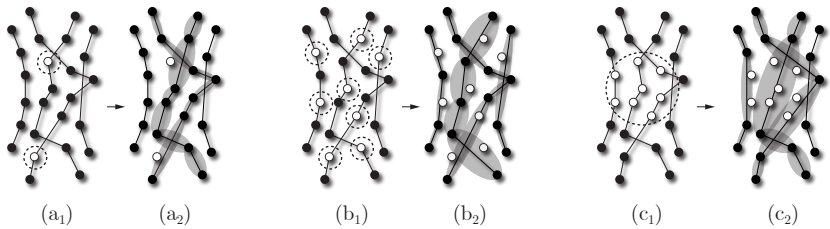

Figure 5.1 Examples of Ruin Methods which Introduce Insufficient Slack
Note. Removal of too few customers (a), randomly-selected customers (b) and radially-selected customers (c). Figures indexed by 1 illustrate the partial solution wherein the customers selected for removal are colored white and outlined by dashed lines. Figures indexed by 2 illustrate the ruined state wherein spatial slack relative to the original state is shaded gray.

If one would remove multiple strings which are geographically close to each other - removal of adjacent strings - a very effective yet simple $\mathcal{R}^-$ method may be obtained. An example of such adjacent string removals is depicted in Fig. 5.2. Only two adjacent strings are selected for removal in (a) and the ruined state of the solution is illustrated in (b). Notice that (i) spatial slack is greatly introduced in only two tours and (ii) the spatial slacks of both overlap significantly. Ideally, the ruined state obtained by the $\mathcal{R}^-$ method (b), is recreated by $\mathcal{R}^+$ into the solution illustrated in (c).

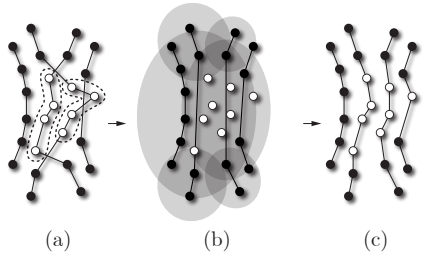

Figure 5.2 An Example of Adjacent String Removals
Note. Selection of two adjacent strings (a), ruined state with significant spatial slack induced (b) and an improved recreated state (c).

5.3 Adjacent String Removal ($\mathcal{R}^-$)

The proposed $\mathcal{R}^-$ method is based on Premises 1 to 3 as follows. The number of removed customers is managed by means of the average number of removed customers $\bar{c}$. In addition, the $\mathcal{R}^-$ method may be set for removing many strings of small cardinality or few strings of high cardinality by means of the maximum cardinality of the removed strings L^{max}. By fixing the average computational effort to an average number of removed customers $\bar{c}$, it is possible to investigate the influence of removing many strings of small cardinality versus few strings of high cardinality. Parameter $\bar{c}$ may be expressed as a function of the average number of strings $\bar{k}$ and the average string length $\bar{l}$, as demonstrated in Eq. 5.1.

$$\bar{c} = \bar{k}.\bar{l} = \frac{K^{min} + K^{max}}{2}.\frac{L^{min} + L^{max}}{2} \qquad K, L \in \mathbb{N}^+ \qquad (5.1)$$

Solutions may consist of many short tours serving few customers or only a few long tours serving many customers, depending upon the specific problem instance. Therefore, it may not be possible to remove K^{max} strings if the solution contains fewer tours, or to remove strings of cardinality L^{max} in cases where all tours are short. Let s be the solution being ruined. A valid number of removed strings k_s and cardinality l_t for every string (each of which are removed from a different tour t) is selected as follows. The maximum string cardinality l_s^{max} is initially set to the minimum of L^{max} and the solution's average tour cardinality $\overline{|t \in T|}$ (Eq. 5.2) before the maximum number of strings k_s^{max} is derived (Eq. 5.3).

$$l_s^{max} = \min\{ L^{max}, \overline{|t \in T|}\} \qquad l_s^{max} \in \mathbb{R}^+ \qquad (5.2)$$

$$k_s^{max} = \frac{4\,\bar{c}}{1 + l_s^{max}} - 1 \qquad k_s^{max} \in \mathbb{R}^+ \qquad (5.3)$$

$$k_s = \lfloor\, U(1,\, k_s^{max} + 1\,) \,\rfloor \qquad\qquad k_s \in \mathbb{N}^+ \tag{5.4}$$

The number of strings k_s to be removed is obtained by rounding down a randomly selected real value from the continuous uniform distribution $U(1, k_s^{max} + 1)$ by way of Eq. 5.4. The maximum cardinality of each string is limited by the initial limit l_s^{max} and the tour's cardinality $|t|$ from which the string is to be removed (Eq. 5.5). Finally, the cardinality for the string to be removed from tour t is obtained by rounding down a real value selected from $U(L^{min}, l_t^{max} + 1)$ Eq. 5.6.

$$l_t^{max} = \min\{\, |t|,\, l_s^{max}\} \qquad\qquad l_t^{max} \in \mathbb{R}^+ \tag{5.5}$$

$$l_t = \lfloor\, U(1,\, l_t^{max} + 1\,) \,\rfloor \qquad\qquad l_t \in \mathbb{N}^+ \tag{5.6}$$

Before the procedural steps of the ruin phase $\mathcal{R}^-$ are detailed by way of Algorithm 3, two assumptions are made: (i) For each customer c an adjacency list $adj(c)$ is assumed to be available for identifying adjacent strings. The list contains all customers ordered by increasing distance from c, which includes c as its first element. (ii) Each string being removed originates from a different tour such that spatial slack as well as capacity slack is spread across a number of tours, controlled by k_s. Therefore a tour may be ruined at most once and removing k_s strings implies ruining k_s tours.

Algorithm 3 SISRs' Ruin Method

Input: Solution $s = \{T, A\}$ where T is the set of tours and A the set of absent customers

1: **procedure** RUIN(s)
2: $\quad l_s^{max},\ k_s^{max},\ k_s \leftarrow calculate(Eq.\ 5.2, 5.3 \text{ and } 5.4)$
3: $\quad c_s^{seed} \leftarrow randomCustomer(s)$
4: $\quad R \leftarrow \emptyset$
5: $\quad$ **for** $c \in adj(c_s^{seed})$ **and** $|R| < k_s$ **do**
6: $\quad\quad$ **if** $c \notin A$ **and** $t \notin R$ **then**
7: $\quad\quad\quad c_t^* \leftarrow c$
8: $\quad\quad\quad l_t^{max},\ l_t \leftarrow calculate(Eq.\ 5.5 \text{ and } 5.6)$
9: $\quad\quad\quad A \leftarrow A \cup removeSelected(t,\ l_t,\ c_t^*)$
10: $\quad\quad\quad R \leftarrow R \cup \{t\}$
11: **end procedure**

The procedural steps of the ruin phase $\mathcal{R}^-$ are given by Algorithm 3. First, values for l_s^{max}, k_s^{max} and k are derived by Equations 1 to 3 (line 2). A randomly selected customer serves as the seed customer c_s^{seed} from which string removals

are initiated (line 3). The auxiliary set R, for storing ruined tours, is initially empty (line 4). The adjacency list from c_s^{seed} is iterated over from the first customer to the last until set R contains k_s ruined tours indicating that k_s strings have been removed (lines 5 to 10). In each iteration, if the customer c has not been removed ($c \notin A$) and its serving tour t has not been ruined ($t \notin R$) (line 6), a string is removed from this tour (lines 7 to 10). Such a customer is special in the sense that it is the closest customer to c_s^{seed} out of all customers served by t, denoted as c_t^* (line 7). The maximum cardinality l_t^{max} and final cardinality l_t of the string to be removed from tour t is obtained by way of Equations 4 and 5 (line 8). The string is removed via the 'string' or 'split string' procedure based on c_t^* and l_t. In either case, l_t customers are selected and removed from t before adding them to set A (line 9) and the ruined tour is added to set R (line 10).

The computational complexity of Algorithm 3 depends on the considered VRP variant. Regarding CVRP, a string can be removed from a tour (line 9) in constant time, while identifying the string's last customer requires l operations. Meanwhile, the best case for identifying k adjacent strings (line 5) may require only k iterations when the first k customers in the adjacency list are served by different tours. The worst-case scenario requires n operations when the solution contains exactly k tours where one tour serves a single customer which appears as the last element in the adjacency list. Therefore the best case for Algorithm 3 requires $O(kl)$ operations while the worst-case requires $O(n)$ operations. Regarding variants with time windows, after removing a string in tour t, $O(|t|)$ additional operations are required for updating the temporal variables associated with the remaining customers in t.

The 'string' procedure removes a randomly selected string of cardinality l_t in tour t which includes customer c_t^*. By removing a string which includes c_t^*, other customers inside this string are also likely to be located close to c_{seed}. Therefore, every call to this procedure will result in a string removal of customers close to the seed customer or simply, strings adjacent to each other. An example where $l_t = 3$ is illustrated in Fig. 5.3. One of the three possible strings is randomly selected for removal.

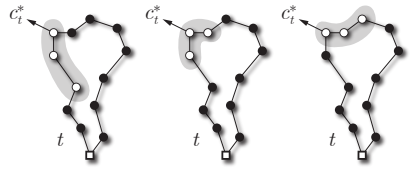

Figure 5.3 Example of 'String' Removal
Note. There are three possible string selections which include c_t^* from tour t when $l_t = 3$.

The 'split string' procedure begins much like 'string', by randomly selecting a string of cardinality $l + m$ which includes customer c_t^*, see Fig. 5.4(a). However, the ruin phase bypasses and preserves a random substring of m intervening customers as shown in Fig. 5.4(b). The number of preserved customers m is determined as follows. Initially $m = 1$ and its current value is maintained if either a random number $U(0, 1)$ is larger than β, which is referred to as the split depth, or once the maximum value for m is reached ($m = m^{max} = |t| - l$). If neither of these conditions is satisfied, m is incremented ($m = m + 1$) and the process repeats.

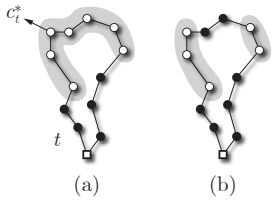

Figure 5.4 Example of 'Split String' Removal
Note. Random selection of a string when $l = 5$ and $m = 2$ (a) and preserving a substring of $m = 2$ customers (b).

The probability of executing 'split string' is referred to as the split rate α. An example is illustrated in Fig. 5.5(a) for 'string' and (b_1), (b_2) for 'split string' with small and large m-values. Customers being removed are colored white. The 'string' procedure removes a string, introducing spatial slack, see Fig. 5.5(a). In the 'split string' removal procedure, a small value for m is obtained with a very low probability. When this occurs, customers close to c_t^* are removed while remaining spatially bonded to the same regions as the preserved customers (b_1). Otherwise $m = m^{max}$ is obtained with much greater probability, indicating that the customers removed are close to the depot (b_2).

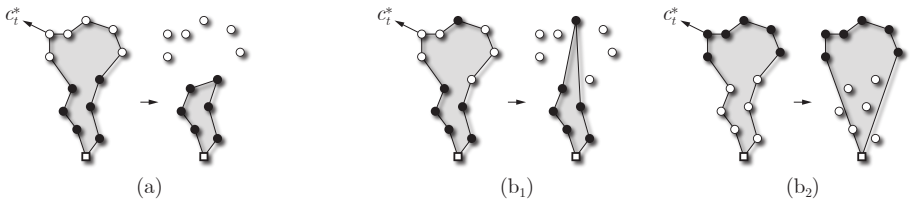

Figure 5.5 Examples of 'String' and 'Split String' Removals
Note. 'String' removal (a) and 'split string' removal when m equals the lowest (b_1) or highest (b_2) possible value.

Fig. 5.6 presents an example of the ruin phase where two strings ($k_s = 2$) are removed from solution s. First, the seed customer c_s^{seed} is randomly selected

(a). Following this, the list $adj(c_s^{seed})$, is iterated over. Since c_s^{seed} is always the first element in the list and no tours are ruined, $c_{t1}^* = c_s^{seed}$. A string of length $l_{t1} = 4$ is removed by the 'string' procedure which includes c_{t1}^* (b-c). In (d) $adj(c_s^{seed})$ is iterated over until the next customer (c_{t2}^*) is found in a not-yet-ruined tour. The second string is also removed by the 'string' method and l_{t2} is, coincidentally, 4 again (e). The final ruined state is illustrated in (f).

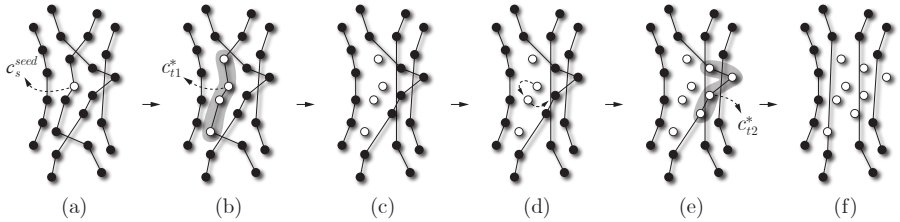

Figure 5.6 An Example of SISRs' Ruin Method

5.4 Greedy Insertion with Blinks ($\mathcal{R}^+$)

While basic greedy methods insert each customer sequentially at the best position, one may deviate slightly from the best position. Pisinger and Røpke (2007) apply *greedy insertion with noise function* by adding a randomized noise term to the insertion cost. This thesis, by contrast, introduces *greedy insertion with blinks* (Algorithm 4), where the best option for inserting a customer is skipped over with a small probability.

First, the set of absent customers A is sorted according to one of the following orders: *Random*, *Demand*, *Far* or *Close* (line 2). 'Random' enables the set's insertion without any ordering. 'Demand' sorts customers by demand, placing those with the largest demand first. 'Far' inserts the most distant customers from the depot first. Finally, 'Close' inserts customers closest to the depot first. Set A is sorted by Random, Demand, Far and Close by weights equal to four, four, two and one, respectively. Each customer $c \in A$ (line 3) is inserted into solution s at the 'best' position P as follows. Initially, P is set to NULL, indicating that no position for c is selected (line 4). All current tours which are part of the solution ($t \in T$) are iterated over in a random order (line 5). When a tour t has sufficient capacity slack to serve c, all positions P_t inside this tour are iterated over (line 6). Each position is evaluated with a probability of $1 - \gamma$, otherwise skipping the position as if the algorithm 'blinks' and therefore γ represents the blink rate (line 7). If a position P_t is found

for which the cost of inserting c is lower than the current best position P, P_t becomes the new best position (lines 8 and 9). If no position was found in existing tours, a new empty tour is created to serve c (lines 10 to 12). Finally c is inserted at P and removed from the set of absent customers (lines 13 and 14). Blink rate $\gamma = 0$ denotes that customers are always inserted at the best position, a strategy proven to be sub-optimal after experimentation. This may appear somewhat counter-intuitive given that it is more logical to assume that consistently inserting each customer at the best possible position would result in the highest quality solution. However, better positions may appear only after inserting other customers and vice versa, inserting a customer may affect the best positions for subsequent customers.

Algorithm 4 SISRs' Recreate Method

Input: Ruined solution $s = \{T, A\}$ where T is the set of tours and A the set of absent customers

```
1:  procedure RECREATE(s)
2:      sort(A)
3:      for c ∈ A do
4:          P ← NULL
5:          for t ∈ T (which can serve c) do
6:              for P_t in t do
7:                  if U(0,1) < 1 − γ then
8:                      if P = NULL or costAt(P_t) < costAt(P) then
9:                          P ← P_t
10:         if P = NULL then
11:             T ← T ∪ {new tour t}
12:             P ← position in t
13:         insert c at P
14:         A ← A \ {c}
15: end procedure
```

Similar to Algorithm 3, the computational complexity of Algorithm 4 depends on the considered VRP variant. Regarding the CVRP, sorting the list of absent customers requires $O(|A|log(|A|))$ operations. Finding the best position for a single customer requires one to iterate over all positions inside at least one tour (line 5) and in the worst-case inside all tours which requires n operations. Inserting a customer at this best position (line 13) is performed in constant time. Therefore, the worst-case complexity of Algorithm 4 requires $O(|A|n)$ operations. Regarding problem variants with time windows, inserting a customer at the selected position in tour t requires $O(|t|)$ additional operations for updating the

temporal variables. Finally, regarding PDPs, iterating over all positions inside a tour requires $O(|t|^2)$ operations for each pickup and delivery pair. Therefore the worst-case complexity for PDPs, where the solution contains only one tour, is $O(n^2)$.

Out of all feasible options for inserting a customer, one may make a random choice or utilize a heuristic to pick the best option (greedy insertion). The effect of blinking is reflected by the probability p of selecting a specific option based on its rank r. A blink rate of γ implies the best option, which is ranked first ($r = 1$), is selected with probability $p(1) = (1 - \gamma)$. If the best ranked option is blinked over, the probability of selecting the second best option is $p(2) = (1 - \gamma)\,\gamma$ and the third ranked option equals $p(3) = (1 - \gamma)\,\gamma^2$. The selection probability for each rank is expressed by exponential function:

$$p(r) = (1 - \gamma)\,\gamma^{(r-1)} \qquad r \in \{1, ..., \infty\} \tag{5.7}$$

Notice how the blinking algorithm itself is unaware of each option's rank, it only blinks with probability γ while iterating over all options. Thus, blinking results in rank-based selection probabilities without requiring one to rank the options first, in contrast to Heuristic-Biased Stochastic Sampling (HBSS) (Bresina 1996), a closely related selection algorithm. A comparison between HBSS and the blinking algorithm can be found below.

Iterative sampling (Langley 1992) builds an empty solution into a complete solution, which represents a 'sample' of the solution space, by incrementally making random decisions at each decision point. For example when applied to VRP, an unserved customer is randomly selected and inserted at a random position in the partial solution until all customers have been inserted. By contrast, decisions may be made based solely on a specific heuristic. For example, an unserved customer is inserted at the best position by *greedy insertion*. Meanwhile, in HBSS (Bresina 1996), the heuristic's preferred options are applied with a certain probability as follows. The heuristic ranks all R options at a decision point. A rank-based weight $bias(r)$ is assigned to each option by way of a specific *bias function*. The probability $p(r)$ of selecting an option based on its rank r is obtained by dividing its weight by the sum of all option weights:

$$p(r) = \Big(\sum_{k=1}^{R} bias(k)\Big)^{-1} bias(r) \qquad r \in \{1, ..., R\} \tag{5.8}$$

Table 5.1 presents the rank-based probabilities obtained by the following bias functions: constant, logarithmic, linear, cubic and exponential. For each bias

function, the left column represents $p(r)$ assuming there exist only five options ($R = 5$) at the considered decision point. Thirty options are assumed ($R = 30$) within the right hand column and the remaining probability of selecting one of the options $r = 6..30$ is detailed in the last row.

Table 5.1 HBSS rank selection probabilities

	Constant		Logarithmic		Linear		Cubic		Exponential	
r	1		$\log^{-1}(r+1)$		r^{-1}		r^{-3}		e^{-r}	
1	0.200	0.033	0.339	0.109	0.438	0.250	0.843	0.832	0.636	0.632
2	0.200	0.033	0.214	0.069	0.219	0.125	0.105	0.104	0.234	0.233
3	0.200	0.033	0.170	0.055	0.146	0.083	0.031	0.031	0.086	0.086
4	0.200	0.033	0.146	0.047	0.109	0.063	0.013	0.013	0.032	0.031
5	0.200	0.033	0.131	0.042	0.088	0.050	0.007	0.007	0.012	0.012
6-30	-	0.833	-	0.678	-	0.429	-	0.013	-	0.006

Clearly, these probabilities depend upon the number of options. With regard to the logarithmic bias function, the probability $p(r)$ of selecting the best option $r = 1$ is three times smaller when thirty options are available (presented in the right column) compared to when only five options are available (presented left). This influence is negligible when the bias function quickly converges to zero and the partial sum $\sum_{k=1}^{R} bias(k)$ converges to the infinite sum ($R = \infty$). This occurs for strong bias functions, such as the cubic and exponential bias function.

In contrast to HBSS, *greedy insertion with blinks* does not require one to explicitly rank the options in advance to obtain rank-based selection probabilities. Nevertheless, the exponential rank-based probability function of the blinking algorithm (Eq. 5.9) may be expressed as HBSS with exponential bias function as follows:

$$p(r) = (1-\gamma)\,\gamma^{(r-1)} \qquad r \in \{1, ..., \infty\} \tag{5.9}$$

$$= \left(\frac{1-\gamma}{\gamma}\right)\gamma^{r} \tag{5.10}$$

$$= \left(\frac{\gamma}{1-\gamma}\right)^{-1}\gamma^{r} \tag{5.11}$$

Based on the sum of an infinite geometric sequence $\sum_{k=0}^{\infty} ab^k = \frac{a}{1-b}$ for $|b| < 1$ and $a = b$:

$$= \left(\sum_{k=0}^{\infty} \gamma^{(k+1)} \right)^{-1} \gamma^r \tag{5.12}$$

$$= \left(\sum_{k=1}^{\infty} \gamma^{k} \right)^{-1} \gamma^r \tag{5.13}$$

which enables the blinking algorithm's probability function (Eq. 5.9) to be expressed as HBSS (Eq. 5.8) with exponential bias function ($bias(r) = \gamma^r$) when there exists an inifinite number of options ($R = \infty$):

$$p(r) = \left(\sum_{k=1}^{R} bias(k) \right)^{-1} bias(r) \qquad \begin{cases} r \in \{1, ..., R\} \\ R = \infty \\ bias(r) = \gamma^r = \left(\frac{1}{\gamma}\right)^{-r} \end{cases} \tag{5.14}$$

The condition $R = \infty$ implies HBSS bias weights should be divided by the infinite sum $\sum_{k=1}^{\infty} bias(k)$, a value approximated by the partial sum for strong bias functions. For example, HBSS probabilities with $bias(r) = e^{-r}$ are equal to those obtained by the blinking algorithm when $\gamma = \frac{1}{e}$, while the blink rate $\gamma = 0.01$ (applied in Section 5.3) is equal to HBSS with $bias(r) = 100^{-r}$. These results are presented in Table 5.2.

Table 5.2 Comparison of rank selection probabilities for HBSS and Blinks

	HBSS: $bias(r)$			Blinks: γ	
r	e^{-r}	100^{-r}	r	$\frac{1}{e}$	0.010
1	0.632	0.990	1	0.632	0.990
2	0.233	0.010	2	0.233	0.010
3	0.086	0.000	3	0.086	0.000
4	0.031	0.000	4	0.031	0.000
5	0.012	0.000	5	0.012	0.000
6-30	0.006	0.000	6-∞	0.006	0.000

5.5 Simulated Annealing for SISRs

The SISRs neighborhood search, detailed in Sections 5.3 and 5.4, is guided by SA. This well known metaheuristic - introduced by Kirkpatrick et al. (1983) - is based on the statistical model concerning energy changes in annealing systems (Metropolis et al. 1953). In the present study, the system's initial temperature $\mathcal{T}_0$ is reduced by the exponential cooling schedule (Equation 5.15) to its final temperature $\mathcal{T}_f$ in f iterations. Cooling constant c is obtained by way of Equation 5.16.

$$\mathcal{T}_{k+1} = c\,\mathcal{T}_k \qquad 0 \leq k < f \tag{5.15}$$

where:

$$c = \left(\frac{\mathcal{T}_f}{\mathcal{T}_0}\right)^{1/f} \qquad \mathcal{T}_0 > \mathcal{T}_f > 0 \tag{5.16}$$

The probability h for accepting a neighbor solution s^* is dependent upon the current temperature $\mathcal{T}$ and the difference in energy $\Delta E = dist(s^*) - dist(s)$ relative to the current solution s (Equation 5.17). Neighbor solution s^* is accepted if a random selected value from the continuous uniform distribution $U(0,1)$ is less than the probability $h(\Delta E, \mathcal{T})$. Therefore, the acceptance criterion is expressed by way of Equation 5.18.

$$h(\Delta E, \mathcal{T}) = exp(-\Delta E/\mathcal{T}) \tag{5.17}$$

$$dist(s^*) < dist(s) - \mathcal{T}\ \ln(\,U(0,1)) \tag{5.18}$$

A solution s is represented by two sets $s = \{T, A\}$. Set T is the set of tours, each of which serves at least one customer, while set A is the set of absent customers, all of whom are not served by any tour $t \in T$. The *initial solution* contains an equal number of tours as there are customers defined in the problem, therefore only one customer is served in each tour. This initial solution is passed to the local search procedure, detailed by way of Algorithm 5. First, the initial solution is saved as the best solution s^{best} found thus far (line 2). Temperature $\mathcal{T}$ is set to the initial temperature $\mathcal{T}_0$ (line 3). Next, f improvement iterations are performed (lines 4 to 10). During each iteration, a neighbor solution s^* is generated from current solution s by employing SISRs' R&R procedure (line 5). This neighbor is accepted as the new current solution s if SA's acceptance criterion is satisfied (lines 6 and 7) and saved if it improves upon the best solution (lines 8 and 9). Finally the system is cooled down (line 10) before proceeding onto the next iteration.

Algorithm 5 Local Search Metaheuristic

Input: Initial solution $s = \{T, A\}$ where T is the set of tours and A the set of absent customers

```
1:  procedure LOCAL-SEARCH(s)
2:      s^best ← s
3:      T ← T_0
4:      for f iterations do
5:          s* ← SISRS-RUIN-RECREATE(s)
6:          if dist(s*) < dist(s) − T ln(U(0,1)) then
7:              s ← s*
8:          if dist(s*) < dist(s^best) then
9:              s^best ← s*
10:         T ← cT
11: end procedure
```

5.6 Fleet Minimisation

This work introduces a two-stage approach to accommodate a hierarchical objective which is primarily the minimisation of the number of vehicles. When the number of vehicles is minimised by way of accepting neighbor solutions with an equal or smaller number, there is nothing that will force the heuristic towards actually creating solutions with fewer vehicles. The newly developed *absences-based* neighbor acceptance criterion is employed during the vehicle minimisation stage, detailed by way of Algorithm 6. The solution which serves all customers with the lowest number of vehicles is represented by s^{best} (line 2). For each customer c the number of solutions where c was not served by any tour (absent from the solution) is tracked by its associated absence counter abs_c. This counter is initially set to 0 for all customers (lines 3 and 4). In each iteration the following steps are performed. First, a neighbor solution s^* is obtained by SISRs' $\mathcal{R}^-$ and $\mathcal{R}^+$ procedures[1] (line 6). This neighbor is accepted if (i) the number of absent customers decreased or (ii) the sum of absence counters associated with the absent customers $sumAbs(A) = \sum_{c \in A}(abs_c)$ decreased (lines 7 and 8). If the current solution is feasible ($A^* = \emptyset$), a copy is saved as s^{best}. Afterwards the tour, whose served customers represent the lowest sum of absence counters, is removed from the current solution (lines 9 to 11). Finally, each absent customer

[1] Notice that in the $\mathcal{R}^+$ procedure (Algorithm 4), lines 11-14 should not be executed during the fleet minimisation phase if line 10 is true and if s^* would employ more vehicles than s as a result of creating a new tour.

in the neighbor solution has their associated absence counter incremented (lines 12 and 13).

Algorithm 6 Fleet Minimisation

Input: Solution $s = \{T, A\}$ where T is the set of tours and A the set of absent customers

```
1:  procedure FLEET-MINIMISATION(s)
2:      s^best ← s
3:      for c defined in the problem do
4:          abs_c ← 0
5:      while minimising tours do
6:          s* ← SISRS-RUIN-RECREATE(s)
7:          if |A*| < |A| or sumAbs(A*) < sumAbs(A) then
8:              s ← s*
9:          if A* = ∅ then
10:             s^best ← s*
11:             remove t ∈ T with the lowest sumAbs(t)
12:         for c ∈ A* do
13:             abs_c ← abs_c + 1
14: end procedure
```

In essence, neighbor s^* is accepted only if the number of absent customers $|A^*|$ decreased or if the sum of absence counters associated with the absent customers $sumAbs(A^*)$ decreased. This *absences-based* acceptance criterion for minimising fleet size proved to be very efficient without complicating the overall method. In fact, it only requires the replacement of SA's *distance-based* acceptance criterion by this *absences-based* acceptance criterion during the fleet minimisation phase.

5.7 Parameters

Parameter variations for both the $\mathcal{R}^-$ and $\mathcal{R}^+$ methods are evaluated for the CVRP, VRPTW and PDPTW. At most two parameters are varied simultaneously to determine their influence upon the algorithm's performance while the values of all other parameters remain fixed. These predefined values, presented in Table 5.3 and 5.4, were observed as high performing while developing the proposed algorithm and are therefore employed throughout the experiments reported in Section 5.8.

Table 5.3 Parameter Settings for SISRs $\mathcal{R}^-$ and $\mathcal{R}^+$ methods

SISRs' $\mathcal{R}^-$
$\bar{c} = 10$
An average of 10 removed customers has been proven a good ruin parameter. When $\bar{c}$ is equal to 10 it implies the removal of at least one and at most 19 customers. Decreasing this parameter, for example to five, was observed to be insufficient for diversification. By contrast, increasing $\bar{c}$ for example to 20, results in extensively ruined solutions. Such solutions cannot easily be recreated into improving solutions and most likely result in highly divergent neighbors, which disables intensification of potentially good solutions.
$L^{max} = 10$
The cardinality of removed strings is limited to 10 since removing longer strings fails to improve final solution quality. This confirms the observations made by Fahrion and Wrede (1990) whose chain-exchange procedure limits string cardinality to half the average tour length. By employing Equation 5.4 the number of removed strings k_s will range from one to three for solutions with large tours ($\overline{\lvert t \in T \rvert} \geq L^{max}$) and one to 19 for solutions whose tours all have a cardinality of one ($\overline{\lvert t \in T \rvert} = 1$). This situation almost always only occurs during the first iteration of the search (see Section 5.5).
$\alpha = 0.5$
By setting the split rate to 50% ($\alpha = 0.5$), both 'string' and 'split string' procedures are executed with equal probability.
$\beta = 0.01$
This setting results in values for the number of preserved customers $m = 1, 2, 3, ..., m^{max}$ having respective probabilities $p = 1.00\%, 0.99\%, 0.9801\%, ..., (100 - p_{m=1} - p_{m=2} - p_{m=3} - ... - p_{m=m^{max}-1})\%$. As such, it is very likely that customers removed within a 'split string removal' are those at the start or end in the tour's sequence.
SISRs' $\mathcal{R}^+$
$\gamma = 0.01$
It was observed that a blink rate of 1% ($\gamma = 0.01$) was the most effective in terms of final solution quality.

Table 5.4 Parameter Settings for SISRs' Fleet Minimisation and SA

Fleet minimisation
$iterations = 10\%\ of\ all\ iterations$
Generally the minimum number of vehicles is found within three minutes. Therefore, dedicating 10% of all iterations to the fleet minimisation is considered to be sufficient.
SA
$\mathcal{T}_0 = 100, \mathcal{T}_f = 1$
The search begins at an initial temperature $\mathcal{T}_0 = 100$ and ends at the final temperature $\mathcal{T}_f = 1$.
$iterations = it(v)$
The number of iterations it is determined as a function of problem size v by linear interpolation. The minimum and maximum problem size is 100 and 1000 respectively, as per Uchoa et al. (2017). The present study set $it(100) = 3 \times 10^7$ and $it(1000) = 3 \times 10^8$, thus enabling the direct comparison of SISRs' calculation times against those from the aforementioned paper.

5.7.1 Parameters for Distance Minimisation

The benchmark sets employed throughout this analysis were introduced by Uchoa et al. (2017) for the CVRP, Gehring and Homberger (2001) for the VRPTW and Li and Lim (2001) for the PDPTW. While results for the CVRP can be directly compared by the solution's distance, results for the VRPTW and PDPTW are not directly comparable when the number of utilized vehicles differs. Solutions employing fewer vehicles usually result in much longer travel distances. Therefore, the number of utilized vehicles is fixed to the most frequently obtained number per instance, thereby ensuring a fair comparison. For each evaluated parameter setting, the complete benchmark set is solved once. Results for all settings are compared by way of their average ranks, presented throughout Tables 5.5, 5.6 and 5.7. For each benchmark instance, every parameter setting is ranked by its obtained result with respect to those obtained by all other settings. The average rank for each setting is the average of all its obtained ranks. Low average ranks represent the best performing settings and are colored dark gray. Meanwhile, the highest average ranks correspond to the worst performing settings and are colored light gray.

Since maximum string cardinality L^{max} and the average number of removed

customers $\bar{c}$ both influence the maximum number of strings K^{max}, the values of both these parameters are varied simultaneously in order to determine which combination performs best. Given that average computational runtime is proportional to $\bar{c}$, changes to this value will result in varying runtimes. This is compensated for by adjusting the number of iterations inversely proportional to $\bar{c}$. This enables a comparison of different settings for $\bar{c}$ while maintaining equal computational effort or calculation time. Average ranks are presented in Table 5.5. Note that not all combinations are feasible. Given that at least one string is removed in each iteration, it is impossible to obtain an average number of removed customers less than $(1+L^{max})/2$. For example, when L^{max} is set to 5, $\bar{c}$ should be equal to $(1+5)/2 = 3$ or greater. Therefore, smaller values for $\bar{c}$ such as 2 have not been considered. It is observed that for both the CVRP and PDPTW the best parameter combinations are identical ($\bar{c} = 10$, $L^{max} = 10$), while slightly larger values ($\bar{c} = 15$ and $L^{max} = 20$) perform better when solving the VRPTW.

Table 5.5 Maximum String Cardinality and Average Number of Removed Locations

	CVRP					VRPTW					PDPTW				
	$\bar{c}$					$\bar{c}$					$\bar{c}$				
L^{max}	2	5	10	15	30	2	5	10	15	30	2	5	10	15	30
2	12.2	8.1	8.9	11.0	12.6	13.4	10.9	11.1	11.7	12.7	10.1	8.6	9.2	9.3	10.8
5	-	4.7	4.7	5.4	8.9	-	7.2	6.3	6.1	8.2	-	5.3	5.5	6.3	7.6
10	-	-	4.1	5.5	7.8	-	-	5.0	4.1	5.7	-	-	4.8	5.2	6.6
20	-	-	-	5.3	7.0	-	-	-	3.6	4.6	-	-	-	6.1	6.6
50	-	-	-	-	8.0	-	-	-	-	4.9	-	-	-	-	7.1

Next, the split rate α and split depth β are varied simultaneously. The average rank obtained by each combination is presented in Table 5.6. In general, the best average ranks are obtained when split rate α is set within the range 0.25 to 0.75 in combination with high values for split depth β such as 0.9 or 0.99. While this well-performing region is less pronounced for the PDPTW, it is distinctive for both the CVRP and VRPTW.

Finally, blink rate γ is varied while all other parameters remain fixed. Average ranks obtained for CVRP exhibit two peaks. Further analysis of these results showed that the peak at $\gamma = 0$ is introduced by results on the small instances while the second peak at $\gamma = 0.05$ emerges due to those for the large instances. For both the VRPTW and PDPTW the best value tends towards $\gamma = 0.05$.

Table 5.6 Split String Parameters α and β

	CVRP					VRPTW					PDPTW				
	α					α					α				
β	0.0	0.25	0.50	0.75	1.0	0.0	0.25	0.50	0.75	1.0	0.0	0.25	0.50	0.75	1.0
0.00	14.2	12.3	12.5	13.8	13.4	12.7	12.1	12.3	12.1	13.6	11.0	10.5	11.1	11.3	11.3
0.50	14.3	13.2	12.2	12.8	11.7	12.8	12.4	11.2	12.5	12.6	10.6	10.6	9.8	11.4	11.1
0.90	14.5	10.5	10.4	8.6	9.7	13.0	10.5	10.8	10.3	12.3	12.0	10.0	9.2	10.2	9.9
0.99	14.1	9.4	9.0	10.7	12.5	13.6	10.1	10.3	10.9	14.2	11.2	10.6	10.6	11.3	13.2
1.00	13.2	11.2	10.0	10.1	13.7	12.9	10.1	11.6	12.4	17.2	11.5	11.0	10.9	12.5	18.0

Table 5.7 Blink Rate Parameter γ

CVRP					VRPTW					PDPTW				
γ					γ					γ				
0.0	0.001	0.01	0.05	0.10	0.0	0.001	0.01	0.05	0.10	0.0	0.001	0.01	0.05	0.10
2.57	2.76	2.87	2.53	3.12	2.91	2.89	2.75	2.72	3.06	3.00	2.79	2.72	2.43	2.63

5.7.2 Parameters for Fleet Minimisation

The performance of Algorithm 6 is evaluated on the 1000 customers benchmark set of Gehring and Homberger (2001) by taking the best of five runs per instance. The cumulative number of vehicles (CNV) for each set is compared against the (best of five) CNV obtained by NB-f(n) (Nagata et al. 2010) and Hybrid Genetic Search with Advanced Diversity Control (HGSADC) (Vidal et al. 2013) in Table 5.8. In addition, results of SISRs with different L^{max} settings are reported. The best results are marked dark gray while results of SISRs are marked light gray when better than the competing algorithms but not the best of the different L^{max} settings. The average calculation time (T) is specified in minutes which is at most 10 minutes for NB-f(n) and unknown for HGSADC.

These results indicate the best settings may vary for different sets. For example, removing more strings of lower cardinality ($L^{max} = 2$) enables to find the best CNV for set C2 while removing strings of intermediate cardinality ($L^{max} = 5$) obtained a better CNV than NB-f(n) and HGSADC for set C1. However, after conducting a parameter analysis similar to that performed in Section 5.7.1, no superior regions were observed. Consequently, the parameter settings presented in Table 5.3 are employed during the fleet minimisation phase when solving the VRPTW and PDPTW.

Table 5.8 Fleet Minimisation with Different L^{max} Settings

		NB-f(n)	HGSADC	SISRs		
				$L^{max} = 10$	$L^{max} = 5$	$L^{max} = 2$
Set	n	CNV (T)	CNV	CNV (T)	CNV (T)	CNV (T)
R1	1000	919 ($\leq$ 10)	919	919 (0.2)	919 (0.2)	919 (0.1)
R2	1000	190 ($\leq$ 10)	190	190 (0.0)	190 (0.0)	190 (0.0)
C1	1000	941 ($\leq$ 10)	941	939 (2.8)	938 (2.4)	939 (2.7)
C2	1000	291 ($\leq$ 10)	288	291 (9.1)	292 (3.7)	288 (2.5)
RC1	1000	900 ($\leq$ 10)	900	900 (0.1)	900 (0.0)	900 (0.0)
RC2	1000	183 ($\leq$ 10)	182	182 (0.3)	182 (0.3)	188 (0.4)

Note. Results are ranked first (best), second and third place by way of dark gray, gray, and white, respectively.

Table 5.9 Maximum String Cardinality and Average Number of Removed Locations

	CVRP					VRPTW					PDPTW				
	$\bar{c}$					$\bar{c}$					$\bar{c}$				
L^{max}	2	5	10	15	30	2	5	10	15	30	2	5	10	15	30
2	1.5	1.2	1.0	1.1	1.0	2.1	2.1	2.1	2.1	2.3	6.4	5.7	5.9	5.7	6.1
5	-	1.4	1.2	1.2	1.2	-	1.2	1.3	1.2	1.4	-	1.5	1.5	1.5	2.1
10	-	-	1.4	1.2	1.1	-	-	1.3	1.3	1.6	-	-	1.4	1.5	2.3
20	-	-	-	1.3	1.1	-	-	-	1.5	2.0	-	-	-	1.7	2.5
50	-	-	-	-	1.3	-	-	-	-	2.0	-	-	-	-	2.6

Table 5.10 Split String Parameters α and β

	CVRP					VRPTW					PDPTW				
	α					α					α				
β	0.0	0.25	0.50	0.75	1.0	0.0	0.25	0.50	0.75	1.0	0.0	0.25	0.50	0.75	1.0
0.00	1.5	1.1	1.3	1.1	1.3	1.3	1.4	1.3	1.3	1.5	2.4	2.1	2.6	2.2	2.1
0.50	1.1	1.1	1.1	1.3	1.3	1.4	1.5	1.3	1.4	1.4	2.5	2.3	2.1	1.9	2.1
0.90	1.1	1.1	1.5	1.1	1.1	1.3	1.3	1.5	1.5	1.6	2.7	1.5	2.1	1.7	1.7
0.99	1.1	1.3	1.3	1.1	1.1	1.2	1.3	1.6	1.5	2.0	2.5	1.9	1.7	2.1	2.8
1.00	1.1	1.1	1.3	1.1	1.3	1.4	1.3	1.6	1.8	6.1	2.5	2.3	2.4	2.5	9.8

Table 5.11 Blink Rate Parameter γ

CVRP					VRPTW					PDPTW				
γ					γ					γ				
0.0	0.001	0.01	0.05	0.10	0.0	0.001	0.01	0.05	0.10	0.0	0.001	0.01	0.05	0.10
1.05	1.02	1.03	1.07	1.09	1.04	1.03	1.02	1.08	1.10	1.20	1.22	1.19	1.15	1.17

5.8 Computational Results

SISRs' computational results are documented throughout the following pages. First, SISRs' results for the most relevant CVRP benchmark set are compared against those obtained by current state-of-the-art methods within Section 5.8.1. Following this, although the CVRP constitutes the present work's core focus, SISRs' general applicability is verified by way of solving numerous CVRP variants. The very minor adjustments necessary to enable SISRs to solve each of these problems is detailed within Section 5.8.2.

5.8.1 CVRP

Uchoa et al. (2017) introduced a new benchmark set due to the fact that *"the existing sets became too easy, are too artificial or do not cover the wide range of characteristics found in real applications"*. This recent benchmark set contains 100 instances whose sizes range from 100 to 1000 customers and which cover a wide variety of characteristics. Uchoa et al. (2017) report the detailed results obtained by two state-of-the-art heuristics, both conducted on Xeon CPU @ 3.07 GHz:

- ILS-SP: Iterated Local Search with Set Partitioning (Subramanian et al. 2013)
- UHGS: Unified Hybrid Genetic Search (Vidal et al. 2014)

This study's experiments were performed on Xeon(R) CPU E5-2650 v2 @ 2.60GHz and compared against the results of ILS-SP and UHGS which were taken from Uchoa et al. (2017). No reimplementation of these methods was therefore required. The full benchmark set is divided into three sets for comparison based on problem size n: Small, Medium and Large (Table 5.12).

Table 5.12 Benchmark Subsets

	n	Instances	# Instances
Small	100 – 250	X-n101-k25 – X-n247-k47	32
Medium	250 – 500	X-n251-k28 – X-n491-k59	36
Large	500 – 1000	X-n502-k39 – X-n1001-k43	32
All	100 – 1000	X-n101-k25 – X-n1001-k43	100

The complete results obtained by each heuristic are documented in Appendix A, where Tables A1, A2 and A3 report the results for the Small, Medium and Large instances, respectively. The average distance (Avg), the best distance (Best) and the average calculation time in minutes (T) obtained after 50 runs per instance

are presented in these tables. Heuristics are ranked first (best), second and third place by way of dark gray, gray and light gray respectively for the columns Avg, Best and T. The distance (Dist) and number of vehicles (NV) of the best know solution (BKS) are also reported. Throughout the entire experimental campaign, 40 BKSs were improved by SISRs and 40 solutions of equal quality were found which are highlighted in dark and light gray respectively in the New BKS column.

To provide a clear overview, average calculation times and gaps to the best known solution (BKS) in percentage terms $(100(result - BKS)/BKS)$ for the average and best results are summarized by way of the minimum (Min), maximum (Max), average (Avg) and median (Median) values in Table 5.13. These values are taken directly from the bottom of Tables B1, B2 and B3.

Table 5.13 Summarized Computational Results

	ILS-SP			UHGS			SISRs		
	Avg	Best	T	Avg	Best	T	Avg	Best	T
Small (100-250)									
Min	0.00	0.00	0.1	0.00	0.00	1.4	0.00	0.00	0.2
Max	2.50	1.19	17.8	0.30	0.06	20.4	0.50	0.16	18.4
Avg	0.31	0.12	2.4	0.07	0.00	6.0	0.11	0.01	5.2
Median	0.19	0.00	1.6	0.03	0.00	5.4	0.05	0.00	5.0
Medium (250-500)									
Min	0.00	0.00	2.0	0.00	0.00	6.5	0.01	-0.18	9.8
Max	1.54	0.73	60.6	0.49	0.10	86.7	1.35	0.28	58.4
Avg	0.54	0.17	23.1	0.24	0.02	30.3	0.20	0.02	27.0
Median	0.42	0.12	16.8	0.26	0.00	22.4	0.15	0.01	22.4
Large (500-1000)									
Min	0.00	0.00	27.3	0.11	0.00	33.1	-0.26	-0.42	60.9
Max	1.74	1.45	792.8	0.68	0.33	560.8	0.38	0.15	412.7
Avg	0.76	0.49	195.7	0.29	0.04	268.6	-0.01	-0.13	152.0
Median	0.75	0.39	144.7	0.26	0.02	258.6	0.01	-0.14	144.8
All (100-1000)									
Min	0.00	0.00	0.1	0.00	0.00	1.4	-0.26	-0.42	0.2
Max	2.50	1.45	792.8	0.68	0.33	560.8	1.35	0.28	412.7
Avg	0.53	0.26	71.7	0.20	0.02	98.8	0.10	-0.03	60.0
Median	0.40	0.11	17.7	0.20	0.00	22.4	0.07	0.00	22.4

Results are ranked first (best), second and third place by use of the colors dark gray, gray and light gray respectively.

UHGS outperforms the other methods for Small instances. For the Medium set, UHGS and SISRs perform equally well insofar as obtaining the best solutions (Best). Additionally, it was observed that SISRs was the most robust method. Meanwhile, for the Large set, SISRs outperforms both heuristics on all aspects - it is the most robust method which finds the best solutions in the least amount of computation time.

The average solution quality (Avg columns of Tables B1, B2 and B3) expressed as gaps have also been visualized by way of boxplots for each heuristic and problem set in Fig. 5.7. These boxplots demonstrate SISRs' efficiency and robustness by exhibiting both a small median and spread of average gaps.

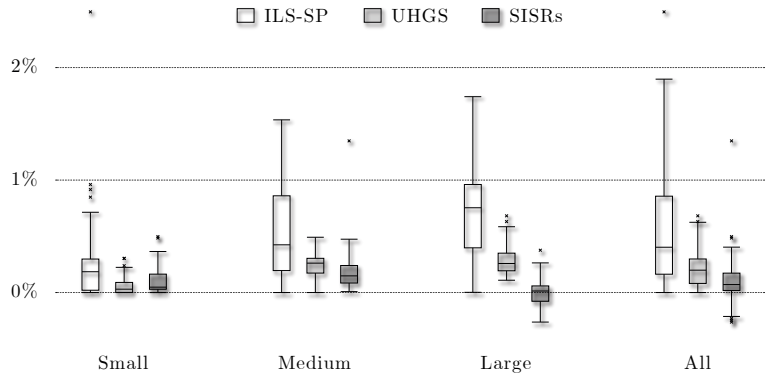

Figure 5.7 Gaps of the Average Results
Note. Gaps of the average results to the best known solutions (Uchoa et al. 2017) visualized by way of boxplots.

Finally, average solution quality is analyzed by conducting a one-tailed Wilcoxon signed-rank test (Wilcoxon 1945), which is a nonparametric test for comparing two independent groups and performed by the software environment R (R Core Team 2017). The only assumption required for this test is that observations are independent. The following null-hypothesis H_0 and alternative hypothesis H_1 is considered, where X represents ILS-SP or UHGS.

$$H_0 : \quad \text{Avg(SISRs)} \quad = \quad \text{Avg(X)}$$

$$H_1 : \quad \text{Avg(SISRs)} \quad > \quad \text{Avg(X)}$$

The average results obtained by SISRs are compared against the average results of ILS-SP and UHGS. Rejecting H_0 implies that the average results obtained

by SISRs are different from those obtained by alternative heuristic X. Rejecting H_1 implies that the values of the average results obtained by SISRs are lower than those obtained by X or that SISRs' average solution quality is better than X'. The obtained p-values are presented in Table 5.14.

Table 5.14 p-values of the Wilcoxon Rank-Sum Test

	Small		Medium		Large		All	
	ILS-SP	UHGS	ILS-SP	UHGS	ILS-SP	UHGS	ILS-SP	UHGS
H_0	0.029239	0.022526	0.000013	0.002624	0.000000	0.000000	0.000000	0.00000
H_1	0.014620	0.989361	0.000007	0.001312	0.000000	0.000000	0.000000	0.00000

While the overall significance level α is set to 2.5%, Bonferroni correction (Dunn 1961) is applied in order to compensate for the family-wise error rate. Bonferroni correction sets the significance level for testing individual hypotheses to α/m where m is the number of tests conducted using the same data. Since four tests are performed using the Small, Medium and Large data sets and four again using All data, a total of eight hypotheses are tested each time with the same data. Therefore p-values are assessed with respect to the Bonferroni adjusted level $0.025/8 = 0.003125$ or 0.3125%. If the p-value is lower than this level, the hypothesis is rejected and the cell is colored gray. The null-hypothesis H_0 and alternative hypothesis H_1 are both rejected when considering Medium, Large and All benchmark sets indicating that SISRs significantly differs from both other heuristics and that its average results are significantly better.

In addition to the performance analysis employing the benchmark set introduced by Uchoa et al. (2017), SISRs is applied to the classical benchmarks listed in Table 5.15. Detailed results obtained by SISRs with respect to these multiple benchmarks are documented within Appendix B. These experiments are performed only in the interests of completeness. Going forward the authors highly recommend that further research utilize the most recent benchmark set of Uchoa et al. (2017). The reasons for why immediately become clear upon providing a brief comparative overview of the various benchmarks.

Table 5.15 CVRP Benchmark Sets

C69	Christofides and Eilon (1969)	R95	Rochat and Taillard (1995)
C79	Christofides et al. (1979)	G98	Golden et al. (1998)
F94	Fisher (1994)	L05	Li et al. (2005)
A95	Augerat et al. (1995)	U17	Uchoa et al. (2017)

The composition of each set is summarized in Table 5.16: the number of instances (#Inst), a boxplot which represents problem size distribution, the distribution of customers (Cust) and the cumulative distance (CD) obtained by SISRs. Customer distribution is categorized as either random (R), clustered (C), real geographic coordinates (G) or artificially distributed along geometric shapes such as concentric circles or squares (A). All benchmark sets prior to R95 contain very few instances and are of very small size. The first medium-sized instances were introduced by G98, wherein customers are distributed along geometric shapes which are very unlikely to appear in real-world environments. L05 contains large instances which are based upon those of G98 with similarly unrealistic customer distributions. U17 is the only benchmark set which exhibits a wide range of problem sizes with relevant customer distributions, vehicle capacities, as well as many other realistic characteristics. Interested readers are referred to Uchoa et al. (2017) for full details. SISRs' results for each benchmark are summarized by the average gap values (Gap) in Table 5.16.

Table 5.16 Results for the Classical Benchmark Sets

Set	#Inst	Distribution of instance size n	Cust	Gap
		0 200 400 **600** 800 1000 1200		
C69	13		R, G	0.00
C79 (M)	5		R, C	0.13
C79 (CMT)	14		R, C	0.05
F94	3		G	0.02
A95 (A)	27		R	0.00
A95 (B)	23		C	0.00
A95 (P)	23		R	0.00
R95	13		C, G	0.05
G98	20		A	0.44
L05	12		A	0.82
U17	100		R, C, RC	0.10

Average gaps obtained by SISRs are at most 0.13% for all benchmark sets except for G98 and L05 where average gaps were 0.44% and 0.82%, respectively. Note that initial calculation times for L05 were considerably longer than for other instances of comparable size. This is very likely due to the high number of customers per tour compared to all other benchmarks. Therefore, the number of executed iterations was divided by ten for this benchmark.

5.8.2 CVRP Variants

Various sets of benchmark instances from the literature were employed to evaluate SISRs' performance. Minimal adjustments were necessary to enable SISRs to solve the CVRP variants addressed and therefore a structured overview for the adjustments related to each problem attribute is provided by way of Table 5.18. If more than one change was necessary then the changes take the form of a numbered list. The computational results were then compared against state-of-the-art methods for each CVRP variant. A detailed breakdown of the results obtained by both SISRs and these state-of-the-art methods for each individual instance is presented in Appendix C. A quick consultation of these results immediately reveals SISRs' impressive dominance. Despite the various state-of-the-art methods it is compared against, being general frameworks or dedicated single-problem heuristics, SISRs' proves highly competitive: very often matching or improving the best known solutions across all variants. Even more impressively, for large-scale instances - which are more representative of real-world environments - SISRs produces many new best known solutions. This is especially true for the PDPTW, for which SISRs improves more than half of the considered benchmark set's instances (178 out of 298, with 58 of these solutions utilizing fewer vehicles). Full details concerning these improved solutions can be found in Table D11, where improved solutions are colored dark gray and also marked with an asterisk when the number of vehicles has been reduced. Meanwhile, solutions of equal quality are colored light gray.

Table 5.17 Adjustments Accommodating VRP Variants

Multi-Depot
1) ($\mathcal{R}^-$) While sorting customers by 'Far' or 'Close' to depot (see Section 5.4), the distance to the closest depot is employed. 2) ($\mathcal{R}^+$) When a new vehicle must be used to include customer c, the closest depot to c which has an unused vehicle is selected.
Backhauls
1) ($\mathcal{R}^-$) If a tour only contains backhaul customers after a string is removed, all remaining customers are removed. 2) (SA) A solution associated with a higher number of tours which exclusively contain backhaul customers is never accepted.
Mixed/Simultaneous Deliveries and Pickups.
For each tour location two additional values are maintained which correspond to the maximum capacity consumption in all previous/following locations. This preprocessing requires $O(\|t\|)$ time and enables a capacity check in $O(1)$ time ($\mathcal{R}^+$).
Open
A dummy location to which all other locations have a distance equal to zero is introduced. This location represents the end depot in each tour.
Cumulative
($\mathcal{R}^+$) The delta evaluation method, which is employed to select the best insertion position, is modified to represent the cumulative objective.
Service Time
($\mathcal{R}^+$) An additional ordering in terms of decreasing service time is employed when each customer has an individual service time.
Time Windows
1) Feasibility checks concerning time windows are based on the 'push forward' global variables (Savelsbergh 1990) which require $O(\|t\|)$ time for preprocessing. These variables enable time window feasibility checks in $O(1)$ time. 2) ($\mathcal{R}^+$) In addition to the sorting orders introduced in Section 5.4 (Random, Demand, Far and Close), customers are sorted with respect to increasing time window length, increasing time window start and decreasing time window end.

Table 5.18 Adjustments Accommodating VRP Variants (continued)

Maximum Ride Times
The procedure for deriving the global variables concerning time windows is adapted for accommodating maximum ride times[a], increasing its complexity to $O(\|t\|Q)$. In addition, this procedure is employed for deriving the maximum time span between one location and all successive location. Since it takes $\|t\|$ executions for deriving the maximum time span between all pairs, preprocessing of a tour requires $O(\|t\|^2 Q)$ time, which again enables feasibility checks in $O(1)$. A similar approach was reported by Gschwind (2018).
Pickup & Deliveries
1) ($\mathcal{R}^-$) Strings are removed as described in Section 5.3. In addition, the associated pickup/delivery locations which were not included by the string are also removed. 2) ($\mathcal{R}^+$) Greedy insertion with blinks (Algorithm 4) is modified to iterate over feasible combinations for pickup and delivery insertion positions.
Fleet Minimisation
1) Minimisation of fleet size is performed by employing the *absences-based* acceptance criterion during the fleet minimisation phase which is executed during the first 10% of iterations. Algorithm 6 provides details concerning this procedure. 2) A solution with an increased number of tours is never accepted in the distance minimisation phase.

[a] This corresponds to the procedure by Tang et al. (2010) who assume a worst-case complexity of $O(\|t\|^2)$. Nevertheless, if $\|t\|/2 > Q$, a worst case complexity of $O(\|t\|Q)$ may be assumed since a service's timing is adjusted at most Q times.

5.9 Conclusions

This chapter introduced Slack Induction by String Removals (SISRs), a heuristic procedure capable of achieving highly competitive and new best results within short computation times for both the Capacitated Vehicle Routing Problem and a wide range of associated problems such as the VRP with Time Windows, the Pickup & Delivery Problem with Time Windows and the Dial-a-Ride Problem. SISRs is composed of a ruin method which generates spatial slack, followed by a greedy recreate heuristic which leaves sufficient room for solution space exploration. In addition to these components, this thesis also contributes a fleet minimization approach to VRP variants focusing primarily on minimising their number of vehicles employed.

In many respects Laporte (2009) provided much of the motivation behind original contribution of this chapter when writing of how *"There is ... a sense that several of the most successful metaheuristics are over-engineered and one should now attempt to produce simple and flexible algorithms capable of handling a larger variety of constraints, even if this were to translate into a small loss in accuracy"*. While SISRs exhibits the simplicity and flexibility Laporte recommends, it also serves as a counter argument that this necessitates a loss in accuracy. As demonstrated by this thesis, no such trade-off occurs.

Furthermore, performance with respect to large-scale academic instances was particularly impressive and exceeded expectations. Therefore, there is also room for further research in terms of even larger instances that are more representative of real-world conditions, the results for which should be broadly similar to those obtained for the largest instances addressed within the present study. Finally, all the newly introduced heuristic components demonstrate an impressive convergence speed which engenders an array of future real-world applications, providing a multitude of exciting avenues for future research.

Chapter 6

Conclusions and Afterthoughts

The core contribution of this thesis is a state-of-the-art heuristic for solving a wide range of VRPs: SISRs. The development of SISRs ended up taking a somewhat controversial path. The natural first step of tackling optimisation problems - which are new to the researcher - would be to investigate the existing academic literature. Such a review provides guidance with respect to situating the problem in question, ascertaining its general definition and which solution approaches are among the state of the art. However, modeling and solving the problem before reviewing the literature may result in a fresh perspective which leads to the development of original approaches. Indeed, putting oneself in the position of having to accommodate every aspect of the problem forces one to acquire a solid understanding of the entire problem. Such an experience may provide a sharper perspective and greater appreciation of other approaches for the problem once one does review the literature. This somewhat controversial research trajectory of practical implementation before academic research was the key ingredient behind SISRs.

Despite the excellent solutions obtained by SISRs, publishing those results was a surprisingly difficult process. The initial version of the algorithm was applied only to the CVRP in order to compare its results against those of the previous state-of-the-art algorithms. Against all odds, very promising results were obtained and some best-known solutions were even improved upon. This motivated for improving and streamlining the algorithm for publication. It felt only natural that the algorithm should be published accompanied by its outstanding results on the new CVRP benchmark set of Uchoa et al. (2017), for which it equalled 40% and improved another 40% of the best-known solutions. Given that this benchmark set was published rather recently, one could argue that improving such large number of solutions does not demonstrate that the algorithm furthers upon the state of the art. Still, the benchmark set was accompanied by the impressive results of three state-of-the-art algorithms (Vidal et al. 2014, Subramanian et al. 2013, Pecin et al. 2014). Since we were originally aiming for a straightforward and compact publication introducing the algorithm accompanied by its superior results for this new CVRP benchmark set, a fast publication without any major roadblocks was expected. However, many concerns were raised by the reviewers. Although resolving these took some effort, they contributed greatly to this thesis.

Comparing solution quality against other methods is what defines and furthers the state of the art. Finding such results in relevant publications and combining them is made easy for researchers thanks to websites which collect all of the best-known results for a specific problem. Although it is obviously a very rewarding experience to have some best-known results attributed to one's algorithm, these websites also serve as a comprehensive overview of current best methods. Still, such an overview may become irrelevant unless the algorithms

which generated the best-known results are also published, along with their experimental conditions. For example, some companies abuse these websites for advertising purposes. If they manage to improve some results and are listed in tables of best-know results, they can provide "evidence" to their customers of their superiority. This leads to an unfair competition with academic research, where other metrics, such as computation time, are important for fair comparison.

This mixing of corporate interests and academic integrity leads researchers to focusing on improving end results over all else, a desire which unfortunately clouds the picture of what constitutes the important scientific advancement. Indeed, while numerous solutions have been improved by SISRs, some more recent methods have improved upon these results by a single unit after giving their algorithm a warm start employing SISRs' solution as a starting point. Perhaps future publications of best-known results should be divided into two sets: those obtained by algorithms which comply with standards typically demanded by peer-reviewed journals (reproducibility and so on), and those obtained by methods which deviate from these standards by permitting extremely long calculation times or by relying on solutions obtained by other methods.

As Laporte (2009) stated: "one should now attempt to produce simple and flexible algorithms ... even if this were to translate into a small loss in accuracy". This thesis provides one such uncomplicated algorithm, one which does not even result in a loss in accuracy. Due to SISRs' simplicity, it can be easily implemented and is being deployed widely. As of now, its reproducibility has been demonstrated by researchers obtaining results with less than a 0.01% gap with respect to those reported. Indeed, even master students have been able to reproduce the method's excellent results.

In the end, researchers ought to recognise the fact that their solution methods do not exist in an academic vacuum. Solution requirements may change over time and practitioners will always want to reuse existing algorithms for the new problems which are constantly emerging in what is always a brave new world. Algorithmic simplicity helps immeasurably in this respect and ought to always be recognised as a positive, not a negative. In this sense, the contributions of this thesis should prove of great value not only for future academic research, but also for the solution frameworks currently employed in practice.

Appendix A

Detailed Results for CVRP

Tables A1, A2 and A3 report the results for the Small, Medium and Large instances, respectively. The average distance (Avg), the best distance (Best) and the average calculation time in minutes (T) obtained after 50 runs per instance are presented in these tables. Heuristics are ranked first (best), second and third place by way of dark gray, gray and light gray respectively for the columns Avg, Best and T. The distance (Dist) and number of vehicles (NV) of the best know solution (BKS) are also reported.

Table A1 Results for the CVRP, benchmark set of Uchoa et al. (2017) - Small

Instance		BKS		New BKS		ILS-SP[a] (50 runs)			UHGS[b] (50 runs)			SISRs (50 runs)		
#	Name	Dist	NV	Dist	NV	Avg	Best	Time	Avg	Best	Time	Avg	Best	Time
1	X-n101-k25	27591	26	27591	26	27591.0	27591	0.13	27591.0	27591	1.43	27591.0	27591	0.82
2	X-n106-k14	26362	14	26362	14	26375.9	26362	2.01	26381.8	26378	4.04	26381.5	26362	1.34
3	X-n110-k13	14971	13	14971	13	14971.0	14971	0.20	14971.0	14971	1.58	14971.1	14971	0.96
4	X-n115-k10	12747	10	12747	10	12747.0	12747	0.18	12747.0	12747	1.81	12747.0	12747	0.18
5	X-n120-k6	13332	6	13332	6	13337.6	13332	1.69	13332.0	13332	2.31	13332.0	13332	1.65
6	X-n125-k30	55539	30	55539	30	55673.8	55539	1.43	55542.1	55539	2.66	55556.3	55542	3.06
7	X-n129-k18	28940	18	28940	18	28998.0	28948	1.92	28948.5	28940	2.71	28948.8	28940	1.47
8	X-n134-k13	10916	13	10916	13	10947.4	10916	2.07	10934.9	10916	3.32	10940.1	10916	2.80
9	X-n139-k10	13590	10	13590	10	13603.1	13590	1.60	13590.0	13590	2.28	13595.4	13590	1.98
10	X-n143-k7	15700	7	15700	7	15745.2	15726	1.64	15700.2	15700	3.10	15705.8	15700	2.14
11	X-n148-k46	43448	47	43448	47	43452.1	43448	0.84	43448.0	43448	3.18	43469.2	43448	2.82
12	X-n153-k22	21220	23	21220	23	21400.0	21340	0.49	21226.3	21220	5.47	21229.4	21220	5.56
13	X-n157-k13	16876	13	16876	13	16876.0	16876	0.76	16876.0	16876	3.19	16878.6	16876	3.72
14	X-n162-k11	14138	11	14138	11	14160.1	14138	0.54	14141.3	14138	3.32	14157.1	14138	3.36
15	X-n167-k10	20557	10	20557	10	20608.7	20562	0.86	20563.2	20557	3.73	20560.8	20557	3.16
16	X-n172-k51	45607	53	45607	53	45616.1	45607	0.64	45607.0	45607	3.83	45619.2	45607	5.29
17	X-n176-k26	47812	26	47812	26	48249.8	48140	1.11	47957.2	47812	7.56	47849.6	47812	5.19
18	X-n181-k23	25569	23	25569	23	25571.5	25569	1.59	25591.1	25569	6.28	25579.8	25569	5.54
19	X-n186-k15	24145	15	24145	15	24186.0	24145	1.72	24147.2	24145	5.92	24178.4	24149	4.03
20	X-n190-k8	16980	8	16980	8	17143.1	17085	2.10	16987.9	16980	12.08	16984.9	16980	9.08
21	X-n195-k51	44225	53	44225	53	44234.3	44225	0.87	44244.1	44225	6.10	44298.5	44241	6.10
22	X-n200-k36	58578	36	58578	36	58697.2	58626	7.48	58626.4	58578	7.97	58636.1	58578	6.69
23	X-n204-k19	19565	19	19565	19	19625.2	19570	1.08	19571.5	19565	5.35	19662.3	19565	4.89
24	X-n209-k16	30656	16	30656	16	30765.4	30667	3.80	30680.4	30656	8.62	30669.4	30656	5.95
25	X-n214-k11	10856	11	10856	11	11126.9	10985	2.26	10877.4	10856	10.22	10908.6	10873	8.66
26	X-n219-k73	117595	73	117595	73	117595.0	117595	0.85	117604.9	117595	7.73	117650.4	117595	8.02
27	X-n223-k34	40437	34	40437	34	40533.5	40471	8.48	40499.0	40437	8.26	40529.9	40448	7.57
28	X-n228-k23	25742	23	25742	23	25795.8	25743	2.40	25779.3	25742	9.80	25790.9	25744	10.47
29	X-n233-k16	19230	17	19230	17	19336.7	19266	3.01	19288.4	19230	6.84	19269.7	19232	8.11
30	X-n237-k14	27042	14	27042	14	27078.8	27042	3.46	27067.3	27042	8.90	27089.7	27042	7.24
31	X-n242-k48	82751	48	82751	48	82874.2	82774	17.83	82948.7	82804	12.42	82884.4	82775	9.92
32	X-n247-k47	37274	51	37274	51	37507.2	37289	2.06	37284.4	37274	20.41	37323.2	37274	18.44
					Min:	0.00	0.00	0.13	0.00	0.00	1.43	0.00	0.00	0.18
					Max:	2.50	1.19	17.83	0.30	0.06	20.41	0.50	0.16	18.44
					Avg:	0.31	0.12	2.41	0.07	0.00	6.01	0.11	0.01	5.19
					Median:	0.19	0.00	1.62	0.03	0.00	5.41	0.05	0.00	5.04

[a] ILS-SP (Subramanian et al. 2013)

[b] UHGS (Vidal et al. 2014)

Results produced by the methods considered are ranked first (best), second and third place by way of dark gray, gray and light gray respectively. New best solutions obtained by SISRs and solutions of equal quality are highlighted in dark and light gray respectively in the New BKS column.

Table A2 Results for the CVRP, benchmark set of Uchoa et al. (2017) - Medium

Instance		BKS		New BKS		ILS-SP[a] (50 runs)			UHGS[b] (50 runs)			SISRs (50 runs)		
#	Name	Dist	NV	Dist	NV	Avg	Best	Time	Avg	Best	Time	Avg	Best	Time
33	X-n251-k28	38684	28	38684	28	38840.0	38727	10.77	38796.4	38699	11.69	38791.0	38687	9.75
34	X-n256-k16	18880	17	18880	17	18883.9	18880	2.02	18880.0	18880	6.52	18888.9	18880	11.51
35	X-n261-k13	26558	13	26558	13	26869.0	26706	6.67	26629.6	26558	12.67	26642.3	26558	11.82
36	X-n266-k58	75478	58	75478	58	75563.3	75478	10.03	75759.3	75517	21.36	75617.8	75478	10.81
37	X-n270-k35	35291	36	35291	36	35363.4	35324	9.07	35367.2	35303	11.25	35362.2	35323	11.42
38	X-n275-k28	21245	28	21245	28	21256.0	21245	3.59	21280.6	21245	12.04	21268.6	21245	13.35
39	X-n280-k17	33503	17	33503	17	33769.4	33624	9.62	33605.8	33505	19.09	33628.1	33529	17.68
40	X-n284-k15	20226	15	20226	15	20448.5	20295	8.64	20286.4	20227	19.91	20286.6	20240	15.30
41	X-n289-k60	95151	61	95151	61	95450.6	95315	16.11	95469.5	95244	21.28	95352.2	95233	14.28
42	X-n294-k50	47167	51	47167	51	47254.7	47190	12.42	47259.0	47171	14.70	47274.5	47210	14.67
43	X-n298-k31	34231	31	34231	31	34356.0	34239	6.92	34292.1	34231	10.93	34276.0	34234	14.47
44	X-n303-k21	21744	21	21744	21	21895.8	21812	14.15	21850.9	21748	17.28	21776.5	21751	17.28
45	X-n308-k13	25859	13	25859	13	26101.1	25901	9.53	25895.4	25859	15.31	26207.7	25931	25.69
46	X-n313-k71	94044	72	94044	72	94297.3	94192	17.50	94265.2	94093	22.41	94182.4	94063	18.93
47	X-n317-k53	78355	53	78355	53	78356.0	78355	8.56	78387.8	78355	22.37	78392.4	78355	21.99
48	X-n322-k28	29866	28	29848	28	29991.3	29877	14.68	29956.1	29870	15.16	29927.6	29849	16.89
49	X-n327-k20	27546	20	27546	20	27812.4	27599	19.13	27628.2	27564	18.19	27631.4	27608	21.65
50	X-n331-k15	31102	15	31102	15	31235.5	31105	15.70	31159.6	31103	24.43	31128.2	31122	20.43
51	X-n336-k84	139172	86	139135	86	139461.0	139197	21.41	139534.9	139210	37.96	139373.4	139209	22.80
52	X-n344-k43	42092	43	42068	43	42284.0	42146	22.58	42208.8	42099	21.67	42158.5	42079	21.53
53	X-n351-k40	25946	41	25928	41	26150.3	26021	25.21	26014.0	25946	33.73	25982.1	25938	26.50
54	X-n359-k29	51509	29	51505	29	52076.5	51706	48.86	51721.7	51509	34.85	51577.8	51505	23.14
55	X-n367-k17	22814	17	22814	17	23003.2	22902	13.13	22838.4	22814	22.02	22833.4	22814	36.10
56	X-n376-k94	147713	94	147713	94	147713.0	147713	7.10	147750.2	147717	28.26	147783.6	147721	31.96
57	X-n384-k52	66081	53	65943	53	66372.5	66116	34.47	66270.2	66081	40.20	66107.4	65963	25.91
58	X-n393-k38	38269	38	38269	38	38457.4	38298	20.82	38374.9	38269	28.65	38394.1	38331	30.42
59	X-n401-k29	66243	29	66187	29	66715.1	66453	60.36	66365.4	66243	49.52	66248.5	66189	38.05
60	X-n411-k19	19718	19	19718	19	19954.9	19792	23.76	19743.8	19718	34.71	19768.5	19731	58.42
61	X-n420-k130	107798	130	107798	130	107838.0	107798	22.19	107924.1	107798	53.19	107879.2	107817	47.91
62	X-n429-k61	65483	62	65483	62	65746.6	65563	38.22	65648.5	65501	41.45	65593.6	65485	35.00
63	X-n439-k37	36391	37	36391	37	36441.6	36395	39.63	36451.1	36395	34.55	36473.8	36426	42.13
64	X-n449-k29	55355	29	55269	29	56204.9	55761	59.94	55553.1	55378	64.92	55411.2	55272	38.00
65	X-n459-k26	24181	26	24169	26	24462.4	24209	60.59	24272.6	24181	42.80	24242.2	24175	56.55
66	X-n469-k138	221909	140	221909	140	222182.0	221909	36.32	222617.1	222070	86.65	222227.1	221984	48.03
67	X-n480-k70	89535	70	89458	70	89871.2	89694	50.40	89760.1	89535	66.96	89559.2	89458	50.46
68	X-n491-k59	66633	60	66510	59	67226.7	66965	52.23	66898.0	66633	71.94	66645.5	66517	51.35
					Min:	0.00	0.00	2.02	0.00	0.00	6.52	0.01	-0.18	9.75
					Max:	1.54	0.73	60.59	0.49	0.10	86.65	1.35	0.28	58.42
					Avg:	0.54	0.17	23.12	0.24	0.02	30.30	0.20	0.02	27.01
					Median:	0.42	0.12	16.80	0.26	0.00	22.39	0.15	0.01	22.40

[a] ILS-SP (Subramanian et al. 2013)

[b] UHGS (Vidal et al. 2014)

Results produced by the methods considered are ranked first (best), second and third place by way of dark gray, gray and light gray respectively. New best solutions obtained by SISRs and solutions of equal quality are highlighted in dark and light gray respectively in the New BKS column.

Table A3 Results for the CVRP, benchmark set of Uchoa et al. (2017) - Large

Instance		BKS		New BKS		ILS-SP[a] (50 runs)			UHGS[b] (50 runs)			SISRs (50 runs)		
#	Name	Dist	NV	Dist	NV	Avg	Best	Time	Avg	Best	Time	Avg	Best	Time
69	X-n502-k39	69253	39	69230	39	69346.8	69284	80.75	69328.8	69253	63.61	69274.7	69243	60.90
70	X-n513-k21	24201	21	24201	21	24434.0	24332	35.04	24296.6	24201	33.09	24292.1	24238	77.12
71	X-n524-k137	154594	155	154594	155	155005.0	154709	27.27	154979.5	154774	80.70	154807.2	154651	151.41
72	X-n536-k96	95122	97	94988	96	95700.7	95524	62.07	95330.6	95122	107.53	95173.2	95006	74.72
73	X-n548-k50	86710	50	86701	50	86874.1	86710	63.95	86998.5	86822	84.24	86798.0	86710	64.50
74	X-n561-k42	42756	42	42722	42	43131.3	42952	68.86	42866.4	42756	60.60	42868.1	42774	73.80
75	X-n573-k30	50780	30	50719	30	51173.0	51092	112.03	50915.1	50780	188.15	50804.6	50737	112.98
76	X-n586-k159	190491	159	190423	159	190919.0	190612	78.54	190838.0	190543	175.29	190600.7	190484	86.28
77	X-n599-k92	108736	94	108490	93	109384.0	109056	72.96	109064.2	108813	125.91	108688.6	108548	75.41
78	X-n613-k62	59584	62	59556	62	60444.2	60229	74.80	59960.0	59778	117.31	59731.3	59585	88.09
79	X-n627-k43	62366	43	62210	43	62905.6	62783	162.67	62524.1	62366	239.68	62317.1	62219	89.28
80	X-n641-k35	63839	35	63725	35	64606.1	64462	140.42	64192.0	63839	158.81	63850.3	63750	92.53
81	X-n655-k131	106780	131	106780	131	106782.0	106780	47.24	106899.1	106829	150.48	106844.6	106813	109.60
82	X-n670-k126	146705	134	146451	133	147676.0	147045	61.24	147222.7	146705	264.10	146720.4	146451	198.89
83	X-n685-k75	68415	75	68240	75	68988.2	68646	73.85	68654.1	68425	156.71	68369.0	68271	135.10
84	X-n701-k44	82254	44	81934	44	83042.2	82888	210.08	82487.4	82293	253.17	82065.4	81974	122.53
85	X-n716-k35	43517	35	43414	35	44171.6	44021	225.79	43641.4	43525	264.28	43483.8	43426	158.32
86	X-n733-k159	136366	160	136250	160	137045.0	136832	111.56	136587.6	136366	244.53	136389.3	136255	143.23
87	X-n749-k98	77700	98	77365	98	78275.9	77952	127.24	77864.9	77715	313.88	77509.2	77380	146.28
88	X-n766-k71	114683	71	114525	71	115738.0	115443	242.11	115147.9	114683	382.99	114761.1	114590	174.42
89	X-n783-k48	72715	48	72445	48	73722.9	73447	235.48	73009.6	72781	269.70	72660.7	72492	170.22
90	X-n801-k40	73564	40	73331	40	74005.7	73830	432.64	73731.0	73587	289.24	73436.7	73347	137.11
91	X-n819-k171	158611	173	158267	172	159425.0	159164	148.91	158899.3	158611	374.28	158423.0	158305	172.48
92	X-n837-k142	194187	142	193813	142	195027.0	194804	173.17	194476.5	194266	463.36	193976.9	193824	166.85
93	X-n856-k95	89060	95	89007	95	89277.6	89060	153.65	89238.7	89118	288.43	89131.3	89050	160.04
94	X-n876-k59	99653	59	99331	59	100417.0	100177	409.31	99884.1	99715	495.38	99483.2	99388	217.39
95	X-n895-k37	54071	38	53902	38	54958.5	54713	410.17	54439.8	54172	321.89	54085.8	53993	212.47
96	X-n916-k207	329752	208	329247	207	330948.0	330639	226.08	330198.3	329836	560.81	329509.5	329299	215.35
97	X-n936-k151	133046	159	132926	158	134530.0	133592	202.50	133512.9	133140	531.50	133117.3	133014	412.66
98	X-n957-k87	85663	87	85482	87	85936.6	85697	311.20	85822.6	85672	432.90	85620.0	85546	202.39
99	X-n979-k58	119194	58	119005	58	120253.0	119994	687.22	119502.1	119194	553.96	119120.4	119065	276.55
100	X-n1001-k43	72719	43	72404	43	73985.4	73776	792.75	72956.0	72742	549.03	72528.1	72415	284.34
					Min:	0.00	0.00	27.27	0.11	0.00	33.09	-0.26	-0.42	60.90
					Max:	1.74	1.45	792.75	0.68	0.33	560.81	0.38	0.15	412.66
					Avg:	0.76	0.49	195.67	0.29	0.04	268.61	-0.01	-0.13	151.98
					Median:	0.75	0.39	144.66	0.26	0.02	258.63	0.01	-0.14	144.76

[a] ILS-SP (Subramanian et al. 2013)

[b] UHGS (Vidal et al. 2014)

Results produced by the methods considered are ranked first (best), second and third place by way of dark gray, gray and light gray respectively. New best solutions obtained by SISRs and solutions of equal quality are highlighted in dark and light gray respectively in the New BKS column.

Appendix B

Detailed Results for Classical CVRP benchmarks

Detailed results for 10 runs for each of the benchmark sets are summarized in Tables B1 to B10. The instance name and best known solution (BKS) is provided for each problem in addition to the following problem characteristics: the number of customers n, vehicle capacity Q, maximum tour distance MD, service time ST and number of vehicles k. Calculation times (Time) are reported in minutes.

Table B1 Results for the CVRP, benchmark set of Christofides and Eilon (1969)

							SISRs (10 runs)		
Instance	n	Q	MD	ST	k	BKS	Best	Avg.	Time
E-n13-k4	12	6000	-	-	= 4	247.00	247.00	247.00	0.00
E-n22-k4	21	6000	-	-	= 4	375.00	375.00	375.00	0.00
E-n23-k3	22	4500	-	-	= 3	569.00	569.00	569.00	0.00
E-n30-k3	29	4500	-	-	= 3	534.00	534.00	534.00	0.00
E-n31-k7	30	140	-	-	= 7	379.00	379.00	379.00	0.00
E-n33-k4	32	8000	-	-	= 4	835.00	835.00	835.00	0.00
E-n51-k5	50	160	-	-	= 5	521.00	521.00	521.00	0.00
E-n76-k7	75	220	-	-	= 7	682.00	682.00	682.00	0.45
E-n76-k8	75	180	-	-	= 8	735.00	735.00	735.00	1.13
E-n76-k10	75	140	-	-	= 10	830.00	830.00	830.50	0.93
E-n76-k14	75	100	-	-	= 14	1021.00	1021.00	1021.00	0.76
E-n101-k8	100	200	-	-	= 8	815.00	815.00	815.00	1.06
E-n101-k14	100	112	-	-	= 14	1067.00	1067.00	1067.70	1.70

Table B2 Results for the CVRP, benchmark set of Christofides et al. (1979)

							SISRs (10 runs)		
Instance	n	Q	MD	ST	k	BKS	Best	Avg.	Time
M-n101-k10	100	200	-	-	= 10	820.00	820.00	820.00	0.04
M-n121-k7	120	200	-	-	= 7	1034.00	1034.00	1034.00	0.61
M-n151-k12	150	200	-	-	= 12	1015.00	1015.00	1016.90	4.57
M-n200-k16	199	200	-	-	= 16	1274.00	1275.00	1280.20	9.85
M-n200-k17	199	200	-	-	= 17	1275.00	1275.00	1276.00	10.49

Table B3 Results for the CVRP, benchmark set of Christofides et al. (1979)

							SISRs (10 runs)		
Instance	n	Q	MD	ST	k	BKS	Best	Avg.	Time
CMT1	50	160	-	-	$\leq$ 5	524.61	524.61	524.61	0.00
CMT2	75	140	-	-	$\leq$ 10	835.26	835.26	835.40	0.96
CMT3	100	200	-	-	$\leq$ 8	826.14	826.14	826.14	1.16
CMT4	150	200	-	-	$\leq$ 12	1028.42	1028.42	1030.68	7.13
CMT5	199	200	-	-	$\leq$ 17	1291.29	1291.50	1292.50	12.80
CMT11	120	200	-	-	$\leq$ 7	1042.12	1042.12	1042.12	1.80
CMT12	100	200	-	-	$\leq$ 10	819.56	819.56	819.56	0.08
CMT6	50	160	200	10	$\leq$ 6	555.43	555.43	555.43	0.01
CMT7	75	140	160	10	$\leq$ 11	909.68	909.68	909.68	0.09
CMT8	100	200	230	10	$\leq$ 9	865.94	865.94	865.94	0.70
CMT9	150	200	200	10	$\leq$ 14	1162.55	1162.55	1162.69	14.35
CMT10	199	200	200	10	$\leq$ 18	1395.85	1397.60	1400.03	20.58
CMT13	120	200	720	50	$\leq$ 11	1541.14	1542.86	1542.86	9.14
CMT14	100	200	1040	90	$\leq$ 11	866.37	866.37	866.37	0.35

Table B4 Results for the CVRP, benchmark set of Fisher (1994)

							SISRs (10 runs)		
Instance	n	Q	MD	ST	k	BKS	Best	Avg.	Time
F-n45-k4	44	2010	-	-	= 4	724.00	724.00	724.00	0.00
F-n72-k4	71	30000	-	-	= 4	237.00	237.00	237.00	0.00
F-n135-k7	134	2210	-	-	= 7	1162.00	1162.00	1162.70	4.48

Table B5 Results for the CVRP, benchmark set of Augerat et al. (1995)

							SISRs (10 runs)		
Instance	n	Q	MD	ST	k	BKS	Best	Avg.	Time
A-n32-k5	31	100	-	-	= 5	784.00	784.00	784.00	0.00
A-n33-k5	32	100	-	-	= 5	661.00	661.00	661.00	0.00
A-n33-k6	32	100	-	-	= 6	742.00	742.00	742.00	0.00
A-n34-k5	33	100	-	-	= 5	778.00	778.00	778.00	0.01
A-n36-k5	35	100	-	-	= 5	799.00	799.00	799.00	0.00
A-n37-k5	36	100	-	-	= 5	669.00	669.00	669.00	0.00
A-n37-k6	36	100	-	-	= 6	949.00	949.00	949.00	0.01
A-n38-k5	37	100	-	-	= 5	730.00	730.00	730.00	0.00
A-n39-k5	38	100	-	-	= 5	822.00	822.00	822.00	0.01
A-n39-k6	38	100	-	-	= 6	831.00	831.00	831.00	0.01
A-n44-k6	43	100	-	-	= 6	937.00	937.00	937.00	0.02
A-n45-k6	44	100	-	-	= 6	944.00	944.00	944.00	0.01
A-n45-k7	44	100	-	-	= 7	1146.00	1146.00	1146.00	0.01
A-n46-k7	45	100	-	-	= 7	914.00	914.00	914.00	0.00
A-n48-k7	47	100	-	-	= 7	1073.00	1073.00	1073.00	0.01
A-n53-k7	52	100	-	-	= 7	1010.00	1010.00	1010.00	0.04
A-n54-k7	53	100	-	-	= 7	1167.00	1167.00	1167.00	0.03
A-n55-k9	54	100	-	-	= 9	1073.00	1073.00	1073.00	0.02
A-n60-k9	59	100	-	-	= 9	1354.00	1354.00	1354.00	0.17
A-n61-k9	60	100	-	-	= 9	1034.00	1034.00	1034.00	0.04
A-n62-k8	61	100	-	-	= 8	1288.00	1288.00	1288.00	0.22
A-n63-k9	62	100	-	-	= 9	1616.00	1616.00	1616.00	0.07
A-n63-k10	62	100	-	-	= 10	1314.00	1314.00	1314.50	0.40
A-n64-k9	63	100	-	-	= 9	1401.00	1401.00	1401.00	0.33
A-n65-k9	64	100	-	-	= 9	1174.00	1174.00	1174.00	0.05
A-n69-k9	68	100	-	-	= 9	1159.00	1159.00	1159.00	0.18
A-n80-k10	79	100	-	-	= 10	1763.00	1763.00	1763.00	1.22

Table B6 Results for the CVRP, benchmark set of Augerat et al. (1995)

							SISRs (10 runs)		
Instance	n	Q	MD	ST	k	BKS	Best	Avg.	Time
B-n31-k5	30	100	-	-	= 5	672.00	672.00	672.00	0.00
B-n34-k5	33	100	-	-	= 5	788.00	788.00	788.00	0.00
B-n35-k5	34	100	-	-	= 5	955.00	955.00	955.00	0.00
B-n38-k6	37	100	-	-	= 6	805.00	805.00	805.00	0.01
B-n39-k5	38	100	-	-	= 5	549.00	549.00	549.00	0.01
B-n41-k6	40	100	-	-	= 6	829.00	829.00	829.00	0.03
B-n43-k6	42	100	-	-	= 6	742.00	742.00	742.00	0.01
B-n44-k7	43	100	-	-	= 7	909.00	909.00	909.00	0.00
B-n45-k5	44	100	-	-	= 5	751.00	751.00	751.00	0.01
B-n45-k6	44	100	-	-	= 6	678.00	678.00	678.00	0.04
B-n50-k7	49	100	-	-	= 7	741.00	741.00	741.00	0.00
B-n50-k8	49	100	-	-	= 8	1312.00	1312.00	1312.00	0.63
B-n51-k7	50	100	-	-	= 7	1032.00	1032.00	1032.00	0.02
B-n52-k7	51	100	-	-	= 7	747.00	747.00	747.00	0.03
B-n56-k7	55	100	-	-	= 7	707.00	707.00	707.00	0.05
B-n57-k7	56	100	-	-	= 7	1153.00	1153.00	1153.00	0.06
B-n57-k9	56	100	-	-	= 9	1598.00	1598.00	1598.00	0.11
B-n63-k10	62	100	-	-	= 10	1496.00	1496.00	1496.00	0.07
B-n64-k9	63	100	-	-	= 9	861.00	861.00	861.00	0.07
B-n66-k9	65	100	-	-	= 9	1316.00	1316.00	1316.00	0.83
B-n67-k10	66	100	-	-	= 10	1032.00	1032.00	1032.00	0.59
B-n68-k9	67	100	-	-	= 9	1272.00	1272.00	1272.00	0.49
B-n78-k10	77	100	-	-	= 10	1221.00	1221.00	1221.00	0.40

Table B7 Results for the CVRP, benchmark set of Augerat et al. (1995)

							SISRs (10 runs)		
Instance	n	Q	MD	ST	k	BKS	Best	Avg.	Time
P-n16-k8	15	35	-	-	= 8	450.00	450.00	450.00	0.00
P-n19-k2	18	160	-	-	= 2	212.00	212.00	212.00	0.00
P-n20-k2	19	160	-	-	= 2	216.00	216.00	216.00	0.00
P-n21-k2	20	160	-	-	= 2	211.00	211.00	211.00	0.00
P-n22-k2	21	160	-	-	= 2	216.00	216.00	216.00	0.00
P-n22-k8	21	3000	-	-	= 8	603.00	603.00	603.00	0.00
P-n23-k8	22	40	-	-	= 8	529.00	529.00	529.00	0.00
P-n40-k5	39	140	-	-	= 5	458.00	458.00	458.00	0.00
P-n45-k5	44	150	-	-	= 5	510.00	510.00	510.00	0.00
P-n50-k7	49	150	-	-	= 7	554.00	554.00	554.00	0.01
P-n50-k8	49	120	-	-	= 8	631.00	631.00	631.00	0.08
P-n50-k10	49	100	-	-	= 10	696.00	696.00	696.00	0.16
P-n51-k10	50	80	-	-	= 10	741.00	741.00	741.00	0.02
P-n55-k7	54	170	-	-	= 7	568.00	568.00	568.00	0.08
P-n55-k8	54	160	-	-	= 8	588.00	576.00	576.00	0.03
P-n55-k10	54	115	-	-	= 10	694.00	694.00	694.00	0.14
P-n55-k15	54	70	-	-	= 15	989.00	989.00	989.00	0.05
P-n60-k10	59	120	-	-	= 10	744.00	744.00	744.00	0.29
P-n60-k15	59	80	-	-	= 15	968.00	968.00	968.00	0.13
P-n65-k10	64	130	-	-	= 10	792.00	792.00	792.00	0.29
P-n70-k10	69	135	-	-	= 10	827.00	827.00	827.00	0.27
P-n76-k4	75	350	-	-	= 4	593.00	593.00	593.00	0.97
P-n76-k5	75	280	-	-	= 5	627.00	627.00	627.00	0.16
P-n101-k4	100	400	-	-	= 4	681.00	681.00	681.20	1.93

Table B8 Results for the CVRP, benchmark set of Rochat and Taillard (1995)

							SISRs (10 runs)		
Instance	n	Q	MD	ST	k	BKS	Best	Avg.	Time
tai75a	75	1445	-	-	-	1618.36	1618.36	1618.36	0.29
tai75b	75	1679	-	-	-	1344.62	1344.62	1344.62	0.23
tai75c	75	1122	-	-	-	1291.01	1291.01	1291.01	0.19
tai75d	75	1699	-	-	-	1365.42	1365.42	1365.42	0.69
tai100a	100	1409	-	-	-	2041.34	2041.34	2042.34	2.22
tai100b	100	1842	-	-	-	1939.90	1939.90	1940.54	2.48
tai100c	100	2043	-	-	-	1406.20	1406.20	1406.23	2.66
tai100d	100	1297	-	-	-	1580.46	1580.46	1580.46	1.46
tai150a	150	1544	-	-	-	3055.23	3055.23	3055.23	14.26
tai150b	150	1918	-	-	-	2727.03	2727.88	2728.66	9.52
tai150c	150	2021	-	-	-	2358.66	2358.66	2358.68	11.37
tai150d	150	1874	-	-	-	2645.39	2645.39	2651.93	5.13
tai385	385	65	-	-	-	24366.41	24399.13	24432.09	112.55

Table B9 Results for the CVRP, benchmark set of Golden et al. (1998)

							SISRs (10 runs)		
Instance	n	Q	MD	ST	k	BKS	Best	Avg.	Time
Golden1	240	550	650	-	$\leq$ 10	5623.47	5643.24	5654.31	26.83
Golden2	320	700	900	-	$\leq$ 10	8404.61	8449.12	8451.90	66.55
Golden3	400	900	1200	-	$\leq$ 10	11036.20	11041.02	11042.81	127.47
Golden4	480	1000	1600	-	$\leq$ 12	13590.00	13626.92	13630.99	178.40
Golden5	200	900	1800	-	$\leq$ 5	6460.98	6460.98	6464.67	6.16
Golden6	280	900	1500	-	$\leq$ 8	8412.90	8412.90	8413.41	67.10
Golden7	360	900	1300	-	$\leq$ 9	10102.70	10169.44	10182.00	68.39
Golden8	440	900	1200	-	$\leq$ 11	11635.30	11663.55	11679.33	94.22
Golden9	255	1000	-	-	$\leq$ 14	579.71	583.48	584.88	23.79
Golden10	323	1000	-	-	$\leq$ 16	735.66	739.60	743.57	40.11
Golden11	399	1000	-	-	$\leq$ 18	912.03	917.24	920.72	79.25
Golden12	483	1000	-	-	$\leq$ 19	1101.50	1108.78	1112.26	120.85
Golden13	252	1000	-	-	$\leq$ 27	857.19	861.02	862.95	29.16
Golden14	320	1000	-	-	$\leq$ 30	1080.55	1082.08	1084.78	52.47
Golden15	396	1000	-	-	$\leq$ 34	1337.87	1343.55	1347.79	100.83
Golden16	480	1000	-	-	$\leq$ 38	1611.56	1616.13	1624.47	160.35
Golden17	240	200	-	-	$\leq$ 22	707.76	707.79	708.06	29.51
Golden18	300	200	-	-	$\leq$ 28	995.13	996.10	997.53	57.46
Golden19	360	200	-	-	$\leq$ 33	1365.60	1366.36	1367.62	95.52
Golden20	420	200	-	-	$\leq$ 41	1817.59	1820.11	1820.64	123.01

Table B10 Results for the CVRP, benchmark set of Li et al. (2005)

							SISRs (10 runs)		
Instance	n	Q	MD	ST	k	BKS	Best	Avg.	Time
Li21	560	1200	1800	-	-	16212.83	16220.02	16223.70	33.66
Li22	600	900	1000	-	-	14499.04	14639.70	14645.90	69.32
Li23	640	1400	2200	-	-	18801.13	18807.12	18831.29	40.74
Li24	720	1500	2400	-	-	21389.43	21465.67	21505.93	35.66
Li25	760	900	900	-	-	16668.51	16930.18	16969.78	157.58
Li26	800	1700	2500	-	-	23977.73	24147.03	24178.73	11.40
Li27	840	900	900	-	-	17372.64	17699.17	17728.88	177.42
Li28	880	1800	2800	-	-	26566.03	26785.23	26823.61	107.59
Li29	960	2000	3000	-	-	29154.34	29187.10	29277.73	145.91
Li30	1040	2100	3200	-	-	31742.64	31777.80	31930.20	163.08
Li31	1120	2300	3500	-	-	34330.94	34389.28	34521.24	244.08
Li32	1200	2500	3600	-	-	37159.41	37399.73	37662.64	278.04

Appendix C

Detailed Results for CVRP Variants

Detailed results of 10 runs per instance are summarized in Tables C1 to C8. The instance name and best known solution (BKS) is provided for each problem in addition to, where applicable, the following problem characteristics: the number of customers n, vehicle capacity Q, maximum tour duration D, service time e (if equal for all customers), number of depots G, number of vehicles m (in each depot), number of linehaul l and backhaul b customers. Average calculation times (Time) and average times for finding the final solutions (Time*) are reported in minutes. Both times are provided since it remains unclear which is published by the authors of many of the compared methods. Best results are highlighted in gray and new best results obtained by SISRs are underlined.

MDVRP (Results: Table C1)
The benchmark set employed for MDVRP was introduced by Cordeau et al. (1997) with problem sizes varying between 48 and 360. The set is subdivided into three blocks. The first block of problems have no tour duration limits and therefore service time for each customer is zero. Route duration for the second block is limited, while customers have no service time. The third block of problems specify a route duration limit and an individual service time for each customer. Results are compared against those achieved by ALNS 50K (Pisinger and Røpke 2007), HGSADC (Vidal et al. 2012) and ILS-RVND-SP (Subramanian et al. 2013).

VRPB (Results: Table C2 and C3)
Goetschalckx and Jacobs-Blecha (1989) introduced a benchmark set for VRPB which contains problems containing a small number of customers, ranging from 25 to 150. Results are compared against those attained by MACS (Gajpal and Abad 2009), APA (Zachariadis and Kiranoudis 2012) and UHGS (Vidal et al. 2014).

VRPMDP and VRPSDP (Results: Table C4 and C5)
Benchmark instances employed for both problems are introduced by Salhi and Nagy (1999) and were reproduced using the instructions supplied. The authors based their single depot instances on those introduced by Christofides et al. (1979) which have problem sizes from 50

to 199 customers. The problems are divided into two groups: the first with and the second without tour duration and service time constraints. Results are compared against those produced by ILS-RVND-SP (Subramanian et al. 2013) and UHGS (Vidal et al. 2014).

MDVRPMDP (Results: Table C6)
The benchmark set introduced by Salhi and Nagy (1999) was employed and reproduced accordance with their instructions. This benchmark set is based on the Multi-Depot instances introduced by Gillett and Johnson (1976) with the number of customers ranging from 50 to 249 and the number of depots from 2 to 5. MDVRPMDP results are compared against those achieved by LNS (6R no learning) (Røpke and Pisinger 2006b) and ILS-RVND-SP (Subramanian et al. 2013).

OVRP (Results: Table C7)
Three benchmark sets were employed for OVRP: Christofides et al. (1979), Fisher (1994) and Golden et al. (1998) with the number of customers ranging from 50 to 480 customers. Results are compared against those attained by ILS-RVND-SP (Subramanian et al. 2013) and UHGS (Vidal et al. 2014).

CCVRP (Results: Table C8)
Benchmark sets introduced by Christofides et al. (1979) and Golden et al. (1998) with problem sizes from 50 to 483 customers were employed. Results are compared against those produced by UHGS (Vidal et al. 2014).

The best results of five runs per instance are summerized in Tables C9 and C10. Results are divided in sets of problem sizes n 200, 400, 600, 800, and 1000. For each problem set, the cumulative number of vehicles (CNV), the cumulative travel distance (CTD) and calulation average calculation time (T) in minutes are reported. Best results are marked gray.

VRPTW (Results: Table C9)
The benchmark set introduced by Gehring and Homberger (2001) was employed with the customers ranging from 200 to 1000. The number of iterations is set to 25×10^6, 50×10^6, 75×10^6, 100×10^6 and 125×10^6 for small to large-scale instances. The best of five runs per instance are compared against those of RTI (Repoussis et al. 2009), NB-100 and NB-f(n) (Nagata et al. 2010) and HGSADC (Vidal et al. 2013).

PDPTW (Results: Table C10 and C11)
The benchmark set introduced by Li and Lim (2001) was employed with problem sizes varying between 200 and 1000. The number of iterations is set to 6×10^6, 12×10^6, 18×10^6, 24×10^6 and 30×10^6 for small to large-scale instances. The best result of five runs per instance are compared against those attained by BH (Bent and Hentenryck 2006), RP (Røpke and Pisinger 2006a), NK (Nagata and Kobayashi 2010) and CLS (Curtois et al. 2018) (Table C10). Best results obtained during all experiments can be found in Table C11.

DARP (Results: Table C12)
The benchmark set introduced by Cordeau and Laporte (2003) was employed with problem sizes varying between 48 and 288. The number of iterations is set to 3×10^6, 3.7×10^6, 4.4×10^6, 5.8×10^6, 7.2×10^6, 7.9×10^6, 8.6×10^6 and 10×10^6 for problem sizes 48, 72, 96, 144, 192, 216, 240 and 288 respectively. Results of five runs are compared against those achieved by ELS (Chassaing et al. 2016), HGA (Masmoudi et al. 2017) and H-ALNS (Gschwind and Drexl 2019).

Table C1 Results for the MDVRP, benchmark set of Cordeau et al. (1997)

							ALNS 50K[a]		HGSADC[b]		ILS-RVND-SP[c]			SISRs			
Inst.	n	Q	D	m	G	BKS	Avg.	Time	Avg.	T	Best	Avg.	T	Best	Avg.	T*	T
p01	50	80	-	4	4	576.87	576.87	0.48	576.87	0.23	576.87	576.87	0.05	576.87	576.87	0.12	1.32
p02	50	160	-	2	4	473.53	473.53	0.47	473.53	0.21	473.53	473.53	0.04	473.53	473.53	0.09	1.22
p03	75	140	-	3	5	641.19	641.19	1.07	641.19	0.43	641.19	641.19	0.12	641.19	641.19	0.40	2.54
p12	80	60	-	5	2	1318.95	1319.13	1.25	1318.95	0.52	1318.95	1318.95	0.10	1318.95	1318.95	0.00	3.17
p04	100	100	-	8	2	1001.04	1006.09	1.47	1001.23	1.94	1001.04	1001.04	0.86	1001.04	1001.16	3.08	3.28
p05	100	200	-	5	2	750.03	752.34	2.00	750.03	1.06	750.03	750.21	0.53	750.03	750.19	3.29	4.72
p06	100	100	-	6	3	876.50	883.01	1.55	876.5	1.14	876.5	876.5	0.43	876.5	876.5	2.12	3.70
p07	100	100	-	4	4	881.97	889.36	1.47	884.43	1.55	881.97	881.97	0.36	881.97	881.97	2.35	3.52
p15	160	60	-	5	4	2505.42	2519.64	4.22	2505.42	1.92	2505.42	2505.42	0.81	2505.42	2505.42	0.05	9.99
p18	240	60	-	5	6	3702.85	3736.53	6.98	3702.85	4.52	3702.85	3702.85	17.00	3702.85	3702.85	2.30	20.39
p21	360	60	-	5	9	5474.84	5501.58	9.70	5476.41	10.00	5474.84	5474.84	42.41	5474.84	5476.61	11.48	41.26
p13	80	60	2000	5	2	1318.95	1318.95	1.00	1318.95	0.57	1318.95	1318.95	0.05	1318.95	1318.95	0.08	3.22
p14	80	60	1800	5	2	1360.12	1360.12	0.97	1360.12	0.55	1360.12	1360.12	0.32	1360.12	1360.12	0.16	3.25
p16	160	60	2000	5	4	2572.23	2573.95	3.13	2572.23	1.97	2572.23	2572.23	4.13	2572.23	2572.23	0.14	12.28
p17	160	60	1800	5	4	2709.09	2709.09	2.98	2709.09	2.14	2709.09	2710.21	24.14	2709.09	2709.09	0.11	12.36
p19	240	60	2000	5	6	3827.06	3838.76	5.25	3827.06	4.20	3827.06	3827.55	20.24	3827.06	3827.06	0.91	28.67
p20	240	60	1800	5	6	4058.07	4064.76	5.00	4058.07	4.37	4058.07	4058.07	9.08	4058.07	4058.07	0.48	31.93
p08	249	500	3100	14	2	4372.78	4421.03	5.55	4397.42	10.00	4379.46	4393.7	20.74	4375.49	4381.91	16.40	18.90
p09	249	500	3100	12	3	3858.66	3892.5	6.02	3868.59	9.50	3859.54	3864.22	23.86	3859.54	3870.47	15.51	22.39
p10	249	500	3100	8	4	3631.11	3666.85	6.05	3636.08	9.82	3631.37	3634.72	23.71	3636.47	3647.09	18.83	21.35
p11	249	500	3100	6	5	3546.06	3573.23	5.95	3548.25	7.14	3546.06	3546.15	20.29	3546.06	3547.45	11.12	23.55
p22	360	60	2000	5	9	5702.16	5722.19	7.70	5702.16	10.00	5702.15	5705.84	14.10	5702.16	5703.7	3.67	65.34
p23	360	60	1800	5	9	6078.75	6092.66	7.38	6078.75	10.00	6078.75	6078.75	16.99	6078.75	6078.75	1.69	72.99
pr01	48	200	5000	1	4	861.32	861.32	0.50	861.32	0.17	861.32	861.32	0.02	861.32	861.32	0.03	1.55
pr07	72	200	5000	1	6	1089.56	1089.56	0.97	1089.56	0.34	1089.56	1089.56	0.06	1089.56	1089.56	0.11	2.83
pr02	96	195	4800	2	4	1307.34	1308.17	1.72	1307.34	0.76	1307.34	1308.53	0.21	1307.34	1307.34	0.32	4.68
pr03	144	190	4600	3	4	1803.80	1810.66	3.57	1803.8	1.91	1803.81	1804.09	0.92	1803.8	1803.8	7.79	10.02
pr08	144	190	4750	2	6	1664.85	1675.74	3.45	1665.05	2.05	1664.85	1665.08	6.57	1664.85	1666.55	2.55	8.15
pr04	192	185	4400	4	4	2058.31	2073.16	4.93	2059.36	5.22	2058.31	2060.93	12.99	2058.31	2058.76	7.81	15.82
pr09	216	180	4500	3	6	2133.20	2144.84	5.83	2134.17	6.10	2133.2	2135.37	17.84	2133.2	2135.05	13.13	18.10
pr05	240	180	4200	5	4	2331.20	2350.31	6.20	2340.29	9.56	2331.2	2338.12	22.29	2331.2	2338.19	18.75	21.56
pr06	288	175	4000	6	4	2676.30	2695.74	7.75	2681.93	10.00	2680.77	2685.23	38.29	2680.74	2686.77	30.15	31.61
pr10	288	170	4250	4	6	2868.26	2905.43	7.58	2886.59	10.00	2874.28	2882.41	50.16	**2867.46**	2873.8	23.63	28.91
							0.44	3.94	0.08	4.24	0.02	0.07	11.81	0.01	0.07	6.02	16.81

[a] ALNS 50K (Pisinger and Røpke 2007)

[b] HGSADC (Vidal et al. 2012)

[c] ILS-RVND-SP (Subramanian et al. 2013)

Best results produced by the methods considered are highlighted in gray, new best solutions obtained by SISRs are underlined.

Table C2 Results for the VRPB, benchmark set of Goetschalckx and Jacobs-Blecha (1989) - part 1

							MACS[a]			APA[b]			UHGS[c]			SISRs			
Inst.	n	Q	l	b	m	BKS	Best	Avg.	T	Best	Avg.	T	Best	Avg.	T	Best	Avg.	T*	T
A1	25	1550	20	5	8	229.89	229.89	229.89	0.02	229.89	229.89	0.08	229.89	229.89	0.11	229.89	229.89	0.00	0.60
A2	25	2550	20	5	5	180.12	180.12	180.12	0.03	180.12	180.12	0.05	180.12	180.12	0.12	180.12	180.12	0.00	0.62
A3	25	4050	20	5	4	163.41	163.41	163.41	0.05	163.41	163.41	0.03	163.41	163.41	0.13	163.41	163.41	0.00	0.55
A4	25	4050	20	5	3	155.80	155.80	155.80	0.03	155.80	155.80	0.05	155.80	155.80	0.16	155.80	155.80	0.00	0.55
B1	30	1600	20	10	7	239.08	239.08	239.08	0.04	239.08	239.08	0.23	239.08	239.08	0.14	239.08	239.08	0.00	0.78
B2	30	2600	20	10	5	198.05	198.05	198.05	0.04	198.05	198.05	0.20	198.05	198.05	0.15	198.05	198.05	0.00	0.82
B3	30	4000	20	10	3	169.37	169.37	169.37	0.03	169.37	169.37	0.08	169.37	169.37	0.18	169.37	169.37	0.00	0.70
C1	40	1800	20	20	7	250.56	250.56	250.56	0.06	250.56	250.56	0.37	250.56	250.56	0.22	250.56	250.56	0.01	1.05
C2	40	2600	20	20	5	215.02	215.02	215.02	0.07	215.02	215.02	0.33	215.02	215.02	0.24	215.02	215.02	0.00	1.08
C3	40	4150	20	20	5	199.35	199.35	199.35	0.08	199.35	199.35	0.20	199.35	199.35	0.23	199.35	199.35	0.00	1.16
C4	40	4150	20	20	4	195.37	195.37	195.37	0.06	195.37	195.37	0.15	195.37	195.37	0.24	195.37	195.37	0.00	1.19
D1	38	1700	30	8	12	322.53	322.53	322.53	0.10	322.53	322.53	0.30	322.53	322.53	0.18	322.53	322.53	0.01	1.08
D2	38	1700	30	8	11	316.71	316.71	316.71	0.10	316.71	316.71	0.25	316.71	316.71	0.17	316.71	316.71	0.00	1.01
D3	38	2750	30	8	7	239.48	239.48	239.48	0.09	239.48	239.48	0.17	239.48	239.48	0.19	239.48	239.48	0.00	0.97
D4	38	4075	30	8	5	205.83	205.83	205.83	0.11	205.83	205.83	0.18	205.83	205.83	0.24	205.83	205.83	0.00	1.03
E1	45	2650	30	15	7	238.88	238.88	238.88	0.11	238.88	238.88	0.45	238.88	238.88	0.27	238.88	238.88	0.00	1.17
E2	45	4300	30	15	4	212.26	212.26	212.26	0.11	212.26	212.26	0.33	212.26	212.26	0.32	212.26	212.26	0.00	1.27
E3	45	5225	30	15	4	206.66	206.66	206.66	0.17	206.66	206.66	0.37	206.66	206.66	0.36	206.66	206.66	0.00	1.29
F1	60	3000	30	30	6	263.17	263.17	263.17	0.19	263.17	263.27	0.57	263.17	263.17	0.38	263.17	263.17	0.02	1.82
F2	60	3000	30	30	7	265.21	265.21	265.21	0.15	265.21	265.66	0.60	265.21	265.21	0.39	265.21	265.21	0.01	2.01
F3	60	4400	30	30	5	241.12	241.12	241.48	0.19	241.12	241.12	0.47	241.12	241.12	0.49	241.12	241.12	0.11	2.12
F4	60	5500	30	30	4	233.86	233.86	233.86	0.25	233.86	234.60	0.53	233.86	233.86	0.54	233.86	233.86	0.00	2.17
G1	57	2700	45	12	10	306.31	306.31	306.54	0.30	306.98	306.31	0.63	-	-	-	306.31	306.31	0.03	1.65
G2	57	4300	45	12	6	245.44	245.44	245.44	0.17	245.44	245.44	0.53	245.44	245.44	0.38	245.44	245.44	0.00	1.74
G3	57	5300	45	12	5	229.51	229.51	229.51	0.24	229.51	229.97	0.52	229.51	229.51	0.43	229.51	229.51	0.03	1.75
G4	57	5300	45	12	6	232.52	232.52	232.52	0.36	232.52	232.52	0.63	232.52	232.52	0.45	232.52	232.52	0.10	1.86
G5	57	6400	45	12	5	221.73	221.73	221.73	0.34	221.73	222.87	0.53	221.73	221.73	0.46	221.73	221.73	0.06	2.00
G6	57	8000	45	12	4	213.46	213.46	213.46	0.34	213.46	214.38	0.42	213.46	213.46	0.54	213.46	213.46	0.01	2.08
H1	68	4000	45	23	6	268.93	268.93	269.00	0.41	268.93	270.06	0.63	268.93	268.93	0.62	268.93	268.93	0.01	2.10
H2	68	5100	45	23	5	253.37	253.37	253.37	0.36	253.37	253.91	0.58	253.37	253.37	0.57	253.37	253.37	0.01	2.25
H3	68	6100	45	23	4	247.45	247.45	247.45	0.34	247.45	247.45	0.60	247.45	247.45	0.64	247.45	247.45	0.01	2.55
H4	68	6100	45	23	5	250.22	250.22	250.22	0.45	250.22	251.09	0.67	250.22	250.22	0.59	250.22	250.22	0.02	2.59
H5	68	7100	45	23	4	246.12	246.12	246.12	0.43	246.12	246.12	0.58	246.12	246.12	0.62	246.12	246.12	0.03	2.58
H6	68	7100	45	23	5	249.14	249.14	249.14	0.50	249.14	250.06	0.70	249.14	249.14	0.59	249.14	249.14	0.02	2.67

[a] MACS (Gajpal and Abad 2009)

[b] APA (Zachariadis and Kiranoudis 2012)

[c] UHGS (Vidal et al. 2014)

Best results produced by the methods considered are highlighted in gray.

Table C3 Results for the VRPB, benchmark set of Goetschalckx and Jacobs-Blecha (1989) - part 2

							MACS[a]			APA[b]			UHGS[c]			SISRs			
Inst.	n	Q	l	b	m	BKS	Best	Avg.	T	Best	Avg.	T	Best	Avg.	T	Best	Avg.	T*	T
I1	90	3000	45	45	10	350.25	350.25	350.40	0.69	350.25	351.08	1.12	350.25	350.37	0.89	350.25	350.25	0.88	3.48
I2	90	4000	45	45	7	309.94	309.94	310.32	0.63	309.94	309.98	1.02	309.94	309.94	0.85	309.94	309.94	0.02	3.68
I3	90	5700	45	45	5	294.51	294.51	294.84	0.73	294.51	294.79	0.92	294.51	294.51	0.99	294.51	294.51	0.01	3.91
I4	90	5700	45	45	6	295.99	295.99	296.13	0.78	295.99	297.91	1.10	295.99	295.99	0.92	295.99	295.99	0.02	4.38
I5	90	5700	45	45	7	301.24	301.24	301.83	0.79	301.24	303.49	1.20	301.24	301.24	0.82	301.24	301.24	0.17	4.45
J1	94	4400	75	19	10	335.01	335.01	335.12	1.11	335.01	335.78	1.62	335.01	335.01	0.83	335.01	335.01	0.29	3.51
J2	94	5600	75	19	8	310.42	310.42	310.42	0.99	310.42	312.51	1.50	310.42	310.42	0.84	310.42	310.42	0.52	3.94
J3	94	8200	75	19	6	279.22	279.22	279.34	1.29	279.22	280.43	1.30	279.22	279.22	0.93	279.22	279.22	0.09	4.17
J4	94	6600	75	19	7	296.53	296.53	296.58	1.05	296.53	298.32	1.47	296.53	296.53	1.12	296.53	296.53	1.10	4.22
K1	113	4100	75	38	10	394.07	394.07	396.14	1.75	394.07	397.38	1.68	394.07	394.35	1.33	394.07	394.09	2.72	5.27
K2	113	5200	75	38	8	362.13	362.13	362.56	1.61	362.13	365.46	1.37	362.13	362.13	1.40	362.13	362.13	0.89	5.32
K3	113	5200	75	38	9	365.69	365.69	366.71	1.90	365.69	369.44	1.50	365.69	365.69	1.30	365.69	365.69	0.80	5.68
K4	113	6200	75	38	7	348.95	348.95	350.32	1.77	348.95	349.72	1.45	348.95	348.95	1.29	348.95	348.95	0.54	6.09
L1	150	**4400**	75	75	10	417.90	417.90	420.06	2.47	417.90	421.68	2.75	417.90	418.16	3.94	417.90	417.90	3.91	9.34
L2	150	5000	75	75	8	401.23	401.23	401.36	2.37	401.23	405.20	2.27	401.23	401.23	2.96	401.23	401.23	2.87	9.42
L3	150	5000	75	75	9	402.68	402.68	404.32	2.68	402.68	405.76	2.88	402.68	402.68	2.73	402.68	402.68	2.69	9.47
L4	150	6000	75	75	7	384.64	384.64	384.83	2.66	384.64	388.14	2.33	384.64	384.64	2.35	384.64	384.64	0.49	10.69
L5	150	6000	75	75	8	387.56	387.56	390.33	2.63	387.57	390.46	2.57	387.56	387.56	2.72	387.56	387.56	0.93	10.62
M1	125	5200	100	25	11	398.59	398.59	399.12	3.83	398.59	400.50	2.20	398.59	398.66	1.58	398.59	398.59	4.07	6.05
M2	125	5200	100	25	10	396.92	396.92	398.16	3.06	396.92	401.91	2.00	396.92	396.93	2.39	396.92	397.24	4.11	5.68
M3	125	6200	100	25	9	375.70	375.70	377.81	3.06	375.70	378.07	1.77	375.70	375.93	2.70	375.70	375.76	4.54	6.91
M4	125	8000	100	25	7	348.14	348.14	348.46	2.74	348.14	352.03	1.57	348.14	348.20	1.72	348.14	348.31	3.18	7.42
N1	150	5700	100	50	11	408.10	408.10	408.17	3.56	408.10	411.72	3.20	408.10	408.10	2.72	408.10	408.10	4.49	9.31
N2	150	5700	100	50	10	408.07	408.07	408.25	3.57	408.07	412.31	2.82	408.07	408.13	2.55	408.07	408.07	4.48	9.09
N3	150	6600	100	50	9	394.34	394.34	394.70	3.33	394.34	398.76	2.40	394.34	394.94	2.46	394.34	394.34	2.95	10.58
N4	150	6600	100	50	10	394.79	394.79	394.87	3.65	394.79	396.16	3.27	394.79	395.13	2.37	394.79	394.79	2.89	10.63
N5	150	8500	100	50	7	373.48	373.48	374.12	4.53	373.48	376.90	2.53	373.48	373.55	3.12	373.48	373.48	2.89	10.54
N6	150	8500	100	50	8	373.76	373.76	374.79	4.25	373.76	379.78	2.83	373.76	373.76	3.42	373.76	373.76	3.46	10.02
							0.00	0.09	1.13	0.00	0.38	1.09	0.00	0.01	1.52	0.00	0.00	0.91	3.95

[a] MACS (Gajpal and Abad 2009)

[b] APA (Zachariadis and Kiranoudis 2012)

[c] UHGS (Vidal et al. 2014)

Best results produced by the methods considered are highlighted in gray.

Table C4 Results for the VRPMDP, benchmark set of Salhi and Nagy (1999)

						ILS-RVND-SP[a]			UHGS[b]			SISRs			
Inst.	n	Q	D	e	BKS	Best	Avg.	T	Best	Avg.	T	Best	Avg.	T*	T
CMT01H	50	160	-	-	3 465.02	465.02	465.03	0.03	465.02	465.02	0.69	465.02	465.02	0.29	2.50
CMT01Q	50	160	-	-	4 489.74	489.74	489.74	0.03	489.74	489.74	0.55	489.74	489.74	0.00	2.30
CMT01T	50	160	-	-	5 520.06	520.06	520.06	0.03	520.06	520.06	0.52	520.06	520.06	0.00	2.07
CMT02H	75	140	-	-	6 662.63	662.63	662.63	0.09	662.63	662.63	1.02	662.63	662.63	0.45	4.02
CMT02Q	75	140	-	-	8 731.26	731.26	731.40	0.13	731.26	731.45	1.46	731.26	731.62	1.85	3.57
CMT02T	75	140	-	-	9 782.77	782.77	782.77	0.13	782.77	782.77	0.83	782.77	782.77	1.30	3.11
CMT03H	100	200	-	-	4 700.94	700.94	700.94	0.29	700.94	700.94	1.94	700.94	700.94	4.78	7.43
CMT03Q	100	200	-	-	6 747.15	747.15	747.15	0.16	747.15	747.15	1.33	747.15	747.15	2.06	7.87
CMT03T	100	200	-	-	7 798.07	798.07	798.07	0.48	798.07	798.07	1.62	798.07	798.65	3.95	6.17
CMT12H	100	200	-	-	5 629.37	629.37	629.37	0.23	629.37	629.37	1.49	629.37	629.37	1.78	7.27
CMT12Q	100	200	-	-	7 729.25	729.25	729.25	0.29	729.25	729.25	1.53	729.25	729.54	2.24	6.13
CMT12T	100	200	-	-	9 787.52	787.52	787.52	0.11	787.52	787.52	0.95	787.52	787.52	0.42	5.86
CMT11H	120	200	-	-	4 818.05	818.05	818.06	1.05	818.05	818.05	4.53	818.05	818.05	5.28	11.56
CMT11Q	120	200	-	-	6 939.36	939.36	939.36	0.34	939.36	939.36	2.99	939.36	939.36	8.20	10.20
CMT11T	120	200	-	-	7 998.80	998.80	998.81	0.33	998.80	998.80	2.34	998.80	998.81	6.76	8.46
CMT04H	150	200	-	-	6 828.12	828.12	831.59	1.34	828.12	829.64	5.67	828.12	829.81	9.62	16.01
CMT04Q	150	200	-	-	9 913.93	915.27	915.27	0.98	915.27	915.46	4.18	915.27	915.31	11.41	16.54
CMT04T	150	200	-	-	11 990.39	990.39	990.39	0.84	990.39	990.39	2.50	990.42	992.53	12.18	14.47
CMT05H	199	200	-	-	9 978.74	978.74	978.74	25.53	978.74	980.58	4.95	978.74	981.79	19.49	28.04
CMT05Q	199	200	-	-	12 1104.87	1104.87	1105.79	27.13	1104.87	1106.95	5.79	1106.39	1108.09	21.14	25.44
CMT05T	199	200	-	-	15 1218.77	1218.77	1220.24	30.05	1218.77	1220.53	5.96	1222.29	1222.96	20.12	21.86
CMT06H	50	160	200	10	6 555.43	555.43	557.35	0.02	555.43	555.43	0.45	555.43	555.43	0.01	2.73
CMT06Q	50	160	200	10	6 555.43	555.43	557.15	0.02	555.43	555.43	0.45	555.43	555.43	0.01	2.51
CMT06T	50	160	200	10	6 555.43	555.43	556.64	0.02	555.43	555.43	0.46	555.43	555.43	0.01	2.52
CMT07H	75	140	160	10	11 900.12	900.54	900.84	0.07	900.12	900.59	1.03	900.12	900.33	2.78	5.59
CMT07Q	75	140	160	10	11 900.69	900.69	902.62	0.08	900.69	901.09	1.19	900.69	900.69	2.71	5.27
CMT07T	75	140	160	10	11 903.05	903.05	903.05	0.08	903.05	903.05	0.73	903.05	903.05	0.48	4.57
CMT08H	100	200	230	10	9 865.50	865.50	865.50	0.13	865.50	865.50	1.31	865.50	865.50	1.78	9.65
CMT08Q	100	200	230	10	9 865.50	865.50	865.50	0.13	865.50	865.50	1.32	865.50	865.50	1.62	9.46
CMT08T	100	200	230	10	9 865.54	865.54	865.54	0.12	865.54	865.54	1.31	865.54	865.54	2.05	8.60
CMT14H	100	200	1040	90	10 821.75	821.75	821.75	0.09	821.75	821.75	0.96	821.75	821.75	0.97	9.38
CMT14Q	100	200	1040	90	10 821.75	821.75	821.75	0.09	821.75	821.75	0.97	821.75	821.75	0.68	9.11
CMT14T	100	200	1040	90	10 826.77	826.77	826.77	0.11	826.77	826.77	1.15	826.77	826.77	0.95	7.70
CMT13H	120	200	720	50	11 1542.86	1542.86	1544.54	1.23	1542.86	1542.86	2.81	1542.86	1542.88	11.95	13.20
CMT13Q	120	200	720	50	11 1542.86	1542.86	1544.05	1.16	1542.86	1542.86	2.85	1542.86	1542.92	11.49	12.96
CMT13T	120	200	720	50	11 1542.86	1542.86	1544.11	1.23	1542.86	1542.95	2.86	**1541.14**	1542.69	11.38	12.75
CMT09H	150	200	200	10	14 1160.68	1160.68	1162.17	1.30	1160.68	1160.68	3.66	1160.68	1160.68	20.40	23.91
CMT09Q	150	200	200	10	14 1161.24	1161.24	1161.69	1.34	1161.24	1161.32	3.46	1161.24	1161.24	15.46	23.18
CMT09T	150	200	200	10	14 1162.55	1162.55	1164.37	1.39	1162.55	1162.55	3.68	1162.55	1162.93	18.90	21.96
CMT10H	199	200	200	10	17 1372.47	1372.52	1377.23	9.17	1372.47	1381.47	6.56	**1372.20**	1372.75	40.37	45.22
CMT10Q	199	200	200	10	17 1374.18	1374.18	1379.47	8.96	1374.18	1378.41	7.91	1376.15	1379.40	38.50	43.98
CMT10T	199	200	200	10	18 1381.04	1381.04	1388.17	8.36	1381.04	1390.82	5.49	1381.04	1387.80	32.67	35.96
						0.00	0.09	2.97	0.00	0.06	2.46	0.01	0.06	8.30	12.41

[a] ILS-RVND-SP (Subramanian et al. 2013)

[b] UHGS (Vidal et al. 2014)

Best results produced by the methods considered are highlighted in gray, new best solutions obtained by SISRs are underlined.

Table C5 Results for the VRPSDP, benchmark set of Salhi and Nagy (1999)

							ILS-RVND-SP[a]			UHGS[b]			SISRs			
Inst.	n	Q	D	e		BKS	Best	Avg.	T	Best	Avg.	T	Best	Avg.	T*	T
CMT01X	50	160	-	-	3	466.77	466.77	466.77	0.03	466.77	466.77	0.72	466.77	466.77	0.01	2.24
CMT01Y	50	160	-	-	3	466.77	466.77	466.77	0.03	466.77	466.77	0.71	466.77	466.77	0.01	2.26
CMT02X	75	140	-	-	6	684.21	684.21	684.78	0.21	684.21	684.43	1.32	684.21	684.21	1.80	3.43
CMT02Y	75	140	-	-	6	684.21	684.21	684.59	0.18	684.21	684.36	1.35	684.21	684.24	1.68	3.49
CMT03X	100	200	-	-	5	721.27	721.27	721.46	0.29	721.27	721.27	1.69	721.27	722.69	4.78	8.26
CMT03Y	100	200	-	-	5	721.27	721.27	721.50	0.29	721.27	721.27	1.79	721.27	722.33	5.41	8.10
CMT12X	100	200	-	-	5	662.22	662.22	663.44	0.15	662.22	662.22	1.74	662.22	662.22	2.12	6.57
CMT12Y	100	200	-	-	5	662.22	662.22	663.12	0.16	662.22	662.22	1.69	662.22	662.22	2.19	6.53
CMT11X	120	200	-	-	4	833.92	846.23	848.65	0.86	833.92	833.92	2.75	846.23	846.23	6.42	10.42
CMT11Y	120	200	-	-	4	833.92	846.23	848.74	0.81	833.92	833.92	2.66	833.92	845.00	6.07	10.39
CMT04X	150	200	-	-	7	852.46	852.46	853.02	1.63	852.46	852.65	4.16	852.46	852.50	11.37	15.72
CMT04Y	150	200	-	-	7	852.46	852.46	852.73	1.34	852.46	852.72	3.95	852.46	852.50	11.22	15.89
CMT05X	199	200	-	-	10	1029.25	1029.25	1029.52	29.78	1029.25	1029.60	7.99	1029.25	1029.28	20.18	25.21
CMT05Y	199	200	-	-	10	1029.25	1029.25	1029.25	28.77	1029.25	1029.79	6.60	1029.25	1029.27	21.28	25.16
							0.21	0.30	4.61	0.00	0.01	2.79	0.11	0.23	6.75	10.26
CMT06X	50	160	200	10	6	555.43	555.43	557.35	0.02	-	-	-	555.43	555.43	0.01	2.69
CMT06Y	50	160	200	10	6	555.43	555.43	557.10	0.02	-	-	-	555.43	555.43	0.01	2.71
CMT07X	75	140	160	10	11	900.12	900.12	901.02	0.08	-	-	-	900.12	900.29	3.06	5.31
CMT07Y	75	140	160	10	11	900.12	900.12	901.08	0.08	-	-	-	900.12	900.20	3.25	5.30
CMT08X	100	200	230	10	9	865.50	865.50	865.50	0.12	-	-	-	865.50	865.50	1.61	9.48
CMT08Y	100	200	230	10	9	865.50	865.50	865.50	0.13	-	-	-	865.50	865.50	2.06	9.48
CMT14X	100	200	1040	90	10	821.75	821.75	821.75	0.09	-	-	-	821.75	821.75	0.78	9.28
CMT14Y	100	200	1040	90	10	821.75	821.75	821.75	0.09	-	-	-	821.75	821.75	0.89	9.23
CMT13X	120	200	720	50	11	1542.86	1542.86	1543.54	1.15	-	-	-	1542.86	1542.88	11.58	12.79
CMT13Y	120	200	720	50	11	1542.86	1542.86	1544.42	1.22	-	-	-	1542.86	1542.88	11.51	12.72
CMT09X	150	200	200	10	14	1160.68	1160.68	1161.77	1.07	-	-	-	1160.68	1160.68	19.03	23.94
CMT09Y	150	200	200	10	14	1160.68	1160.68	1162.59	1.35	-	-	-	1160.68	1160.68	19.95	23.83
CMT10X	199	200	200	10	17	1373.40	1373.40	1379.19	9.21	-	-	-	1374.00	1376.47	39.46	43.36
CMT10Y	199	200	200	10	17	1373.40	1373.40	1377.03	9.12	-	-	-	1373.72	1375.97	40.18	41.58
							0.00	0.14	1.70				0.00	0.03	10.96	15.12

[a] ILS-RVND-SP (Subramanian et al. 2013)

[b] UHGS (Vidal et al. 2014)

Best results produced by the methods considered are highlighted in gray.

Table C6 Results for the MDVRPMDP, benchmark set of Salhi and Nagy (1999)

							ILS-RVND-SP[a]			SISRs			
Inst.	n	Q	D	m	G	BKS	Best	Avg.	T	Best	Avg.	T*	T
GJ01H	50	80	-	4	4	499.12	-	-	-	499.12	499.12	0.02	2.44
GJ01Q	50	80	-	4	4	528.30	528.30	528.30	0.05	528.30	528.30	0.32	2.44
GJ01T	50	80	-	4	4	569.43	569.43	569.43	0.05	569.43	569.43	0.01	2.17
GJ02H	50	160	-	2	4	440.00	440.00	440.00	0.05	440.00	440.00	0.01	2.24
GJ02Q	50	160	-	2	4	449.72	449.72	449.72	0.04	449.72	449.72	0.01	2.34
GJ02T	50	160	-	2	4	464.13	464.13	464.13	0.04	464.13	464.13	0.00	2.17
GJ03H	75	140	-	3	5	579.45	579.45	579.45	0.16	579.45	579.45	0.52	5.06
GJ03Q	75	140	-	3	5	605.25	605.25	605.25	0.15	605.25	605.25	0.23	4.60
GJ03T	75	140	-	3	5	624.44	624.44	624.44	0.18	624.44	624.44	1.53	4.07
GJ04H	100	100	-	8	2	789.19	789.19	789.30	0.53	789.19	789.33	4.40	6.52
GJ04Q	100	100	-	8	2	874.78	874.78	874.79	0.70	874.78	874.78	3.69	5.87
GJ04T	100	100	-	8	2	962.25	962.25	962.65	0.63	962.25	964.20	3.20	4.92
GJ05H	100	200	-	5	2	676.81	676.81	676.91	0.37	676.81	676.81	4.54	8.38
GJ05Q	100	200	-	5	2	700.15	700.15	700.15	0.33	700.15	700.15	2.11	7.06
GJ05T	100	200	-	5	2	733.17	733.17	733.18	0.66	733.17	733.19	2.91	5.95
GJ06H	100	100	-	6	3	742.18	742.18	742.18	0.54	742.18	742.30	3.84	6.75
GJ06Q	100	100	-	6	3	793.85	793.85	793.87	0.42	793.85	793.85	2.53	6.17
GJ06T	100	100	-	6	3	850.82	850.82	850.82	0.45	850.82	850.82	2.84	5.28
GJ07H	100	100	-	4	4	732.73	732.73	732.73	0.41	732.73	732.73	1.91	7.28
GJ07Q	100	100	-	4	4	801.91	801.91	801.94	0.64	801.91	802.03	4.00	6.15
GJ07T	100	100	-	4	4	853.54	853.54	853.54	0.34	853.54	853.54	2.48	5.15
GJ08H	249	500	310	14	2	3320.39	3320.39	3342.91	23.92	**3303.39**	3317.32	21.59	36.89
GJ08Q	249	500	310	14	2	3745.18	3745.18	3769.01	21.48	**3732.56**	3734.17	18.06	34.66
GJ08T	249	500	310	14	2	4110.78	4110.78	4120.27	21.21	**4107.43**	4115.18	18.58	29.28
GJ09H	249	500	310	12	3	2990.92	2990.92	3005.52	24.64	**2988.24**	2991.52	13.00	40.18
GJ09Q	249	500	310	12	3	3351.18	3351.18	3361.23	22.70	3369.44	3377.57	8.15	38.69
GJ09T	249	500	310	12	3	3656.03	3656.03	3661.62	21.95	3657.81	3658.69	17.67	33.89
GJ10H	249	500	310	8	4	2894.71	2894.71	2905.23	24.21	**2890.84**	2898.84	20.82	44.45
GJ10Q	249	500	310	8	4	3220.79	3220.79	3226.79	21.93	3221.10	3233.11	15.49	41.78
GJ10T	249	500	310	8	4	3470.70	3470.70	3477.99	21.37	3478.11	3493.40	13.60	36.36
GJ11H	249	500	310	6	5	2842.79	2842.79	2845.71	22.63	**2840.21**	2843.71	17.36	43.21
GJ11Q	249	500	310	6	5	3138.64	3138.64	3143.33	21.12	3143.85	3152.85	24.01	42.74
GJ11T	249	500	310	6	5	3360.48	3360.48	3367.63	19.69	3365.39	3370.03	16.13	34.44
							0.00	0.12	8.55	0.00	0.08	7.44	16.96

[a] ILS-RVND-SP (Subramanian et al. 2013)

Best results produced by the methods considered are highlighted in gray, new best solutions obtained by SISRs are underlined.

Table C7 Results for the OVRP, benchmark set of Christofides et al. (1979), Fisher (1994), Golden et al. (1998)

							ILS-RVND-SP[a]			UHGS[b]			SISRs			
Inst.	n	Q	D	e		BKS	Best	Avg.	T	Best	Avg.	T	Best	Avg.	T*	T
CMT01	50	160	-	-	5	416.06	416.06	416.06	0.03	416.06	416.06	0.41	416.06	416.06	0.02	1.33
F11	71	30000	-	-	4	177.00	177.00	177.21	0.07	177.00	177.00	0.65	177.00	177.00	0.31	3.19
CMT02	75	140	-	-	10	567.14	567.14	567.14	0.11	567.14	568.15	0.51	567.14	567.14	0.60	2.26
CMT03	100	200	-	-	8	639.74	639.74	639.81	0.26	639.74	639.74	0.85	639.74	639.74	3.22	5.41
CMT12	100	200	-	-	10	534.24	534.24	534.24	0.10	534.24	534.24	0.61	534.24	534.24	1.90	5.27
CMT11	120	200	-	-	7	682.12	682.12	682.12	0.39	682.12	682.12	1.51	682.12	682.12	3.87	5.73
F12	134	2210	-	-	7	769.55	769.55	770.00	0.68	769.55	769.68	1.71	786.14	787.39	9.32	10.84
CMT04	150	200	-	-	12	733.13	733.13	733.13	0.46	733.13	733.13	1.73	733.13	733.13	9.17	11.63
CMT05	199	200	-	-	16	883.50	883.50	895.55	26.32	884.08	890.15	4.13	884.01	886.26	8.21	10.38
CMT06	50	160	180	10	6	412.96	412.96	412.96	0.02	412.96	412.96	0.55	412.96	412.96	0.01	1.87
CMT07	75	140	144	10	10	583.19	583.19	582.07	0.12	583.19	584.59	0.77	583.19	583.19	0.88	2.56
CMT08	100	200	207	10	9	644.63	644.63	644.95	0.15	644.63	644.79	1.79	644.63	644.63	3.70	6.70
CMT14	100	200	936	90	11	591.87	591.87	591.87	0.38	591.87	591.87	1.70	591.87	591.87	2.32	6.02
CMT13	120	200	648	50	11	899.16	899.16	904.02	0.62	899.16	900.22	3.39	902.25	909.17	7.99	9.51
CMT09	150	200	180	10	13	757.07	757.91	759.38	0.65	757.07	760.75	5.18	757.07	757.07	9.06	15.33
CMT10	199	200	180	10	17	874.71	874.71	877.68	7.88	874.71	875.49	6.10	874.71	874.71	14.97	18.62
GWKC5	200	900	-	-	5	6018.52	6018.52	6018.52	21.18	6018.52	6018.52	5.71	6018.52	6018.52	0.18	28.58
GWKC1	240	550	-	-	9	4543.00	4544.46	4551.74	20.79	4543.00	4546.35	8.20	4546.94	4556.00	16.26	24.34
GWKC6	280	900	-	-	7	7704.59	7721.16	7728.77	33.48	7704.59	7704.91	11.23	7705.97	7709.74	17.26	63.33
GWKC2	320	700	-	-	10	7213.56	7215.48	7229.56	44.39	7213.56	7218.41	14.94	7213.56	7216.62	37.02	56.31
GWKC7	360	900	-	-	8	9127.70	9180.93	9205.01	95.04	9127.70	9132.27	19.01	9127.70	9138.40	22.48	38.92
GWKC3	400	900	-	-	9	9750.63	9773.83	9784.52	84.43	9750.63	9763.39	22.66	9750.63	9757.72	45.04	82.49
GWKC8	440	900	-	-	10	10289.70	10326.57	10342.10	102.34	10289.70	10316.60	26.43	10293.87	10307.55	62.67	97.15
GWKC4	480	1000	-	-	10	12380.66	12389.43	12393.40	110.89	12380.66	12387.82	26.41	**12372.97**	12390.73	66.28	138.65
							0.07	0.22	22.95	0.00	0.11	6.92	0.12	0.20	14.28	26.93

[a] ILS-RVND-SP (Subramanian et al. 2013)

[b] UHGS (Vidal et al. 2014)

Best results produced by the methods considered are highlighted in gray, new best solutions obtained by SISRs are underlined.

Table C8 Results for the CCVRP, benchmark set of Christofides et al. (1979), Golden et al. (1998)

					UHGS[a]			SISRs			
Inst.	n	Q	m	BKS	Best	Avg.	T	Best	Avg.	T*	T
CMT1	50	160	5	2230.35	2230.35	2230.35	0.44	2230.35	2230.35	0.04	0.53
CMT2	75	140	10	2391.63	2391.63	2394.00	0.70	2391.63	2391.63	0.14	0.85
CMT3	100	200	8	4045.42	4045.42	4045.42	0.93	4045.42	4045.42	0.35	2.14
CMT4	150	200	12	4987.52	4987.52	4987.52	1.84	4987.52	4987.52	1.02	4.49
CMT5	199	200	17	5806.02	5806.02	5809.94	4.02	5806.02	5809.11	2.41	6.84
CMT11	120	200	7	7314.55	7314.55	7314.55	1.35	7314.55	7314.55	0.54	1.99
CMT12	100	200	10	3558.92	3558.93	3558.93	0.64	3558.92	3558.92	0.55	2.02
GWKC1	240	550	9	54739.85	54739.85	54742.20	7.40	54739.84	54742.20	6.66	11.21
GWKC2	320	700	10	100560.16	100560.16	100562.52	11.00	100560.17	100575.37	17.59	30.13
GWKC3	400	900	10	170923.53	170923.53	170964.42	23.95	170923.55	170958.03	28.38	49.82
GWKC4	480	1000	10	261993.33	261993.33	262044.19	26.14	261993.34	262031.30	32.15	62.89
GWKC5	200	900	5	114163.63	114163.63	114163.63	6.95	114163.63	114170.48	4.39	12.47
GWKC6	280	900	7	140430.08	140430.08	140430.08	12.65	140430.08	140430.78	11.65	23.88
GWKC7	360	900	8	178880.44	178880.44	178976.20	26.66	179292.14	179487.17	7.52	14.76
GWKC8	440	900	10	193659.14	193659.14	193683.21	26.07	193661.16	193738.98	26.42	42.48
GWKC9	255	1000	14	4722.06	4722.06	4724.01	6.97	4724.36	4725.23	9.33	11.59
GWKC10	323	1000	16	6713.26	6713.26	6720.04	10.30	**6713.12**	6715.38	17.82	20.55
GWKC11	399	1000	18	9214.07	9219.42	9222.92	14.61	**9212.90**	9214.70	29.69	33.73
GWKC12	483	1000	19	12500.52	12500.52	12516.98	28.66	**12488.05**	12492.29	25.03	31.75
GWKC13	252	1000	26	3627.45	3627.45	3632.63	7.16	3628.21	3630.74	4.50	7.11
GWKC14	320	1000	29	5187.56	5187.56	5206.53	16.25	5189.71	5198.27	5.39	7.75
GWKC15	396	1000	33	7005.47	7005.47	7015.51	18.70	**7001.25**	7011.07	10.56	14.20
GWKC16	480	1000	37	9239.10	9239.10	9247.68	27.28	**9236.44**	9241.93	23.17	30.42
GWKC17	240	200	22	3060.14	3060.14	3061.28	6.06	**3059.82**	3061.45	4.56	6.74
GWKC18	300	200	27	4199.43	4199.43	4211.80	11.64	4199.42	4204.36	4.80	6.85
GWKC19	360	200	33	5496.39	5496.39	5502.59	17.06	5500.05	5508.30	11.03	15.35
GWKC20	420	200	38	7184.19	7184.19	7188.59	21.82	7186.75	7190.16	13.45	16.29
					0.00	0.07	12.49	0.01	0.05	11.08	17.36

[a] UHGS (Vidal et al. 2014)

Best results produced by the methods considered are highlighted in gray, new best solutions obtained by SISRs are underlined.

Table C9 Results for the VRPTW, benchmark set of Gehring and Homberger (2001)

Inst.	n	RTI[a] Best 3	NB-100[b] 1 run	NB-f(n)[b] Best 5	HGSADC[c] Best 5	SISRs Best 5
R1	200	18.2 | 3640.11	18.2 | 3615.15	18.2 | 3612.36	18.2 | 3613.16	18.2 | 3619.20
R2	200	4.0 | 2941.99	4.0 | 2930.04	4.0 | 2929.41	4.0 | 2929.41	4.0 | 2939.40
C1	200	18.9 | 2721.90	18.9 | 2718.44	18.9 | 2718.41	18.9 | 2718.41	18.9 | 2718.50
C2	200	6.0 | 1833.36	6.0 | 1831.64	6.0 | 1831.64	6.0 | 1831.59	6.0 | 1837.80
RC1	200	18.0 | 3224.63	18.0 | 3182.48	18.0 | 3178.68	18.0 | 3180.48	18.0 | 3184.30
RC2	200	4.3 | 2554.33	4.3 | 2536.54	4.3 | 2536.22	4.3 | 2536.20	4.3 | 2545.60
CNV		694	694	694	694	694
CTD		169163	168143	168067	168092	168447
T		90	4.7	5x4.1	5x8.40	5x11.0
R1	400	36.4 | 8514.11	36.4 | 8413.23	36.4 | 8403.24	36.4 | 8402.57	36.4 | 8399.00
R2	400	8.0 | 6258.82	8.0 | 6149.49	8.0 | 6148.57	8.0 | 6152.92	8.0 | 6196.20
C1	400	37.6 | 7273.90	37.6 | 7179.71	37.6 | 7175.72	37.6 | 7170.47	37.6 | 7200.90
C2	400	11.7 | 3941.70	11.7 | 3898.02	11.7 | 3899.00	11.6 | 3952.95	11.8 | 3927.10
RC1	400	36.0 | 8088.46	36.0 | 7931.66	36.0 | 7922.23	36.0 | 7907.14	36.0 | 7906.60
RC2	400	8.4 | 5516.59	8.4 | 5293.74	8.4 | 5297.86	8.5 | 5215.21	8.4 | 5356.40
CNV		1381	1381	1381	1381	1382
CTD		395936	388548	388466	388013	389861
T		180	34.0	5x16.2	5x34.1	5x42.0
R1	600	54.5 | 18781.79	54.5 | 18194.38	54.5 | 18186.24	54.5 | 18023.18	54.5 | 17946.80
R2	600	11.0 | 12804.60	11.0 | 12319.75	11.0 | 12330.49	11.0 | 12352.38	11.0 | 12485.80
C1	600	57.3 | 14236.86	57.4 | 14054.70	57.4 | 14067.34	57.4 | 14058.46	57.4 | 14076.60
C2	600	17.4 | 7729.80	17.4 | 7601.94	17.4 | 7605.07	17.4 | 7594.41	17.4 | 7774.20
RC1	600	55.0 | 16767.72	55.0 | 16179.39	55.0 | 16183.95	55.0 | 16097.05	55.0 | 16018.50
RC2	600	11.4 | 11311.81	11.4 | 10591.87	11.4 | 10586.14	11.5 | 10511.86	11.4 | 10776.50
CNV		2066	2067	2067	2068	2067
CTD		816326	789420	789592	786373	790784
T		270	80.4	5x25.3	5x99.4	5x95.3
R1	800	72.8 | 32734.57	72.8 | 31486.74	72.8 | 31492.81	72.8 | 31311.38	72.8 | 31208.50
R2	800	15.0 | 20618.21	15.0 | 19873.04	15.0 | 19914.97	15.0 | 19933.39	15.0 | 20082.50
C1	800	75.2 | 25911.44	75.3 | 24990.42	75.2 | 25151.83	75.4 | 24876.38	74.9 | 25530.30
C2	800	23.4 | 11835.72	23.4 | 11438.52	23.4 | 11447.27	23.3 | 11475.05	23.4 | 11638.20
RC1	800	72.0 | 33795.61	72.0 | 31020.22	72.0 | 31278.28	72.0 | 29404.32	72.8 | 28665.70
RC2	800	15.5 | 17536.54	15.4 | 16438.90	15.4 | 16484.31	15.4 | 16495.82	15.4 | 16655.10
CNV		2739	2739	2738	2739	2743
CTD		1424321	1352478	1357695	1334963	1337802
T		360	126.8	5x27.6	5x215	5x216.6
R1	1000	91.9 | 51414.26	91.9 | 48287.98	91.9 | 48369.71	91.9 | 47759.66	91.9 | 47338.80
R2	1000	19.0 | 30804.79	19.0 | 28913.40	19.0 | 29003.42	19.0 | 29076.45	19.0 | 29246.40
C1	1000	94.2 | 43111.60	94.1 | 41683.29	94.1 | 41748.60	94.1 | 41572.86	93.9 | 41827.70
C2	1000	29.3 | 16810.22	29.1 | 16498.61	29.1 | 16534.36	28.8 | 16796.45	29.1 | 16733.70
RC1	1000	90.0 | 46753.61	90.0 | 44743.18	90.0 | 44860.60	90.0 | 44333.40	90.0 | 44150.80
RC2	1000	18.4 | 25588.52	18.3 | 23939.62	18.3 | 24055.31	18.2 | 24131.13	18.2 | 24338.60
CNV		3428	3424	3424	3420	3421
CTD		2144830	2040661	2045720	2036700	2036358
T		450	186.4	5x35.3	5x349	5x324.2

[a] RTI (Repoussis et al. 2009)

[b] NB-100 and NB-f(n) (Nagata et al. 2010)

[c] HGSADC (Vidal et al. 2013)

Best results produced by the methods considered are highlighted in gray.

Table C10 Results for the PDPTW, benchmark set of Li and Lim (2001)

	BH[a] (10 runs)			RP[b] (5 runs)			NK[c] (5 runs)			CLS[d] (All runs)			SISRs (5 runs)		
n	CNV	CTD	T	CNV	CTD	T	CNV	CTD	T	CNV	CTD	T	CNV	CTD	T
200	614	180358	65.0	606	180931	4.4	606	179996	7.6	601	186159	15.0	600	185257	6.3
400	1188	423636	100.0	1158	422201	14.7	1158	418455	40.2	1142	447627	15.0	1133	444247	23.7
600	1718	879940	100.0	1679	863442	37.0	1670	851174	81.1	1643	935948	30.0	1629	906717	66.0
800	2245	1480767	135.0	2208	1432078	65.3	2189	1416710	227.8	2146	1551495	60.0	2119	1526559	140.1
1000	2759	2225190	135.0	2652	2137034	89.5	2659	2120760	275.1	2634	2310830	60.0	2577	2276009	226.6

[a] BH (Bent and Hentenryck 2006)

[b] RP (Røpke and Pisinger 2006a)

[c] NK (Nagata and Kobayashi 2010)

[d] CLS (Curtois et al. 2018)

Best results produced by the methods considered are highlighted in gray.

Table C11 Best results for the PDPTW, benchmark set of Li and Lim (2001)

200	R1		R2		C1		C2		RC1		RC2	
1	20	4819.12	5	4073.10	20	2704.57	6	1931.44	19	3606.06	6	3595.18
2	17	4666.09	4	3796.00	19	2764.56	6	1881.40	15	3671.02	5	3158.25
3	14	4402.38	4	3098.41	17	3127.78	6	1844.33	13	3154.92	4	2881.99
4	10	3031.20	3	2486.15	17	2693.41	6	1767.12	10	2631.82	3	2845.62
5	16	4773.00	4	3438.39	20	2702.05	6	1891.21	16	3715.81	5	2778.28
6	13	4800.94	3	4651.59	20	2701.04	6	1857.78	16	3572.26	5	2707.96
7	12	3543.36	3	3100.33	20	2701.04	6	1850.13	14	3666.34	4	3011.69
8	9	2759.44	2	2454.21	19	3354.27	6	1824.34	13	3145.74	4	2401.13
9	13	5050.75	3	3923.38	18	2724.24	6	1854.21	13	3181.60	4	2208.49
10	11	3664.08	3	3255.45	17	2947.64	6	1817.45	12	2932.17	3	2438.90
400	R1		R2		C1		C2		RC1		RC2	
1	40	10639.75	8	9726.88	40	7152.06	12	4116.33	36	9124.52	* 11	10154.36
2	* 30	11009.75	7	9426.78	38	8007.79	12	4144.29	31	8346.47	10	7184.61
3	22	9261.48	5	10597.89	32	8753.61	12	4401.08	24	7811.53	8	6431.43
4	* 15	7077.05	4	6397.04	30	6910.46	12	4038.00	19	5827.87	5	5251.77
5	28	11374.06	6	9931.76	40	7150.00	12	4030.63	32	8852.83	10	7381.91
6	* 23	11778.11	5	8979.78	40	7154.02	12	3900.29	30	8394.47	9	6337.13
7	18	8771.76	* 4	8086.60	40	7149.43	12	3962.51	28	8121.69	8	6331.56
8	14	5842.41	4	5306.60	* 38	8305.42	12	3844.45	26	7961.80	7	5776.31
9	24	9946.66	6	7946.23	36	7451.20	12	4190.73	25	8031.77	6	6480.65
10	20	8245.87	5	7715.93	* 34	8010.54	12	3828.44	23	7109.99	6	5555.84
600	R1		R2		C1		C2		RC1		RC2	
1	59	22821.65	11	22140.32	60	14095.64	19	7977.98	52	18293.72	16	14598.54
2	45	20158.04	9	23114.74	57	15062.27	18	10009.77	43	16554.50	13	14003.83
3	37	17903.39	7	18393.42	* 49	15086.28	17	8570.14	36	14011.69	10	12488.43
4	28	13179.56	6	10949.87	47	13329.51	17	7904.58	25	10847.32	7	10071.24
5	* 37	24352.75	* 8	23490.88	60	14086.30	* 18	9051.53	* 45	17586.78	13	14698.23
6	31	21832.07	7	19980.43	60	14090.79	18	8910.65	* 41	19037.12	12	15484.92
7	24	17405.34	6	15357.60	60	14083.76	* 18	9854.18	37	16022.35	10	14660.61
8	18	12018.71	* 4	12512.46	58	14880.70	18	7580.43	33	15365.04	9	13331.23
9	31	22578.26	8	18724.38	* 53	16106.46	18	8845.43	* 33	15642.29	* 8	16788.75
10	26	18568.61	7	16779.85	* 51	15688.66	17	7954.74	29	14272.34	7	13334.28
800	R1		R2		C1		C2		RC1		RC2	
1	80	39291.32	14	44179.61	80	25184.38	24	11687.06	66	32252.28	20	23074.22
2	59	34198.47	* 11	38178.37	* 76	30605.50	24	13925.75	56	27943.70	17	22284.26
3	44	29380.19	9	27417.48	63	26461.77	24	12990.87	48	24537.67	14	20598.05
4	25	20808.93	6	21338.07	* 59	22787.23	24	12307.93	34	18346.76	11	14931.64
5	48	41859.16	11	36392.46	80	25211.22	25	12298.75	58	31270.47	16	23633.03
6	* 38	38801.12	9	31280.26	80	25164.25	24	12654.39	54	29025.74	15	22662.26
7	30	27592.35	7	27017.95	80	25158.38	* 24	15133.86	* 50	29126.72	* 13	26866.58
8	* 19	21605.57	5	19012.77	* 77	27377.32	24	11454.90	* 44	28048.82	* 11	23860.47
9	40	39004.71	10	31029.95	72	25889.48	24	11642.36	44	25082.93	* 10	23383.52
10	31	30465.39	* 8	31666.01	69	26145.15	23	12350.24	40	23776.90	9	20663.31
1000	R1		R2		C1		C2		RC1		RC2	
1	100	56745.45	* 17	63441.60	100	42488.66	30	16879.24	82	49175.44	22	34625.64
2	80	49445.58	14	54367.96	94	44556.97	* 30	21074.51	* 71	45583.72	* 19	39112.91
3	54	41750.18	* 10	44305.20	* 79	45099.17	* 29	19458.08	53	35864.47	16	27650.54
4	* 27	31442.54	8	27735.86	* 73	37865.98	29	17994.89	40	27534.52	11	23531.32
5	* 58	62627.18	* 13	56851.53	100	42477.41	31	17137.73	72	50628.04	* 16	41779.00
6	47	50630.53	11	46692.87	101	42838.39	31	17195.29	* 67	45154.07	17	30927.49
7	* 35	39427.57	* 8	43654.04	100	42854.99	31	18653.14	* 60	41890.14	* 15	33737.91
8	* 24	30469.38	* 6	27225.97	98	42951.73	30	17018.59	* 55	42101.70	-	-
9	* 48	54392.71	* 12	52225.09	91	42586.27	30	18237.81	53	39477.68	-	-
10	* 38	47033.43	* 10	46788.87	* 87	42226.13	29	17220.03	* 47	37667.36	11	29910.68

Equal solutions are highlighted in light gray.

New best solutions are highlighted in dark gray.

Reduced number of vehicles are marked with an asterisk.

Table C12 Results for the DARP, benchmark set of Cordeau and Laporte (2003)

							ELS[a] (5 runs)			HGA[b] (5 runs)			H-ALNS[c] (5 runs)			SISRs (5 runs)			
Inst.	n	Q	D	L	m	BKS	Best	Avg.	T	Best	Avg.	T	Best	Avg.	T	Best	Avg.	T*	T
R1a	48	6	480	90	3	190.02	190.02	190.02	0.25	190.02	190.02	0.35	190.02	190.02	0.24	190.02	190.02	0.03	1.87
R2a	96	6	480	90	5	301.34	301.34	301.34	1.25	301.34	301.34	0.89	301.34	301.34	0.64	301.34	301.34	0.49	5.16
R3a	144	6	480	90	7	532.00	532.43	533.86	2.30	532.00	534.08	1.04	532.00	532.00	0.92	532.00	532.00	1.42	9.64
R4a	192	6	480	90	9	570.25	570.54	574.47	7.37	570.25	571.45	1.59	570.29	570.86	2.37	570.25	572.65	2.42	16.26
R5a	240	6	480	90	11	625.64	630.82	637.59	12.07	628.48	631.39	1.90	629.14	632.67	4.94	628.98	632.45	10.78	24.17
R6a	288	6	480	90	13	**782.98**	792.80	796.10	21.92	787.41	788.52	2.28	788.86	790.25	6.17	790.48	791.14	17.35	34.73
R7a	72	6	480	90	4	291.71	291.71	292.96	0.47	291.71	291.79	0.48	291.71	291.71	0.32	291.71	291.71	0.07	3.00
R8a	144	6	480	90	6	487.84	491.60	493.16	2.68	488.89	491.53	1.03	489.89	491.44	1.18	487.84	488.86	1.76	9.81
R9a	216	6	480	90	8	653.94	672.86	681.35	11.25	658.31	660.24	1.54	658.90	661.37	2.52	659.51	662.18	3.52	19.94
R10a	288	6	480	90	10	845.47	857.36	860.68	21.33	853.16	859.91	2.43	854.60	858.65	6.15	847.12	855.86	6.84	34.15
R1b	48	6	480	90	3	164.46	164.46	164.46	0.28	164.46	164.46	0.51	164.46	164.46	0.28	164.46	164.46	0.04	1.96
R2b	96	6	480	90	5	295.66	295.66	295.72	1.37	295.66	295.66	1.09	295.66	296.18	0.74	295.66	295.86	1.37	5.49
R3b	144	6	480	90	7	484.83	489.00	490.70	3.70	484.83	487.23	1.62	484.83	485.15	1.58	484.93	488.05	3.23	9.97
R4b	192	6	480	90	9	529.33	531.08	531.98	10.20	531.86	532.19	2.49	530.88	531.94	3.67	530.22	530.91	4.38	16.86
R5b	240	6	480	90	11	573.56	578.44	580.23	19.93	579.03	582.06	3.30	576.88	577.27	7.83	578.09	579.25	6.54	26.78
R6b	288	6	480	90	13	725.22	731.25	736.59	32.32	737.03	741.06	3.84	737.09	740.12	11.47	735.39	736.64	14.06	36.55
R7b	72	6	480	90	4	248.21	248.21	248.21	0.58	248.21	248.29	0.72	248.21	248.21	0.37	248.21	248.21	0.06	3.39
R8b	144	6	480	90	6	458.73	461.21	462.40	4.32	461.11	463.32	1.72	461.48	461.64	1.66	461.07	461.57	2.12	10.71
R9b	216	6	480	90	8	**590.49**	595.39	597.53	12.43	593.49	595.37	3.12	594.14	596.02	5.73	594.14	595.59	6.57	20.71
R10b	288	6	480	90	10	783.81	796.60	803.99	31.45	791.01	793.64	3.46	784.57	790.01	7.87	787.27	791.45	13.46	33.88
							0.66	1.06	9.87	0.39	0.67	1.77	0.37	0.57	3.33	0.32	0.57	4.83	16.25

[a] ELS (Chassaing et al. 2016)

[b] HGA (Masmoudi et al. 2017)

[c] H-ALNS (Gschwind and Drexl 2019)

Best results produced by the methods considered are highlighted in gray, new best solutions obtained by SISRs are underlined (BKS).

Bibliography

Hussein Al-Omari and Khair Eddin Sabri. New graph coloring algorithms. *American Journal of Mathematics and Statistics*, 2 (4):739–741, 2006.

P. Augerat, Jose Manuel Belenguer, Enrique Benavent, A. Corberan, D. Naddef, and G. Rinaldi. Computational results with a branch-and-cut code for the capacitated vehicle routing problem, 1995.

G. Babin, S. Deneault, and G. Laporte. Improvements to the or-opt heuristic for the symmetric travelling salesman problem. *The Journal of the Operational Research Society*, 58(3):402–407, 2007. ISSN 01605682, 14769360. URL http://www.jstor.org/stable/4622707.

Alexandre Beaudry, Gilbert Laporte, Teresa Melo, and Stefan Nickel. Dynamic transportation of patients in hospitals. *OR Spectrum*, 32(1):77–107, 2010.

Russell Bent and Pascal Van Hentenryck. A two-stage hybrid algorithm for pickup and delivery vehicle routing problems with time windows. *Computers & Operations Research*, 33(4):875–893, 2006. ISSN 0305-0548. Part Special Issue: Optimization Days 2003.

John L Bresina. Heuristic-biased stochastic sampling. In *AAAI/IAAI, Vol. 1*, pages 271–278, 1996.

Ann Melissa Campbell, Dieter Vandenbussche, and William Hermann. Routing for relief efforts. *Transportation Science*, 42(2):127–145, 2008.

M. Chassaing, C. Duhamel, and P. Lacomme. An els-based approach with dynamic probabilities management in local search for the dial-a-ride problem. *Engineering Applications of Artificial Intelligence*, 48:119–133, 2016. ISSN 0952-1976.

N. Christofides and S. Eilon. An algorithm for the vehicle-dispatching problem. *Journal of the Operational Research Society*, 20(3):309–318, 1969.

N. Christofides, A. Mingozzi, and P. Toth. The vehicle routing problem. In *Combinatorial Optimization*, chapter 11, pages 315–338. Wiley, Chichester, UK, 1979.

Jean-François Cordeau and Gilbert Laporte. A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research Part B: Methodological*, 37 (6):579–594, 2003. ISSN 0191-2615.

Jean-François Cordeau, Michel Gendreau, and Gilbert Laporte. A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks*, 30(2):105–119, 1997. ISSN 1097-0037.

Teodor Gabriel Crainic and Michel Toulouse. *Handbook of Metaheuristics*, chapter Parallel Strategies for Meta-Heuristics, pages 475–513. Springer US, Boston, MA, 2003.

G. A. Croes. A method for solving traveling-salesman problems. *Operations Research*, 6(6):791–812, 1958. ISSN 0030364X, 15265463.

Timothy Curtois, Dario Landa-Silva, Yi Qu, and Wasakorn Laesanklang. Large neighbourhood search with adaptive guided ejection search for the pickup and delivery problem with time windows. *EURO Journal on Transportation and Logistics*, pages 1—-42, Jan 2018.

G. Dantzig, R. Fulkerson, and S. Johnson. Solution of a large-scale traveling-salesman problem. *Journal of the Operations Research Society of America*, 2(4):393–410, 1954. ISSN 00963984. URL http://www.jstor.org/stable/166695.

W. A. Dees and R. J. Smith. Performance of interconnection rip-up and reroute strategies. In *18th Design Automation Conference*, pages 382–390, June 1981.

Ulrich Derigs, Jens Gottlieb, Jochen Kalkoff, Michael Piesche, Franz Rothlauf, and Ulrich Vogel. Vehicle routing with compartments: applications, modelling and heuristics. *OR Spectrum*, 33(4):885–914, 2011.

Guy Desaulniers. Branch-and-price-and-cut for the split-delivery vehicle routing problem with time windows. *Operations Research*, 58(1):179–192, 2010. ISSN 0030364X.

Edsger W Dijkstra et al. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.

Yvan Dumas, Jacques Desrosiers, and François Soumis. The pickup and delivery problem with time windows. *European Journal of Operational Research*, 54(1): 7–22, 1991. ISSN 0377-2217.

Olive Jean Dunn. Multiple comparisons among means. *Journal of the American Statistical Association*, 56(293):52–64, 1961.

Burak Eksioglu, Arif Volkan Vural, and Arnold Reisman. The vehicle routing problem: A taxonomic review. *Computers & Industrial Engineering*, 57(4):1472–1483, 2009.

R. Fahrion and M. Wrede. On a principle of chain-exchange for vehicle-routeing problems (1-vrp). *The Journal of the Operational Research Society*, 41(9): 821–827, 1990. ISSN 01605682, 14769360.

Abdellah El Fallahi, Christian Prins, and Roberto Wolfler Calvo. A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem. *Computers & Operations Research*, 35(5):1725–1741, 2008. ISSN 0305-0548.

Marshall L. Fisher. Optimal solution of vehicle routing problems using minimum k-trees. *Operations Research*, 42(4):626–642, 1994. ISSN 0030364X.

Merrill M. Flood. The traveling-salesman problem. *Operations Research*, 4(1):61–75, 1956.

Yuvraj Gajpal and Prakash Abad. An ant colony system (acs) for vehicle routing problem with simultaneous delivery and pickup. *Computers & Operations Research*, 36(12):3215–3223, 2009. ISSN 0305-0548.

H. Gehring and J. Homberger. A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows. In *University of Jyväskyla*, pages 57–64, 1999.

H. Gehring and J. Homberger. A parallel two-phase metaheuristic for routing problems with time windows. *Asia - Pacific Journal of Operational Research*, 18(1):35–47, 2001.

Billy E Gillett and Jerry G Johnson. Multi-terminal vehicle-dispatch algorithm. *Omega*, 4(6):711–718, 1976.

Fred Glover. Applications of integer programming future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*, 13(5): 533–549, 1986. ISSN 0305-0548.

Marc Goetschalckx and Charlotte Jacobs-Blecha. The vehicle routing problem with backhauls. *European Journal of Operational Research*, 42(1):39–51, 1989. ISSN 0377-2217.

Bruce L. Golden, Edward A. Wasil, James P. Kelly, and I-Ming Chao. *The Impact of Metaheuristics on Solving the Vehicle Routing Problem: Algorithms, Problem Sets, and Computational Results*, pages 33–56. Springer US, Boston, MA, 1998.

Timo Gschwind. Route feasibility testing and forward time slack for the synchronized pickup and delivery problem. *OR Spectrum*, 2018. ISSN 1436-6304.

Timo Gschwind and Michael Drexl. Adaptive large neighborhood search with a constant-time feasibility test for the dial-a-ride problem. *Transportation Science*, 2019.

Pierre Hansen, Alain Hertz, and Julio Kuplinsky. Bounded vertex colorings of graphs. *Discrete Mathematics*, 111(1-3):305–312, 1993.

W. D. Harvey and M. Ginsberg. Limited discrepancy search. In *IJCAI*, 1995.

Keld Helsgaun. An effective implementation of the lin–kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126(1):106–130, 2000. ISSN 0377-2217. doi: https://doi.org/10.1016/S0377-2217(99)00284-2. URL https://www.sciencedirect.com/science/article/pii/S0377221799002842.

Tino Henke, M. Grazia Speranza, and Gerhard Wäscher. The multi-compartment vehicle routing problem with flexible compartment sizes. *European Journal of Operational Research*, 246(3):730–743, 2015. ISSN 0377-2217.

Jang-Jei Jaw, A. R. Odoni, H. N. Psaraftis, and N. H. M. Wilson. A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows. *Transportation Research Part B: Methodological*, 20(3):243–257, 1986. ISSN 0191-2615.

Gerard A. P Kindervater and Martin W. P Savelsbergh. Vehicle routing: handling edge exchanges. In *Local Search in Combinatorial Optimization*, pages 337–360. Princeton University Press, Princeton, 2018. ISBN 0691187568.

S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *SCIENCE*, 220(4598):671–680, 1983.

Rahma Lahyani, Leandro C Coelho, Mahdi Khemakhem, Gilbert Laporte, and Frédéric Semet. A multi-compartment vehicle routing problem arising in the collection of olive oil in tunisia. *Omega*, 51:1–10, 2015.

Pat Langley. Systematic and nonsystematic search strategies. In *Proceedings of the First International Conference on Artificial Intelligence Planning Systems*, pages 145–152, 1992.

Gilbert Laporte. Fifty years of vehicle routing. *Transportation Science*, 43(4):408–416, 2009.

Gilbert Laporte, Roberto Musmanno, and Francesca Vocaturo. An adaptive large neighbourhood search heuristic for the capacitated arc-routing problem with stochastic demands. *Transportation Science*, 44(1):125–135, 2010. ISSN 00411655, 15265447.

Feiyue Li, Bruce Golden, and Edward Wasil. Very large-scale vehicle routing: new test problems, algorithms, and results. *Computers & Operations Research*, 32 (5):1165–1179, 2005. ISSN 0305-0548.

H. Li and A. Lim. A metaheuristic for the pickup and delivery problem with time windows. In *Proceedings 13th IEEE International Conference on Tools with Artificial Intelligence. ICTAI 2001*, pages 160–167, 2001.

S. Lin and B. W. Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, 21(2):498–516, 1973.

Shen Lin. Computer solutions of the traveling salesman problem. *Bell System Technical Journal*, 44(10):2245–2269, 1965. ISSN 1538-7305.

Mohamed Amine Masmoudi, Kris Braekers, Malek Masmoudi, and Abdelaziz Dammak. A hybrid genetic algorithm for the heterogeneous dial-a-ride problem. *Computers & Operations Research*, 81:1–13, 2017. ISSN 0305-0548.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

Hokey Min. The multiple vehicle routing problem with simultaneous delivery and pick-up points. *Transportation Research Part A: General*, 23(5):377–386, 1989.

Yves Molenbruch, Kris Braekers, An Caris, and Greet Vanden Berghe. Multi-directional local search for a bi-objective dial-a-ride problem in patient transportation. *Computers & Operations Research*, 77:58–71, 2017.

Yuichi Nagata and Shigenobu Kobayashi. A memetic algorithm for the pickup and delivery problem with time windows using selective route exchange crossover. In Robert Schaefer, Carlos Cotta, Joanna Kołodziej, and Günter Rudolph, editors, *Parallel Problem Solving from Nature, PPSN XI*, pages 536–545, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

Yuichi Nagata, Olli Bräysy, and Wout Dullaert. A penalty-based edge assembly memetic algorithm for the vehicle routing problem with time windows. *Computers & Operations Research*, 37(4):724–737, 2010. ISSN 0305-0548.

Sandra Ulrich Ngueveu, Christian Prins, and Roberto Wolfler Calvo. An effective memetic algorithm for the cumulative capacitated vehicle routing problem. *Computers & Operations Research*, 37(11):1877–1885, 2010.

Temel Öncan, İ Kuban Altınel, and Gilbert Laporte. A comparative analysis of several asymmetric traveling salesman problem formulations. *Computers & Operations Research*, 36(3):637–654, 2009.

Johan Oppen and Arne Løkketangen. A tabu search approach for the livestock collection problem. *Computers & Operations Research*, 35(10):3213–3229, 2008.

Johan Oppen, Arne Løkketangen, and Jacques Desrosiers. Solving a rich vehicle routing and inventory problem using column generation. *Computers & Operations Research*, 37(7):1308–1317, 2010.

Ilhan Or. *Traveling salesman-type combinatorial problems and their relation to the logistics of regional blood banking*. PhD thesis, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, ILy, 1976.

Dimitris C. Paraskevopoulos, Gilbert Laporte, Panagiotis P. Repoussis, and Christos D. Tarantilis. Resource constrained routing and scheduling: Review and research prospects. *European Journal of Operational Research*, 263(3):737–754, 2017. ISSN 0377-2217.

Sophie N Parragh, Jean-François Cordeau, Karl F Doerner, and Richard F Hartl. Models and algorithms for the heterogeneous dial-a-ride problem with driver-related constraints. *OR spectrum*, 34(3):593–633, 2012.

Sophie N. Parragh, Jorge Pinho de Sousa, and Bernardo Almada-Lobo. The dial-a-ride problem with split requests and profits. *Transportation Science*, 49(2):311–334, May 2015. ISSN 1526-5447.

JG Partyka and RW Hall. On the road to service. *ORMS Today*, 27(4):26–30, 2000.

Diego Pecin, Artur Pessoa, Marcus Poggi, and Eduardo Uchoa. *Improved Branch-Cut-and-Price for Capacitated Vehicle Routing*, pages 393–403. Springer International Publishing, 2014.

David Pisinger and Stefan Røpke. A general heuristic for vehicle routing problems. *Computers & Operations Research*, 34(8):2403–2435, 2007. ISSN 0305-0548.

Chris N Potts and Mikhail Y Kovalyov. Scheduling with batching: A review. *European Journal of Operational Research*, 120(2):228–249, 2000.

Jean-Yves Potvin and Jean-Marc Rousseau. A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *European Journal of Operational Research*, 66(3):331–340, 1993. ISSN 0377-2217.

Jean-Yves Potvin and Jean-Marc Rousseau. An exchange heuristic for routeing problems with time windows. *The Journal of the Operational Research Society*, 46(12):1433–1446, 1995. ISSN 01605682, 14769360.

Patrick Prosser and Paul Shaw. Study of greedy search with multiple improvement heuristics for vehicle routing problems, 1996.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2017. URL https://www.R-project.org/.

Abdur Rais, Filipe Alvelos, and Maria Sameiro Carvalho. New mixed integer-programming model for the pickup-and-delivery problem with transshipment. *European Journal of Operational Research*, 235(3):530–539, 2014.

P. P. Repoussis, C. D. Tarantilis, and G. Ioannou. Arc-guided evolutionary algorithm for the vehicle routing problem with time windows. *IEEE Transactions on Evolutionary Computation*, 13(3):624–647, June 2009.

Glaydston Mattos Ribeiro and Gilbert Laporte. An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem. *Computers & Operations Research*, 39(3):728–735, 2012. ISSN 0305-0548.

Yves Rochat and Éric D. Taillard. Probabilistic diversification and intensification in local search for vehicle routing. *Journal of Heuristics*, 1(1):147–167, Sep 1995.

Stefan Røpke and David Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4):455–472, 2006a.

Stefan Røpke and David Pisinger. A unified heuristic for a large class of vehicle routing problems with backhauls. *European Journal of Operational Research*, 171(3): 750–775, 2006b. ISSN 0377-2217.

Matteo Salani and Ilaria Vacca. Branch and price for the vehicle routing problem with discrete split deliveries and time windows. *European Journal of Operational Research*, 213(3):470–477, 2011. ISSN 0377-2217.

S. Salhi and G. Nagy. A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling. *Journal of the Operational Research Society*, 50(10):1034–1042, 1999. ISSN 1476-9360.

D. Sariklis and S. Powell. A heuristic method for the open vehicle routing problem. *Journal of the Operational Research Society*, 51(5):564–573, 2000. ISSN 1476-9360.

Martin W. P. Savelsbergh. *Computer Aided Routing*. Ph.D. Dissertation, Centrum voor Wiskunde en Informatica, Amsterdam, Amsterdam, 1988.

Martin WP Savelsbergh. The vehicle routing problem with time windows: Minimizing route duration. *ORSA Journal on Computing*, 4(2):146–154, 1992.

M.W.P. Savelsbergh. An efficient implementation of local search algorithms for constrained routing problems. *European Journal of Operational Research*, 47(1): 75–85, 1990. ISSN 0377-2217.

Gerhard Schrimpf, Johannes Schneider, Hermann Stamm-Wilbrandt, and Gunter Dueck. Record breaking optimization results using the ruin and recreate principle. *Journal of Computational Physics*, 159(2):139–171, 2000.

Paul Shaw. Using constraint programming and local search methods to solve vehicle routing problems. In *Proceedings of the 4th International Conference on Principles and Practice of Constraint Programming*, CP '98, pages 417–431, London, UK, UK, 1998. Springer-Verlag.

Anand Subramanian, Eduardo Uchoa, and Luiz Satoru Ochi. A hybrid algorithm for a class of vehicle routing problems. *Computers & Operations Research*, 40(10"): 2519–2531, 2013. ISSN 0305-0548.

Eric Taillard, Philippe Badeau, Michel Gendreau, François Guertin, and Jean-Yves Potvin. A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation Science*, 31(2):170–186, 1997.

El-Ghazali Talbi. *Metaheuristics: From Design to Implementation*, volume 74. 06 2009. ISBN 9780470278581. doi: 10.1002/9780470496916.

Jiafu Tang, Yuan Kong, Henry Lau, and Andrew W.H. Ip. A note on "efficient feasibility testing for dial-a-ride problems". *Operations Research Letters*, 38(5): 405–407, 2010. ISSN 0167-6377.

Paolo Toth and Daniele Vigo. Heuristic algorithms for the handicapped persons transportation problem. *Transportation Science*, 31(1):60–71, 1997.

Eduardo Uchoa, Diego Pecin, Artur Pessoa, Marcus Poggi, Thibaut Vidal, and Anand Subramanian. New benchmark instances for the capacitated vehicle routing problem. *European Journal of Operational Research*, 257(3):845–858, 2017. ISSN 0377-2217.

Alex Van Breedam. *An Analysis of the Behavior of Heuristics for the Vehicle Routing Problem for a selection of problems with Vehicle-related, Customer-related, and Time-related Constraints*. Ph.d. dissertation, University of Antwerp, Faculty of Applied Economics, Antwerp, 1994.

Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau, Nadia Lahrichi, and Walter Rei. A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. *Operations Research*, 60(3):611–624, 2012. ISSN 0030364X.

Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau, and Christian Prins. A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Computers & Operations Research*, 40(1):475–489, 2013. ISSN 0305-0548.

Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau, and Christian Prins. A unified solution framework for multi-attribute vehicle routing problems. *European Journal of Operational Research*, 234(3):658–673, 2014. ISSN 0377-2217.

C. D. J. Waters. A solution procedure for the vehicle-scheduling problem based on iterative route improvement. *The Journal of the Operational Research Society*, 38(9):833–839, 1987. ISSN 01605682, 14769360.

Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1 (6):80–83, 1945. ISSN 00994987.

N. H. M. Wilson, J. M. Sussman, H. K. Wang, and B. T. Higonnet. Scheduling algorithms for a dial-a-ride system. Technical Report USL TR-70-13, Massachusetts Institute of Technology, Cambridge, MA, 1971.

Emmanouil E. Zachariadis and Chris T. Kiranoudis. An effective local search approach for the vehicle routing problem with backhauls. *Expert Systems with Applications*, 39(3):3174–3184, 2012. ISSN 0957-4174.

List of Publications

Christiaens, J., & Vanden Berghe, G. (2020). Slack Induction by String Removals for Vehicle Routing Problems. Transportation Science, 54(2), 417-433. doi:10.1287/trsc.2019.0914

Christiaens, J., Çalik, H., Wauters, T., Chirayil Chandrasekharan, R., & Vanden Berghe, G. (2020). The prisoner transportation problem. European Journal Of Operational Research. doi:10.1016/j.ejor.2020.01.027

Toffolo, T. A. M., Christiaens, J., Spieksma, F., & Vanden Berghe, G. (2019). The sport teams grouping problem. Annals of Operations Research, 275(1), 223-243. doi:10.1007/s10479-017-2595-z

Toffolo, T. A. M., Christiaens, J., Van Malderen, S., Wauters, T., & Vanden Berghe, G. (2018). Stochastic local search with learning automaton for the swap-body vehicle routing problem. Computers & Operations Research , 89, 68-81. doi:10.1016/j.cor.2017.08.002

Wauters, T., Villa, F., Christiaens, J., Álvarez-Valdés, R., & Vanden Berghe, G. (2016). A decomposition approach to dual shuttle automated storage and retrieval systems. Computers & Industrial Engineering , 101, 325-337.

Toffolo, T., Christiaens, J., Spieksma, F., & Vanden Berghe, G. (2016). Grouping sport teams into round robin competitions. In E. K. Burke, L. D. Gaspero, E. Özcan, B. McColumn, & A. Schaerf (Eds.), Proceedings of the 11th international conference of the practice and theory of automated timetabling (PATAT 2016) (pp. 353-369). Udine, Italy: PATAT.



FACULTY OF ENGINEERING TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE
CODES KU LEUVEN
Celestijnenlaan 200A box 2402
B-3001 Leuven
j.christiaens@kuleuven.be