# New Challenges in Time-Definite Vehicle Routing

Vom Fachbereich Wirtschaftswissenschaften
der Technischen Universität Kaiserslautern
zur Verleihung des akademischen Grades
Doctor rerum politicarum (Dr. rer. pol.)
genehmigte

D i s s e r t a t i o n

vorgelegt von

Dipl.-Inf. Dipl.-Kfm. Michael David Schneider

| | |
|---|---|
| Tag der mündlichen Prüfung: | 16. Juli 2012 |
| Dekan: | Prof. Dr. Stefan Roth |
| Vorsitzender: | Prof. Dr. Stefan Roth |
| Berichterstatter: | 1. Prof. Dr. Oliver Wendt |
| | 2. Prof. Dr. Daniele Vigo |

D 386
(2012)

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

In order to enhance the readability of this thesis, the meaning of abbreviations is restated on first usage in each chapter. The complete list of abbreviations is as follows.

| | |
|---|---|
| ACO | ant colony optimization |
| AFS | alternative fuel stations |
| AFV | alternative fuel vehicles |
| BEV | battery electric vehicle |
| BKS | best-known solution |
| CNG | compressed natural gas |
| CNV | cumulated number of vehicles |
| ConVRP | consistent VRP |
| CPLEX | IBM ILOG CPLEX Optimizer |
| CTD | cumulated traveled distance |
| CVRP | capacitated VRP |
| DBCA | density-based clustering algorithm |
| DHL | DHL International |
| DPD | DPD Dynamic Parcel Distribution |
| EC | evolutionary computation |
| E-VRPTW | electric VRP with time windows and recharging stations |
| FTS | free tabu search |
| FABRA | fixed-area-based routing approach |
| G-VRP | green VRP |
| ILS | iterated local search |
| MCWS | modified Clarke and Wright savings algorithm |
| MDVRP | multi-depot VRP |
| MDVRPI | multi-depot VRP with inter-depot routes |
| NP | nondeterministic polynomial time |
| OR | operations research |
| PVRP | periodic VRP |

| | |
|---|---|
| RR | route reoptimization |
| SA | simulated annealing |
| SPS | small package shipping |
| SFSTR | semi-fixed service territory routing |
| TS | tabu search |
| TS-DST | tabu search for solving VRPTWDST |
| TTS | territory tabu search |
| TWD | time window density |
| TWW | time window width |
| UPS | United Parcel Service |
| VNS | variable neighborhood search |
| VRP | vehicle routing problem |
| VRPIRF | VRP with intermediate replenishment facilities |
| VRPPD | VRP with pickup and delivery |
| VRPTW | VRP with time windows |
| VRPTWDST | VRP with time windows and driver-specific times |

# List of Notations

To keep the list of notations concise, notations that are only used in direct proximity to their definition or whose meaning is always restated before use are not included in the list.

| | |
|---|---|
| $V$ | set of customers |
| $N$ | total number of customers |
| $A$ | set of arcs |
| $d_{ij}$ | distance between vertex $i$ and vertex $j$ |
| $t_{ij}$ | travel time between vertex $i$ and vertex $j$ |
| $K$ | set of vehicles |
| $C$ | capacity of a vehicle |
| $q_i$ | demand of vertex $i$ |
| $e_i$ | earliest time to start service at vertex $i$ |
| $l_i$ | latest time to start service at vertex $i$ |
| $s_j$ | service time at vertex $j$ |
| $F$ | set of recharging stations |
| $F'$ | set of visits to recharging stations in $F$ |
| $h$ | charge consumption rate |
| $Q$ | battery capacity |
| $g$ | recharging rate |
| $S$ | solution |
| $\mathcal{N}(S)$ | neighborhood of solution $S$ |
| $f$ | objective function |
| $Vert$ | set of vertices visited in a route or complete solution |
| $m$ | number of routes/vehicles |
| $P_c$ | capacity violation penalty |
| $a_v$ | earliest start time of service at vertex $v$ |
| $P_{tw}$ | time window violation penalty |
| $\tilde{a}_v$ | extended earliest start time of service at vertex $v$ |
| $\mathrm{TW}_v^{\rightarrow}$ | forward time window penalty slack of vertex $v$ |

| | |
|---|---|
| $\tilde{z}_v$ | extended latest start time of service at vertex $v$ |
| $\mathrm{TW}_v^{\leftarrow}$ | backward time window penalty slack of vertex $v$ |
| $EZ$ | exclusion zone |
| $\omega$ | percentage of total customers in exclusion zone |
| $\tau$ | number of sample days |
| $V_t$ | subset of customers that require service on day $t$ |
| $m_{st}$ | number of service territories |
| $\theta$ | service level for sample days |
| $I$ | set of seed customers |
| $\mathfrak{A}$ | adjacency graph |
| $ST$ | service territory |
| $\rho$ | percentage of customers assigned to service territories |
| $\mu$ | expected value of normal distribution |
| $\sigma$ | standard deviation of normal distribution |
| $\alpha, \beta, \gamma$ | penalty factors |
| $\vartheta$ | tabu tenure |
| $\nu$ | factor to prioritize customers located far from the depot |
| $TD$ | secondary objective of minimizing traveled distance |
| $WT$ | secondary objective of minimizing working time |
| $WD$ | secondary objective of minimizing working duration |

# Chapter 1

# Introduction

Small package shipping (SPS) is a fast-growing market with an increase in package volume of 37% from 2000 to 2010 in Germany (Esser and Kurte 2011). In general, SPS companies perform last-mile deliveries from a set of local depots (see, e.g., Zhong et al. 2007, Haase and Hoppe 2008) and the associated pickup and delivery costs are estimated to amount to 35-60% of the total transportation cost (Wasner and Zäpfel 2004). The SPS sector is characterized by a highly competitive market situation, especially since the deregulation in the US and Europe (see, e.g., Figliozzi et al. 2007), which was even intensified by the financial crisis in 2008/2009 (Esser and Kurte 2011). Moreover, rising fuel and labor costs constantly decrease the profit margins per delivered package.

In recent years, the pressure has further increased as laws and regulations to reduce greenhouse gas pollution have already been passed or are currently under debate. For example, to stop the increasing emissions of light commercial vehicles (<3.5t), EU regulation No 510/2011 imposes a penalty of 95 Euro for each gram $CO_2$/km above 147 g $CO_2$/km of the manufacturers' average emissions starting in 2020 (European Parliament and European Council 2011). The white book of the European Commission even envisages a mostly emission-free city logistics until 2030 (European Commission 2011).

Besides optimizing daily delivery schedules concerning routing costs, SPS companies are forced to use every possible advantage to render their local delivery operations profitable. On the one hand, they pay more and more attention to customer service and effective workforce management practices to gain advantage over their competitors. On the other hand, battery electric vehicles (BEVs) are employed for last-mile deliveries to reduce energy costs and to meet future emission standards.

Enhanced customer service quality and an effective utilization of drivers can be achieved by having the same driver visit the same set of customers regularly as the

driver becomes acquainted with the region and the customer locations therein. Experienced drivers use shortcuts, know about traffic light intervals, anticipate road or traffic problems, find parking space more easily and know alternative delivery possibilities in case a customer is absent, which leads to reduced travel and service times. The regularity of service creates a bond between customer and driver, which yields improved customer service and a competitive advantage resulting from higher customer loyalty and improved reputation (Wong 2008, Smilowitz et al. 2012).

To achieve such consistency benefits, the routing operations of SPS companies are commonly based on the division of the depot area into fixed service territories, each visited by a single driver (Malandraki et al. 2001, Wong 2008). Due to the pre-assignment of customers to drivers, such fixed-area-based routing approaches (FABRAs) offer the following advantages (Wong and Beasley 1984): 1) they implicitly achieve service consistency as a customer requiring service is visited by the same driver every time he requires service, 2) they decrease routing complexity as routes can be planned independently for each driver, and 3) they reduce administrative costs, e.g., by reducing the effort for sorting and assigning packages to drivers or for the instruction of drivers. The drawback of FABRAs is the decline in routing flexibility, which yields route configurations that are suboptimal concerning route efficiency (for example, measured in total traveled distance) when faced with varying demand and/or numerous and possibly tight time window requirements.

For reducing energy costs and complying with emission regulations, the use of BEVs is a promising alternative as EU regulation No 510/2011 defines BEVs to have zero emissions. In earlier years, BEVs failed due to exorbitant battery prices and very short driving ranges. However, as BEVs have become one of the major research areas in the automotive sector and more and more BEVs are developed, the magnitude of these problems diminishes. In the SPS industry, several big companies, like DHL International (DHL)[1], United Parcel Service (UPS)[2], and DPD Dynamic Parcel Distribution (DPD)[3] already started using

[1] `http://www.dhl.com/en/press/releases/releases_2011/group/040711.html`

[2] `http://pressroom.ups.com/Press+Releases/Archive/2011/Q3/UPS+Purchasing+100+All-Electric+Vehicles+for+California+Deployment`

[3] `http://www.dpd.com/de/Home/About-DPD/Press-Centre/Press-Releases/Archive/Archive-2011/EN-press-releases-2011/DPD-in-Germany-emissions-free-parcel-deliveries-with-electric-vehicles-in-Hamburg`

BEVs for last-mile deliveries from depots to customers, in particular in urban areas. Governments in all parts of the world promote the electrification trend and plan to provide the required infrastructure, e.g., "Source London", a city wide electric vehicle charging network will go into operation in London in 2013[4].

Operations research (OR) techniques are generally applied for the effective management of local delivery tasks (Toth and Vigo 2002). In the scientific literature, route planning tasks are represented as Vehicle Routing Problems (VRPs). The basic VRP seeks to minimize the costs for visiting a set of customers by means of delivery routes starting and ending at a depot. Many variants of the VRP incorporating real-world constraints and conditions have been proposed, among them the Capacitated VRP (CVRP), where vehicles have a limited freight capacity (see, e.g., Toth and Vigo 2002, Cordeau and Laporte 2005) and the VRP with Time Windows (VRPTW), where customers have to be reached within a specified time interval (see, e.g., Bräysy and Gendreau 2005a,b, Gendreau and Tarantilis 2010). The VRPTW is an NP-hard problem of which only small-sized instances can be solved by means of exact solution methods (Baldacci et al. 2012). Consequently, a huge amount of metaheuristic solution methods have been proposed (Gendreau et al. 2008).

In order to find a tradeoff between achieving service consistency and maintaining routing flexibility, some recent scientific approaches present VRP models for SPS routing problems that explicitly integrate consistency requirements and are solved without fixing service territories (see Groër et al. 2009, Sungur et al. 2010, Smilowitz et al. 2012). On the other hand, FABRAs are adapted by allowing to adjust them based on the daily demand, i.e., by excluding a percentage of customers from being assigned to fixed areas as done at UPS and described in Zhong et al. (2007). However, to the best of our[5] knowledge, all published works on FABRAs neglect the existence of time windows and of the above mentioned approaches forgoing service territories only Sungur et al. (2010) consider soft time windows. This strongly conflicts with recent practical developments. Our industry contacts state that up to 60% of their orders are time-definite, which is consistent with the statistics given in Campbell and Thomas (2009).

[4] `https://www.sourcelondon.net/`

[5] As commonly done in scientific publications to enhance readability, the first person plural is used to indicate the author of this thesis throughout the work.

If time windows are considered, routing flexibility is not only needed to achieve distance-efficient route configurations but also to fulfill customer delivery time requirements. Thus, the value of routing flexibility should increase, which is likely to have a negative effect on the solution quality of FABRAs. However, no study exists on the magnitude of this effect and the factors that influence it. And nonetheless, several private communications with employees of German SPS companies and the literature on SPS routing indicate that, despite the high percentage of time-definite deliveries, FABRAs are utilized.

Concerning the utilization of BEVs, route planning models have to incorporate the specifics of BEVs in order to be beneficial. For example, the maximum driving range of BEVs is potentially not sufficient to perform the typical delivery tour of a small package shipper in one run or to reach customers located far from the depot. Since reducing the number of deliveries performed by one vehicle or excluding customers is clearly not a profitable option, visits to recharging stations along the routes are required. If theses are not properly integrated in the route planning method, this can lead to long detours, especially if available recharging stations are scarce.

However, to the best of our knowledge, only one routing model that considers recharging stations exists. Erdogan and Miller-Hooks (2012) propose the Green VRP (G-VRP), a routing model for Alternative Fuel Vehicles (AFVs). The G-VRP considers a limited fuel capacity of the vehicles and the possibility to refuel at Alternative Fuel Stations (AFSs). For each refueling as well as for each customer visit, a fixed service time is considered and the maximum duration of a route is restricted. However, no capacity restrictions and no time window constraints are considered.

## 1.1 Contribution

This thesis addresses the above described challenges faced by SPS companies and investigates the integration of 1) service consistency and driver knowledge aspects and 2) the utilization of electric vehicles into the route planning of small package shippers. We use OR models and solution methods to gain insights into the newly arising problems and thus support managerial decisions concerning these issues.

The first question studied in this work is: How well do FABRAs perform in the presence of time window requirements and how strong is the influence of factors like the size of the service territories or the variance of the daily demand on this performance? To address this question, we develop a FABRA to solve a series of VRPTW that are linked by a common customer base set of which every day a random subset requires service. This corresponds to the practical problem faced by many SPS companies. The daily problems are solved completely independent of one another, i.e., we are not optimizing consistency goals like, e.g., driver familiarity over the considered time period, but our optimization goal is to minimize the traveled distance on each day while adhering to the restriction posed by the service territories.

The developed FABRA is a two-phase method called *Semi-Fixed Service Territory Routing* (SFSTR) that first divides the delivery area into service territories and second carries out the daily routing based on these territories. It aims at providing a performance that is good enough to achieve meaningful results in the studies while maintaining (relative) simplicity. SFSTR is used in numerical studies to investigate the suitability of such an approach when faced with time window constraints in the daily routing. More precisely, we study to what extent the goals of feasible and distance-efficient routes can be achieved with a FABRA under time window constraints and how well the implicit realization of familiarity benefits (without directly optimizing consistency in any way but only due to the restriction to service territories) is achieved.

Second, we investigate a route planning model that integrates driver learning aspects in order to generate efficient routes based on different extents of driver knowledge. This is achieved by means of driver-specific travel and service times reflecting the knowledge of the different drivers. In this way, routing flexibility is maintained while drivers have an incentive to stay in familiar areas due to shorter driving and service times. Note that contrary to the SFSTR approach, the goal is not to promote the learning of the drivers but to design efficient routes based on already existing different extents of driver knowledge.

We provide a mathematical formulation of the so-called *VRP with Time Windows and Driver-Specific Times* (VRPTWDST). As VRPTWDST extends the VRPTW, the high complexity of the problem renders exact solution methods inadequate for solving realistically sized problem instances. Therefore, we de-

velop a high-quality Tabu Search (TS) for solving VRPTWDST and study its performance on benchmark instances of the closely related VRPTW and a comprehensive set of newly generated VRPTWDST test instances. Moreover, the effect of different distributions of the "learned" customers and the level of driver knowledge is investigated.

Third, we study a route planning model that considers the special characteristics of BEV. The *Electric Vehicle Routing Problem with Time Windows and Recharging Stations* (E-VRPTW) incorporates the possibility of recharging at any of a set of available stations using an appropriate recharging scheme, i.e., recharging times depend on the battery charge of the vehicle on arrival at the station. Moreover, the most important practical requirements of SPS companies using BEVs, namely capacity constraints on vehicles and customer time windows are included.

As E-VRPTW is also an extension of the VRPTW and thus NP-hard, we develop a hybrid metaheuristic to solve it. The hybrid combines a Variable Neighborhood Search (VNS) heuristic with a TS method for the intensification phase of the VNS. In numerical studies, we prove the quality and efficiency of our VNS/TS on test instances of related problems, namely the G-VRP and the Multi-Depot VRP with Inter-Depot Routes (MDVRPI). Moreover, we design two sets of benchmark instances for E-VRPTW: A set of small-sized instances that we can solve exactly with the optimization software IBM ILOG CPLEX Optimizer (CPLEX)[6] in order to assess the performance of VNS/TS on E-VRPTW and a set of more realistically sized instances, on which we study the effectiveness of every component of our hybrid solution method.

Last, we contribute by identifying an error in a recently proposed approach for efficiently handling time window constraints in local search by Nagata et al. (2010). We study the cases in which the proposed formula offers incorrect results and present a sound formula. The corrected approach is adapted for all the solution methods presented in this work.

One should note that the findings of this thesis are not only relevant for SPS companies but also for other logistics service providers. The insights about fixed-area routing in the presence of time windows as well as the decision support offered by the VRPTWDST routing model and solution method can be of

[6] `http://www.ibm.com/software/integration/optimization/cplex-optimizer/`

importance to any operations whose routing creates driver learning effects. This is always the case if repeated delivery to a base set of customers or locations takes place, like, e.g., in meal delivery, product maintenance, transport of disabled persons or bus routing. Moreover, the E-VRPTW model and solution method can be of relevance for any company employing electric vehicles in their delivery fleet to fulfill time-definite deliveries.

## 1.2 Organization

This thesis is structured as follows. In Chapter 2, the fundamental concepts and the relevant literature are introduced. As the VRPTW lies at the core of all problems considered in this work, we give a thorough description of the problem, an overview of the most successful solution methods and a precise description of the most established VRPTW benchmarks. Moreover, the metaheuristic components used in this work are introduced. Subsequently, techniques for handling capacity and time window constraints are discussed and a correction of the time travel approach of Nagata et al. (2010) is presented. Finally, the relevant literature concerning consistency aspects and the employment of electric vehicles in vehicle routing is presented.

In Chapter 3, we detail how SFSTR designs service territories and conducts the daily routing based on the created territories. The TS used to generate the solutions for the created sample days and for carrying out the daily routing is introduced. Further, we present computational studies that use several 100-day series of VRPTW problems with different customer distributions to study the performance of SFSTR concerning route efficiency and consistency measures in comparison to an RR strategy. Additionally, we analyze the influence of the size of the territories, i.e., the number of customers fixedly assigned to a driver, and different variabilities in the number of customers requiring service on the overall performance of SFSTR.

Chapter 4 provides a mathematical formulation of the VRPTWDST and describes an efficient TS method to solve the problem. Moreover, we generate several sets of benchmark instances for VRPTWDST and perform extensive numerical studies on the created instances and on standard VRPTW instances. The

performance of the developed TS is assessed and the impact of different distributions of learned customers and different levels of driver knowledge is studied.

In Chapter 5, we present a mixed-integer model for E-VRPTW and describe the VNS/TS hybrid for solving the problem. Experimental results obtained on newly designed E-VRPTW instances as well as on benchmark sets of related problems are presented. Finally, we summarize the findings of this thesis and give an outlook on future research opportunities in Chapter 6.

# Chapter 2

# Fundamental Concepts and Literature

This section introduces some basic concepts and commonalities that are relevant for the problems addressed in this work and gives an overview of the related literature. In Section 2.1, the Vehicle Routing Problem with Time Windows (VRPTW), which lies at the core of all problems studied in the thesis, is introduced. We provide an exact mathematical formulation of the problem, an overview of the most successful metaheuristic solution methods, paying special attention to the approaches applied in the thesis, and a detailed description of the most relevant benchmark problems, which are later adapted for the problems addressed in this work.

Section 2.2 investigates the efficient constraint handling in local-search-based solution methods for the VRPTW. We present techniques to efficiently compute the change in capacity and time window violation caused by application of a local search move. Moreover, we propose a correction of a recently presented approach by Nagata et al. (2010). In Section 2.3, we give an overview of the literature addressing consistency and driver learning aspects in route planning methods, setting the background for Chapters 3 and 4. Section 2.4 discusses the literature relevant for integrating the specifics of electric vehicles into route planning, laying the foundation for the Electric Vehicle Routing Problem with Time Windows and Recharging Stations (E-VRPTW) studied in Chapter 5.

## 2.1 The Vehicle Routing Problem with Time Windows (VRPTW)

More than fifty years have elapsed since the VRP was first introduced in Dantzig and Ramser (1959), providing a useful abstraction of important real-world problems like waste collection or grocery distribution. Since then, the VRP and

its variants have become some of the most studied combinatorial optimization problems in OR literature, inspiring with their practical relevance and intrinsic difficulty (Laporte 2009).

Among the VRP variants, the VRPTW is the most important and widely studied, with hundreds of published papers (Nagata et al. 2010, Gendreau and Tarantilis 2010). It considers time slots (so called time windows) in which service at each customer has to take place and it can thus be used to model many real-world distribution management problems more accurately, like, e.g., parcel deliveries in various industries, bank deliveries, school bus routing and transports for just-in-time manufacturing (Bräysy and Gendreau 2005a). The significant academic value of the VRPTW also stems from the fact that it is often used as a benchmark problem when solution approaches for new, real-world inspired VRPTW variants are suggested (Nagata et al. 2010).

The solution of a VRPTW calls for the determination of a cost-minimal set of routes carried out by a set of homogeneous vehicles located at a single depot. Each route starts and ends at the depot within a given scheduling horizon and the cumulated demand of the customers visited on a route does not exceed vehicle capacity. Each customer is served by exactly one vehicle. In this work, the focus lies on hard time windows requiring that in the final solution of a problem each customer is served within the given time window. By contrast, a small part of the literature on VRPTW considers soft time windows, i.e., the final solution is allowed to violate time windows and a cost penalty is generally added to the objective function (see, e.g., Taillard et al. 1997, Figliozzi 2010). In the following, VRPTW refers to the problem with hard time windows if not explicitly stated otherwise.

The VRPTW is a challenging optimization problem generalizing the NP-hard Capacitated VRP (CVRP). Even finding a feasible solution to the VRPTW given a fixed vehicle number is an NP-hard problem as it includes the one-dimensional bin packing problem as a special case (Garey and Johnson 1979, Savelsbergh 1985). Contrary to the Traveling Salesman Problem, for which instances with several thousand customers are regularly solved (see, e.g., Gutin and Punnen 2002), only small to medium-sized instances of VRPTW can be solved by exact methods even within large amounts of computing time (see, e.g., Baldacci et al. 2012). Consequently, intensive research effort has been given to meta-

heuristic optimization approaches for VRPTW, which resulted in a large number of successful methods that are able to produce high-quality solutions for more realistically sized instances in reasonable time.

Unfortunately, the objective function of the VRPTW differs between exact solution approaches and heuristics, which makes a comparison between methods of the different strands or the utilization of bounds determined by exact methods to assess the quality of a heuristic method impossible. While in the literature on exact approaches the minimization of traveled distance is traditionally considered as objective (see, e.g., Kallehauge 2008, Baldacci et al. 2012), most heuristic approaches use a hierarchical objective (see, e.g., Bräysy and Gendreau 2005a, Gendreau and Tarantilis 2010). First, the number of vehicles is minimized and only in the second step the traveled distance. Consequently, a solution with fewer vehicles is always superior, independent of the traveled distance. The hierarchical objective used in heuristic approaches can be traced back to the paper of Solomon (1987), who presented a set of construction heuristics for VRPTW and compared their performance on newly generated (and now well-known) VRPTW benchmark instances. If not explicitly stated otherwise, the hierarchical objective function is used in the remainder of this work.

Cordeau et al. (2002) provide a comprehensive overview of exact and heuristic methods for the VRPTW proposed up to the end of the twentieth century. A detailed and updated description of heuristic solution methods for VRPTW is given in the two-part review of Bräysy and Gendreau (2005a,b). The first part deals with constructive heuristics and local search methods while the second part is devoted to metaheuristics. The review of metaheuristics is again updated in Gendreau et al. (2008). Gendreau and Tarantilis (2010) provide an overview and analysis of advanced heuristics for solving large-scale VRPTW. A recent survey on exact algorithms is given by Baldacci et al. (2012).

### 2.1.1 Problem Definition

This section provides a formal definition of the VRPTW as mixed-integer program. Let $V = \{1, \ldots, N\}$ denote the set of $N$ customers and let 0 and $N+1$ denote instances of the same depot, where every route starts at 0 and ends at $N+1$. Further, let the indices 0 and $N+1$ indicate that a set contains the respective instance of the depot, e.g., $V_0 = V \cup \{0\}$. Then, VRPTW can be

defined on a complete directed graph $G = (V_{0,N+1}, A)$, with the set of arcs $A = \{(i,j) \mid i,j \in V_{0,N+1}, i \neq j\}$.

With each arc, a distance $d_{ij}$ and a travel time $t_{ij}$ are associated. At the depot, a set $K$ of homogeneous vehicles with a maximal capacity of $C$ are available. Each vertex $i \in V_{0,N+1}$ is assigned a positive demand $q_i$, which is set to 0 for the depot. Moreover, each vertex $i \in V_{0,N+1}$ has a time window $[e_i, l_i]$ and an associated service time $s_j$ ($s_0, s_{N+1} = 0$). Service cannot begin before $e_i$, which might cause waiting time and is not allowed to start after $l_i$ but might end later.

Instead of a three-index formulation, we use decision variables associated with vertices to keep track of vehicle states, thus keeping the number of required variables lower. Variable $\tau_j$ specifies the time of arrival and $u_j$ the remaining load on arrival at vertex $j \in V_{0,N+1}$. The decision variables $x_{ij}, i \in V_0, j \in V_{N+1}, i \neq j$ are binary and equal 1 if an arc $(i,j)$ is traveled and 0 otherwise. Using the introduced notation, the mathematical model of VRPTW is formulated as mixed-integer program as follows:

$$\min \sum_{i \in V_0, j \in V_{N+1}, i \neq j} d_{ij} x_{ij} \tag{2.1}$$

$$\sum_{j \in V_{N+1}, i \neq j} x_{ij} = 1 \qquad \forall i \in V \tag{2.2}$$

$$\sum_{j \in V_0, i \neq j} x_{ji} = 1 \qquad \forall i \in V \tag{2.3}$$

$$\tau_i + (t_{ij} + s_i) x_{ij} - l_0(1 - x_{ij}) \leq \tau_j \qquad \forall i \in V_0, \forall j \in V_{N+1}, i \neq j \tag{2.4}$$

$$e_j \leq \tau_j \leq l_j \qquad \forall j \in V_{0,N+1} \tag{2.5}$$

$$0 \leq u_j \leq u_i - q_i x_{ij} + C(1 - x_{ij}) \qquad \forall i \in V_0, \forall j \in V_{N+1}, i \neq j \tag{2.6}$$

$$0 \leq u_0 \leq C \tag{2.7}$$

$$x_{ij} \in \{0,1\} \qquad \forall i \in V_0, j \in V_{N+1}, i \neq j \tag{2.8}$$

The objective function is defined in (2.1). In the mathematical model, the objective to minimize traveled distance is given. Constraints (2.2) ensure that each customer has exactly one successor, thus enforcing connectivity. Flow conservation is established by Constraints (2.3), guaranteeing that at each vertex the number of incoming arcs is equal to the number of outgoing arcs. Constraints (2.4) enforce

time feasibility for arcs leaving customers and the depot. Constraints (2.5) ensure that each vertex is visited within its time window. The formation of subtours is prevented by Constraints (2.4) and (2.5). Capacity constraints are given in (2.6) and (2.7), assuring a nonnegative load upon arrival at any vertex. Finally, binary variables are defined in (2.8).

As far as we are aware, no precise mathematical notation exists to define the desired hierarchical objective function. However, we can use a cost-based formulation to realize the hierarchical objective as follows:

$$\min \sum_{j \in V} x_{0j} \cdot M + \sum_{i \in V_0, j \in V_{N+1}, i \neq j} c_{ij} x_{ij}.$$

By setting the cost $c_{ij}$ for traveling arc $(i, j)$ to the distance $d_{ij}$ and choosing an appropriately large value for the cost $M$ of an arc leaving the depot, we obtain a hierarchical objective function. For example, the cost $M$ can be set to $2 \cdot N \cdot \max_{i \in V_0, j \in V_{N+1}}(c_{ij})$ as this is an upper bound of the travel cost of any solution.

### 2.1.2 Metaheuristic Solution Methods for the VRPTW

Metaheuristics are solution methods that use higher-level strategies to guide underlying heuristics in order to achieve a robust search of the solution space (Talbi 2009, Gendreau and Potvin 2010b). Contrary to exact solution methods, metaheuristics are approximate in the sense that they do not guarantee to find an optimal solution to the problem tackled. Their goal is to achieve a good trade-off between achieved solution quality and the computational effort spent on the problem solving.

Metaheuristics can be classified into single-solution and population-based methods (see, e.g. Blum and Roli 2003, Gendreau and Potvin 2005, Talbi 2009). Single-solution metaheuristics, also called trajectory methods, are algorithms working on a single solution at any time of the search process (Blum and Roli 2003). They encompass local-search-based methods like Variable Neighborhood Search (VNS), Iterated Local Search (ILS) or Tabu Search (TS), whose goal is to guide local search methods to overcome local optima and to efficiently explore the search space. All solution methods developed in this work, or more precisely all their components, stem from this class of metaheuristics. Many single-solution

metaheuristics have been successfully applied to the VRPTW, e.g., TS (Taillard et al. 1997, Cordeau et al. 2001), ILS (Ibaraki et al. 2005, 2008), large neighborhood search (Bent and Van Hentenryck 2004, Pisinger and Ropke 2007, Prescott-Gagnon et al. 2009), and multi-start local search (Ibaraki et al. 2005, Lim and Zhang 2007).

Population-based metaheuristics consider a set of solutions instead of a single solution at any iteration of the optimization process (Blum and Roli 2003). The most studied population-based methods for solving combinatorial optimization problems are Evolutionary Computation (EC) and Ant Colony Optimization (ACO). As no methods of this class of metaheuristics are used in this work, they are not described in further detail. However, it should be mentioned that population-based methods have also been applied to the VRPTW with great success, e.g., evolution strategies and memetic algorithms, which both belong to the class of EC methods (Mester and Bräysy 2005, Homberger and Gehring 2005, Nagata et al. 2010), and ACO (Gambardella et al. 1999).

In the following paragraphs, we describe the single-solution metaheuristic concepts relevant for this work in more detail and give references to their most successful applications to the VRPTW. We refer the reader to Gendreau et al. (2008) for a categorized bibliography of metaheuristic solution methods for the VRPTW. For a more complete view on metaheuristic solution methods, we refer to Reeves (1993), Osman and Laporte (1996), Corne et al. (1999), Voss et al. (1999), Resende and Pardalos (2002), Ribeiro and Hansen (2001), Glover and Kochenberger (2003), Blum and Roli (2003), Gendreau and Potvin (2005), Talbi (2009) and Gendreau and Potvin (2010b).

#### 2.1.2.1 Tabu Search

TS is a local-search-based metaheuristic whose principles were first proposed in Glover (1986). In each iteration, the search considers the neighborhood $\mathcal{N}(S)$ of the current solution $S$. In general, the neighborhood is defined implicitly by means of neighborhood operators, which describe modifications to the current solution. In order to be able to escape low-quality local optima, TS selects the best neighbor of the current solution at each iteration, even if it is of lower quality. In addition, recently visited solutions or attributes of the solutions are stored in a tabu list and are prohibited for some iterations called the tabu tenure. In this

way, short term cycling of the search process is avoided (Gendreau and Potvin 2005). To illustrate this, imagine a situation where the search process has reached a local optimum and the least deteriorating move is chosen. Without the tabu list, the search is trapped as the next move returns to the local optimum because it is then the best move available.

Several stopping criteria are possible but typically the search stops after a fixed number of iterations or if the best found solution has not improved for a certain number of iterations. A pseudocode overview of a TS is given in Figure 2.1 (cf., e.g., Blum and Roli 2003).

$S \leftarrow$ generateInitialSolution()
$tabuList \leftarrow \emptyset$
**while** termination criteria not satisfied **do**
  $S \leftarrow$ chooseBestOf($\mathcal{N}(S) \setminus tabuList$)
  update($tabuList$)
**end while**

**Figure 2.1:** Tabu search in pseudocode

A drawback of all solution methods that are based on local search is that they tend to be too local and thus only search relatively small regions of the solution space. Therefore, diversification techniques are often applied to improve the efficiency of TS methods, see Soriano and Gendreau (1996) and Gendreau and Potvin (2010b) for an introduction to the most commonly applied methods.

TS has successfully been applied to various combinatorial optimization problems providing near-optimal solutions in reasonable computing times (Gendreau and Potvin 2005). For a detailed description of TS and its extensions, we refer to Glover (1989, 1990), Glover and Laguna (1997, 2002), Gendreau (2001) and Gendreau (2003). For the most successful applications of TS to the VRPTW, see Garcia et al. (1994), Rochat and Taillard (1995), Potvin (1996), Taillard et al. (1997), Badeau et al. (1997), Cordeau et al. (2001) and Bräysy and Gendreau (2002).

#### 2.1.2.2 Variable Neighborhood Search

VNS, introduced by Mladenović and Hansen (1997), performs a local search on systematically changing (often nested) neighborhoods in order to escape from local optima. The idea is to switch to the next neighborhood as soon as a local

optimum for the current neighborhood is reached. Starting from the current solution $S$, the perturbation phase (also called shaking) randomly generates a new solution $S'$ in its first neighborhood and from there a local descent is performed. If the obtained local optimum $S''$ is of lower quality than the current solution, the next neighborhood is chosen to repeat the procedure. In this way, VNS explores solutions increasingly distant from the current (Mladenović and Hansen 1997). If a better solution is found or all neighborhoods have been searched, the procedure is restarted with the first neighborhood. VNS terminates if a stopping criterion like, e.g., reaching a pre-defined number of iterations, is satisfied. Figure 2.2 shows a pseudocode overview of the VNS method (cf., e.g., Hansen and Mladenović 2001b).

Define a set of neighborhood structures $\mathcal{N}_\kappa$ for $\kappa = 1, ..., \kappa_{max}$
$S \leftarrow$ generateInitialSolution()
$\kappa \leftarrow 1$
**while** termination criteria not satisfied **do**
  $S' \leftarrow$ generateRandomPoint($\mathcal{N}_\kappa(S)$) {Shaking}
  $S'' \leftarrow$ applyLocalDescent($S'$) {Local Search}
  **if** $f(S'') < f(S)$ **then**
    $S \leftarrow S''$
    $\kappa \leftarrow 1$
  **else**
    $\kappa \leftarrow \kappa + 1 \mod \kappa_{max}$
  **end if**
**end while**

**Figure 2.2:** Variable neighborhood search in pseudocode

Surveys and tutorials on VNS are presented in Hansen and Mladenović (1999, 2002, 2001a,b, 2003) and Hansen et al. (2010). VNS yields excellent results when applied to vehicle routing problems, in particular VRPs with time windows and/or multiple depots, see Bräysy (2003), Polacek et al. (2004), Tarantilis et al. (2008) and Stenger et al. (2011).

#### 2.1.2.3 Simulated Annealing

Simulated Annealing (SA) is a randomized local search method introduced by Kirkpatrick et al. (1983). With some probability, SA also accepts moves to neighboring solutions with a worse objective function value and is thus able to escape from low-quality local optima. SA is inspired by the physical annealing process of

glass or metal, which aims at generating solids with low-energy states by carefully reducing the temperature in a stepwise fashion. In the solution method context, a solution corresponds to a state and the objective function value to its energy (Talbi 2009).

In each iteration, the current solution $S$ is modified by randomly selecting a neighboring solution. Next, the objective function values $f(S')$ and $f(S)$ of the new solution $S'$ and the current solution are compared. Better solutions are always accepted, worse solutions are accepted with the Metropolis probability (Metropolis et al. 1953). The Metropolis probability depends on a parameter called the temperature $T$ and the difference between the objective function values of the current and the newly generated solution as follows:

$$p = e^{-\frac{\Delta f}{T}} = e^{-\frac{f(S')-f(S)}{T}}.$$

Basically, a move is more likely to be accepted if the temperature is high and the magnitude of the cost increase is low. The SA procedure starts with an initial temperature $T_0$ that gradually decreases according to some predefined cooling schedule. At each temperature level, a certain number of iterations are performed. Thus, the probability of accepting worse solutions "cools" down during the solution process and the method stops in a local optimum (Gendreau and Potvin 2005). The SA algorithm is presented in pseudocode in Figure 2.3 (see, e.g., Blum and Roli 2003).

$S \leftarrow generateInitialSolution()$
$T \leftarrow T_0$
**while** termination criteria not satisfied **do**
  $S' \leftarrow$ generateRandomPoint($\mathcal{N}(S)$)
  **if** $f(S') < f(S)$ **then**
    $S \leftarrow S'$
  **else**
    Accept $S'$ as new solution with probability $p(\Delta f, T)$
  **end if**
  update($T$)
**end while**

**Figure 2.3:** Simulated annealing in pseudocode

Contrary to most metaheuristics, it can be proven that SA converges to a global optimum for an infinite number of iterations (see, e.g., Nikolaev and Jacobson

2010). Unfortunately, this is not true for finite-time implementations, which significantly lowers the practical impact of this finding. For the most successful applications of SA to the VRPTW, we refer to Czech and Czarnas (2002), Li and Lim (2003), Debudaj-Grabysz and Czech (2005) and Woch and Łebkowski (2009).

### 2.1.3 Description of Benchmark Instances

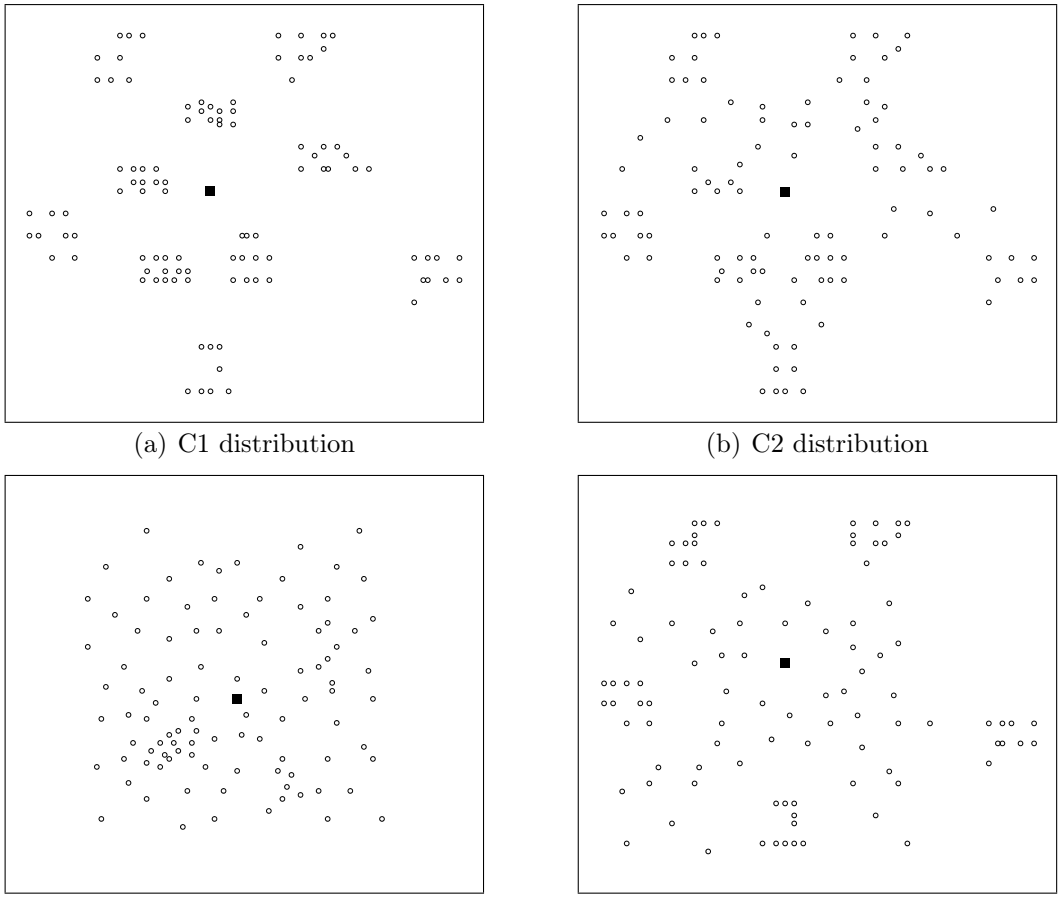
As mentioned above, the VRPTW has been studied extensively in the literature and benchmark instances have been proposed in order to make the performance of different solution methods comparable. In this section, we describe the most important benchmark problems proposed for VRPTW as they are used to assess the performance of solution methods presented in this work and also as a basis for generating the test instances for the addressed routing problems.

The best-known VRPTW instances stem from Solomon (1987). In the original work, instances with different customer numbers have been proposed (25, 50, 75, 100 customers), but in recent years only the 56 largest instances with 100 customers have received attention. In the remainder of this work, they are simply referred to as Solomon instances. They can be downloaded from the website of the Transportation Optimization Portal of SINTEF Applied Mathematics[1].

The proposed test instances were generated to represent different problem characteristics. They are divided into six groups: C1, C2, R1, R2, RC1, RC2, each of them containing between 8 and 12 test problems. The groups are based on four different geographical distributions of the customer locations: two clustered distributions (C1 and C2), one uniform (R) and one combination of clustered and uniform (RC) as shown in Figure 2.4. Filled squares mark the locations of depots, blank circles customer locations.

The instance groups differ with regard to the scheduling horizon of the depot $l_0 - e_0$ and the capacities of vehicles. Groups C1, R1 and RC1 have a relatively short scheduling horizon and low-capacity vehicles, generally resulting in solutions with a high number of short routes containing only few customers. Groups C2, R2, RC2 have a considerably longer scheduling horizon and high-capacity vehicles, resulting in solutions with a comparatively low vehicle number performing longer routes containing a higher number of customers.

[1] http://www.sintef.no/Projectweb/TOP/Problems/VRPTW/Solomon-benchmark/



(a) C1 distribution

(b) C2 distribution

(c) R distribution

(d) RC distribution

**Figure 2.4:** The distribution of customers for the different problem groups of the 100-customer Solomon VRPTW benchmark. Customers are represented as white circles, depots as black squares.

The instances within a group differ in terms of time window density (TWD), i.e., the percentage of customers with a time window $(25\%, 50\%, 75\%, 100\%)$, and time window width (TWW). TWW is defined as the average width of $(l_v - e_v)$ across all customers in the problem instance. Table 2.1 reports the TWD and TWW of the Solomon instances, which are uniquely identified with a combination of their group indicator and their number as shown in the table.

In Table 2.2, we present the best known metaheuristic solutions for the Solomon instances assuming the hierarchical objective function described above. The list is compiled from the SINTEF website and reports the number of vehicles (#Rts.) and traveled distance (TD) for each instance. Moreover, for each solution the first published article in which the respective solution was reported is given in

| | C1 | | C2 | | R1 | | R2 | | RC1 | | RC2 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| # | TWD | TWW | TWD | TWW | TWD | TWW | TWD | TWW | TWD | TWW | TWD | TWW |
| 01 | 100% | 60.76 | 100% | 160.00 | 100% | 10.00 | 100% | 115.96 | 100% | 30.00 | 100% | 120.00 |
| 02 | 75% | 61.27 | 75% | 160.00 | 75% | 10.00 | 75% | 115.23 | 75% | 30.00 | 75% | 120.00 |
| 03 | 50% | 59.90 | 50% | 160.00 | 50% | 10.00 | 50% | 117.34 | 50% | 30.00 | 50% | 120.00 |
| 04 | 25% | 60.64 | 25% | 160.00 | 25% | 10.00 | 25% | 111.80 | 25% | 30.00 | 25% | 120.00 |
| 05 | 100% | 121.61 | 100% | 320.00 | 100% | 30.00 | 100% | 240.00 | 100% | 54.33 | 100% | 223.06 |
| 06 | 100% | 156.15 | 100% | 486.64 | 75% | 30.00 | 75% | 240.00 | 100% | 60.00 | 100% | 240.00 |
| 07 | 100% | 180.00 | 100% | 612.32 | 50% | 30.00 | 50% | 240.00 | 100% | 88.21 | 100% | 349.50 |
| 08 | 100% | 243.28 | 100% | 640.00 | 25% | 30.00 | 25% | 240.00 | 100% | 112.33 | 100% | 471.93 |
| 09 | 100% | 360.00 | | | 100% | 58.89 | 100% | 349.50 | | | | |
| 10 | | | | | 100% | 86.50 | 100% | 383.27 | | | | |
| 11 | | | | | 100% | 93.10 | 100% | 471.94 | | | | |
| 12 | | | | | 100% | 117.64 | | | | | | |

**Table 2.1:** Time window density (TWD) and average time window width (TWW) of the Solomon instances

column AUT, using the following abbreviations depending on author names: **BB** (Berger and Barkaoui 2004), **BGGPT** (Badeau et al. 1997), **BVH** (Bent and Van Hentenryck 2004), **CC** (Czech and Czarnas 2002), **CLM** (Cordeau et al. 2001), **DGC** (Debudaj-Grabysz and Czech 2005), **GH** (Gehring and Homberger 1999), **GTA** (Gambardella et al. 1999), **H** (Homberger 2000), **IKMUY** (Ibaraki et al. 2005), **LL** (Li and Lim 2003), **MBD** (Mester et al. 2007), **PR** (Pisinger and Ropke 2007), **RGP** (Rousseau et al. 2002), **RT** (Rochat and Taillard 1995), **S97** (Shaw 1997), **S98** (Shaw 1998), **SSSD** (Schrimpf et al. 2000) and **WL** (Woch and Łebkowski 2009). Using the best-known solution for every instance leads to a cumulated number of vehicles (CNV) of 405 and a cumulated traveled distance (CTD) of 57180.84.

The second set of well-known benchmark instances was suggested by Gehring and Homberger (1999), who were the first authors to explicitly consider large-scale VRPTW instances. They developed a parallel hybrid evolutionary algorithm and generated large-scale instances to test their method. The Gehring and Homberger benchmarks are an extension to the Solomon instances, consisting of five sets with customer numbers of 200, 400, 600, 800 and 1000 respectively. Each set contains 60 instances, resulting in 300 instances in total. The instances are structured similarly to the Solomon instances, differing in customer distribution (R, C, RC), scheduling horizon, TWD, and TWW. A description of the instances, the instances themselves for downloading and a list of the best known solutions to the instances can be found on the SINTEF website[2].

[2]http://www.sintef.no/Projectweb/TOP/Problems/VRPTW/Homberger-benchmark/

| C | | | | R | | | | RC | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | #Rts. | TD | AUT | | #Rts. | TD | AUT | | #Rts. | TD | AUT |
| 101 | 10 | 828.94 | RT | 101 | 19 | 1645.79 | H | 101 | 14 | 1696.94 | BGGPT |
| 102 | 10 | 828.94 | RT | 102 | 17 | 1486.12 | RT | 102 | 12 | 1554.75 | BGGPT |
| 103 | 10 | 828.06 | RT | 103 | 13 | 1292.68 | LL | 103 | 11 | 1261.67 | S98 |
| 104 | 10 | 824.78 | RT | 104 | 9 | 1007.24 | MBD | 104 | 10 | 1135.48 | CLM |
| 105 | 10 | 828.94 | RT | 105 | 14 | 1377.11 | RT | 105 | 13 | 1629.44 | BB |
| 106 | 10 | 828.94 | RT | 106 | 12 | 1251.98 | MBD | 106 | 11 | 1424.73 | BB |
| 107 | 10 | 828.94 | RT | 107 | 10 | 1104.66 | S97 | 107 | 11 | 1230.48 | S97 |
| 108 | 10 | 828.94 | RT | 108 | 9 | 960.88 | BB | 108 | 10 | 1139.82 | BGGPT |
| 109 | 10 | 828.94 | RT | 109 | 11 | 1194.73 | GH | | | | |
| | | | | 110 | 10 | 1118.59 | MBD | 201 | 4 | 1406.91 | MBD |
| 201 | 3 | 591.56 | RT | 111 | 10 | 1096.72 | RGP | 202 | 3 | 1365.65 | DGC |
| 202 | 3 | 591.56 | RT | 112 | 9 | 982.14 | GTA | 203 | 3 | 1049.62 | CC |
| 203 | 3 | 591.17 | RT | | | | | 204 | 3 | 798.41 | MBD |
| 204 | 3 | 590.60 | RT | 201 | 4 | 1252.37 | GH | 205 | 4 | 1297.19 | MBD |
| 205 | 3 | 588.88 | RT | 202 | 3 | 1191.70 | RGP | 206 | 3 | 1146.32 | H |
| 206 | 3 | 588.49 | RT | 203 | 3 | 939.5 | WL | 207 | 3 | 1061.14 | BVH |
| 207 | 3 | 588.29 | RT | 204 | 2 | 825.52 | BVH | 208 | 3 | 828.14 | IKMUY |
| 208 | 3 | 588.32 | RT | 205 | 3 | 994.42 | RGP | | | | |
| | | | | 206 | 3 | 906.14 | SSSD | | | | |
| | | | | 207 | 2 | 890.61 | PR | | | | |
| | | | | 208 | 2 | 726.75 | MBD | | | | |
| | | | | 209 | 3 | 909.16 | H | | | | |
| | | | | 210 | 3 | 939.34 | MBD | | | | |
| | | | | 211 | 2 | 885.71 | WL | | | | |

**Table 2.2:** Compilation of the best known solutions for the Solomon instances. #Rts. denotes the number of vehicles in the respective solution, TD the traveled distance.

## 2.2 Constraint Handling in Local Search for the VRPTW

Restricting the search to feasible solutions often hinders a sufficient exploration of the solution space. Therefore, most successful metaheuristics for VRPTW allow infeasible solutions along the search trajectory, i.e., the inner local search is allowed to apply moves that lead to solutions violating capacity and/or time window constraints (Nagata et al. 2010). One way to guide the search towards feasible solutions is to penalize an infeasible solution according to the magnitude of the violations it causes (Talbi 2009). In this case, it is vital to calculate the amount of violations and thus the penalties in an efficient manner. In this section, we introduce the procedures for determining capacity and time window violations that are applied throughout this thesis.

### 2.2.1 Capacity Handling

We define a vehicle route $r$ as a sequence of customers $\langle v_0, v_1, \ldots, v_n, v_{n+1} \rangle$, with $v_0$ and $v_{n+1}$ representing the depot (cf. Nagata et al. 2010). A solution $S$ to a VRPTW instance is defined as a set containing $m(S)$ routes, i.e., $S = \{r_k, k = 1, \ldots, m(S)\}$. $Vert(r)$ denotes the set of vertices that are part of route $r$, $Vert(S)$ the set of vertices visited in a solution $S$.

With these definitions, the capacity violation $P_c$ of route $r$ can be calculated as:

$$P_c(r) = \max \Big( \sum_{v \in Vert(r)} q_v - C, 0 \Big).$$

The total capacity violation of a solution $S$ can then be computed from the individual violations of all contained routes:

$$P_c(S) = \sum_{k=1}^{m(S)} P_c(r_k).$$

In order to efficiently compute the change of capacity violation of a route generated by applying a local search move, we save forward and backward capacity slacks for all vertices in the affected routes. In this way, it is possible to determine capacity violations for inter-route moves in constant time $\mathcal{O}(1)$ if the conventional neighborhood operators Relocate, Exchange, Or-Opt or 2-opt* are utilized (see, e.g., Kindervater and Savelsbergh 1997, Ibaraki et al. 2005). In this work, only these operators are used so that the presented technique covers all relevant cases. Obviously, no computations for intra-route moves are necessary as the capacity of a route does not change by moving vertices inside a route.

For calculating the change of violation caused by inter-route moves, we define the forward capacity slack of the vertices inside a route $r$ as follows:

$$CAP^{\rightarrow}_{v_0} = 0$$
$$CAP^{\rightarrow}_{v_i} = CAP^{\rightarrow}_{v_{i-1}} + q_{v_i}, \quad i = 1, \ldots, n+1.$$

The backward penalty slack is computed analogously:

$$CAP^{\leftarrow}_{v_{n+1}} = 0$$
$$CAP^{\leftarrow}_{v_i} = CAP^{\leftarrow}_{v_{i+1}} + q_{v_i}, \quad i = 0, \ldots, n.$$

Now, let $CAP^{\rightarrow}_v$ and $CAP^{\leftarrow}_v$ be known for all vertices $v \in \mathit{Vert}(S)$ of a current solution $S$. Then, the capacity violation for inter-route moves can be calculated in constant time using the following two rules:

- If a route $r_{new} = \langle v_0, \ldots, x, y, \ldots, v_{n+1} \rangle$ is constructed from two partial paths $\langle v_0, \ldots, x \rangle$ and $\langle y, \ldots, v_{n+1} \rangle$, the capacity violation of the route can be determined as follows:

$$P_c(r_{new}) = \max\left(CAP^{\rightarrow}_x + CAP^{\leftarrow}_y - C, 0\right).$$

- If a route $r_{new} = \langle v_0, \ldots, x, v, y, \ldots, v_{n+1} \rangle$ is constructed by inserting a vertex $v$ in between two partial paths $\langle v_0, \ldots, x \rangle$ and $\langle y, \ldots, v_{n+1} \rangle$, the capacity violation of the new route can be calculated as:

$$P_c(r_{new}) = \max\left(CAP^{\rightarrow}_x + CAP^{\leftarrow}_y + q_v - C, 0\right).$$

### 2.2.2 Time Window Handling

Handling time window violations is a more complex task as the removal or insertion of a vertex affects the time window violations at all subsequent vertices. The arrival time at each vertex has to be known to compute the violation. Given a route $r$, the earliest departure time at the depot $a_{v_0}$, the earliest start time of service $a_{v_i}$ at every vertex $v_i, i = 1, \ldots, n$, and the earliest arrival time at the depot $a_{v_{n+1}}$ are defined as follows (cf. Nagata et al. 2010):

$$a_{v_0} = e_0$$
$$a_{v_i} = \max\left(a_{v_{i-1}} + s_{v_{i-1}} + t_{v_{i-1}v_i}, e_{v_i}\right) \quad i = 1, \ldots, n+1.$$

A time window violation at a customer $v_i$ occurs if the vehicle arrives after the latest arrival time $l_{v_i}$. Thus, a route is feasible with respect to time window violations if $a_{v_i} \leq l_{v_i}, i = 0, \ldots, n+1$. Traditionally, the difference between

arrival time and latest feasible arrival time is cumulated along the entire route in order to compute the time window penalty of a route (see, e.g., Badeau et al. 1997, Berger and Barkaoui 2004, Bouthillier and Crainic 2005, Ibaraki et al. 2005, 2008):

$$P_{tw}^{traditional}(r) = \sum_{i=0}^{n+1} \max\left(a_{v_i} - l_{v_i}, 0\right).$$

The time window violation of a solution $S$ can be computed from the violations in the routes of $S$:

$$P_{tw}(S) = \sum_{k=1}^{m(S)} P_{tw}(r_k). \tag{2.9}$$

Although sophisticated methods have been proposed to determine the change of time window violation caused by applying one of the conventional neighborhood operators introduced above (see, e.g., Irnich et al. 2006, Irnich 2008), it still takes $\mathcal{O}(n)$ time to compute the change. More precisely, if two routes $r_1$ and $r_2$ are affected by a move, the computational complexity of calculating the change in time window violation of the routes is $\mathcal{O}(|\mathit{Vert}(r_1)| + |\mathit{Vert}(r_2)|)$ (Nagata 2007). To improve efficiency, all algorithms presented in this work use the *time travel approach* proposed by Nagata et al. (2010) for penalizing time window violations, which is described in the following.

#### 2.2.2.1 The Time Travel Approach

Nagata et al. (2010) introduce a new, highly efficient approach to calculate time window violations, which is based on the concept of time travel. To put it short, if a time window violation occurs at a certain customer, a penalty of the actual arrival time minus the latest feasible arrival time is assigned to the route. However, the calculation of violations for the following customers is executed as if a travel back in time to the latest feasible arrival time had taken place. In this way, none of the subsequent customer visits are affected by the actual delay at the customer.

This approach has two important advantages compared to the traditional approach of propagating time window violations along the route. First, it allows to

evaluate potential time window violations for inter-route moves in constant time $\mathcal{O}(1)$ for the conventional neighborhood operators. This significantly speeds up the calculation compared to the traditional approach described above. Second, the approach penalizes time window violations only at customers where they originate and feasible parts of routes are no longer penalized by earlier time window violations.

To formally explain the approach of Nagata et al. (2010), we define for a route $r$ the extended earliest departure time at the depot $\tilde{a}_{v_0}$, the extended earliest start time of service $\tilde{a}_{v_i}$ at every vertex $v_i, i = 1, \ldots, n$, and the extended earliest arrival time at the depot $\tilde{a}_{v_{n+1}}$ as follows:

$$
\begin{aligned}
&\tilde{a}_{v_0} = e_0, \tilde{a}'_{v_0} = e_0 \\
&\tilde{a}'_{v_i} = \tilde{a}_{v_{i-1}} + s_{v_{i-1}} + t_{v_{i-1}v_i}, \quad i = 1, \ldots, n{+}1 \\
&\tilde{a}_{v_i} = \begin{cases} \max\left(\tilde{a}'_{v_i}, e_{v_i}\right) & \text{if } \tilde{a}'_{v_i} \leq l_{v_i}, \\ l_{v_i} & \text{if } \tilde{a}'_{v_i} > l_{v_i}, \end{cases} \quad i = 1, \ldots, n{+}1.
\end{aligned}
$$

The extended earliest start time of service $\tilde{a}_{v_i}$ at vertex $v_i$ describes the start of service after waiting for the time window at $v_i$ to open or after traveling back in time in case of a time window violation. The definition uses the extended earliest arrival time $\tilde{a}'_{v_i}$ at vertex $v_i$, which describes the arrival time at vertex $v_i$ before waiting for the time window to open or before traveling back in time in case a time window is violated. In the latter case, a travel back in time to the latest feasible arrival time $l_{v_i}$ is assumed for which a penalty $\tilde{a}'_{v_i} - l_{v_i}$ has to be paid.

The time window penalty of a route $r$ can be calculated as follows:

$$
P_{tw}(r) = \sum_{i=0}^{n+1} \max\left(\tilde{a}'_{v_i} - l_{v_i}, 0\right),
$$

and the total time window penalty of a solution can be determined using Equation (2.9).

Figure 2.5 demonstrates the difference between the time travel approach and traditional time window handling as explained above, using an example route $\langle v_0, v_1, v_2, v_3, v_4, v_5, v_{n+1} \rangle$ with 5 customers. Lines represent the time flow and brackets are used to indicate time windows of customers and depot. Travel between customers is represented by dotted lines, waiting times at customer loca-

tions as zig-zagged lines and service times as dashed lines. Time window penalties are marked by arrows. A blank circle describes the arrival time at a customer, a filled circle the begin of service. Note that for the traditional approach the latter corresponds to the $a_{v_i}$ variables and for the time travel approach to the $\tilde{a}_{v_i}$ variables.

In the traditional approach depicted in Figure 2.5(a), the time window violation at customer $v_2$ is propagated along the remaining route, leading to time window violations at customers $v_3$ and $v_4$, although the partial route $\langle v_2, v_3, v_4 \rangle$ is a feasible sequence. The time travel approach in Figure 2.5(b) is able to identify the feasible partial path. The time window of customer $v_2$ is still violated, but after paying the penalty for traveling back in time, the time windows at $v_3$ and $v_4$ are no longer violated.

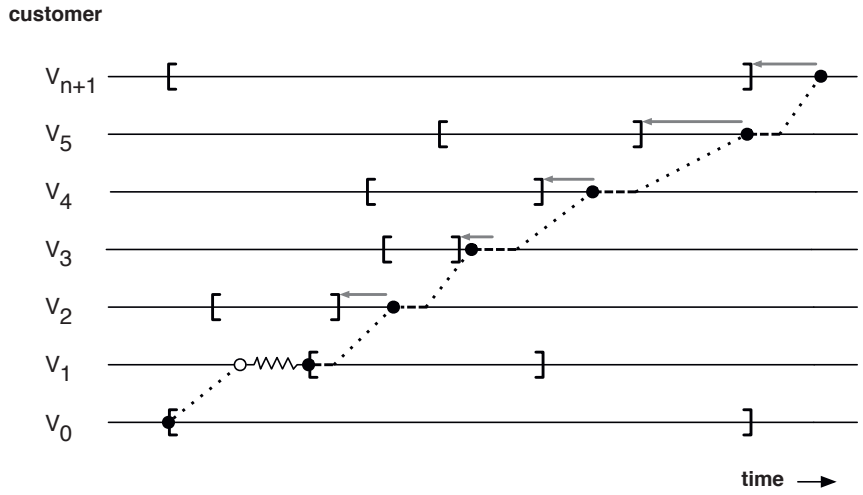
As mentioned above, the time travel approach allows for the efficient calculation of time window penalties. This is achieved by means of slack variables similar to the capacity slacks defined above. The forward time window penalty slack $\mathrm{TW}^{\rightarrow}_{v_i}$ of vertex $v_i$ is the time window violation that occurs in the partial route from $v_0$ to $v_i$. More precisely, it is the penalty that the vehicle must pay on the way from $v_0$ to $v_i$ in order to be able to start service at customer $v_i$ at time $\tilde{a}_{v_i}$:

$$\mathrm{TW}^{\rightarrow}_{v_i} = \sum_{j=0}^{i} \max\left(\tilde{a}'_{v_j} - l_{v_j}, 0\right), \quad i = 0, \ldots, n+1.$$

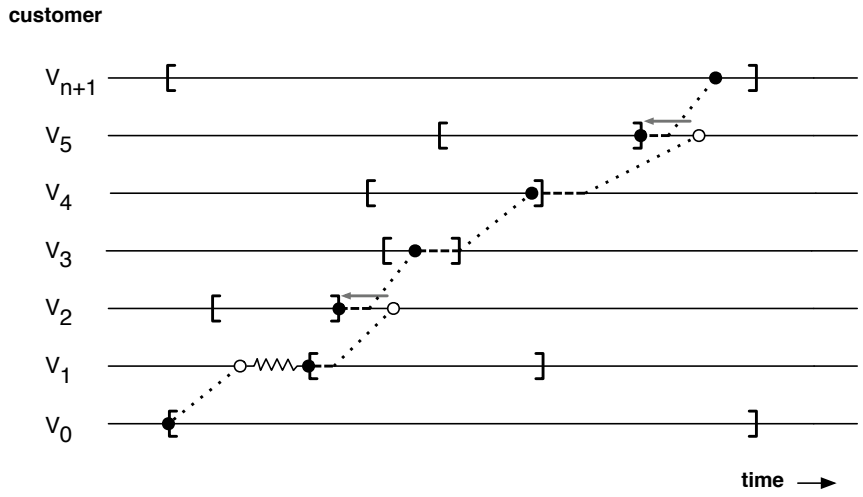
In order to be able to determine the analogical backward penalty slack, the extended latest arrival time at the depot $\tilde{z}_{v_{n+1}}$, the extended latest start time of service at customer $v_i$, $\tilde{z}_{v_i}$ and the extended latest departure time at the depot $\tilde{z}_{v_0}$ are defined recursively as follows:

$$\begin{aligned}
&\tilde{z}_{v_{n+1}} = l_0, \tilde{z}'_{v_{n+1}} = l_0 \\
&\tilde{z}'_{v_i} = \tilde{z}_{v_{i+1}} - t_{v_i v_{i+1}} - s_{v_i}, \quad i = 0, \ldots, n \\
&\tilde{z}_{v_i} = \begin{cases} \min\left(\tilde{z}'_{v_i}, l_{v_i}\right) & \text{if } \tilde{z}'_{v_i} \geq e_{v_i}, \\ e_{v_i} & \text{if } \tilde{z}'_{v_i} < e_{v_i}, \end{cases} \quad i = 0, \ldots, n.
\end{aligned}$$

The backward time window penalty slack represents the time window violation occurring on the partial route from customer $v_i$ to $v_{n+1}$ even if the vehicle starts




(a) Traditional




(b) Time travel

**Figure 2.5:** Comparison of approaches for handling time windows: (a) traditional vs. (b) time travel (cp. Nagata et al. 2010). Brackets denote time windows, dotted lines travel between customers, zig-zagged lines waiting times, dashed lines service times, blank circles the arrival time at a customer and filled circle the begin of service. Time window penalties are shown as arrows.

the service at customer $v_i$ at the earliest possible moment $e_{v_i}$:

$$\mathrm{TW}_{v_i}^{\leftarrow} = \sum_{j=i}^{n+1} \max\left(e_{v_j} - \tilde{z}'_{v_j}, 0\right), \quad i = 0, \ldots, n{+}1.$$

Figure 2.6 illustrates the definition of the extended earliest start time of service $\tilde{a}_{v_i}$ and extended latest start time of service $\tilde{z}_{v_i}$. The arrows indicate the forward and backward penalties that are cumulated in the forward and backward time window penalty slacks.

Let the variables $\tilde{a}_{v_i}$, $\tilde{z}_{v_i}$, $\mathrm{TW}_{v_i}^{\rightarrow}$ and $\mathrm{TW}_{v_i}^{\leftarrow}$ be known for all routes in the current solution. Nagata et al. (2010) offer the following two rules to efficiently calculate the change in time window violation caused by an inter-route local search move:

- If a route $r_{new} = \langle 0, \ldots, x, y, \ldots, n{+}1 \rangle$ is constructed from two partial paths $\langle 0, \ldots, x \rangle$ and $\langle y, \ldots, n{+}1 \rangle$, the time window penalty can be determined by:

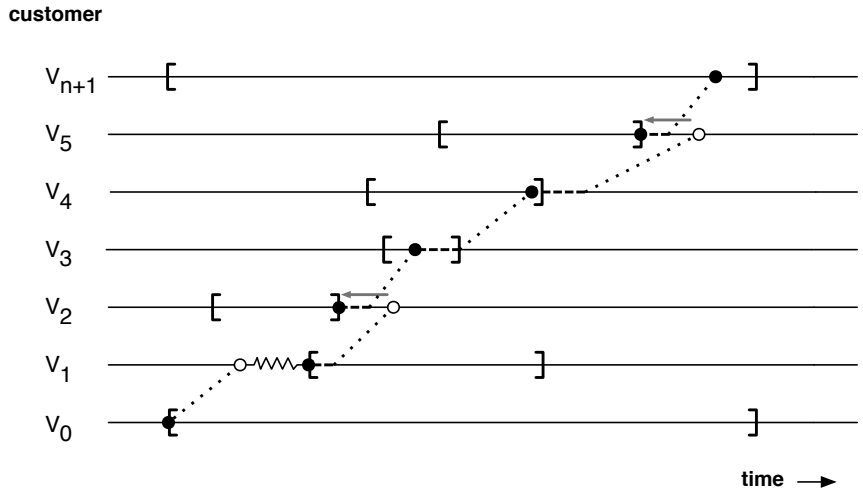
$$P_{tw}(r_{new}) = \mathrm{TW}_x^{\rightarrow} + \mathrm{TW}_y^{\leftarrow} + \max\left(\tilde{a}_x + s_x + t_{xy} - \tilde{z}_y, 0\right). \tag{2.10}$$

- If a route $r_{new} = \langle v_0, \ldots, x, v, y, \ldots, v_{n+1} \rangle$ is constructed by inserting a vertex $v$ in between two partial paths $\langle v_0, \ldots, x \rangle$ and $\langle y, \ldots, v_{n+1} \rangle$, the time window violation of the newly generated route is:

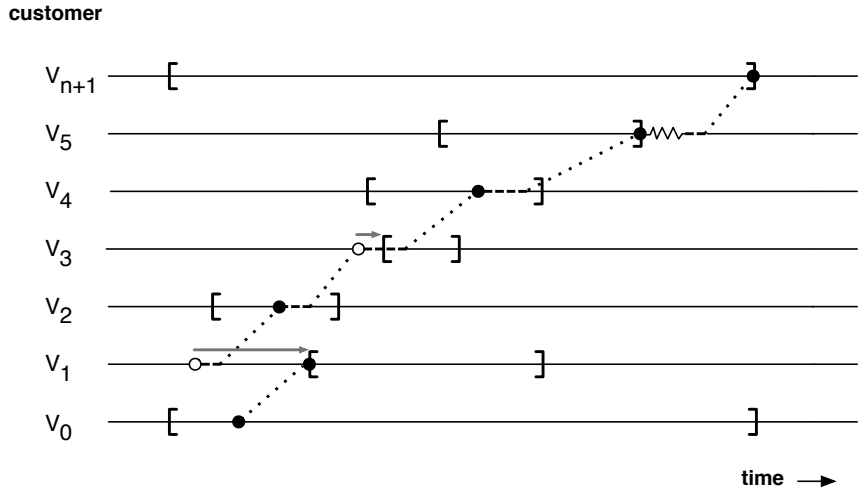
$$P_{tw}(r_{new}) = \mathrm{TW}_x^{\rightarrow} + \mathrm{TW}_y^{\leftarrow} + \max\left(\tilde{a}_x + s_x + t_{xv} - (\tilde{z}_y - t_{vy} - s_v), 0\right). \tag{2.11}$$

In principle, the changes in time window penalties of inter-route moves generated with the conventional operators 2-opt*, Relocate, Exchange and Or-Opt can be calculated in constant time using the presented rules. However, we found that the second rule presented by Nagata et al. (2010) yields incorrect results under certain conditions. In the worst case, this leads to a situation where infeasible solutions are assigned a time window violation of 0. In the following section, we identify the cases in which the given equation is wrong and present the corrected formula for each case. Simple examples are used to illustrate the calculations.

Note that for intra-route moves, there are no distinct parts of routes to be merged and thus the rules are not applicable. However, even then, it is generally not required to walk through the modified route completely. Instead, it is suffi-




(a) Forward penalties




(b) Backward Penalties

**Figure 2.6:** Illustration of the definition of (a) extended earliest start time of service $\tilde{a}_{v_i}$ and (b) extended latest start time of service $\tilde{z}_{v_i}$ depicted as filled circles (cp. Nagata et al. 2010). The calculation of the associated penalties is described by the arrows.

cient to determine the first and last customer that is influenced by a move and to recalculate the penalty variables for the two customers and those in between. The calculation for intra-route moves still has a computational complexity of $\mathcal{O}(n)$ (Nagata et al. 2010).

#### 2.2.2.2 Correction of the Time Travel Approach

We show that the rule presented in Equation (2.11) does not cover all cases concerning the time window characteristic of the customer $v$ to be inserted. To this end, we distinguish three insertion cases:

1. Customer $v$ is reached within its time window, i.e., we neither have to wait nor do we have to time travel backward.

2. Customer $v$ is reached before the start of the associated time window, i.e., we have to wait before service can be started.

3. Customer $v$ cannot be reached before the end of its time window, i.e., we have to time travel backward at $v$.

A simple example with customer set $V = \{1, 2, 3, 4\}$ and a single depot 0 is used to show in which cases Equation (2.11) fails. Let us assume the depot to have a scheduling horizon of $[0, 100]$. All customers have a service time $s_i = 10, i \in V$ $(s_0 = 0)$ and the distances $d_{ij},\ i, j \in V \cup \{0\}, j \neq i$ are all 10. Travel times are assumed to be equivalent to traveled distances. In our example, we use the initial route $\langle 0, 1, 2, 0\rangle$ as starting point for all insertion cases. The remaining customers are individually inserted into the route between customers 1 and 2 to illustrate the cases where an error is produced. The time window handling related values for all customers are given in Table 2.3. In all cases, we compare the time window penalty given by Equation (2.11) to a primitive calculation that has to walk through the complete newly created route. Contrary to traditional time window handling, this primitive calculation uses backward time travel to penalize violations in order to produce results that are comparable to those obtained by Equation (2.11). For all insertion cases, we provide the corrected formula valid for the specific case and integrate these formulas into one equation that covers all cases afterwards.

| $i$ | $e$ | $l$ | $a$ | $\tilde{a}$ | $\tilde{a}'$ | $\tilde{z}$ | $\tilde{z}'$ | $\mathrm{TW}_i^{\rightarrow}$ | $\mathrm{TW}_i^{\leftarrow}$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 10 | 10 | 10 | 10 | 10 | 20 | 0 | 0 |
| 2 | 35 | 40 | 35 | 35 | 30 | 40 | 80 | 0 | 0 |
| 3 | 35 | 45 | – | – | – | – | – | – | – |
| 4 | 10 | 15 | – | – | – | – | – | – | – |

**Table 2.3:** Time window handling related values for all customers in the example. Variables before insertion of the customers 3 and 4 into the initial route $\langle 0, 1, 2, 0\rangle$.

**2.2.2.2.1 Case 1: Reach Customer within Time Window** If the customer $v$ to be inserted is reached within its time window, Equation (2.11) provides the correct result. The following step by step derivation illustrates this. Given the time window violation up to customer $x$ ($\mathrm{TW}_x^{\rightarrow}$), we start service there at $\tilde{a}_x$ and arrive at customer $v$ at $\tilde{a}_x + s_x + t_{xv}$. We reach $v$ within its time window and the arrival time at $y$ can be calculated by adding $s_v + t_{vy}$. For the time window violation $\mathrm{TW}_y^{\leftarrow}$ to be valid for completing the route from $y$ onwards, we have to arrive at $y$ before $\tilde{z}_y$. If the arrival time at $y$ is later, we have to add the difference as time window violation.

Thus, if arrival at customer $v$ is within the given time window, i.e., $e_v \leq \tilde{a}_x + s_x + t_{xv} \leq l_v$, the following holds (which is obviously equivalent to Equation (2.11)):

$$P_{tw}(r) = \mathrm{TW}_x^{\rightarrow} + \mathrm{TW}_y^{\leftarrow} + \max\left(\tilde{a}_x + s_x + t_{xv} + s_v + t_{vy} - \tilde{z}_y, 0\right).$$

**2.2.2.2.2 Case 2: Wait at Customer** If we have to wait to start the service at the customer $v$ to be inserted, Equation (2.11) does not always provide the correct result as the example of route $r_2 = \langle 0, 1, 3, 2, 0\rangle$ shows. The primitive calculation is as follows: We arrive at customer 1 at time 10, leave at 20, arrive at customer 3 at 30, have to wait till 35, leave at 45 and arrive at customer 2 at 55. We have to time travel back 15 units, leave customer 2 at 50 and arrive at the depot at 60. The route $r_2$ has a time window violation of the amount we had to time travel, i.e., $\mathrm{TW}(r_2) = 15$. According to Equation (2.11), the violation is $\mathrm{TW}(r_2) = 0 + 0 + \max(10 + 10 + 10 - (40 - 10 - 10), 0) = 10$, i.e., the amount of violation at $y$ that is caused by the wait at $v$ is not detected. Note that in the worst case, this means that the equation returns a time window violation of 0 for an infeasible route with a time window violation at customer $y$.

To correct the formula, we make the following considerations: At customer $x$ with $\mathrm{TW}_x^{\rightarrow}$, we start service at $\tilde{a}_x$ and arrive at customer $v$ at $\tilde{a}_x + s_x + t_{xv}$. We reach $v$ before its time window, wait until $e_v$ and the arrival time at $y$ can be calculated by adding $s_v + t_{vy}$ to $e_v$. As mentioned above, we have to arrive at $y$ before $\tilde{z}_y$ for the time window violation $\mathrm{TW}_y^{\leftarrow}$ for completing the route from $y$ onwards to be valid. If the arrival time at $y$ is later, we add the difference as time window violation:

$$P_{tw}(r) = \mathrm{TW}_x^{\rightarrow} + \mathrm{TW}_y^{\leftarrow} + \max\left(e_v + s_v + t_{vy} - \tilde{z}_y, 0\right), \text{ if } \tilde{a}_x + s_x + t_{xv} < e_v. \quad (2.12)$$

Note that Equation (2.11) is only incorrect for the considered case, if the wait at the inserted customer causes an arrival after the extended latest start time of the service at the following customer. Otherwise, it holds that $e_v + s_v + t_{vy} - \tilde{z}_y < 0$, and since $\tilde{a}_x + s_x + t_{xv} < e_v$, it follows that $\tilde{a}_x + s_x + t_{xv} - (\tilde{z}_y - t_{vy} - s_v) < 0$. Consequently, Equations (2.11) and (2.12) can both be simplified to $P_{tw}(r) = \mathrm{TW}_x^{\rightarrow} + \mathrm{TW}_y^{\leftarrow}$ in this case.

**2.2.2.2.3 Case 3: Arrive Late at Customer** The example route $r_3 = \langle 0, 1, 4, 2, 0\rangle$ shows that Equation (2.11) can also yield incorrect results if the time window at the inserted customer $v$ is violated. The primitive calculation is: We arrive at customer 1 at time 10, leave at 20, arrive at customer 4 at 30, have to time travel backwards 15 units, leave at 25 and arrive at customer 2 at 35, leave at 45 and arrive at the depot at 55. Consequently, route $r_3$ has a time window violation of $\mathrm{TW}(r_3) = 15$. Equation (2.11) returns an incorrect amount of violation of $\mathrm{TW}(r_3) = 0 + 0 + \max(10 + 10 + 10 - (40 - 10 - 10), 0) = 10$. Note that again, in the worst case, a solution can be marked feasible according to Equation (2.11) although the time window at customer $v$ is violated.

The correct equation for this case can be derived as follows: At customer $x$ with $\mathrm{TW}_x^{\rightarrow}$, we start service at $\tilde{a}_x$ and arrive at customer $v$ at $\tilde{a}_x + s_x + t_{xv}$. We reach $v$ after its time window, time travel back to $l_v$ and add this as time window violation. The arrival time at $y$ can be calculated by adding $s_v + t_{vy}$ to $l_v$. Thus, the following holds if we arrive late at customer $v$, i.e., if $\tilde{a}_x + s_x + t_{xv} > l_v$:

$$P_{tw}(r) = \mathrm{TW}_x^{\rightarrow} + \mathrm{TW}_y^{\leftarrow} + (\tilde{a}_x + s_x + t_{xv} - l_v) + \max\left(l_v + s_v + t_{vy} - \tilde{z}_y, 0\right). \quad (2.13)$$

Note that Equation (2.11) is only incorrect for the considered case if $l_v + s_v + t_{vy} - \tilde{z}_y < 0$, i.e., if we arrive before the extended latest start time of service at customer $y$. Otherwise, Equation (2.13) is $P_{tw}(r) = \mathrm{TW}_x^{\rightarrow} + \mathrm{TW}_y^{\leftarrow} + \tilde{a}_x + s_x + t_{xv} - l_v + l_v + s_v + t_{vy} - \tilde{z}_y$ and thus equal to Equation (2.11).

**2.2.2.2.4 Corrected Proposition Covering All Cases** Summing up, the corrected rule is: If a route $r = \langle v_0, \ldots, x, v, y, \ldots, v_{n+1} \rangle$ is generated from two partial paths $\langle v_0, \ldots, x \rangle$ and $\langle y, \ldots, v_{n+1} \rangle$ by inserting customer $v$, the time window penalty of this route is computed by:

$$P_{tw}(r) = \begin{cases} \mathrm{TW}_x^{\rightarrow} + \mathrm{TW}_y^{\leftarrow} + & \\ \max\left(\tilde{a}_x + s_x + t_{xv} + s_v + t_{vy} - \tilde{z}_y, 0\right) & \text{if } e_v \leq \tilde{a}_x + s_x + t_{xv} \leq l_v \\ \mathrm{TW}_x^{\rightarrow} + \mathrm{TW}_y^{\leftarrow} + \max\left(e_v + s_v + t_{vy} - \tilde{z}_y, 0\right) & \text{if } \tilde{a}_x + s_x + t_{xv} < e_v \\ \mathrm{TW}_x^{\rightarrow} + \mathrm{TW}_y^{\leftarrow} + \left(\tilde{a}_x + s_x + t_{xv} - l_v\right) + & \\ \max\left(l_v + s_v + t_{vy} - \tilde{z}_y, 0\right) & \text{if } \tilde{a}_x + s_x + t_{xv} > l_v. \end{cases}$$

This can be reduced to the following equation that covers all three cases:

$$\begin{aligned} P_{tw}(r) =& \mathrm{TW}_x^{\rightarrow} + \mathrm{TW}_y^{\leftarrow} + \max\left(\tilde{a}_x + s_x + t_{xv} - l_v, 0\right) \\ &+ \max\left(\max\left(\min\left(\tilde{a}_x + s_x + t_{xv}, l_v\right), e_v\right) + s_v + t_{vy} - \tilde{z}_y, 0\right). \end{aligned} \tag{2.14}$$

An important thing to note is that the original formula systematically underestimates the time window violation of a local search move, i.e., the time window violation calculated with the original formula is equal or smaller than the violation computed with the corrected formula for all cases.

We use numerical studies detailed in Appendix A to show that a significant proportion of the evaluations performed by a TS for VRPTW falls under the identified two cases yielding incorrect results. Moreover, we demonstrate that the incorrect time window handling has a significant negative impact on the solution quality of the TS. The corrections presented above and the numerical experiments in Appendix A are available in similar form as working paper (Schneider, Sand and Stenger 2012) and have been submitted to an international journal.

## 2.3 Routing with Driver Learning Aspects

In recent years, the research trend goes towards VRPs representing real-world characteristics. Considering the case of the small package shipping (SPS) industry, companies highly valuate driver familiarity with routes and customers (Groër et al. 2009), but relatively few VRP approaches exist that account for this kind of consistency requirements (Wong 2008). We briefly discuss the relevant literature.

Wong and Beasley (1984) divide the complete delivery area into a number of fixed service territories to be serviced by a single vehicle. The areas are determined based on historical demand data. The basic idea of their approach is to aggregate those customers into the same service territory that often appear together on the same route during an initialization phase of several sample days. On each of these sample days, a classical VRP is solved independently without considering any consistency requirements. Their approach is quite simple and provides strong familiarity benefits as the territories are completely fixed. However, the inflexibility of the territories leads to significant problems when demand varies strongly as routes are often infeasible due to capacity constraints. Other early works addressing consistency requirements by means of fixed routes or areas are Christofides (1971) and Beasley (1984).

Zhong et al. (2007) present a two-stage method for solving large-scale vehicle dispatching problems with uncertain customer locations and demand. In the strategic phase, customers are first aggregated in "cells" and a percentage of cells are assigned to "core areas" that serve as service territories. Core areas are always visited by the same driver. Cells located between core areas and those included in the "flex zone" around the depot remain unassigned. In the operational phase, routes within the core areas are constructed and the unassigned cells are inserted at the lowest cost in order to generate the daily routes. The insertion cost incorporates a driver learning model that explicitly considers the familiarity of a driver with a certain cell. Numerical studies show that the approach is well suited to balance the tradeoff between driver familiarity and routing flexibility.

Haughton (2008) studies the effect of exclusive territory assignment on routing efficiency against the benchmark of daily route reoptimization (RR), where drivers are flexibly assigned based on the day-to-day demand situation. The possibility of territory sharing is considered, i.e., a team of drivers visits a territory in order to reduce the negative impact of exclusive assignment on routing efficiency

by means of the pooling effect. In a simulation environment, the influence of the variance of customer demand, vehicle capacity and the size of the team of drivers assigned to each territory on the routing efficiency is investigated. In an earlier work, Haughton (2007) considers assignment rules that operations managers follow with the goal of increasing the familiarity of drivers with the visited customers and analyzes them by means of a statistical model.

Haugland et al. (2007) study the problem of designing fixed service territories for a stochastic VRP. For this two-stage (partitioning and routing) problem, they present a TS and a multistart heuristic and perform numerical studies on classical VRP benchmarks instances that they adapted to their problem. The tests show the superiority of the TS over the multistart heuristic. Further works addressing districting problems are Daganzo and Erera (1999), Erera (2000), Ouyang (2007) and Carlsson (2011).

Groër et al. (2009) introduce the Consistent VRP (ConVRP), a multi-period routing problem that considers service consistency requirements. The problem requires that "the same driver visits the same customers at roughly the same time on each day that these customers need service", in addition to the traditional constraints on capacity and route length. They develop a two-phase algorithm based on record-to-record travel, which first constructs a template route and uses the latter to generate the daily routes by removing non-occurring customers and inserting new ones.

The template route is similar to a priori routes (see, e.g., Campbell and Thomas 2008) and is based on a simple precedence principle. The principle states that if two customers $a$ and $b$ are served by the same vehicle on a specific day, then the vehicle that serves them and the order in which they are served must be the same on all days on which they both require service. Numerical studies on a number of generated and one real-world data set show that their approach is able to produce routes that fulfill the consistency requirement to a high degree. However, compared to the results of a standard heuristic VRP solution, travel distance and the number of vehicles increase slightly.

Sungur et al. (2010) present a model for the courier delivery problem, a multi-day VRP with soft time windows, using robust optimization and scenario-based stochastic programming to represent uncertainty in service time and probabilistic customers. The model generates a master plan and daily routes with the weighted

objective of maximizing customer coverage and similarity between routes while minimizing route duration and earliness/lateness penalties. An insertion heuristic enhanced by a TS improvement method is presented and tested in numerical studies. The results show that compared to an RR approach, their method improves route similarity and lateness penalties at the expense of increased total travel time. Moreover, they are able to outperform current industry practice on two real-world data sets.

Smilowitz et al. (2012) present a method of quantifying the effect of workforce management in route construction, i.e., to place a value on the benefit of visiting a customer repeatedly with the same driver. Thus, they are able to balance the resulting consistency benefits against additional routing costs like, e.g., increased distance. They propose three different periodic vehicle routing problems (PVRPs) that include consistency metrics as a part of the objective function to be compared against a base PVRP minimizing traveled distance. Similar metrics were considered in Francis et al. (2007), however, they were calculated a posteriori to evaluate routing solutions and the presented models did not attempt to optimize these measures.

To solve the PVRPs, an adapted TS is developed and studies are carried out on the multi-day ConVRP benchmarks proposed by Groër et al. (2009). They demonstrate that with the right weighting of consistency metrics, solutions can be obtained that only slightly deteriorate the traveled distance while achieving the consistency metric used in the model to a significantly higher extent. This shows that by adding workforce management metrics to the objective function, an appropriate balance can be obtained between travel cost and consistency metrics. On the other hand, solving the base PVRP and applying post-processing steps to increase consistency does not lead to convincing results.

Another recent approach to address consistency requirements in VRPs is presented by Coelho et al. (2011), who present an inventory routing model that integrates consistency measures, among them driver consistency, and develop a matheuristic to solve the model. As mentioned in Chapter 1, apart from Sungur et al. (2010), none of the above presented works considers the existence of time windows. Sungur et al. (2010) only account for soft time windows and, to the best of our knowledge, no method based on service territories deals with time requirements.

In Chapter 4, we study the Vehicle Routing Problem with Time Windows and Driver-Specific Times (VRPTWDST), a vehicle routing problem that models different levels of driver knowledge using driver-specific travel and service times. Zhong et al. (2007) were the first to integrate driver learning aspects by means of driver-specific costs. Their cell routing heuristic estimates the cost of inserting a cell into a specific driver's route by means of a learn factor that describes the driver's performance level for the considered cell. Schneider, Doppstadt, Sand, Stenger and Schwind (2010) proposed driver-specific times to consider different levels of driver knowledge in time-definite route planning time. The goal here was to design efficient vehicle routes, which have an incentive to visit each driver's familiar customers due to the reduced times, but are not restricted by service territories. In this way, a better tradeoff between routing flexibility and consistency is pursued.

Schneider, Doppstadt, Stenger and Schwind (2010) address a VRP with stochastic and driver-specific travel and service times with deadlines for the service at each customer. An ACO algorithm is presented as solution method: Each ant represents a single driver and creates a route dependent on driver-specific heuristic information and pheromone values that are traded off against each other. In Sand et al. (2011), the price of anarchy for several routing problems is studied, using a TS method as centralized approach and a multi-agent system as decentralized approach. Both methods are able to handle driver-specific times.

The VRPTWDST is strongly related to the well-known class of VRP with heterogeneous fleet (Baldacci et al. 2008), and following problems similar to the VRPTWDST have been studied in this strand. In an early work, Ferland and Michelon (1988) address a vehicle scheduling problem accounting for vehicles of different types with varying travel and service times. They show how to extend some heuristic and exact methods for the standard vehicle scheduling problem to be able to deal with several vehicle types. No computational experiments are reported. Dondo and Cerdá (2006) study a dynamic vehicle routing problem with time windows and later a multi-depot VRPTW with heterogeneous fleet in Dondo and Cerdá (2007). Both problem formulations account for driver-specific times, however, all tests were conducted on problem instances with vehicle-independent times.

## 2.4 Routing Electric Vehicles

In this section, we briefly review the literature related to the routing of electric vehicles addressed in Chapter 5 of this work.

The use of battery electric vehicles (BEVs) requires the integration of distance constraints depending on battery charge. Distance constraints have commonly been used in VRPs to include working hour restrictions. Here, the duration is related to route length by an average speed. Due to the widespread availability of petrol stations and the large cruising range of gasoline powered vehicles, distance constraints, however, have scarcely attracted interest as pure range (fuel) constraints. Some works on military issues propose concepts to extend the length of vehicle chains when fuel can be transferred between vehicles (Mehrez and Stern 1985, Melkman et al. 1986).

Our E-VRPTW routing model is closely related to an extension of the Multi-Depot VRP (MDVRP) described in Crevier et al. (2007). The MDVRP itself is a well-known VRP variant, where vehicles are located at several locally disperse depots and each route has to end at the depot it originated from. For a more detailed introduction to the MDVRP the reader is referred to Crainic et al. (2012), who present a hybrid genetic algorithm that is the most successful solution method for MDVRP proposed in the literature.

The MDVRP inter-depot routes (MDVRPI), the extension by Crevier et al. (2007), is motivated by the deliveries of a grocery in Montreal. The model considers intermediate depots at which vehicles can be replenished with goods during the course of a route. To solve the MDVRPI, Crevier et al. (2007) present a heuristic procedure that combines ideas from adaptive memory programming, described in Rochat and Taillard (1995), TS and integer programming. More precisely, the problem is split into three subproblems: an MDVRP, a VRP and an inter-depot subproblem, for which solutions are determined by means of a TS heuristic and saved in a solution pool. The generated routes are subsequently merged by means of a set-partitioning algorithm, followed by an improvement phase. Although the multi-depot case is described, all proposed benchmark instances consider only one depot at which the vehicle fleet is stationed.

Therefore, Tarantilis et al. (2008) rename the problem to VRP with Intermediate Replenishment Facilities (VRPIF). They propose a hybrid guided local search heuristic that follows a three-step procedure. First, an initial solution

is constructed by means of a cost-savings heuristics. Second, a VNS algorithm is applied using a TS in the local search phase, instead of a greedy procedure. Third, the solution is further improved by means of a guided local search. In numerical tests performed on available benchmark instances, the heuristic clearly outperforms the solution procedure proposed by Crevier et al. (2007). In addition, they present 54 new benchmark instances with up to 175 customers. Problems similar to VPRIRF arise in the collection of waste, for example, described by Kim et al. (2006). In this context, however, the objective is not only to minimize travel distance but also to balance the workload among the vehicles and to obtain a high route compactness. Solutions with a high route compactness are defined to have few crossovers among the routes.

Relatively few literature has been published on optimization problems related to alternative fuels. Most articles deal with the question of how to place refueling stations in an infrastructure-oriented context, either for refueling vehicles using compressed natural gas (CNG) (Boostani et al. 2010) or electricity (Qiu et al. 2011). The development of an infrastructure consisting of refueling stations, differing in terms of refueling speed and capacity, has been realized to be crucial for the promotion of alternative fuel vehicles (AFVs). The models usually use a node or flow-based set covering problem to determine the optimal number and location of the refueling stations. To model the fuel demand for short-distance trips in urban areas, customers are usually aggregated to nodes and a node-based formulation is used. Considering long-distance trips within the location decision, the flow between origin-destination pairs is used as a measure for the demand (Wang and Lin 2009, Wang and Wang 2010).

Other work concentrates on finding the energy shortest path from a given origin to a destination, which can, e.g., be used in navigation systems. Given a battery capacity, the objective is to maximize the energy level at the destination while positive arcs represent energy consumption and negative arcs recuperation (Artmeier et al. 2010). Wang and Shen (2007) propose a scheduling problem for electric buses, called Vehicle Scheduling Problem with Route and Fueling Time Constraints. They assign timetabled trips that are known in advance, to buses with the objective of minimizing total idle time. The travel range is limited by the vehicle's charge so that every vehicle has to be recharged after several trips.

Finally, we are aware of three publications that explicitly consider the specific characteristics of alternative fuels and adopt them to VRPs. Gonçalves et al. (2011) consider a VRP with Pickup and Delivery (VRPPD) with a mixed fleet that consists of BEVs and vehicles using internal-combustion engines. The objective is to minimize total costs, which consist of vehicle-related fixed and variable costs. They consider time and capacity constraints and assume a time for recharging the BEVs, which they calculate from the total distance traveled and the range using one battery charge. However, they do not incorporate the actual location of recharging stations into their model. Thus, they basically propose a mixed-fleet VRPPD with an additional distance-dependent time variable.

To the best of our knowledge, Erdogan and Miller-Hooks (2012) are the first to combine a VRP with the possibility of refueling a vehicle at a station along the route. They are mainly motivated by vehicle fleets operating on a wide geographical region and driving with biodiesel, liquid natural gas or CNG. For these fuels only a limited refueling infrastructure exists, but refueling times may be assumed to be fixed. The proposed Green VRP (G-VRP) considers a maximum route duration and fuel constraint. Fuel is consumed with a given rate per traveled distance and can be replenished at alternative fuel stations (AFS). In principle, the G-VRP is modeled as an extension to the MDVRPI, and the authors propose two heuristics to solve the new problem.

The first heuristic is a Modified Clarke and Wright Savings algorithm (MCWS) which creates routes by establishing feasibility through the insertion of AFSs, merging feasible routes according to savings and removing redundant AFSs. The second heuristics is a Density-Based Clustering Algorithm (DBCA) based on a cluster-first and route-second approach. The DBCA forms clusters of customers such that every vertex within a given radius contains at least a predefined number of neighbors. Subsequently, the MCWS algorithm is applied on the identified clusters. For the numerical studies, they design two sets of problem instances. The first consists of 40 small-sized instances with 20 customers and the second involves 12 instances with up to 500 customers.

# Chapter 3

# Fixed-Area-Based Routing under Time Window Constraints

As described above, small package shipping (SPS) companies are shifting more and more attention to customer service and efficient workforce management. Fixed-area-based routing approaches (FABRAs) are used to achieve high service consistency, among other benefits. Their drawback, however, is a decline in routing flexibility. Consequently, a high percentage of time-definite deliveries, as commonly found in the SPS sector, is expected to have a significant negative effect on the solution quality of FABRAs. To the best of our knowledge, no study exists on the magnitude of this effect and the factors that influence it, although FABRAs are commonly utilized in practice.

In Section 3.1, we introduce Semi-Fixed Service Territory Routing (SFSTR), a FABRA which is suitable for handling time window requirements. SFSTR first divides the delivery area into service territories and second carries out the daily routing based on these territories. The generation of the territories bases on routing solutions obtained on sample days of Vehicle Routing Problems with Time Windows (VRPTW) generated from historical demand data as inspired by Wong and Beasley (1984). We implement a Tabu Search (TS) method to obtain the solutions for the sample days. Additionally, we incorporate spatial aspects in order to generate "well-shaped" service territories as well as more sophisticated concepts like excluding a percentage of customers from being assigned to territories and the utilization of an *exclusion zone* around the depot (cp. Zhong et al. 2007). Moreover, we introduce so-called *semi-fixed* service territories that allow a certain number of customers to be removed and assigned to a different driver if the daily demand scenario requires it, thus increasing routing flexibility. The operational routing based on the generated territories is carried out by an extension of our TS method.

Section 3.2 presents the computational experiments carried out on several 100-day series of VRPTW problems with different customer distributions. The goal is to study the performance of SFSTR concerning route efficiency and consistency measures in comparison to a route reoptimization (RR) strategy. Additionally, we analyze the influence of the size of the service territories, i.e., the number of customers fixedly assigned to a driver, and different variabilities in the number of customers requiring service on the overall performance of SFSTR.

## 3.1 The Semi-Fixed Service Territory Routing Approach

Our SFSTR approach is divided into two phases, a *districting* and a *routing* phase as illustrated in Figure 3.1. The districting phase starts with the definition of an exclusion zone around the depot. The customers within this zone are always assigned to drivers on a daily and not on a fixed basis, i.e., they are never added to one of the territories (see Figure 3.2). Next, the set of territories is constructed on the basis of historical demand data. This method is inspired by the real-world situation, where SPS store detailed delivery data such as customer location, number, size and weight of packages and time of delivery for long periods in order to identify frequent customers and customer sequences.

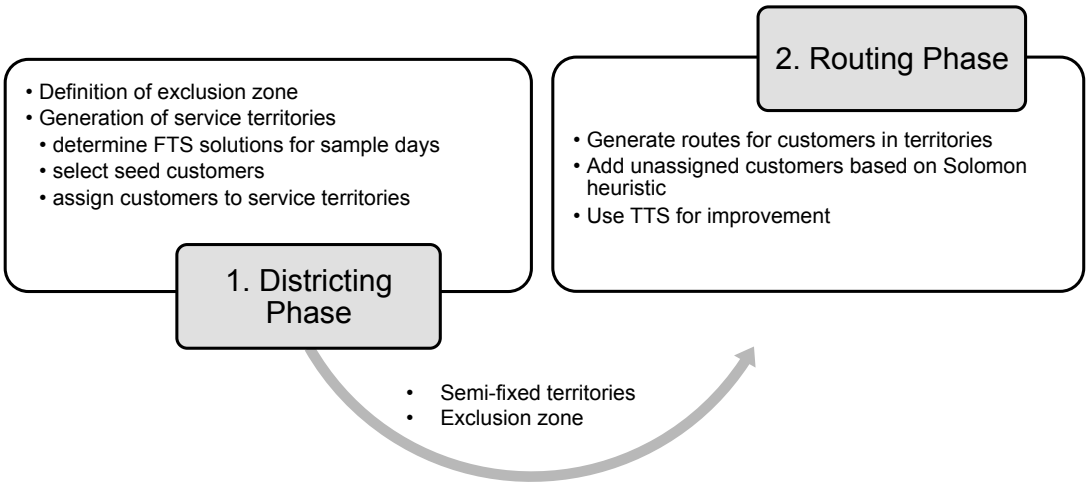



**Figure 3.1:** Overview of the districting and routing phase of SFSTR

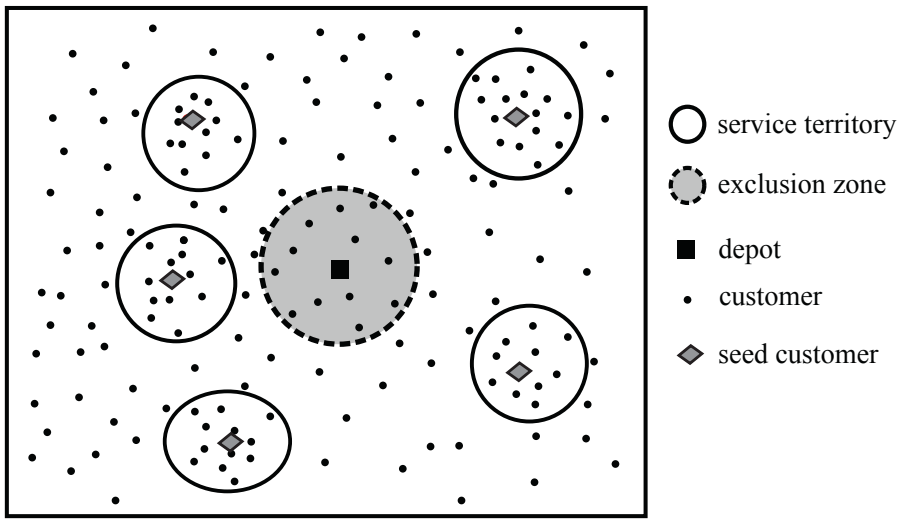



**Figure 3.2:** The delivery area: Service territories, exclusion zone and seed customers

The delivery data is used to generate series of one-day VRPTW instances which we solve by means of a TS heuristic. Note that the routing on each of these days is conducted as an independent problem without considering consistency requirements, that is why the TS used here is denoted as Free TS (FTS). Based on the solutions obtained for the one-day problems and considering additional spatial aspects, we generate the territories by first selecting a set of seed customers and then adding a predefined percentage of the overall customers as illustrated in Figure 3.2. The aim of this step is to create an assignment of customers to service territories, which is of high quality with respect to both consistency and flexibility requirements. The districting phase is detailed in Section 3.1.1.

In the routing phase, daily routing is conducted based on the service territories developed in the districting phase. The territories are normally visited exclusively by a single driver; exemptions from this rule only happen if required by the daily demand situation. More precisely, the problem to be solved in this operational phase is a VRPTW with the additional feature that part of the customers are preassigned to a specific driver. As solution method, we use an adapted version of the TS method from the districting phase denoted as Territory Tabu Search (TTS). This phase is described in Section 3.1.2.

### 3.1.1 The Districting Phase

This section details the districting phase, which consists of the following steps: 1) definition of the exclusion zone, 2) generation of routing solutions for the sample days by means of FTS, 3) selection of seed customers for the service territories, and 4) iterative assignment of customers to the service territories.

#### 3.1.1.1 Definition of Exclusion Zone

As already noted by Beasley (1984), all vehicle routes pass through the area around the depot so that the customers in there can practically be reached from every route without making long detours. Thus, such customers can be used to efficiently balance loads between routes in daily routing. This insight was also used in the creation of a flex zone in Zhong et al. (2007). We assign all customers that are located within a radius $r_{EZ}$ around the depot to an exclusion zone $EZ$ and prohibit the assignment of those customers to a service territory. Based on the historical demand data, the radius is chosen to include a given percentage $\omega$ of the total number of customers $|V|$.

#### 3.1.1.2 Generation of Solutions for a Series of Sample One-Day Problems

Prior to the design of the territories, vehicle routes have to be constructed for a number of sample days. We create a series of $\tau$ sample days, where on each sample day $t = 1, \ldots, \tau$, a subset $V_t$ of all customers require service. As mentioned above, neither consistency requirements nor any other dynamic effects are considered for generating these initial solutions, i.e., the problem to be solved here is a VRPTW.

We follow the hierarchical objective described in Section 2.1 and minimize the number of employed vehicles before minimizing the traveled distance. This is motivated by the fact that later the number of generated territories (the number of employed vehicles) is set to achieve a certain service level over the sample days. Therefore, we are interested in the minimal number of vehicles with which complete service of the customers on each day can be achieved.

FTS is inspired by the TS method proposed in Cordeau et al. (1997). As described above, TS is a potent metaheuristic concept to guide local search heuristics in order to overcome local optima. The primary goal of the TS developed in this section is to achieve VRPTW results of reasonably high quality. This shall

render SFSTR able to design sound service territories, which produce meaningful results in the computational studies. Another goal is to keep the search method as simple as possible. The focus is explicitly not on achieving record-breaking solution quality in competitive run-time.

Although FTS is able to reduce the number of vehicle routes in a solution through inter-route moves, it is not designed to minimize the number of vehicles in a solution. Instead, FTS searches for the best solution with a given vehicle number $m$. We determine the minimal vehicle number of a VRPTW instance by means of a binary search method. We start from the vehicle number given by the capacity restrictions of the problem and multiply this number by a factor $\delta_m$ until a vehicle number is reached for which FTS finds a feasible solution. This vehicle number is used as upper bound of the binary search, the last vehicle number for which FTS was unsuccessful as lower bound.

In the following, we describe the components of FTS in more detail.

**3.1.1.2.1 Generalized Cost Function** Considering only solutions during the search process that are feasible in terms of time window and capacity constraints is too restrictive and can prevent the search process from entering promising regions of the solution space. Thus, during the initialization and improvement process of FTS, both capacity violations and time window violations are permitted as commonly done in the literature (see, e.g., Talbi 2009). In order to guide the search process towards high-quality and feasible solutions, constraint violations are penalized using the following generalized cost function:

$$f_{gen}(S) = f(S) + \alpha \cdot P_c(S) + \beta \cdot P_{tw}(S),$$

where $f(S)$ denotes the traveled distance of solution $S$, and $P_c(S)$ and $P_{tw}(S)$ are the total capacity and time window violations of $S$, which are weighted by the penalty factors $\alpha$ and $\beta$.

Determining fixed values for $\alpha$ and $\beta$ that lead to high quality solutions is a difficult, if not impossible, task as these values determine whether the focus of the search lies on intensification or diversification. Since different values are advisable in different stages of the search process, we use dynamic penalty weights as introduced by Gendreau et al. (1994). Here, the penalty factors are scaled by a factor $\delta \geq 1$ depending on the recent search history. If the

solution of the last iteration was infeasible concerning the capacity, $\alpha$ is set to $\min(\alpha \cdot \delta, \alpha_{max})$. By contrast, if the solution was feasible with respect to the capacity constraint, $\alpha$ is set to $\max(\alpha/\delta, \alpha_{min})$. The technique is used to balance between intensification and diversification of the search. The procedure for updating the time window penalty factor $\beta$ is analogous to the one described for the capacity penalty factor $\alpha$.

For calculating the capacity and time window penalties, we use the methods described in Section 2.2. As explained above, the determination of changes in capacity and time window violation for inter-route moves with the conventional neighborhood operators is thus possible in constant time.

**3.1.1.2.2 Generation of Initial Solution** As initialization, we use a parallel insertion method that makes use of some of the ideas of the I1 heuristic of Solomon (1987). The original heuristic is a simple but effective sequential insertion heuristic that generates good initial VRPTW solutions in very short computing time. Contrary to the original version, our algorithm creates a potentially infeasible solution with a given number of vehicles $m$ instead of a feasible solution with an open number of vehicles.

We start from a set of $m$ seed customers, which are determined sequentially as the customer that is farthest from the depot and all previously chosen seed customers. For each seed customer, an initial route serving only the seed customer is generated. In the next step, for all unrouted customers $v$ the best route $r_{k^*}$ of the available routes $r_k = \langle v_0, v_1^k, v_2^k, \ldots, v_{n_k}^k, v_0 \rangle, k = 1, \ldots, m$ and the best insertion position $p^*$ of customer $v$ in this route is determined according to criterion $c_1$, which we define as follows:

$$(k^*, p^*)(v) = \underset{k=1,\ldots,m,\, p=1,\ldots,n_k}{\arg\min} \left(c_1(r_k, v_{p-1}^k, v, v_p^k)\right)$$

$$c_1(g, x, y, z) = d_{xy} + d_{yz} - d_{xz} + \alpha_{init}\Delta P_c(g) + \beta_{init}\Delta P_{tw}(g).$$

In our algorithm, the $c_1$ criterion determines the increase of the generalized cost function value through the insertion of customer $v$, where the change in capacity violation $\Delta P_c$ and time window violation $\Delta P_{tw}$ is weighted by factors $\alpha_{init}$ and $\beta_{init}$.

For the determined insertion positions, Solomon's criterion $c_2$ is evaluated and the customer $v^*$ maximizing the criterion is inserted at its best insertion position:

$$v^* = \underset{v \text{ unrouted}}{\arg\max} \left( c_2(r_{k^*(v)}, v^{k^*}_{p^*(v)-1}, v, v^{k^*}_{p^*(v)}) \right)$$
$$c_2(g, x, y, z) = \nu \cdot d_{0y} - c_1(g, x, y, z)$$
$$\nu \geq 0$$

The factor $\nu$ is used to prioritize customers that are located far from the depot. Such customers cannot be assigned to different routes as flexibly as customers that are close to the depot and should therefore be routed first. The described steps are repeated until all customers are routed.

**3.1.1.2.3 Neighborhood Generation and Tabu List** For the simplicity reasons mentioned above, our TS uses only one neighborhood operator String-Relocate in the improvement phase. The operator moves a sequence of vertices of a given length from one route to another in case of an inter-route move, or changes the position of the sequence inside the route in case of an intra-route move. An exemplary application of String-Relocate as inter-route operator is depicted in Figure 3.3. Our implementation of the operator limits the length of the sequence to be exchanged to 2.

String-Relocate can be seen as the node-based version of the edge-exchange operator Or-opt (Or 1976). Or-opt is a variant of the more general 3-opt move that is often used for intra-route exchange. It replaces three edges without inverting any of the three segments.

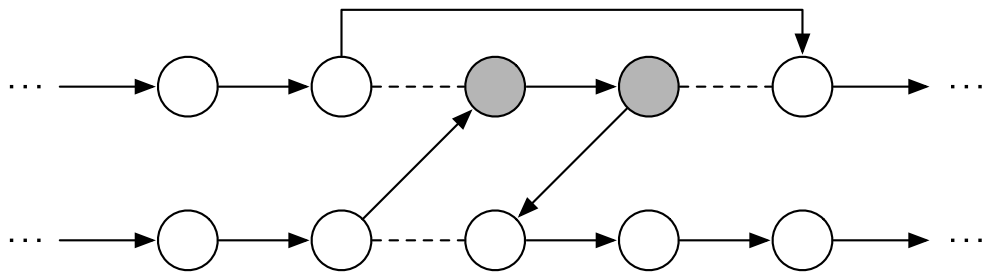


**Figure 3.3:** String-Relocate as inter-route operator with a sequence length of 2. The sequence to be relocated is indicated by the fill of the vertices, dashed lines indicate removed edges.

As described above, the tabu list is a short-term memory structure that prevents the TS from cycling by storing recently visited solutions and setting them tabu for a certain tenure. In general, saving complete solutions in the tabu list to prevent the reversal of moves to these solutions is not very efficient concerning storage and comparison effort. Rather, certain attributes of a solution are stored and moves are tabu if they induce these attributes (Gendreau and Potvin 2010a). We identify a move by attributes $(v, k)$ with customer index $v$ and route index $k$. When customer $v$ is removed from route $k$ or moved within it, attribute $(v, k)$ is set tabu for the next $\vartheta$ moves, i.e., its reinsertion into this route is forbidden (cf. Cordeau et al. 1997).

Besides the described tabu list, we introduce an additional *strong tabu list*, which prohibits any move of a customer $v$ for the next $\vartheta_{strong}$ iterations instead of only setting the attribute $(v, k)$ tabu. Its goal is to avoid wasting iterations on moving around vertices of minor importance, which, e.g., happens if flexible customers near the depot are moved around several routes before the standard tabu list becomes effective. We conducted several tests on the performance of this additional tabu list and found it important to the success of our method. As aspiration criterion, we allow the execution of a tabu move if it leads to a new overall best solution.

**3.1.1.2.4 Diversification Methods** One problem that is inherent in most heuristics based on local search is the tendency of the search to focus too strongly on small parts of the solution space. Although the tabu list is sufficient to prevent the TS from getting stuck in local optima, it is not enough to guarantee a comprehensive searching of the solution space.

To further diversify the search, we use a continuous diversification mechanism similar to the one presented in Cordeau et al. (2001). Continuous diversification constantly tries to guide the search into previously unexplored regions of the solution space by adding a diversification penalty $P_{div}(S)$ to the cost of a solution $S$. The magnitude of the penalty depends on the frequency with which attributes of the solution have been present in solutions previously visited during the search. For each solution $S$, an attribute set $B(S) = \{(v, k)$: customer $v$ is served on route $k\}$ is defined. The frequency with which an attribute $(v, k)$ has been present in solutions during the search is denoted as $\varrho_{vk}$.

Now, for every move deteriorating the current solution, i.e., if $f_{gen}(S') \geq f_{gen}(S)$, the following diversification penalty term is added:

$$P_{div}(S') = \lambda_{div} \cdot \frac{f(S')\sqrt{|V_t|\, m(S')}}{\max_{v \in V_t, k \in 1,\ldots,m(S')}(\varrho_{vk})} \cdot \sum_{(v,k) \in B(S')} \varrho_{vk}, \tag{3.1}$$

where parameter $\lambda_{div}$ is used to control the extent of the diversification and the scaling factor $\frac{f(S')\sqrt{|V_t|m(S')}}{\max_{v \in V_t, k \in 1,\ldots,m}(\varrho_{vk})}$ to adjust the magnitude of the penalty to problem size, solution cost and the frequency of the most common solution attribute. $m(S')$ denotes the number of vehicles in the generated solution. Note that solutions improving the current solution are not penalized.

Moreover, we reset the current solution to the best solution after $\eta_{reset}$ iterations without improvement to guide FTS back to a promising area of the search space. The FTS procedure stops after $\eta_{stop}$ iterations.

After determining VRPTW solutions for the generated sample days, the creation of the service territories continues with the selection of appropriate seed customers as described in the following section.

#### 3.1.1.3 Selection of Seed Customers

Now, the task at hand is to partition a set of customer locations into districts in order to define the service territories. The number of service territories $m_{st}$ is selected based on the solutions obtained for the $\tau$ sample one-day problems. More precisely, $m_{st}$ corresponds to the number of vehicles required by the FTS method to find a feasible solution for a given percentage $\theta$ of the $\tau$ sample days. This procedure is again inspired by practice as it is not realistic to have a number of vehicles/drivers that guarantee a 100% percent service level.

For solving such a districting problem, one needs a well-defined notion of adjacency. By only allowing customers to be added to a service territory if they are adjacent to it, we guarantee the creation of spatially well-defined territories, i.e., we avoid to generate disconnected territories or enclaves (territories within territories), which are very unlikely to be optimal. For our districting approach, we use the concept of adjacency introduced by Haugland et al. (2007).

Here, two customers $v$ and $w$ are called adjacent if the edge between these customers is not intersected by any shorter edge between two other arbitrary customers $u$ and $y$. The adjacency graph $\mathfrak{A}$ consists of all edges connecting adja-

cent nodes. It is a planar graph in which all edges form triangles and consequently a node that has several neighbors in two districts suggests that the node is located close to the border of the two districts (Haugland et al. 2007). In order to reduce computing times, we remove all edges whose length exceeds $d_{max}$ from the adjacency graph regardless of whether they are intersected by any shorter edge or not. Figure 3.4 shows the adjacency graphs for the Solomon problem instance C104 with clustered customers using $d_{max} = 20$.

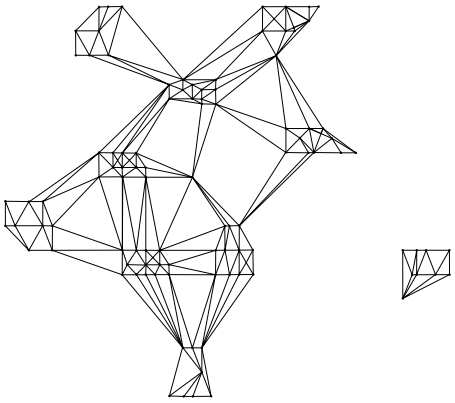


**Figure 3.4:** The adjacency graph of Solomon instance C104 using a maximal distance of $d_{max} = 20$

Using this definition of adjacency, each of the $m_{st}$ service territories is built around a seed customer as depicted in Figure 3.2. The selection of seed customers bases on the following criteria: 1) a large distance to the depot and all other seed customers, 2) a large number of neighboring customers, to which they are 3) spatially close. Customers within the exclusion zone are obviously not eligible as seed customers. Let $I$ be the set of selected seed customers, vertex 0 the depot, $|\mathfrak{A}(v)|$ the number of neighboring customers of customer $v$ in $\mathfrak{A}$ and $\overline{d}(\mathfrak{A}(v))$ the average distance to those neighboring customers. Starting from $I = \emptyset$, we iteratively select $m_{st}$ seed customers according to the following formula and add them to set $I$:

$$v_{add} = \underset{v \in V \setminus (I \cup EZ)}{\arg\max} \left( \frac{|\mathfrak{A}(v)| \cdot \min\left(d_{vw} | w \in (I \cup 0)\right)}{\overline{d}(\mathfrak{A}(v))} \right). \tag{3.2}$$

Only the locations of previously chosen seed customers are taken into account for the selection of a new seed customer. This results in a non-uniform distribution of seed customers over the delivery area as the distance between adjacent seed

customers decreases with every additional seed customer. In order to provide for a more uniform distribution of seed customers over the delivery area and thus a starting point for well-shaped service territories, we use the relocation procedure given in pseudocode in Figure 3.5.

**repeat**
  **for** each customer $v \in I$ **do**
    Determine closest vertex $w_1 \in I \cup 0 \setminus \{v\}$ with $d_{vw_1} = \min d_{vw}$
    Determine second closest vertex $w_2 \in I \cup 0 \setminus \{v, w_1\}$ with $d_{vw_2} = \min d_{vw}$
    $d(v) = d_{vw_2} - d_{vw_1}$
  **end for**
  Select customer $v_{remove} = \arg\max_{v \in I} d(v)$
  Remove $v_{remove}$ from $I$
  Select customer $v_{add}$ according to Equation (3.2) and add it to $I$
**until** $v_{remove} = v_{add}$ or maximal iteration number reached

**Figure 3.5:** Pseudocode for the relocation of seed customers

#### 3.1.1.4 Customer Assignment to Territories

After the set of seed customers $I$ has been selected, unassigned customers are iteratively added to the service territories. We generate a set of $m_{st}$ territories that approximately include a given percentage $\rho$ of all customers and are roughly of equal size, measured by the sum of expected demands of the customers assigned to the territory. Let $E(Q)$ denote the average demand of all customers in the delivery area during the $\tau$ sample days. The capacity constraint of each service territory is calculated by $C_{ST} = \rho \cdot \frac{1}{m_{st}} \cdot E(Q)$.

Starting from the territories initialized with the seed customers, we use an iterative approach to decide which of the unassigned customers to add to which territory. The resulting territories have to be suitable regarding the historical demand data and must also be geographically well-shaped. This is achieved by the procedure given in pseudocode in Figure 3.6, which bases the assignment decision on:

- The average cost of insertion with the neighbors in the service territory as introduced by Wong and Beasley (1984). Let $a_{vw}$ denote the number of times customers $v$ and $w$ are served by the same driver and $h_v$ the total number of times customer $v$ is served during the sample days. Then, the

cost $b_{vw}$ of having customers $v$ and $w$ together in the same service territory is:

$$b_{vw} = \min(h_v, h_w) - a_{vw} \qquad v, w \in V$$

Consequently, an insertion cost of $b_{vw} = 0$ means that customers $v$ and $w$ were visited by the same driver whenever both of them required service on the same day. Note that in this way temporal aspects of a good customer allocation are integrated since, e.g., customer sequences that are infeasible due to time window violations are penalized by this measure.

- The number of neighbors $|\mathfrak{A}(v,k)|$ that each customer $v$ has in service territory $k$.
- The distance $d(v, g(v))$ to the center of gravity $g(v)$ of the nearest territory that customer $v$ is not adjacent to or the distance to the depot $d(v, 0)$. The idea is to prefer customers that are as far away as possible from all other service territories and the depot. Customers close to the depot can easily be added to different routes on a flexible basis. A high distance between service territories enables a more uniform distribution of territories and enough of a flexible buffer zone between them. We use a distance exponent $\psi$ as weighting factor to determine the influence of the distance to the other service territories and the depot.

### 3.1.2 The Routing Phase

In the second phase of SFSTR, the operational routing of vehicles is carried out. The number of vehicles corresponds to the number of service territories, i.e., one vehicle is assigned to each of the territories serving all customers within this area that require service on that day. The opening of further routes is not allowed. However, we allow single vehicles to stay at the depot in case none of the customers of the respective service territory has to be served on that day and the vehicle is not needed to serve any flexible customers.

For the initialization of routes in TTS, we use the insertion heuristic presented above. In a first step, the initialization is individually applied to every territory

$\{ST_k, k = 1, \ldots, m_{st}\} \leftarrow$ set of service territories
$\overline{q}(v) \leftarrow$ average demand of customer $v$ over sample days
$Candidates \leftarrow \mathfrak{A}(I) \setminus \{EZ \cup 0 \cup I\}$
Initialize each territory $ST_k$ with one seed customer $v$ from $I$
Set territory demand $q(ST_k) \leftarrow \overline{q}(v)$
**while** $(\exists\, k : q(ST_k) < C_{ST}) \wedge (Candidates \neq \emptyset)$ **do**
  Remove from *Candidates* all customers adjacent to more than one territory
  $newCustomers = \emptyset$
  **for** each territory $ST_k$ with $q(ST_k) < C_{ST}$ **do**
    $v_{add} = \underset{v \in Candidates}{\arg\max} \left( |\mathfrak{A}(v,k)|^2 \cdot \frac{\min(d(v,g(v)), d(v,0))^{\psi}}{\max\left(\sum_{w \in \mathfrak{A}(v,k)} b_{vw},\, 0.5\right)} \right)$
    $ST_k = ST_k \cup v_{add}$
    $q(ST_k) \leftarrow q(ST_k) + \overline{q}(v_{add})$
    $newCustomers \leftarrow newCustomers \cup v_{add}$
    Update center of gravity of $ST_k$
  **end for**
  $Candidates \leftarrow Candidates \cup \mathfrak{A}(newCustomers) \setminus (0 \cup EZ \cup (\bigcup_{k \in \{1,\ldots,m_{st}\}} ST_k))$
**end while**

**Figure 3.6:** Pseudocode for the assignment of customers to service territories

and all customers assigned to a territory are iteratively inserted into the corresponding vehicle route. Subsequently, the remaining customers are added to the initialized routes one at a time at the best place of insertion. The initialized routes are then iteratively improved by the TTS method. Those customers that are assigned to fixed service territories must stay in the route of the respective territory and must not be considered for relocation. They are simply neglected by inter-route moves.

Since the demand situation in the routing phase is not known during the districting phase, on some days it might not be possible to generate a valid solution that satisfies all capacity and time window constraints and respects the preassignment of customers based on service territories. This is especially true if demand peaks occur, territories are large and time windows are tight. To mitigate the problem, we use semi-fixed service territories, i.e., we allow to expel customers from their fixed territories and assign them to different drivers. The procedure for expelling a customer is as follows:

1. If the stopping criterion of TTS is met, check if the solution is feasible with respect to capacity and time window constraints. If it is, stop, otherwise continue.

2. Find the preassigned customer whose expulsion from his service territory leads to the best improvement of the objective function and allow this customer to be removed from his route.

3. Continue TTS and reset the stopping criterion.

This cycle can be repeated several times. TTS stops either if a feasible solution is found or if an upper limit of $\varsigma$ customers are expelled from their respective service territories. Note that it is nevertheless possible to end up with an infeasible solution. This problem cannot be avoided if the territories are designed in a manner trading off feasibility and efficiency considerations.

## 3.2 Computational Studies

The main purpose of our studies is to examine the efficiency of a FABRA when dealing with time-definite customer requirements. As no adequate test instances with time window requirements are available, we generate new multi-day benchmark problems based on the Gehring and Homberger VRPTW benchmark problems. We evaluate efficiency measures, like, e.g., traveled distance, as well as consistency measures quantifying driver familiarity effects for SFSTR and compare the results to an RR strategy.

Since the flexibility of SFSTR mainly depends on the number of fixedly assigned customers, we study the influence of different territory sizes on the considered efficiency and consistency measures. Furthermore, we analyze the robustness of SFSTR concerning a higher variability in the number of customers requiring service, which is expected to have a negative influence on the performance of a FABRA.

The next section describes the generation of the test data. Afterwards, we give some insights into the parameter setting used for the studies in Section 3.2.2. Section 3.2.3 presents the numerical studies performed to analyze the general performance of SFSTR. Subsequently, we study the influence of parameter changes, namely the size of the service territories and variability in the number of occurring customers, on SFSTR in Section 3.2.4.

### 3.2.1 Generation of Test Problems

In order to study the performance of FABRAs with time-definite routing problems, we require multi-day series of VRPTW one-day problems since otherwise statements about delivery consistency are not possible. Another requirement for consistency evaluation is that the customers for each day stem from the same base set. This mimics the practical situation, where each day a (more or less) random subset of the SPS company's customer base requires service.

We generate several series of $2 \cdot \tau = 100$ VRPTW problems, where the customer set $V_t$ that requires service on day $t = 1, \ldots, 2 \cdot \tau$ is a random subset of the customers of a larger base problem with customer set $V$. Throughout the experimental analysis, we separate the generated one-day problems in two groups. The data of the first group consists of the first 50 days and is used as input to the districting phase and the remaining 50 days are used to evaluate the performance of the service territories generated in the districting phase.

To allow for a certain degree of demand variation, not only the choice of customers is randomized, but the number of selected customers $|V_t|$ for day $t$ is also a stochastic variable. We assume $|V_t|$ to be normally distributed with an expected value of $\mu$ and a standard deviation of $\sigma$. Moreover, we make the assumption that each customer has the same probability of requiring service on a given day. This is the worst case scenario for a FABRA and we did not want to have a distribution that prejudices the generated instances towards problems which are more easily tractable by FABRAs. Note however that such distributions, with several core customers that have high order volumes and frequencies, may well occur in practice.

As base problems, we use a selection of the 1000-customer Gehring and Homberger VRPTW benchmark problems. As described in Section 2.1.3, the instances are similar to the Solomon benchmark and six problem classes exist involving instances with a clustered (C), random (R) and random-clustered (RC) customer distribution. Furthermore, the instances differ in the vehicle capacity and the percentage of customers with time windows (25%, 50%, 75% and 100%). As shown in Table 3.1, we select 12 representative base problems from the different problem classes with different vehicle capacities and time window characteristics. In order to generate unfavorable instances for a FABRA approach, we use problem instances with 50% or 75% time windows. These lie around the upper bound

of the time window density characteristics of real-world SPS companies, which face up to 60% of time-definite deliveries (Campbell and Thomas 2009).

| | Time Window Density | |
|---|---|---|
| Capacity | 50% | 75% |
| low | R1_10_3, C1_10_3, RC1_10_3 | R1_10_2, C1_10_2, RC1_10_2 |
| high | R2_10_3, C2_10_3, RC2_10_3 | R2_10_2, C2_10_2, RC2_10_2 |

**Table 3.1:** Base instances for generating multi-day series of VRPTW problems

### 3.2.2 Parameter Setting

For tuning the parameters of our TS methods, we follow the strategy described in Ropke and Pisinger (2006). First, we use preliminary tests to find a decent parameter setting. Next, this parameter setting is refined by changing the value of one parameter while the rest of the parameters remain fixed. We use a representative subset of the Solomon VRPTW benchmark instances with 100 customers. On those, we conduct 10 runs of FTS with each parameter setting and then choose the setting that produces the best average results. This process is repeated with the next parameter until all parameters are tuned once. The parameters found for FTS are also used for TTS.

For our initialization heuristic, the penalty factors for capacity violation and lateness are set to $\alpha_{init} = 1, \beta_{init} = 20$ and the factor $\nu$ of the Solomon $c_2$ criterion is set to 2. During the improvement phase, the penalty factors are updated with factor $\delta = 1.5$ in the interval $[1, 1500]$. We use a standard tabu tenure of $\vartheta = 125$, while the length of the strong tabu list is $\vartheta_{strong} = 5$. The search stops after $\eta_{stop} = 150000$ iterations. The number of iterations without improvement before resetting to the best solution is $\eta_{reset} = 7500$. Furthermore, we set the value of the diversification factor to $\lambda_{div} = 2$.

For the districting phase, we use a base parameter setting which we describe next. All modifications to this base setting in later studies are described with the respective study. The number of vehicles $m_{st}$ and thus the number of service territories to create corresponds to the number of vehicles that FTS requires to find a feasible solution for $\theta = 95\%$ of the 50 sample days, i.e., the number of vehicles is set to achieve the given service level over the sample days when RR is used.

The service territories are created with $\rho = 0.5$, so as to include half of the customers in set $V$, which we denote as medium-sized territories. We found that the percentage $\omega$ of the total customers $V$ to be included in the exclusion zone has to be chosen in dependence of the service territory size in order to guarantee that enough "free" customers exist in between territories. For medium-sized territories, we use $\omega = 0.05$. The maximum distance $d_{max}$ for the adjacency graph is set to the average distance between customers in the respective base instance divided by four. A distance exponent of $\psi = 2$ is used.

SFSTR is implemented as single-thread code in Java. All tests are performed on a desktop computer equipped with an Intel Core i5 750 processor at 2.67 GHz, 4 GB RAM and running Windows 7 Professional.

### 3.2.3 Performance of Semi-Fixed Service Territory Routing

In this study, we evaluate the performance of SFSTR on the 12 generated benchmark instances that are created from the above described base problems using an expected value of $\mu = 120$ customers and a standard deviation of $\sigma = 10$. The performance is not only assessed concerning standard efficiency measures, like, e.g., traveled distance, but also in terms of achieving consistent vehicle routes. To this end, Table 3.2 presents the results for 50 days of routing with SFSTR compared to a strategy of daily RR (carried out with our FTS method), where vehicle routes are determined without considering any fixed assignment of customers to drivers. The results are grouped according to the structure of the base problem used to create the test series.

For each series, the following efficiency measures are reported for SFSTR and RR: 1) the average number of routes (#Rts.), 2) the average traveled distance (TD), 3) the number of days with an invalid solution before outsourcing customers (Inv), and 4) the average number of customers that have to be outsourced to achieve a feasible solution (OC). The last measure is inspired by practice, where unprofitable or difficult to serve customers are subcontracted (Stenger et al. 2011). We use a straightforward approach to determine the customers to outsource if a solution is infeasible. In each iteration, we find the customer that causes the highest time window and capacity violation and remove it until we end up with a feasible solution.

In order to quantify route consistency, we use the following consistency measures, which are similar to the measures introduced in Smilowitz et al. (2012):

- *Customer familiarity* for a certain customer is defined as the percentage of deliveries to that customer which are carried out by the one driver who visits the customer most frequently. By averaging over customers, we get the customer familiarity values reported in Table 3.2 as CF.
- *Driver diversity* is the number of different drivers serving a customer during the sample period. The average over all customers is provided in column DD in Table 3.2.

It should be noted that these two measures have a different behavior, i.e., a high customer familiarity value (respectively low driver diversity value) indicates route consistency. The gaps of SFSTR to the RR solution for the average number of routes, the average traveled distance and for the two consistency measures ($\Delta$#Rts., $\Delta$TD, $\Delta$CF, $\Delta$DD) are reported in Table 3.2. For all reported values, we provide averages based on the structure and over all entries.

In order to ensure a fair comparison of the consistency results obtained by SFSTR and RR, we apply a post-processing step to the RR solution as commonly done in literature (cp. Zhong et al. 2007, Groër et al. 2009, Sungur et al. 2010, Smilowitz et al. 2012). The set of routes first determined by FTS is subsequently assigned to drivers to maximize consistent deliveries. More precisely, we model the second step as an assignment problem which aims at finding the optimal driver-route assignment in terms of the considered consistency measure and solve the assignment problem by means of the optimization software CPLEX 12.1.

The average number of employed vehicles of SFSTR increases by 9.0% compared to RR and the increase in traveled distance is also rather moderate with 8.5%. Between the problem classes, a slight increase from C to R and from R to RC can be found for $\Delta$#Rts. and $\Delta$TD. Thus, the smallest efficiency forfeits can be found for cluster-based instances, probably due to the natural districting that even RR solutions to these problems show.

In addition, as a result of the limited flexibility of FABRAs, the number of days with invalid solutions more than doubles for SFSTR. However, with only 4 invalid days in 50 on average, the constraint violations seem rather moderate. Especially, if we consider the possibility of subcontracting certain critical customers. Only

| | SFSTR with Medium-Sized Service Territories | | | | | | | | | | Route Reoptimization | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | #Rts. | Δ#Rts. | TD | ΔTD | CF | ΔCF | DD | ΔDD | OC | Inv. | #Rts. | TD | CF | DD | OC | Inv. |
| C110_2 | 25.5 | 8.5% | 13520.6 | 6.2% | 65.7% | 16.1% | 2.71 | -12.7% | 3 | 9 | 23.5 | 12728.7 | 56.6% | 3.10 | 1 | 3 |
| C110_3 | 19.8 | 6.3% | 11316.4 | 8.4% | 69.3% | 16.8% | 2.50 | -16.0% | 3 | 6 | 18.6 | 10439.0 | 59.4% | 2.97 | 3 | 5 |
| C210_2 | 10.0 | 0.2% | 7906.6 | 7.5% | 82.6% | 31.2% | 1.79 | -33.7% | 0 | 0 | 10.0 | 7356.0 | 63.0% | 2.70 | 0 | 0 |
| C210_3 | 8.0 | 0.0% | 6742.1 | 4.0% | 82.7% | 28.9% | 1.78 | -30.1% | 1 | 1 | 8.0 | 6483.0 | 64.2% | 2.55 | 1 | 3 |
| **Avg. C** | **15.8** | **5.3%** | **9871.4** | **6.7%** | **75.1%** | **23.6%** | **2.19** | **-22.5%** | **1.7** | **4.0** | **15.0** | **9251.7** | **60.8%** | **2.83** | **1.3** | **2.8** |
| R110_2 | 17.9 | 9.5% | 12077.3 | 11.9% | 66.0% | 19.2% | 2.66 | -15.3% | 3 | 10 | 16.4 | 10792.2 | 55.4% | 3.14 | 2 | 3 |
| R110_3 | 15.8 | 15.8% | 10056.2 | 11.4% | 74.4% | 29.5% | 2.19 | -25.5% | 1 | 1 | 13.6 | 9024.4 | 57.5% | 2.94 | 3 | 1 |
| R210_2 | 7.0 | 0.0% | 9460.9 | 7.0% | 83.2% | 31.2% | 1.67 | -35.2% | 0 | 0 | 7.0 | 8838.2 | 63.4% | 2.58 | 0 | 0 |
| R210_3 | 6.0 | 0.0% | 7656.6 | 4.4% | 86.2% | 31.9% | 1.56 | -35.7% | 0 | 0 | 6.0 | 7333.1 | 65.3% | 2.43 | 1 | 1 |
| **Avg. R** | **11.7** | **8.7%** | **9812.7** | **9.1%** | **77.4%** | **28.2%** | **2.02** | **-27.1%** | **1.0** | **2.8** | **10.8** | **8997.0** | **60.4%** | **2.77** | **1.5** | **1.3** |
| RC110_2 | 24.4 | 12.3% | 12679.2 | 9.3% | 68.0% | 13.5% | 2.52 | -12.8% | 2 | 9 | 21.7 | 11604.2 | 59.9% | 2.88 | 2 | 5 |
| RC110_3 | 19.5 | 24.1% | 10676.7 | 15.8% | 71.8% | 20.2% | 2.33 | -18.9% | 2 | 6 | 15.7 | 9217.5 | 59.7% | 2.88 | 0 | 0 |
| RC210_2 | 9.0 | 3.0% | 8292.1 | 7.3% | 86.5% | 26.7% | 1.55 | -34.4% | 2 | 3 | 8.7 | 7727.7 | 68.3% | 2.36 | 1 | 1 |
| RC210_3 | 8.0 | 6.4% | 6982.2 | 5.3% | 87.8% | 26.6% | 1.48 | -37.5% | 1 | 3 | 7.5 | 6628.5 | 69.3% | 2.36 | 1 | 1 |
| **Avg. RC** | **15.2** | **13.4%** | **9657.5** | **9.8%** | **78.5%** | **22.1%** | **1.97** | **-24.9%** | **1.6** | **5.3** | **13.4** | **8794.5** | **64.3%** | **2.62** | **1.1** | **1.8** |
| **Tot. Avg.** | **14.3** | **9.0%** | **9780.6** | **8.5%** | **77.0%** | **24.6%** | **2.06** | **-24.8%** | **1.4** | **4.0** | **13.1** | **9014.4** | **61.8%** | **2.74** | **1.3** | **1.9** |

**Table 3.2:** Results for 50 days of SFSTR using medium-sized service territories on test instances created with a standard deviation of 10, compared to an RR strategy. The results are grouped according to the structure of the base instance used to create the test series. For each series, the average number of routes (#Rts.), the average traveled distance (TD), the number of days with an invalid solution before outsourcing customers (Inv), the average number of customers that have to be outsourced to achieve a feasible solution (OC), customer familiarity (CF) and driver diversity (DD) are reported for SFSTR and RR. Moreover, we report the percentage gap (Δ) between the two routing approaches for #Rts., TD, CF and DD. For all reported values, we give averages based on the structure R, RC, C and for all entries.

1.4 customers need to be outsourced on average to obtain feasible vehicle routes with SFSTR, compared to RR, where 1.3 outsourced deliveries are required on average.

Comparing the different problem classes for the number of invalid solutions and the number of outsourced customers, the value obtained on the R instances is significantly lower than for C and RC. A possible explanation is that, for routes serving randomly distributed customers, the exchange of a customer between routes is easier than for routes serving clustered customers. Here, vehicle routes of different drivers seldom overlap and are often spatially distant, which renders an insertion or exchange of a customer quite difficult.

On the other hand, strong improvements in the consistency measures are caused by SFSTR. Customer familiarity increases from about 60% to 80%, while the driver diversity falls from 2.7 to 2.1 different drivers per customer. Note that the familiarity values achieved by RR are already on a relatively high level and can nevertheless be significantly improved with SFSTR. The percentage improvement in customer familiarity and driver diversity is highest for the R instances. The difference to the RC instances seem to be due to the already higher consistency of the RR solutions in RC. An explanation might be that the RR solutions of more clustered instances already respect the locality induced by the clusters, which results in compact routes that are adequate to achieve high consistency. The difference to the C instances stems from a slightly lower absolute consistency of SFSTR on these instances.

Taking into account that daily customer selection is based on a uniform distribution and the instances have very high time window densities, the results indicate an only moderate deterioration in efficiency measures of the FABRA. On the other hand, the implicit consideration of driver familiarity enables the FABRA to improve consistency results significantly. Adding the real-world possibility of subcontracting, which reduces the risk of having unsatisfied customers, the advantages of a FABRA seem to outweigh the efficiency forfeits even in the presence of time window constraints and unfavorable customer occurrence.

### 3.2.4 Influence of Territory Size and Variability of the Number of Customers Requiring Service

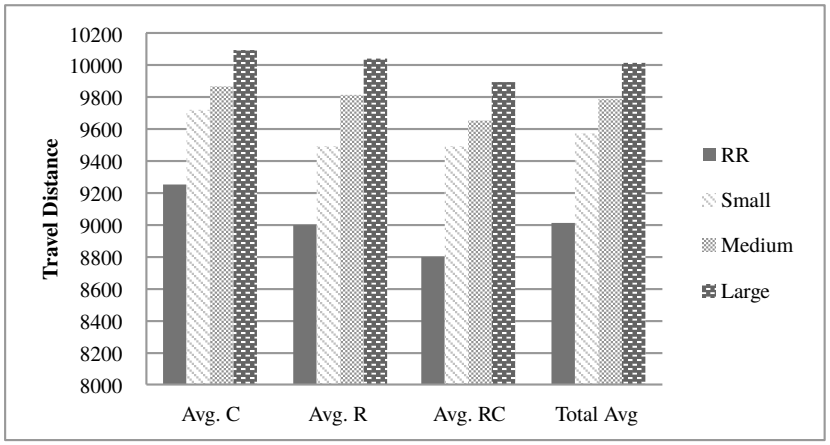
This section examines the impact of the size of the generated service territories and the variability of the number of customers requiring service on routing efficiency and route consistency. In a first study, we generate small territories including $\rho = 33\%$ of customers $V$ and large ones with $\rho = 66\%$ in addition to the medium-sized territories. We compare the results of SFSTR obtained with small, medium-sized and large service territories to an RR strategy on all instances described in the previous section. Note that the number of customers located in the exclusion zone is adapted to the territory size, i.e., the test compares the performance of territories created with the following $(\rho, \omega)$ pairs: $(0.5, 0.05)$, $(0.33, 0.1)$,$(0.66, 0.03)$.

The results are displayed in Figures 3.7 and 3.8, which depict the impact of different territory sizes on the efficiency and consistency measures described above. We show averages over the problem classes as well as overall averages.

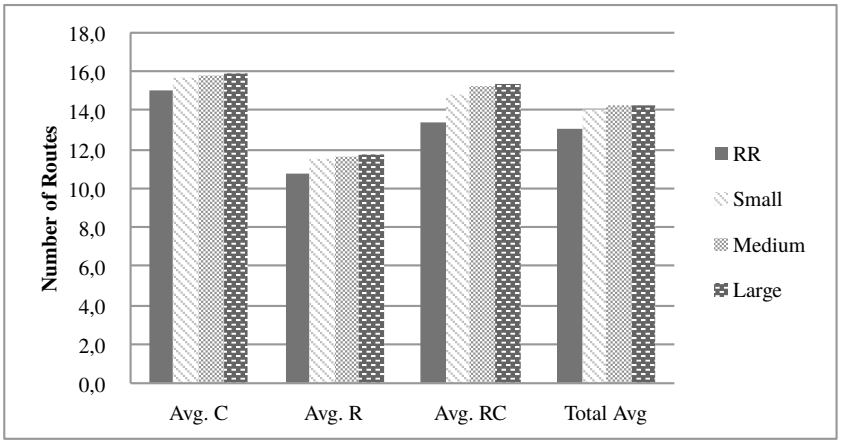
Concerning the total averages, the behavior of all measures is as could be expected. Increasing the size of the service territories leads to a deterioration of efficiency measures, i.e., an increase in traveled distance, number of routes and outsourced deliveries, and an improvement of the familiarity measures, namely an increase in customer familiarity and a decrease in driver diversity. An important point can be noted: The changes in the efficiency and consistency measures behave approximately proportionally to the increase of the customer percentage fixedly assigned to drivers. This suggests that the size of the service territories can be utilized to control the achieved tradeoff between service consistency and route efficiency.

Concerning the averages based on problem classes, only the number of outsourced customers for problem class R shows some rather irregular behavior, indicating that for this type of problems the solutions generated with smaller territories are not necessarily closer to being feasible than those with larger territories. This could probably be due to the strongly unstructured nature of these instances, which might sometimes be favorable for more clustered solutions that emerge when using (larger) territories.

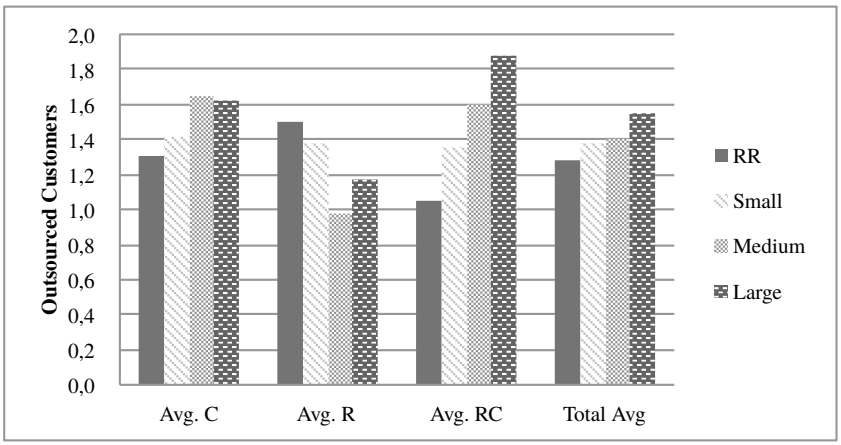
To be better able to study the tradeoff between consistency and efficiency, Figure 3.9 depicts the total averages for all four measures, normalized by using




(a) Distance

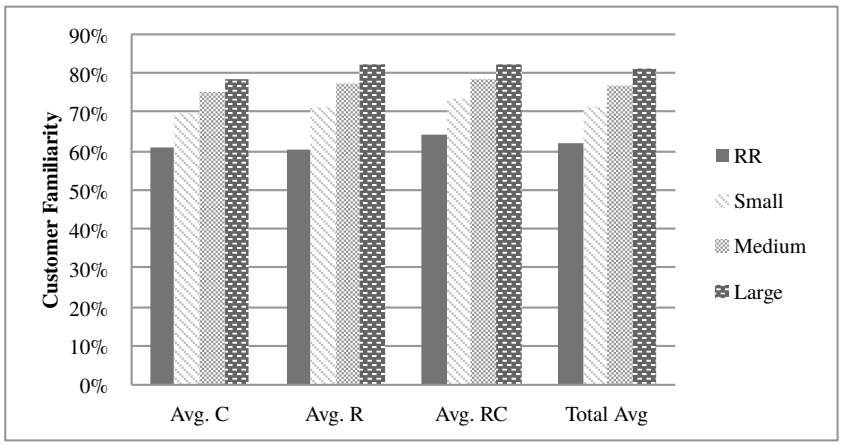



(b) Number of routes

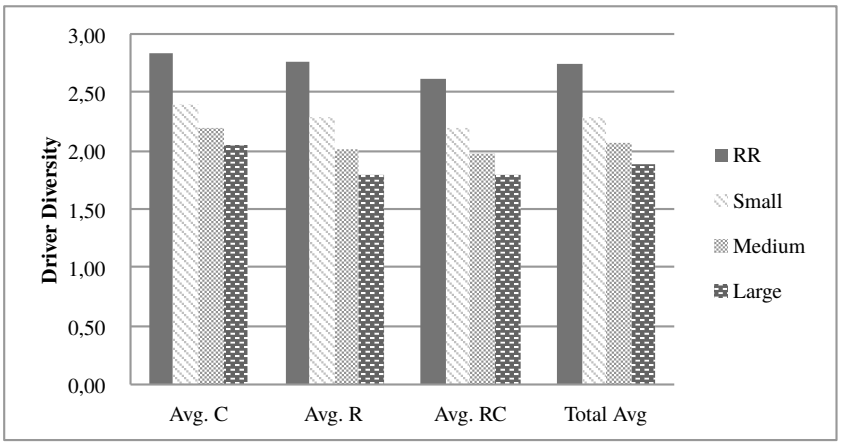



(c) Outsourced customers

**Figure 3.7:** Efficiency measures of SFSTR with different territory sizes and of the RR approach. Averages are grouped according to structure of the base problem and also given as total.




(a) Customer familiarity




(b) Driver diversity

**Figure 3.8:** Consistency measures of SFSTR with different territory sizes and of the RR approach. Averages are grouped according to structure of the base problem and also given as total.

the value achieved with medium-sized territories as 100% level. We can see that from a percentage perspective the consistency improvement achieved by SFSTR seems to outweigh the increase in traveled distance, number of routes and number of outsourced customers. Clearly, the optimal tradeoff can only be determined if a cost value can be put on each of the considered measures.

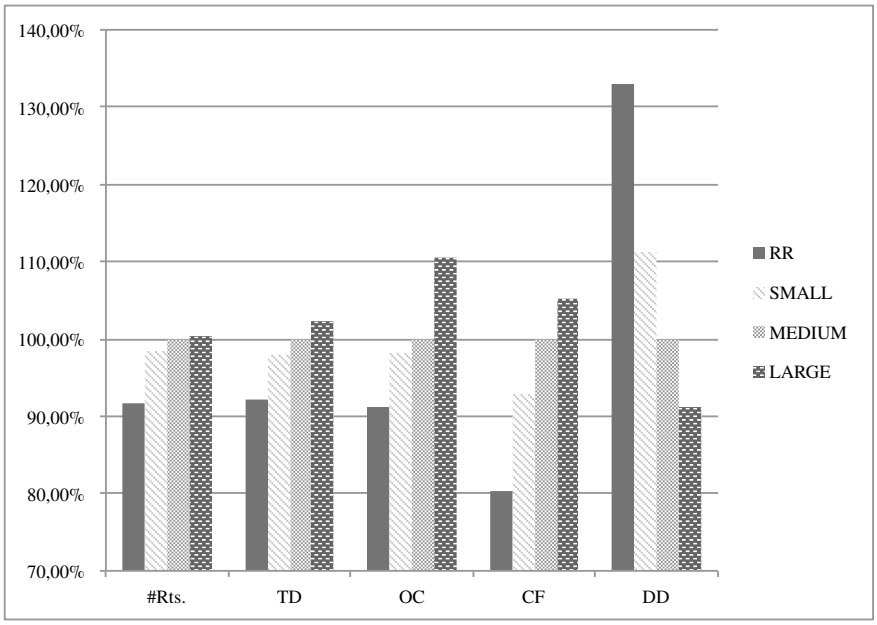



**Figure 3.9:** Comparison of SFSTR and RR concerning efficiency and consistency measures. Total averages normalized by using the value achieved with medium-sized territories as 100% level are depicted.

Last, we study the influence of the variability of the number of customers requiring service on each day on the performance of SFSTR in comparison to an RR approach. To this end, we compare results for one-day problem series generated with $\sigma = 10, 20, 30$. To get an idea of the influence of the standard deviation on the minimal, maximal and average number occurring in a 100-day series, we present these values for all 12 base instances in Table 3.3. The impact of the different standard deviations on the introduced efficiency and consistency measures is depicted graphically in Figures 3.10 and 3.11. For each standard deviation value, we show average values for small, medium-sized and large service territories as well as for the RR approach.

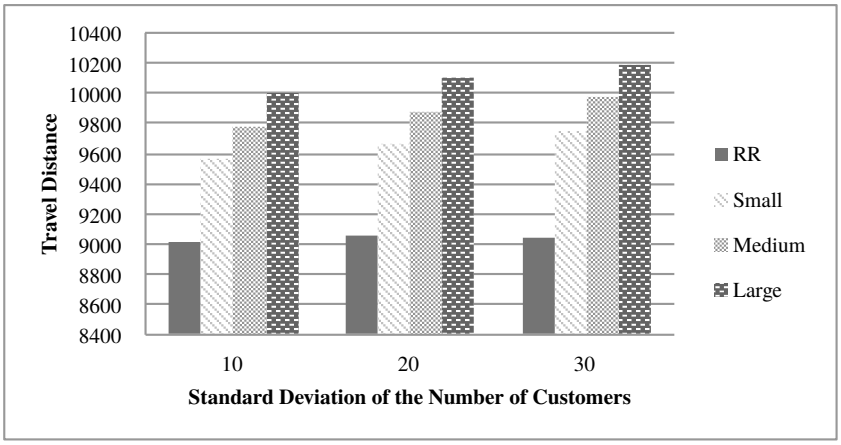
Concerning the traveled distance and number of routes, we can see that a higher standard deviation leads to an increase in the difference between RR and SFSTR

| | $\sigma = 10$ | | | $\sigma = 20$ | | | $\sigma = 30$ | | |
|---|---|---|---|---|---|---|---|---|---|
| Problem | Avg. | Max. | Min. | Avg. | Max. | Min. | Avg. | Max. | Min. |
| C110_2 | 121.3 | 148 | 102 | 119.1 | 164 | 65 | 123.4 | 195 | 58 |
| C110_3 | 121.0 | 143 | 104 | 124.3 | 156 | 97 | 118.6 | 167 | 41 |
| C210_2 | 120.3 | 155 | 93 | 120.8 | 181 | 80 | 121.8 | 188 | 50 |
| C210_3 | 122.7 | 145 | 100 | 118.8 | 159 | 75 | 133.1 | 193 | 90 |
| R110_2 | 120.5 | 139 | 103 | 122.4 | 170 | 80 | 117.3 | 193 | 49 |
| R110_3 | 120.1 | 148 | 97 | 122.7 | 167 | 69 | 125.6 | 175 | 66 |
| R210_2 | 120.2 | 139 | 97 | 116.9 | 156 | 82 | 119.8 | 196 | 63 |
| R210_3 | 120.6 | 146 | 102 | 118.8 | 147 | 84 | 115.4 | 194 | 57 |
| RC110_2 | 122.9 | 144 | 105 | 123.2 | 173 | 76 | 115.8 | 178 | 38 |
| RC110_3 | 120.7 | 146 | 101 | 122.2 | 172 | 84 | 117.9 | 168 | 45 |
| RC210_2 | 120.4 | 142 | 104 | 122.5 | 158 | 89 | 119.8 | 183 | 52 |
| RC210_3 | 121.9 | 139 | 110 | 118.1 | 152 | 62 | 125.8 | 201 | 66 |
| **Total** | | **155** | **93** | | **181** | **62** | | **201** | **38** |

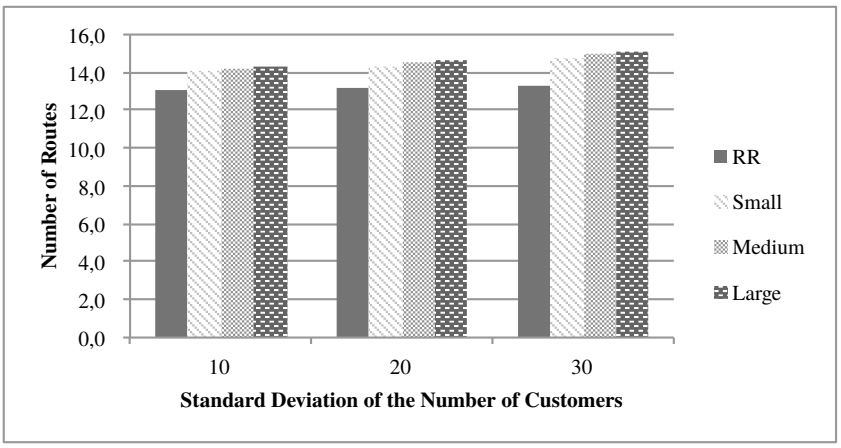
**Table 3.3:** Minimal, maximal and average number of customers requiring service in the 100-day series generated with expected value $\mu = 120$ and standard deviations $\sigma = 10, 20, 30$

for all considered territory sizes. Furthermore, a higher number of customers has to be outsourced on average, which is probably mainly due to the increased total customer demand and the higher number of customers requiring service within a predefined time window. The number of outsourced customers is the only measure that does not show a consistent behavior for the different territory sizes, probably due to specifics of the solution approach and the way that the customers to outsource are determined.

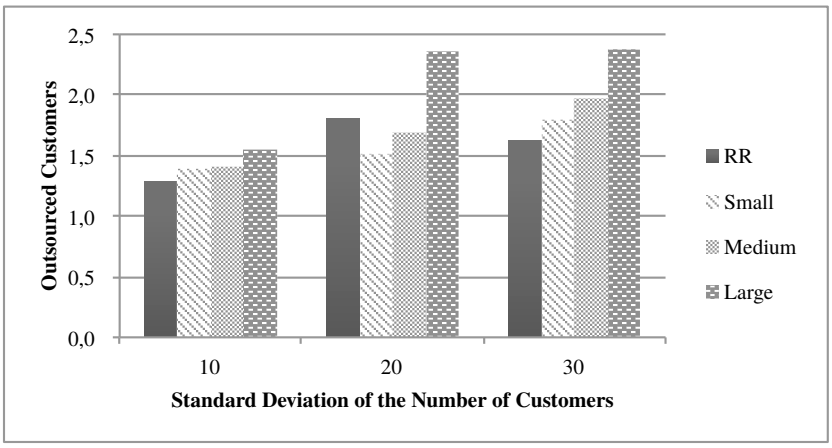
Considering the consistency measures, one can see that the achieved values for customer familiarity and driver diversity remain stable for different standard deviations. On the one hand, this means that although efficiency forfeits increase, no consistency improvement is achieved for higher standard deviations indicating that a stronger demand variation renders FABRAs less suitable. On the other hand, the results show that, even with a high demand variation and the unfavorable uniform distribution of customers, SFSTR achieves significantly higher routing consistency than RR while only having moderate efficiency forfeits due to the partial fixing.




(a) Distance

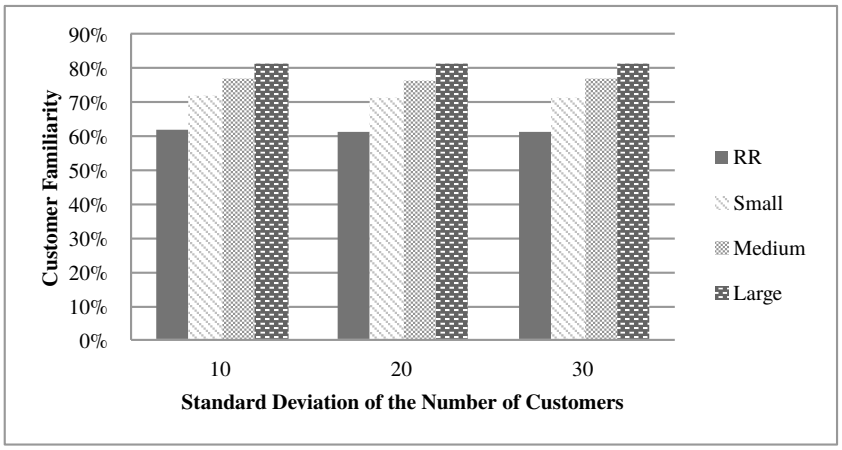



(b) Number of routes

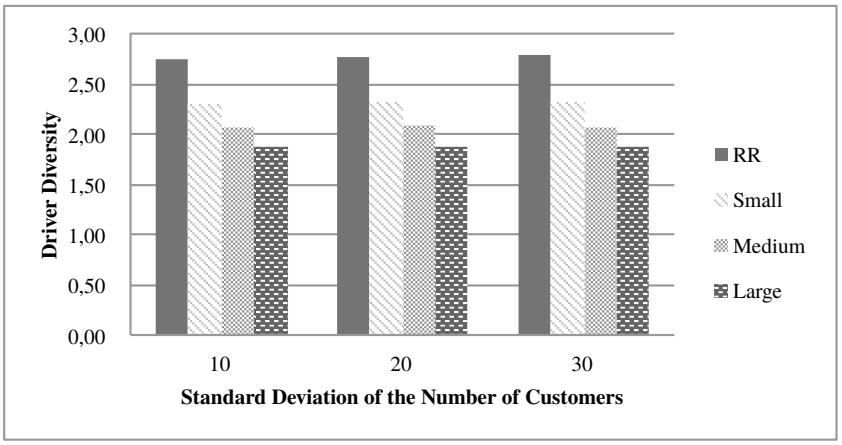



(c) Outsourced customers

**Figure 3.10:** Efficiency measures of SFSTR and an RR approach for test instances generated with expected value $\mu = 120$ and standard deviations $\sigma = 10, 20, 30$




(a) Customer familiarity




(b) Driver diversity

**Figure 3.11:** Consistency measures of SFSTR and an RR approach for test instances generated with expected value $\mu = 120$ and standard deviations $\sigma = 10, 20, 30$

## 3.3 Summary and Conclusion

We develop SFSTR to study the performance of FABRAs in the presence of time window requirements. The districting phase defines an exclusion zone around the depot, selects a set of seed customers and adds a predefined percentage of the overall customers. The customer selection is based on spatial aspects as well as on routing solutions obtained by our FTS on a set of sample days that are generated from historical demand data. In the routing phase, daily routing is conducted based on the service territories developed in the districting phase using TTS.

Our numerical studies on 12 100-day series of VRPTW problems with different customer distributions and time window densities give the following insights:

- Vehicle routes designed with FABRAs are generally characterized by a deterioration of efficiency measures, i.e., higher traveled distances, higher number of routes and a higher number of customers to outsource in order to achieve feasible daily solutions compared to an RR approach. On the other hand, FABRAs achieve considerably higher consistency, namely an increase in customer familiarity and a decrease in driver diversity.

- Increasing the size of the territories leads to further deterioration of efficiency measures while improving consistency. The changes in the efficiency and consistency measures behave approximately proportionally to the increase of the customer percentage fixedly assigned to drivers. This suggests that the size of the territories can be utilized to control the achieved tradeoff between service consistency and route efficiency.

- A higher variance of the number of customers requiring service on each day does not have the detrimental effect that one could expect (independent of the service territory size). Efficiency measures slightly deteriorate compared to RR while customer familiarity and driver diversity remain stable on a high level.

Summing up, even with high variations of the number of customers requiring service, the unfavorable uniform distribution of customers and high time window densities, the FABRA studied shows only moderate efficiency forfeits while achieving significantly higher routing consistency. This is highly valuated by SPS

companies. Adding the real-world possibility of subcontracting, the problem of invalid vehicle routes obtained on some days can easily be solved in practice.

The content of this chapter is available in similar form as technical report (Schneider, Stenger, Lagemann and Vigo 2012) and has been submitted to an international journal.

# Chapter 4

# The Vehicle Routing Problem with Time Windows and Driver-Specific Times

This chapter studies the Vehicle Routing Problem with Time Windows and Driver-Specific Times (VRPTWDST), which allows to incorporate driver knowledge into the route planning decision. Contrary to the Semi-Fixed Service Territory Routing (SFSTR) approach presented in the previous chapter, the goal is not to promote driver learning but to design efficient routes based on already existing different extents of driver knowledge. This is achieved by incorporating the regional and customer-related knowledge of a driver in the form of driver-specific travel and service times. The resulting model can provide decision support for designing high-quality vehicle routes that make use of available driver knowledge. Moreover, routes are likely to achieve a high degree of consistency as drivers have an incentive to visit familiar areas.

As described in Section 2.3, the problem has already been mentioned in the literature as a method to accommodate driver knowledge in a routing model (see, e.g., Schneider, Doppstadt, Sand, Stenger and Schwind 2010, Sand et al. 2011). However, to the best of our knowledge, no work systematically investigating the problem has been presented yet. This is done in the following sections.

Section 4.1 gives a formal definition of the VRPTWDST. To tackle the NP-hard problem, we develop a Tabu Search (TS) method, which is detailed in Section 4.2. Unlike the TS we used for generating routes for the sample days in SFSTR in Chapter 3, the goal here is to develop an efficient metaheuristic providing solutions of the highest quality possible in reasonable computing times. Consequently, the description of the TS is more detailed, putting special emphasis on design aspects. The performance of the proposed method is investigated in extensive numerical studies in Section 4.3. To this end, we perform tests on

a set of newly designed VRPTWDST benchmark instances and on benchmark instances of the closely related VRPTW.

## 4.1 Problem Definition

We define the VRPTWDST as an extension of the classical VRPTW. Contrary to the VRPTW definition given in Section 2.1, we use a three-index formulation as it provides a more natural representation of a problem with driver or vehicle specific aspects (see, e.g., Cordeau et al. 2002).

Let $V = \{1, \ldots, N\}$ denote the set of $N$ customers. 0 and $N+1$ denote instances of the depot. VRPTWDST is defined on a complete directed graph $G = (V_{0,N+1}, A)$, with $A = \{(i,j) \mid i,j \in V_{0,N+1}, i \neq j\}$ denoting the arc set. Homogeneous vehicles with a maximal capacity of $C$ are assumed to be stationed at the depot. Thus, a set $K$ of driver/vehicle pairs is available that can uniquely be identified by the driver as is done in the following. With each vertex $i \in V_{0,N+1}$ are associated a nonnegative demand $q_i$ ($q_0 = q_{N+1} = 0$), a time window $[e_i, l_i]$ (the time window of the depot $[e_0, l_0] = [e_{N+1}, l_{N+1}]$ corresponds to the scheduling horizon of the problem) and nonnegative service times $s_{ik}$ that depend on the driver $k \in K$ visiting the vertex ($s_{0k} = s_{N+1k} = 0, k \in K$). Associated with each arc $(i,j)$ is a distance $d_{ij}$ and travel times $t_{ijk}$ denoting the time it takes driver $k$ to travel the arc. The binary decision variable $x_{ijk}$ equals 1 if driver $k$ visits vertex $j$ after $i$, and 0 otherwise. Variable $\tau_{ik}$ specifies the start of service of vehicle $k$ at vertex $i$.

Like generally done for heuristic methods for the closely related VRPTW, we study hierarchical objective functions for the VRPTWDST, first minimizing the number of vehicles and only considering a secondary objective in case of ties. Contrary to the vast majority of works on the VRPTW, which only consider the minimization of traveled distance as secondary objective, we address the following three secondary objectives:

1. Minimize traveled distance ($TD$):

$$\min \sum_{k \in K} \sum_{(i,j) \in A} d_{ij} x_{ijk}$$

Note that driver-specific times only indirectly influence $TD$, as potentially

different routes with respect to the time window constraints are possible.

2. Minimize working time ($WT$). $WT$ is defined as the sum of travel and service times of a solution and thus the driver-specific times influence the objective function value in a direct fashion.

$$\min \sum_{k \in K} \sum_{(i,j) \in A} (t_{ijk} + s_{jk}) x_{ijk}$$

3. Minimize working duration ($WD$). $WD$ is defined as the difference between the start and end time of the routes. It can alternatively be described as the working time plus waiting times and is obviously also directly influenced by driver-specific times.

$$\min \sum_{k \in K} (\tau_{N+1k} - \tau_{0k})$$

In order to obtain the hierarchical objective function, the technique described in Section 2.1.1 is used. The constraints of the VRPTWDST are as follows:

$$\sum_{k \in K} \sum_{j \in V_{N+1}} x_{ijk} = 1 \qquad \forall i \in V \tag{4.1}$$

$$\sum_{j \in V_{N+1}} x_{0jk} = 1 \qquad \forall k \in K \tag{4.2}$$

$$\sum_{i \in V_0} x_{ijk} - \sum_{i \in V_{N+1}} x_{jik} = 0 \qquad \forall j \in V, k \in K \tag{4.3}$$

$$x_{ijk}(\tau_{ik} + s_{ik} + t_{ijk} - \tau_{jk}) \leq 0 \qquad \forall (i,j) \in A, k \in K \tag{4.4}$$

$$e_i \leq \tau_{ik} \leq l_i \qquad \forall i \in V_{0,N+1}, k \in K \tag{4.5}$$

$$\sum_{(i,j) \in A} q_i x_{ijk} \leq C \qquad \forall k \in K \tag{4.6}$$

$$x_{ijk} \in \{0,1\} \qquad \forall (i,j) \in A, k \in K \tag{4.7}$$

Constraints (4.1) state that each customer is visited by exactly one vehicle. Constraints (4.2) require each vehicle to start from the depot and flow conservation constraints are given in (4.3). Time window and capacity restrictions are enforced by Constraints (4.4) – (4.6). Binary variables are defined in Constraints (4.7).

## 4.2 Tabu Search for the VRPTWDST

This section introduces the TS we develop to solve the VRPTWDST, called TS-DST. An overview of the method in pseudocode is given in Figure 4.1. After eliminating infeasible arcs in a preprocessing step as described in Section 4.2.1, the initial solution is generated by using an adapted version of the I1 heuristic presented in Solomon (1987) (Section 4.2.2). The generated solution as well as the solutions during the TS phase are allowed to be infeasible. They are evaluated based on a generalized cost function penalizing infeasible solutions as described in Section 4.2.3. In each iteration of the TS, a set of neighborhood operators are applied to the current solution $S$ to generate the composite neighborhood $\mathcal{N}(S)$ (Section 4.2.4).

Each possible move is evaluated and the best non-tabu move is performed. By contrast, if the algorithm is in a probabilistic phase, a random of the best $\xi$ non-tabu moves is selected. A move is superior if it either reduces the number of employed vehicles or, in case of ties, has a better objective function value according to the generalized cost function. If a move is tabu but improves the best overall solution, an aspiration criterion is applied that lifts the tabu status of the move. After executing the move, the tabu list, the best overall solution and the penalties are updated and the algorithm decides whether a probabilistic phase is started, continued or ended as described in Section 4.2.5.

Besides the probabilistic phase, we apply further search diversification techniques like, e.g., shaking the solution. The respective methods are applied if the search has not improved the best found solution for a certain number of iterations. The techniques are detailed in Section 4.2.5. The termination criterion of the search depends on the phase of the TS run, i.e., whether it is searching for a feasible solution with a given number of vehicles or minimizing the secondary objective (see Section 4.2.6).

### 4.2.1 Preprocessing Step

In the preprocessing step, we remove those arcs from the solution graph that cannot belong to a feasible solution as often done for VRP and VRPTW (see, e.g., Psaraftis 1983, Savelsbergh 1985, 1990). An arc can either be infeasible due to time window or due to capacity constraints. To identify these arcs, we

```
initialize(tabuList,penaltyWeights)
S ← generateInitialSolution()
S* ← S
probabilisticPhase ← false
repeat
   if probabilisticPhase then
      S ← selectRandomOfBest( 𝒩(S) \ tabuList, ξ)
   else
      S ← selectBestOf( 𝒩(S) \ tabuList)
   end if
   update(tabuList, S*, penaltyWeights, probabilisticPhase)
   if specific diversification criteria satisfied then
      applyFurtherDiversification()
   end if
until termination criteria satisfied
return S*
```

**Figure 4.1:** Tabu search for solving VRPTWDST in pseudocode

use the following rules and remove all arcs $(v, w)$ for which one of the following conditions holds:

$$v, w \in V \wedge q_v + q_w \geq C \tag{4.8}$$

$$v \in V_0, w \in V_{N+1} \wedge \min_{k \in K}(e_v + s_{vk} + t_{vwk}) \geq l_w \tag{4.9}$$

$$v \in V_0, w \in V \wedge \min_{k \in K}(e_v + s_{vk} + t_{vwk} + s_{wk} + t_{wN+1k}) \geq l_0 \tag{4.10}$$

Equation (4.8) describes capacity violations that occur if the cumulated demand of two customers exceeds the vehicle capacity. Obviously, this rule is rather unlikely to remove any arcs as the demands have to be very high compared to the capacity of a vehicle. Equations (4.9) and (4.10) are adaptions of well-known preprocessing steps based on time window violations. Equation (4.9) states that an arc can be removed if even the driver with the minimal travel and service time on this arc is not able to reach $w$ within the given time window when starting service at $v$ at the earliest possible time. Equation (4.10) is similar but considers the depot deadline. If the driver with the minimal travel and service times is not able to return to the depot in time after consecutively visiting customers $v$ and $w$, the arc $(v, w)$ is infeasible. We conducted numerical studies on the Solomon benchmark instances using the driver-unspecific versions of the presented rules,

which show the partially significant effect of the preprocessing step. The results can be found in Appendix B.1.

### 4.2.2 Generation of Initial Solution

To generate an initial solution, we again use a slight modification of the insertion heuristic introduced by Solomon (1987). Like the method described in Chapter 3, the algorithm creates a potentially infeasible solution with a given number of vehicles $m$ instead of a feasible solution with an open number of vehicles as in the original version. The procedure in pseudocode is shown in Figure 4.2.

To be able to use the approach in the presence of driver-specific times, we calculate average travel and service times over the different drivers, thus transforming a VRPTWDST problem instance into a VRPTW instance. After having created the initial solution, we solve an assignment problem to find the cost-optimal assignment of available drivers to the generated routes using IBM ILOG CPLEX Optimizer (CPLEX) 12.1. As potentially fewer drivers are employed for generating the initial solution than available in the problem instance, the assignment problem is rectangular.

$S \leftarrow \emptyset$
$V_{unrouted} \leftarrow V$
**while** $m(S) < m$ **and** $V_{unrouted} \neq \emptyset$ **do**
  $v_{seed} \leftarrow \text{farthestFromDepot}(V_{unrouted})$
  $V_{unrouted} \leftarrow V_{unrouted} \setminus \{v_{seed}\}$
  $r \leftarrow \langle v_0, v_{seed}, v_{n+1} \rangle$
  **repeat**
    **for all** $v \in V_{unrouted}$ **do**
      $p^*(v) \leftarrow \text{bestPosition}(c_1, v, r)$
    **end for**
    $v^* \leftarrow \text{bestCustomer}(c_2, p^*, r)$
    $r \leftarrow \text{insertCustomer}(r, v^*, p^*)$
    $V_{unrouted} \leftarrow V_{unrouted} \setminus \{v^*\}$
  **until** no feasible insertion into $r$ possible
  $S \leftarrow \text{addRoute}(S, r)$
**end while**
**if** $V_{unrouted} \neq \emptyset$ **then**
  $S \leftarrow \text{insertCustomers}(S, V_{unrouted}, f_{gen})$
**end if**
**return** $S$

**Figure 4.2:** Generation of initial VRPTWDST solution in pseudocode

The algorithm sequentially opens routes and fills them with customers according to specific criteria until no more customers can be feasibly inserted. To open a new route, the procedure selects the unrouted customer farthest from the depot $v_{seed}$ and generates a new route $r = \langle v_0, v_{seed}, v_{n+1} \rangle$. In the next step, for each of the still unrouted customers $v \in V_{unrouted}$ the best insertion position $p^*(v)$ between two consecutive customers in $r$ is determined according to criterion $c_1$ in Equation (4.11):

$$p^*(v) = \underset{p=1,\ldots,n_k}{\arg\min}(c_1(v_{p-1}, v, v_p)) \tag{4.11}$$

$$c_1(x, y, z) = \epsilon_1 \cdot c_{11}(x, y, z) + \epsilon_2 \cdot c_{12}(x, y, z) \tag{4.12}$$

$$c_{11}(x, y, z) = d_{xy} + d_{yz} - \iota \cdot d_{xz} \tag{4.13}$$

$$c_{12}(x, y, z) = a_z - a_z^{old} \tag{4.14}$$

$$\text{with} \quad \iota \geq 0, \quad \epsilon_1 + \epsilon_2 = 1, \quad \epsilon_1 \geq 0, \quad \epsilon_2 \geq 0$$

Equation (4.12) considers the increase in traveled distance due to the insertion of the customer (calculated in Equation (4.13)) and the increase of the arrival time at the successive customer (given in Equation (4.14)). Subsequently, the customer $v^*$ that is feasibly insertable and maximizes Solomon's $c_2$ criterion is added to the route at its best insertion position:

$$v^* = \underset{v \in V_{unrouted} \wedge feasibleInsert(v)}{\arg\max} \left( c_2(v_{p^*(v)-1}, v, v_{p^*(v)}) \right)$$

$$c_2(x, y, z) = \nu \cdot d_{0y} - c_1(x, y, z)$$

$$\nu \geq 0$$

These steps are repeated as long as a customer can be inserted into the route without violating constraints. If this is no longer possible, the route is closed, added to the solution $S$ and the procedure continues with the next route until either $m$ routes are created or each customer is routed. After $m$ routes have been created, any customers that are still unrouted are added to the routes at the best insertion position according to the generalized cost function described in the following section.

### 4.2.3 Generalized Cost Function and Penalty Determination

We permit capacity and time window violations during the initialization and improvement phase using the generalized objective function $f_{gen}(S) = f(S) + \alpha \cdot P_c(S) + \beta \cdot P_{tw}(S)$, where $f(S)$ denotes the original objective function value of solution $S$, e.g., traveled distance or one of the other objective functions introduced above.

The penalty factors $\alpha, \beta$ are dynamically updated during the search to allow for more diversification if the current solution is feasible and, in the contrary case, to guide the algorithm to a feasible solution. Similar to the TS described in Section 3.1.1, the time window or capacity violation penalty factor is multiplied by $\delta$ after the respective constraint is violated for $\eta_{penalty}$ iterations. If the constraint is met for $\eta_{penalty}$ iterations, the respective penalty factor is divided by $\delta$. The penalty factors are restricted to the interval $[\alpha_{min}, \alpha_{max}]$ and $[\beta_{min}, \beta_{max}]$ respectively.

As driver-specific times have no influence on capacity restrictions, the capacity penalty $P_c(S)$ of a solution $S$ can be defined and efficiently computed as described in Section 2.2. TS-DST uses a subset of the four conventional neighborhood operators mentioned above, namely the Relocate, Exchange and the 2-opt* operator, which are detailed in the following section. Consequently, the change in $P_c(s)$ can be computed in $\mathcal{O}(1)$ time.

For determining time window penalties, we adapt the time travel approach introduced in Section 2.2 to the VRPTWDST. A straightforward adaption is to determine all time window handling related variables of a route based on the driver-specific times of the driver currently assigned to the route. In this way, the time window handling variables $\tilde{a}_v, \tilde{z}_v, \mathrm{TW}_v^{\rightarrow}$ and $\mathrm{TW}_v^{\leftarrow}$ are still defined for the vertices $v \in \mathit{Vert}(S)$ of all routes. Using this approach, penalties for the Relocate and Exchange operator can be computed in constant time by applying the original rule for merging partial paths given in Equation (2.10) and the corrected rule for inserting a vertex between partial paths presented in Equation (2.14). The reason is that for Relocate and Exchange both partial paths "belong" to the same driver and thus all time window handling related variables are available for the partial paths.

However, this is not true if the rule for merging partial paths is applied for calculating the time window penalty of a 2-opt* move. Here, the first and the second partial path to be merged belong to different routes which are carried out by different drivers. Therefore, the time window handling variables for the

second partial path are not adequate for the driver performing the first partial path and vice versa.

This situation can be remedied by defining all time window handling related variables in a driver-specific manner, i.e., for each driver $k \in K$ and all vertices $v \in \mathit{Vert}(S)$ variables $\tilde{a}_{vk}, \tilde{z}_{vk}, \mathrm{TW}_{vk}^{\rightarrow}$ and $\mathrm{TW}_{vk}^{\leftarrow}$ are kept. After a move is carried out, the variables of the affected routes have to be recalculated for all drivers. In this way, we are still able to compute time window penalties for inter-route moves with conventional operators in constant time. However, we have to accept an increase in updating and storing effort as the time window handling variables of all routes have to be kept for all drivers.

### 4.2.4 Neighborhood Generation and Tabu List

TS-DST combines several neighborhood operators to build a composite neighborhood. To define the neighborhood of a solution, we employ the 2-opt* operator for inter-route moves only and the Relocate and Exchange operators for both inter and intra-route moves. The neighborhood in each generation of the TS is generated by applying the neighborhood operators for each arc of the list of generator arcs. Using arcs as move generators was originally introduced in Toth and Vigo (2003). The so-called generator arc $(v, w) \notin S$ modifies the solution in a manner that after application of the move, vertex $v$ is followed by vertex $w$. In this way, all other arcs to be removed and added are specified for a given operator. In the following, we detail the utilized neighborhood operators putting special emphasis on how a given generator arc $(v, w)$ determines the respective move. We use $v^-$ and $v^+$ to denote the predecessor and successor of a vertex $v$.

#### 4.2.4.1 Relocate Operator

The Relocate operator was introduced in Savelsbergh (1992) and removes one vertex from a route and inserts it into another route at an arbitrary position (inter-route) or into the same route at a different position (intra-route). The generator arc $(v, w)$ defines $v$ as the vertex to be moved as shown in Figure 4.3. More precisely, the move is defined by the following operations:

- Remove arcs $(v^-, v)$, $(v, v^+)$, $(w^-, w)$
- Add arcs $(w^-, v)$, $(v, w)$, $(v^-, v^+)$

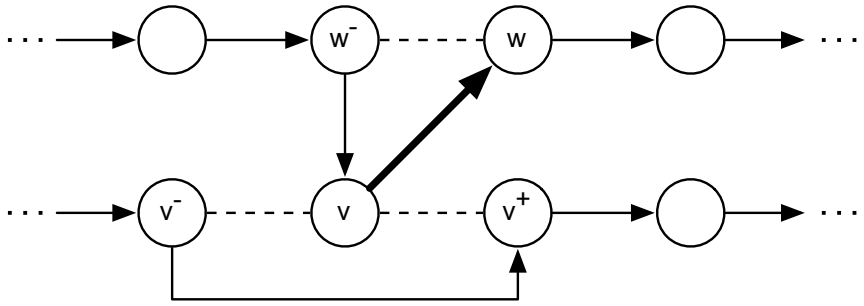



**Figure 4.3:** Relocate move with generator arc in bold and removed arcs as dashed lines

#### 4.2.4.2 Exchange Operator

The Exchange operator was also introduced in Savelsbergh (1992) and swaps the position of two vertices. For the generator arc $(v, w)$, vertex $v$ takes the position of vertex $w^-$ and vice versa as shown in Figure 4.4. The arc operations are as follows:

- Remove arcs $(v^-, v)$, $(v, v^+)$, $(w^{--}, w^-)$, $(w^-, w)$
- Add arcs $(w^{--}, v)$, $(v, w)$, $(v^-, w^-)$, $(w^-, v^+)$

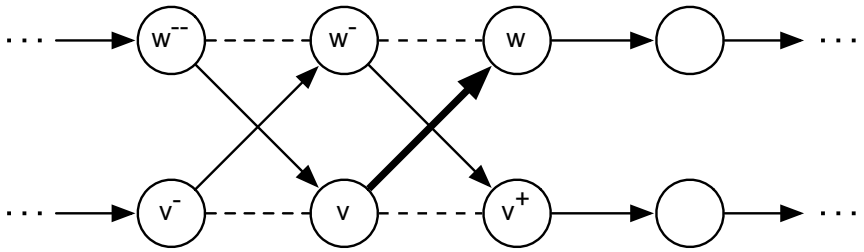



**Figure 4.4:** Exchange move with generator arc in bold and removed arcs as dashed lines

#### 4.2.4.3 2-opt* Operator

The 2-opt* operator was proposed specifically for the VRPTW by Potvin and Rousseau (1995) as a modification to the classical 2-opt operator (Lin 1965). The latter allows the reversal of the direction of partial routes, which is not desirable for routing problems with time windows as the vertices are often already sequenced according to their due time. Thus, reverting part of the route is likely

to violate time windows. The 2-opt* operator overcomes this shortcoming by removing one arc from each route and reconnecting the first part of the first route with the second part of the second route and vice versa. Thus, directions are kept in all involved part routes.

As depicted in Figure 4.5, with arc $(v, w)$ acting as generator arc, arc $(w^-, v^+)$ is unambiguously determined as the second arc to add. More precisely, the move is defined by the following operations:

- Remove arcs $(v, v^+)$, $(w^-, w)$
- Add arcs $(v, w)$, $(w^-, v^+)$

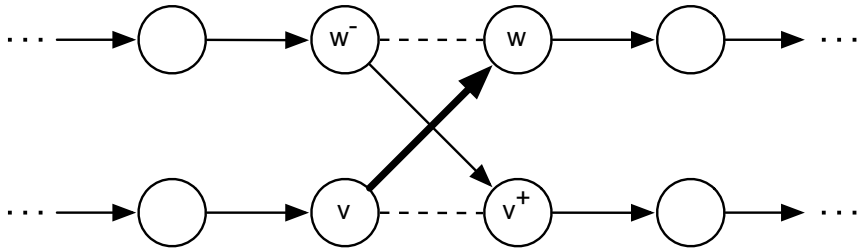



**Figure 4.5:** 2-opt* move with generator arc in bold and removed arcs as dashed lines

#### 4.2.4.4 Tabu List

As described above, saving complete solutions in the tabu list is inefficient concerning storage and comparison effort. Therefore, attributes of a solution are stored and moves are tabu if they induce these attributes. This also leads to a more extensive exploration of the search space as it helps the search to move away from previously visited parts of the search space. We follow the approach of Toth and Vigo (2003) and consider a move tabu if it reinserts an arc that was removed in one of the previous $\vartheta$ iterations, where $\vartheta$ denotes the tabu tenure. It is set to a random value in $[\vartheta_{min}, \vartheta_{max}]$ as proposed by Gendreau et al. (1994). As aspiration criterion, we allow the execution of a tabu move if it leads to a new overall best solution.

### 4.2.5 Diversification Methods

Local-search-based heuristics tend to be too local in the sense that the search focuses on small regions of the search space. Therefore, the incorporation of

diversification mechanisms, which aim at guiding the TS into previously unvisited regions of the solution space, is crucial in the design of efficient solution methods for complex combinatorial problems like the VRPTWDST (Gendreau and Potvin 2010a). We apply several such mechanisms in TS-DST, which are described in the following.

#### 4.2.5.1 Continuous Diversification

As described in Section 3.1.1, continuous diversification adds a diversification penalty $P_{div}(S')$ to the objective function value of each generated solution $S'$ that deteriorates the current solution. The diversification penalty used in TS-DST is very similar to the one given in Equation (3.1) but uses a different scaling factor and has to be defined for several different objective functions.

With the attribute set $B(S) = \{(v, k) : \text{customer } v \text{ is serviced by driver } k\}$, the occurrence frequency $\varrho_{vk}$ of attribute $(v, k)$, the diversification factor $\lambda_{div}$ and the currently used objective function $f$, the diversification penalty is computed as:

$$P_{div}(S') = \lambda_{div} \cdot f(S') \cdot \sqrt{|V| \, m(S')} \cdot \sum_{(v,k) \in B(S')} \varrho_{vk}.$$

#### 4.2.5.2 Probabilistic Phase

To further diversify the search, we use probabilistic phases, in which not the best solution is chosen in each iteration but a random one is picked from the list of the best $\xi$ solutions. Once started, a probabilistic phase runs for at most $\eta_{endProb}$ iterations or is stopped if a new best overall solution is found. The probabilistic phase is initialized as soon as one of the following two criteria is satisfied:

1. If the search has not improved the best overall solution $S^*$ for $\eta_{startProb}$ iterations.

2. If the time window violation $P_{tw}$ of the current solution has not changed for $\eta_{tw}$ iterations. This criterion helps to diversify if the search gets stuck with a certain (generally small) time window violation and is neither able to find a feasible solution nor to move away due to an ever increasing time window penalty factor that prohibits solutions with a higher time window violation.

To further increase the diversification effect of the probabilistic phase, the tabu length $\vartheta$ is randomly chosen from a wider interval $[\vartheta_{minProb}, \vartheta_{maxProb}]$ during this phase.

#### 4.2.5.3 Random Moves and Reset

After the TS has not found a solution improving the best overall solution for $\eta_{startRand}$ iterations, we apply a sequence of $\varphi$ random moves to the current solution (cf. Tan et al. 2001, Nagata and Bräysy 2009). Furthermore, if the TS has not improved $S^*$ for $\eta_{reset}$ iterations, the search is restarted from $S^*$. Due to different tabu list entries and penalties, the search is very likely to take a different path than before. Note that this technique also has an intensification effect as the search is guided to further explore a promising area of the solution space.

### 4.2.6 Minimization of Vehicle Number

Although inter-route Relocate and 2-opt* moves are clearly able to reduce the number of vehicle routes in a solution, TS-DST is not designed to minimize the number of vehicles. As the literature on VRPTW shows, very specialized methods are often necessary to achieve competitive results concerning the reduction of the number of employed vehicles, (see, e.g., Nagata and Bräysy 2009). An approach commonly applied for heuristic VRPTW solution methods is to set the number $m$ of vehicles equal to the number of routes of the best known solution reported in the literature (see, e.g., Taillard et al. 1997, Schrimpf et al. 2000, Cordeau et al. 2001, Czech and Czarnas 2002, Ibaraki et al. 2005, 2008).

From a practical viewpoint, starting the search with a given vehicle number can be justified by the fact that in most real life applications of the VRPTW, the vehicle number is given as exogenous parameter determined by the available fleet size and workforce (Lau et al. 2003, Ibaraki et al. 2005). If the vehicle number is not known, the search has to be applied for various values of $m$. Here, the assumption is made that in practical situations the range of the vehicle number can be appropriately restricted.

For TS-DST, we distinguish the following cases:

1. In case $m$ is fixed, e.g., due to practical considerations, our approach can be adjusted to determine the best possible solution concerning the secondary

objectives and employing exactly $m$ vehicles. This is achieved by prohibiting inter-route moves reducing the vehicle number.

2. If the objective is hierarchical and the number of routes of the best known solution $m_{best}$ is available, e.g., from the literature, we proceed as follows. We initially set $m = m_{best}$ and the search tries to find a feasible solution for $\eta_{feas}$ iterations. In case a feasible solution is found, the vehicle number is further reduced until no feasible solution can be found anymore. If it fails, $m$ is increased by one until a feasible solution is found. Up to this point, the search evaluates solutions based on the traveled distance independent of the secondary objective, as extensive tests showed that this metric is best suited to find a feasible solution with a restricted vehicle number. From there, the search continues optimizing the secondary objective until no improvement has been found for $\eta_{stop}$ iterations and the search stops.

3. If the objective is hierarchical and $m$ is restricted by an interval $[m_{min}, m_{max}]$, for example by some real-life requirement, we use a binary search approach using the so-called fast mode of TS-DST to minimize the vehicle number. For the vehicle number given by the search step, the fast mode tries to find a feasible solution for $\eta_{fast}$ iterations, again evaluating solutions based on the traveled distance. Depending on the success of the run, the next input value is determined by the binary search until a preliminarily minimal vehicle number is found. This number is reduced by one and used as input $m_{best}$ of the procedure described for case 2 to potentially further reduce the number of employed vehicles.

4. If the objective is hierarchical and no restrictions for $m$ are available like in the case of the newly designed problem instances described in the following section, we determine bounds to constrain the search to an interval for which the binary search described in case 3 can be applied. As lower bound, the vehicle number defined by capacity restrictions of the VRPTWDST instance is used. To determine the upper bound, this number is multiplied by a factor $\delta_m$ until the fast mode of the search finds a feasible solution.

## 4.3 Numerical Studies

This section describes the numerical experiments we conducted to assess the performance of the developed TS. First, we detail the parameter setting of TS-DST and the experimental environment in Section 4.3.1. In Section 4.3.2, we design a comprehensive set of benchmark instances for VRPTWDST as the problem has not been thoroughly investigated in the literature yet and thus no benchmarks are available. Section 4.3.3 reports the results of TS-DST on the newly generated VRPTWDST instances for the different objective functions described above. Moreover, we investigate the influence of different learning levels and different distributions of the learned customers. In order to be able to assess the performance of TS-DST, we conduct tests on the Solomon benchmark for the closely related VRPTW and compare the results to the best-performing VRPTW metaheuristics from the literature in Section 4.3.4.

### 4.3.1 Parameter Settings and Experimental Environment

To determine the parameter setting of TS-DST, we conducted a series of pretests on a selection of the VRPTWDST benchmark instances described above. We start from a reasonable parameter setting found during the development of the method. Using this setting, we tune one parameter at a time, while keeping all other parameters fixed as described in Section 3.2.2. This resulted in the setting reported in the overview in Table 4.1.

| Initialization | | General TS | | Penalties | | Prob. Phase | | Diversification | |
|---|---|---|---|---|---|---|---|---|---|
| $\epsilon_1$ | 0.5 | $\eta_{feas}$ | 25000 | $\alpha_0, \beta_0$ | 1, 200 | $\eta_{startProb}$ | 200 | $\eta_{startRand}$ | 1000 |
| $\epsilon_2$ | 0.5 | $\eta_{stop}$ | 2500 | $\alpha_{min}, \beta_{min}$ | 1 | $\eta_{endProb}$ | 50 | $\varphi$ | 100 |
| $\iota$ | 1 | $\eta_{fast}$ | 2500 | $\alpha_{max}, \beta_{max}$ | 6400 | $\eta_{tw}$ | 100 | $\eta_{reset}$ | 2500 |
| $\nu$ | 1 | $\vartheta_{min}$ | 10 | $\delta$ | 1.2 | $\xi$ | 250 | | |
| $\alpha_{init}$ | 1 | $\vartheta_{max}$ | 25 | $\eta_{penalty}$ | 10 | $\vartheta_{minProb}$ | 100 | | |
| $\beta_{init}$ | 5 | | | $\lambda_{div}$ | 2.0 | $\vartheta_{maxProb}$ | 200 | | |

**Table 4.1:** Overview of the parameter setting of TS-DST chosen for the numerical studies

The initialization phase described in Section 4.2.2 uses parameters $\epsilon_1 = \epsilon_2 = 0.5$ and $\nu = \iota = 1.0$ to evaluate the cost of insertion of a customer. For inserting the remaining infeasible customers into the open routes, the generalized cost function with penalty factors $\alpha_{init} = 1$ and $\beta_{init} = 5$ is evaluated. The TS is

run for maximally $\eta_{feas} = 25000$ iterations to find a feasible solution with a given number of vehicles and continues to minimize the secondary objective until no improvement has been found for $\eta_{stop} = 2500$ iterations. If the fast phase of the TS is used for minimizing the number of vehicles as described in Section 4.2.6, it is run for $\eta_{fast} = 2500$ iterations. The tabu tenure $\vartheta$ is drawn from the interval $[10, 25]$.

For setting the penalty factors of the generalized cost function, we achieved the best results by starting with initial values $\alpha_0 = 1$ and $\beta_0 = 200$, which are updated by multiplying our dividing by factor $\delta = 1.2$ every $\eta_{penalty} = 10$ iterations. The values are restricted to the interval $[1, 6400]$. For calculating the diversification penalty, we use a factor $\lambda_{div} = 2.0$.

After $\eta_{tw} = 100$ iterations without a change in the time window violation of the selected solution and after $\eta_{startProb} = 200$ iterations without improving the overall best solution, a probabilistic phase is executed for $\eta_{endProb} = 50$ iterations or until a new best overall solution has been found. The list of solutions considered in each iteration of the phase contains the $\xi = 250$ best solutions. During the phase, the tabu tenure is drawn from the interval $[100, 200]$.

If the best overall solution has not improved for $\eta_{startRand} = 1000$ iterations, we perform $\varphi = 100$ random moves to diversify the search. After $\eta_{reset} = 2500$ iterations without improvement, we reset to the best overall solution.

The TS is coded in Java. Studies on the Solomon VRPTW instances and a small part of the experiments on the generated VRPTWDST benchmark are performed on a standard desktop computer PC with an Intel Core i7 870 processor at 2.93 GHz, running Windows 7 Professional. In this way, we are able to assess computation times of the proposed method. Due to the high number of generated VRPTWDST instances and the variety of different objective functions considered, we conducted the remaining tests on the cluster *Elwetritsch* of the University of Kaiserslautern. The cluster consists of two hardware complexes, one with 8 servers using 4 AMD Opteron 6140 processors at 2.6 GHz each and the other with 24 servers using two Intel Xeon E5345 processors at 2.33 GHz each. More information on the cluster can be found at `https://elwe.rhrk.uni-kl.de`. For these tests, run-times are obviously not comparable due to variations in the allocation of resources.

### 4.3.2 Generation of VRPTWDST Benchmark Instances

Our VRPTWDST benchmark comprises 6 sets with 56 problem instances in each set. All problem instances are created based on the Solomon VRPTW test problems. The driver-specific travel and service times are defined by multiplying the distance and service time values of the Solomon instances with so-called learning factors. More precisely, the time $t_{ijk}$ of driver $k \in K$ for traveling arc $(i, j) \in A$ is defined as $t_{ijk} = d_{ij} \cdot l_{ijk}$, where $l_{ijk}$ is the learning factor of the driver on this arc. In the same manner, the service time $s_{ik}$ for driver $k$ at customer $i$ is computed as $s_{ik} = s_i \cdot l_{ik}$ with $l_{ik}$ denoting the learning factor of the driver for serving this customer. The value of the learning factor equals 1, if the respective arc or vertex has never before been visited by the given driver and takes a real value in $[0.5, 1.0]$ if learning has occurred.

A VRPTWDST problem instance is implemented using two files, the standard VRPTW instance file and a so-called driver file containing for each driver $k \in K$ the customer learning factors $l_{ik}, i \in V$ and the arc learning factors $l_{ijk}, i \in V_0, j \in V_{N+1}$. The drivers in $K$ are numbered from 1 to $|K|$, where the number of maximally available drivers $|K|$ is adopted from the corresponding Solomon instance. Thus, generating a VRPTWDST benchmark problem corresponds to generating a driver file that defines a VRPTWDST problem when "applied" to the original Solomon VRPTW instances.

The generation of a VRPTWDST instance starts from a driver file with all learning factors of all drivers initialized to 1. The file is then modified by means of a learning factor distribution mechanism that determines which driver/customer and driver/arc combinations are assigned a learning factor smaller than one. These are referred to as learned customers and respectively learned arcs of the driver.

More precisely, given a Solomon instance $x$, a learning factor distribution mechanism divides the customer set $V(x)$ into a number of disjoint subsets $V_k(x), k = 1, \ldots, m_{best}(x)$, where $m_{best}(x)$ is the best-known vehicle number for Solomon instance $x$ as reported in Section 2.1.3. The resulting subsets are chosen to be of approximately equal size, containing $|V|/m_{best}$ customers. If $|V|$ is not divisible by $m_{best}$, some arbitrary subsets are chosen to be larger by one. Now, subset $V_k$ is used to define the learned customers of drivers $k = 1, \ldots, m_{best}$. The learning factors of all learned customers and all learned arcs of driver $k$ are set

to a given value $\Gamma$, i.e., $l_{ik} = \Gamma, i \in V_k$ and $l_{ijk} = \Gamma, i \in V_k, j \in V_k$. The remaining drivers $k = m_{best} + 1, \ldots, |K|$ are not affected by the distribution mechanism and their learning factors remain set to 1.

Using the Solomon instances as a base, we select $\Gamma \in \{0.5, 0.7, 0.9\}$ to generate three sets of 56 instances per distribution mechanism. We apply two distribution mechanisms *Random* and *Cluster*, which are described in the following, thus generating a total of $3 \times 56 \times 2 = 324$ instances.

The *Random* mechanism selects the customer subsets in a random fashion, resulting in a geographically dispersed distribution of learned customers for each driver. Although it is difficult to imagine a practical application with such a distribution, we use this mechanism to generate benchmark instances that are hard to tackle since it is difficult to generate routes that exploit learning benefits. Figure 4.6(a) shows the distribution of learned customers as obtained with the *Random* mechanism for Solomon instance C105. The learned customers of different drivers are indicated by the different symbols.

The *Cluster* mechanism is inspired by the real-world background where learned customers are likely to appear in clusters that a driver became familiar with when serving his territory as described in Chapter 3. Therefore, *Cluster* selects the learned customers of a driver to be geographically close to one another. This is achieved in a fashion similar to the sweep heuristic of Gillett and Miller (1974). Here, a line centered at the depot is rotated to partition the customers according to their polar angle into subsets of the desired size.

The goal of the *Cluster* mechanism is to achieve a partitioning into subsets of learned customers that matches the geographical distribution of the customers as close as possible, i.e., we want to avoid situations in which the learned customers of drivers are distributed over two or more different geographical clusters. Figure 4.6(b) depicts the results of the *Cluster* mechanism for Solomon instance C105, which shows a sound matching of the "learning clusters" and the geographical clusters.

To achieve this, the *Cluster* mechanism generates $|V|$ different partitionings by selecting each of the customers in turn to define the zero degree line of the sweep approach as connection of the customer with the depot. For each obtained partitioning, we determine for each subset of learned customers the central customer defined as the customer with the minimal cumulated distance to all other

customers of the subset. The cumulated distance of the central customer to the other customers in the subset is used as an indicator of the quality of the subset's compliance with the geographical distribution. Obviously, the measure increases if the learned customers are distributed over different geographical clusters. Therefore, the partitioning with the minimal cumulated distance over all subsets is finally selected.

In Appendix B.2, we provide additional visualization examples for the two distribution mechanisms on Solomon instances representative of the different groups R1, R2, C1, C2, RC1 and RC2.

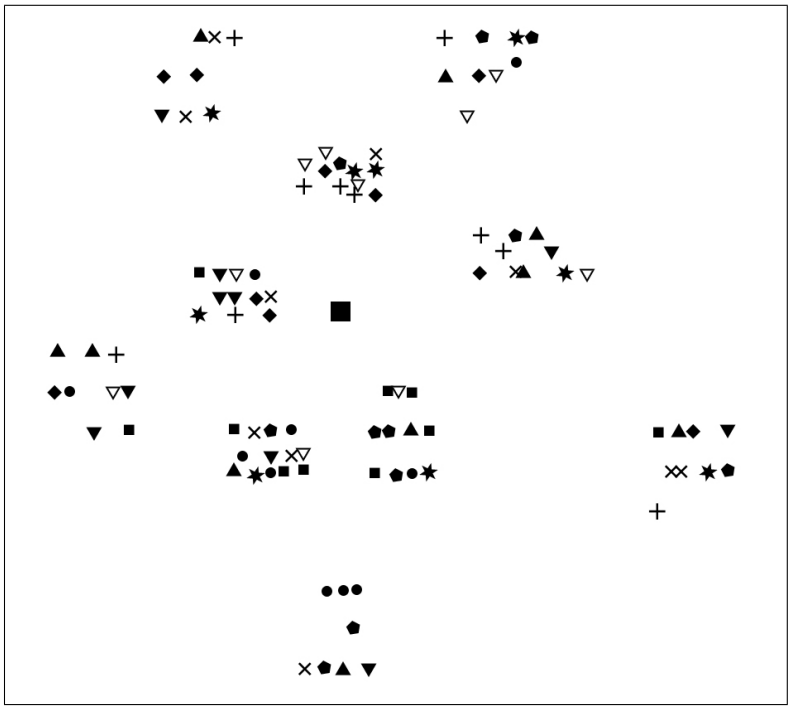
### 4.3.3 Performance on VRPTWDST Instances

To study the performance of TS-DST on benchmarks with driver-specific times, we conduct 10 runs on each of the generated VRPTWDST instances for each of the hierarchical objective functions with $\{TD, WT, WD\}$ as secondary objectives. The tests with secondary objective $TD$ were conducted on our desktop PC so that run-times can be reported. Due to the large number of problem instances and test runs, the studies with the secondary objectives $WT$ and $WD$ were carried out on a computing cluster. No systematic change in the run-time is expected due to a different objective function as confirmed by several pretests with smaller problem sets. The complete results on an instance basis for all objective functions are reported in the appendix in Section B.4. They are provided as comparison values for potential methods addressing the VRPTWDST in the future.

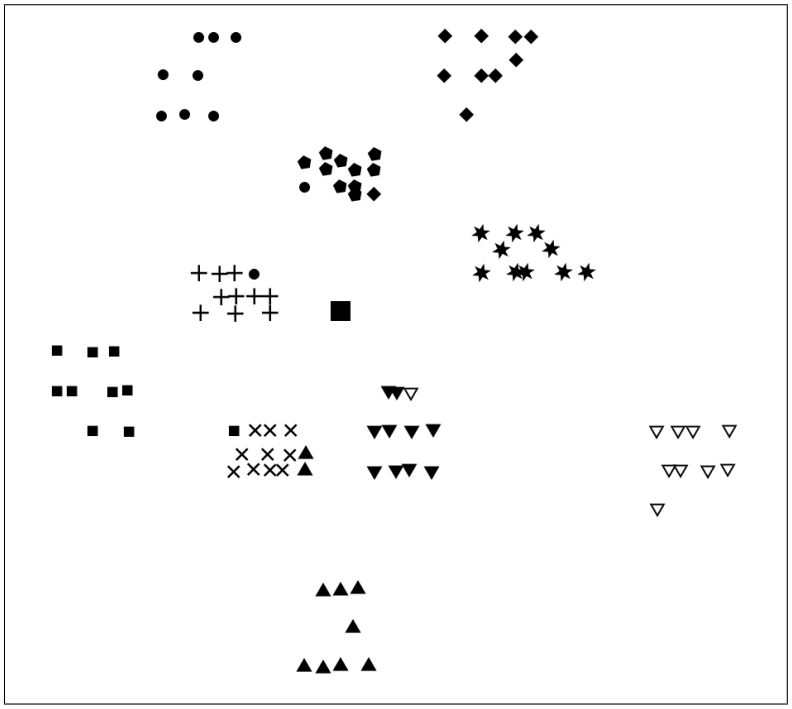
An aggregate view on the obtained results is given in Table 4.2. The considered problem sets are identified by the distribution mechanism (*Random,Cluster*) and the learning factor ($\Gamma = 0.5, 0.7, 0.9$). We always use a hierarchical objective function first minimizing the number of vehicles as described in Section 4.2.6. For each problem set, we report the cumulated number of vehicles (CNV) and the cumulated value of the secondary objective (CTD, CWT, CWD) based on the best solution found in 10 runs. For the secondary objective $TD$, we also report the average run-time.

To be better able to assess the obtained solution values, we report the following two comparison values for each problem set:

1. For each instance in the problem set, we take the solution that TS-DST obtained for the associated VRPTW Solomon instance and evaluate this



(a) *Random* distribution for instance C105



(b) *Cluster* distribution for instance C105

**Figure 4.6:** Distribution of learned customers using the (a) *Random* and (b) *Cluster* mechanism

solution with the given driver-specific times of the considered instance and the given secondary objective. To make the comparison value as fair as possible, we solve an assignment problem to assign the routes of the VRPTW solution to the drivers of the VRPTWDST instance in the manner optimizing the considered secondary objective. The objective function values achieved by this method are cumulated and the gap to this value is reported as $\Delta\text{VRPTW}_{assign}$ in Table 4.2.

2. For each instance in the problem set, the associated VRPTW Solomon instance is modified by setting all travel times $t_{ij}, i \in V_0, j \in V_{n+1}$ to $\overline{\Gamma} \cdot d_{ij}$ and all service times $s_i, i \in V$ to $\overline{\Gamma} \cdot s_i$, where $\overline{\Gamma}$ denotes the learning factor of the considered problem set. The resulting VRPTW is solved by the TS-DST and the gap of the cumulated objective function values is reported as $\Delta\text{VRPTW}_{\overline{\Gamma}}$ in Table 4.2.

| | **Cluster** | | | **Random** | | |
|---|---|---|---|---|---|---|
| | $\Gamma = 0.9$ | $\Gamma = 0.7$ | $\Gamma = 0.5$ | $\Gamma = 0.9$ | $\Gamma = 0.7$ | $\Gamma = 0.5$ |
| **Minimize Vehicle Number** | | | | | | |
| CNV | 396 | 367 | 350 | 402 | 391 | 377 |
| $\Delta\text{VRPTW}_{assign}$(%) | -2.22 | -9.38 | -13.58 | -0.74 | -3.46 | -6.91 |
| $\Delta\text{VRPTW}_{\overline{\Gamma}}$(%) | 4.49 | 7.00 | 8.36 | 6.07 | 13.99 | 16.72 |
| **Minimize Traveled Distance** | | | | | | |
| CTD | 57291 | 56955 | 55494 | 57191 | 57782 | 58375 |
| $\Delta\text{VRPTW}_{assign}$(%) | -0.41 | -0.99 | -3.53 | -0.58 | 0.45 | 1.48 |
| $\Delta\text{VRPTW}_{\overline{\Gamma}}$(%) | -0.35 | 4.08 | 5.02 | -0.53 | 5.59 | 10.47 |
| CPU Time (s) | 89.2 | 84.5 | 76.8 | 70.5 | 88.3 | 86.1 |
| **Minimize Working Time** | | | | | | |
| CWT | 228387 | 186133 | 143008 | 242768 | 228775 | 209913 |
| $\Delta\text{VRPTW}_{assign}$(%) | -0.24 | -0.45 | -1.30 | -0.18 | -0.68 | -3.44 |
| $\Delta\text{VRPTW}_{\overline{\Gamma}}$(%) | 1.71 | 7.77 | 16.82 | 8.12 | 32.46 | 71.47 |
| **Minimize Working Duration** | | | | | | |
| CWD | 240644 | 217936 | 201105 | 250864 | 240140 | 228978 |
| $\Delta\text{VRPTW}_{assign}$(%) | -4.14 | -9.14 | -13.07 | -2.83 | -4.43 | -6.93 |
| $\Delta\text{VRPTW}_{\overline{\Gamma}}$(%) | 2.53 | 12.52 | 26.82 | 6.89 | 23.98 | 44.40 |

**Table 4.2:** Aggregate results of TS-DST on the VRPTWDST benchmark sets generated with distribution mechanisms *Random* and *Cluster* and learn factors $\Gamma = 0.5, 0.7, 0.9$. For each set, we report the CNV and the cumulated value of the secondary objective function based on the best solution found in 10 runs. Average run-times of TS-DST are reported when traveled distance is used as secondary objective.

Concerning the number of employed vehicles, it can be noted that a reduction of the learning factor leads to a decrease of the number of vehicles. As could be expected, the reduced travel and service times allow to generate routes with fewer vehicles that are nevertheless able to fulfill all time window requirements, and TS-DST is able to find these solutions and exploit the reduced times.

The vehicle numbers also show consistent behavior concerning the influence of the distribution mechanism: For each learning factor, the number of vehicles in the *Random* generated instance is higher than in the *Cluster* instance. This could be expected, as it is possible to gather more learning benefits if the reduced times of a driver occur in a clustered region, but it again shows the ability of our algorithm to generate solutions making use of the reduced times. This is also expressed by the gap to the $\text{VRPTW}_{\overline{\Gamma}}$ solutions, which are higher for the *Random* generated test sets as driver-specific times cannot be exploited that strongly.

Consequently, the best solution is obtained with a *Cluster* distribution and a learning factor $\Gamma = 0.5$, reducing the CNV by 13.58% compared to the solutions of the associated VRPTW instances. The obtained solutions use only 8.36% more vehicles than the $\text{VRPTW}_{\overline{\Gamma}}$ solutions with learning factor 0.5.

To illustrate the combined influence of learning factors and distribution mechanisms on the number of employed vehicles, Figure 4.7 depicts the gap of the CNV values obtained for the different sets to the $\text{VRPTW}_{assign}$ solution. The second comparison value $\text{VRPTW}_{\overline{\Gamma}}$ and the CNV given by vehicle capacity restrictions are also provided. The described trends concerning the two distribution mechanisms and changes in the learning factor are clearly visible.

Concerning the secondary objective of minimizing traveled distance, no such clear trend can be found. While the negative gap to $\text{VRPTW}_{assign}$ grows for lower values of the learning factor for the *Cluster* distribution, it even becomes positive for the *Random* distribution. On the other hand, $\Delta\text{VRPTW}_{\overline{\Gamma}}$ is negative for *Cluster*-$\Gamma = 0.9$ and *Random*-$\Gamma = 0.9$. These results can be explained by the interdependency of the first and secondary objectives: It is not necessarily possible to obtain a solution with a reduced secondary objective when employing fewer vehicles.

Minimizing working time and working duration, the negative gaps to $\text{VRPTW}_{assign}$ increase for lower values of the learning factor for both distribution mechanisms although the considered solutions employ fewer vehicles than

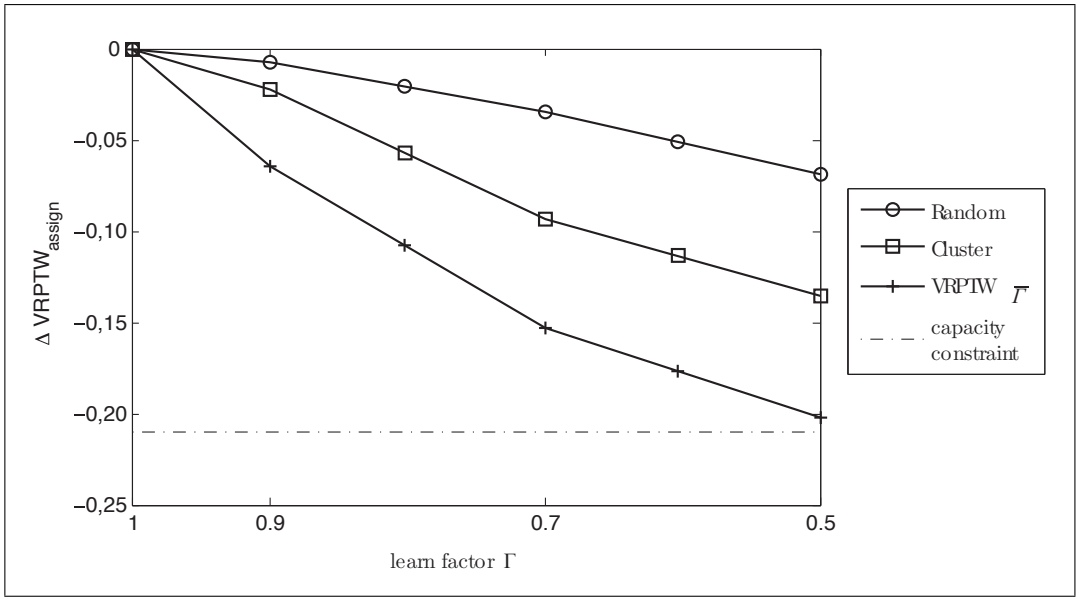



**Figure 4.7:** Influence of different learning factors and distribution mechanisms on the number of employed vehicles. As reference point, we use the CNV of the $\Delta\text{VRPTW}_{assign}$ solution to the respective set. We show the results obtained for the *Random* and *Cluster* distribution and also provide $\text{VRPTW}_{\overline{\Gamma}}$ as further reference point. Moreover, the CNV given by the capacity restrictions is indicated as dashed line.

$\text{VRPTW}_{assign}$. This might be explained by the fact that the reduced times directly influence the objective function value. For both objective functions a clear trend can also be found concerning the values of $\Delta\text{VRPTW}_{\overline{\Gamma}}$. For all considered values of the learning factor, the value of $\Delta\text{VRPTW}_{\overline{\Gamma}}$ for *Cluster* is clearly below that of the *Random* set. By being better able to exploit reduced times in the *Cluster* sets, the solution also gets closer to the $\text{VRPTW}_{\overline{\Gamma}}$ solution concerning working time or working duration.

To illustrate the influence of the characteristics of the Solomon instances on the obtained VRPTWDST solutions, Table 4.3 shows the gaps to $\text{VRPTW}_{assign}$ solutions for the different Solomon groups. For groups C1 and C2, no reduction of the vehicle number is possible as all instances in these two groups are generated in a way that the vehicle number is only limited by the capacity constraint. Time windows in C1 and C2 are positioned around the arrival times at customers obtained by a solution to the problem without time windows. Therefore, reduced times do not allow to generate more efficient routes in terms of traveled distance as time windows are not restrictive for these groups.

| | | Cluster | | | Random | | |
|---|---|---|---|---|---|---|---|
| | | $\Gamma = 0.9$ | $\Gamma = 0.7$ | $\Gamma = 0.5$ | $\Gamma = 0.9$ | $\Gamma = 0.7$ | $\Gamma = 0.5$ |
| **C1** | ΔCNV(%) | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| | ΔCTD(%) | 0.00 | -0.05 | -0.07 | 0.00 | -0.03 | -0.03 |
| **C2** | ΔCNV(%) | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| | ΔCTD(%) | -0.26 | -0.26 | -0.39 | -0.13 | -0.26 | -0.38 |
| **R1** | ΔCNV(%) | -2.10 | -11.19 | -18.18 | -0.70 | -2.80 | -7.69 |
| | ΔCTD(%) | -1.95 | -5.52 | -8.46 | -1.27 | -1.68 | 0.10 |
| **R2** | ΔCNV(%) | -3.33 | -20.00 | -20.00 | 0.00 | -13.33 | -16.67 |
| | ΔCTD(%) | -0.23 | 4.37 | -0.48 | -1.00 | 4.78 | 3.63 |
| **RC1** | ΔCNV(%) | -2.17 | -11.96 | -17.39 | -2.17 | -3.26 | -7.61 |
| | ΔCTD(%) | -3.44 | -6.03 | -10.02 | 0.83 | -2.60 | -2.14 |
| **RC2** | ΔCNV(%) | -11.54 | -19.23 | -26.92 | 0.00 | -11.54 | -19.23 |
| | ΔCTD(%) | 5.21 | 5.10 | 4.38 | -1.42 | 3.32 | 7.85 |
| **Total** | ΔCNV(%) | -2.22 | -9.38 | -13.58 | -0.74 | -3.46 | -6.91 |
| | ΔCTD(%) | -0.41 | -0.99 | -3.53 | -0.58 | 0.45 | 1.48 |

**Table 4.3:** Results of TS-DST on VRPTWDST instances for different instance groups, using traveled distance as secondary objective. For each considered Solomon group, the gap of the TS-DST solution to the VRPTW$_{assign}$ solution concerning CNV and cumulated traveled distance (CTD) is given.

A reduction of the vehicle number is only possible for R and RC instances. Reductions are on average higher for groups R2 and RC2, i.e., the problems with a longer scheduling horizon. One explanation might be that solutions to these instances employ a very low vehicle number so that reductions lead to high percentage improvements. On the other hand, this characteristic seems to lead to an increase in traveled distance for most cases. Note that all Solomon groups individually show the above described trend for the influence of learning factors and distribution mechanism on the CNV.

### 4.3.4 Performance on Standard VRPTW Instances

To be able to assess the solution quality and run-time of TS-DST, we run tests on the Solomon VRPTW instances. In this way, we are able to compare our results to those of the VRPTW methods presented in the literature. TS-DST is run 10 times for each problem instance, starting from the best-known vehicle number reported in the literature for the respective instance as described above.

In Table 4.4, we present the results of TS-DST, in comparison to the best-performing metaheuristics from the literature. For each method, the solution quality in terms of vehicle number and traveled distance is reported. We follow the common procedure and give averages over the problem classes C, R and RC and the CNV and the CTD. For TS-DST, we report the best solution found in the 10 runs. Detailed results and run-times on an instance basis are reported in Appendix B.3.

As comparison methods, we include all solution methods from the literature that are able to find solutions with a CNV of 405. This is only achieved by very few of the numerous methods proposed, namely **HY** (Hashimoto et al. 2008), **LCK** (Bouthillier and Crainic 2005), **BVH** (Bent and Van Hentenryck 2004), **PGDR** (Prescott-Gagnon et al. 2009), **HYI** (Hashimoto et al. 2008), **LZ** (Lim and Zhang 2007), **PR** (Pisinger and Ropke 2007), **RTI** (Repoussis et al. 2009), **NBD** (Nagata et al. 2010) and **B** (Bräysy 2003).

To allow for a fair comparison of the run-times of the different methods, we follow the approach of Gendreau and Tarantilis (2010) and approximately translate the different hardware used in the tests into a comparable time measure. They derive a relative estimated speed compared to a Pentium 4 2,8 GHz processor for each of the used machines based on the performance indicators of Dongarra (2011). We use their reported values and estimate our Intel Core i7 processor at 2.93 GHz with a speed factor of 1.74 in comparison to their Pentium 4 after running the LINPACK benchmark used in Dongarra (2011) on our machine. Clearly, this technique for comparing the run-times of the different methods is not exact, but it provides a good indication about the speed of the heuristics. To translate into the common time measure, the relative speed factor, the number of employed processors, the number of executed runs and the run-time in minutes are multiplied. All necessary values are reported in Table 4.4.

TS-DST achieves a CNV of 405 and a CTD of 57525. Concerning the solution quality, TS-DST ranks $9^{\text{th}}$ among the 11 best performing VRPTW heuristics with only minimal differences to the CTD found by the other methods. The gap to the CTD of the best-known solutions for all instances as reported in Section 2.1.3 is 0.6%. This is a very persuasive result as TS-DST was designed as solution method for VRPTWDST and the parameter tuning was carried out on VRPTWDST instances. TS-DST is the only tabu search able to solve the Solomon VRPTW

| | HY | LCK | BVH | PGDR | HYI | LZ | PR | RTI | NBD | B | TS-DST |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **R1** | 11.92 | 11.92 | 11.92 | 11.92 | 11.92 | 11.92 | 11.92 | 11.92 | 11.92 | 11.92 | 11.92 |
| | 1216.7 | 1214.2 | 1213.25 | 1210.34 | 1213.18 | 1213.61 | 1212.39 | 1210.82 | 1210.34 | 1222.12 | 1218.24 |
| **R2** | 2.73 | 2.73 | 2.73 | 2.73 | 2.73 | 2.73 | 2.73 | 2.73 | 2.73 | 2.73 | 2.73 |
| | 961.28 | 954.32 | 966.373 | 955.74 | 955.61 | 961.05 | 957.72 | 952.67 | 951.03 | 975.12 | 962.46 |
| **C1** | **10** | **10** | **10** | **10** | **10** | **10** | **10** | **10** | **10** | **10** | **10** |
| | **828.38** | **828.38** | **828.38** | **828.38** | **828.38** | **828.38** | **828.38** | **828.38** | **828.38** | **828.38** | **828.38** |
| **C2** | **3** | **3** | **3** | **3** | **3** | **3** | **3** | **3** | **3** | **3** | **3** |
| | **589.86** | **589.86** | **589.86** | **589.86** | **589.86** | **589.86** | **589.86** | **589.86** | **589.86** | **589.86** | **589.86** |
| **RC1** | 11.5 | 11.5 | 11.5 | **11.5** | 11.5 | 11.5 | 11.5 | 11.5 | **11.5** | 11.5 | 11.5 |
| | 1384.17 | 1385.3 | 1384.22 | **1384.16** | 1384.25 | 1385.56 | 1385.78 | 1384.3 | **1384.16** | 1389.58 | 1390.14 |
| **RC2** | 3.25 | 3.25 | 3.25 | 3.25 | 3.25 | 3.25 | 3.25 | 3.25 | 3.25 | 3.25 | 3.25 |
| | 1138.37 | 1129.43 | 1141.24 | 1119.44 | 1120.5 | 1121.82 | 1123.49 | 1119.72 | 1119.24 | 1128.38 | 1127.98 |
| **CNV** | 405 | 405 | 405 | 405 | 405 | 405 | 405 | 405 | 405 | 405 | 405 |
| **CTD** | 57484 | 57360 | 57567 | 57240 | 57282 | 57368 | 57332 | 57216 | 57187 | 57710 | 57525 |
| **Computer** | X2.8G | P850M | SU10 | O2.3G | P2.8G | P2.8G | P3G | P3G | O2.4G | P200M | i7/870 |
| **Processors** | 1 | 5 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **CPU** | 17 | 12 | 120 | 30 | 16.7 | 93.2 | 2.4 | 17.9 | 5 | 82.5 | 1.1 |
| **Runs** | 1 | 1 | 5 | 5 | 3 | 1 | 10 | 3 | 5 | 1 | 10 |
| **Speed** | 1.26 | 0.14 | 0.16 | 1.41 | 1 | 1 | 1.07 | 1.07 | 1.45 | 0.03 | 1.74 |
| **Time** | 21.48 | 8.43 | 94.76 | 211.85 | 50.1 | 93.2 | 25.77 | 57.66 | 36.26 | 2.19 | 19.14 |

**Table 4.4:** The results of TS-DST for VRPTWDST on the Solomon benchmark set in comparison to the best VRPTW methods from the literature and the two most successful tabu searches for VRPTW. Average results are given for each problem group and best-known solutions are indicated in bold. Moreover, the cumulated number of vehicles (CNV), the cumulated traveled distance (CTD) and run-time related info is provided and the translation into a common time measure is given.

benchmark with a CNV of 405 vehicles and thus the best-performing TS for the VRPTW proposed in the literature. The formerly best-performing TS heuristics for VRPTW are Taillard et al. (1997) with a CNV of 410 and a CTD of 57523 and Cordeau et al. (2001) with a CNV of 407 and a CTD of 57556.

Concerning run-time, our approach needs on average 65.7 seconds to solve each instance. This translates to a value of 19.14 for the common time measure and thus, TS-DST is the third fastest of the methods presented in the table. run-times to the formerly best tabu searches are not comparable as Taillard et al. (1997) and Cordeau et al. (2001) do not report the necessary values in their papers. In comparison to the results of the VRPTWDST test sets, a slight reduction of the run-times for VRPTW can be noticed. This can be explained by the extra effort necessary to apply the time travel approach for time window handling in a driver-specific manner.

## 4.4 Summary and Conclusion

In this chapter, we investigate VRPTWDST, a route planning problem that integrates different extents of available driver knowledge by means of driver-specific times. We present a mathematical model of the problem and develop TS-DST as metaheuristic solution method. After a preprocessing step removing infeasible arcs, an initial solution is generated by means of a sequential insertion heuristic inspired by Solomon (1987). TS-DST evaluates solutions based on a generalized cost function penalizing capacity and time window violations. For calculating the latter, the time travel approach is adapted by storing and updating driver-specific time window handling variables. The Relocate and Exchange operator are used for intra- and inter-route moves and 2-opt* for intra-route moves. TS-DST uses probabilistic phases, continuous diversification and random shaking to diversify the search.

We use the mechanisms *Random* and *Cluster* to create several sets of VRPTWDST benchmark instances with different distributions of the learned customers. Tests on the generated instances show the ability of TS-DST to exploit the reduced travel and service times in order to produce efficient vehicle routes. Moreover, we find that, compared to the *Random* instances, the efficiency gains increase for the *Cluster* instances as routes can be designed to reap

learning benefits if the learned customers of a driver are geographically close. As could be expected, a reduction of the learning factor leads to a decrease in the number of required vehicles.

Further tests on benchmark instances of the closely related VRPTW see TS-DST on rank 9 of the best solution methods proposed for VRPTW concerning solution quality and on rank 3 concerning run-time. Moreover, by accomplishing a CNV of 405 vehicles, TS-DST is the most successful TS method ever proposed for the VRPTW.

# Chapter 5

# The Electric Vehicle Routing Problem with Time Windows and Recharging Stations

Driven by new laws and regulations concerning the emission of greenhouse gases, carriers are starting to use battery electric vehicles (BEVs) for last-mile deliveries in order to meet future emission standards and to reduce energy costs. The limited battery capacities of BEVs necessitate visits to recharging stations during delivery tours of industry-typical length, which have to be considered in the route planning in order to avoid inefficient vehicle routes with long detours. Recharging operations take a significant amount of time, especially when compared to the relatively short customer service times of small package shipping (SPS) companies, and thus clearly affect route planning.

Moreover, SPS companies require the incorporation of their most important practical constraints into routing models for electric vehicles, namely limited vehicle freight capacities and time-definite deliveries. The latter requirement is especially challenging as recharging times for BEVs cannot be assumed to be fixed but depend on the current battery charge of the vehicle when arriving at the recharging station. In this chapter, we study the Electric Vehicle Routing Problem with Time Windows and Recharging Stations (E-VRPTW), which 1) incorporates the possibility of recharging at any of the available stations with recharging times depending on the level of charge when arriving at the station, and 2) considers capacity constraints on vehicles as well as customer time windows.

A mathematical description of E-VRPTW is provided in Section 5.1. As solution method for the NP-hard problem, we present a hybrid metaheuristic that combines a Variable Neighborhood Search algorithm (VNS) with a Tabu Search (TS) in Section 5.2. In Section 5.3, we perform tests on newly designed benchmark instances for E-VRPTW in order to assess the performance of the VNS/TS

and to study the effectiveness of every component of our hybrid solution method. Moreover, we prove the quality and efficiency of VNS/TS on test instances of related problems, namely the Green VRP (G-VRP) and the Multi-Depot VRP with Inter-Depot Routes (MDVRPI).

## 5.1 Problem Definition

We formulate the E-VRPTW as an extension of the VRPTW model introduced in Section 2.1.1. The problem description is self-contained, repeating some of the already presented variable definitions. Let $V'$ be a set of vertices with $V' = V \cup F'$, where $V = \{1, \ldots, N\}$ denotes the set of customers and $F'$ a set of dummy vertices generated to permit several visits to each vertex in the set $F$ of recharging stations. Vertices 0 and $N+1$ denote the same depot and every route starts at 0 and ends at $N+1$. In order to indicate that a set contains the respective instance of the depot, the set is subscripted with 0 or $N+1$, e.g., $V'_0 = V' \cup \{0\}$.

Thus, E-VRPTW can be defined on a complete directed graph $G = (V'_{0,N+1}, A)$, with the set of arcs $A = \{(i,j) \mid i,j \in V'_{0,N+1}, i \neq j\}$. With each arc, we associate a distance $d_{ij}$ and a travel time $t_{ij}$. Each traveled arc consumes the amount $h \cdot d_{ij}$ of the remaining battery charge of the vehicle traveling the arc, where $h$ denotes the constant charge consumption rate.

A set of homogeneous vehicles with a maximal capacity of $C$ is positioned at the depot. Each vertex $i \in V'_{0,N+1}$ has a positive demand $q_i$, which is 0 if $i \notin V$. Moreover, a time window $[e_i, l_i]$ in which service has to start is associated with each vertex $i \in V'_{0,N+1}$ and all vertices $i \in V_{0,N+1}$ have a service time $s_i$ ($s_0, s_{N+1} = 0$). At a recharging station, the difference between the present charge level and the battery capacity $Q$ is recharged with a recharging rate of $g$, i.e., the recharging time incurred depends on the energy level of the vehicle when arriving at the respective station.

We use decision variables associated with vertices to keep track of vehicle states. Variable $\tau_i$ specifies the time of arrival, $u_i$ the remaining cargo and $y_i$ the remaining charge level on arrival at vertex $i \in V'_{0,N+1}$. The decision variables $x_{ij}, i \in V'_0, j \in V'_{N+1}, i \neq j$ are binary and equal 1 if an arc is traveled and 0 otherwise. The objective function of E-VRPTW is hierarchical. As commonly done for VRPs with time window constraints (see Section 2.1), our first objective

is to minimize the number of vehicles and the second objective to minimize the total traveled distance.

The mathematical model of E-VRPTW is formulated as mixed-integer program as follows:

$$\min \sum_{i \in V'_0, j \in V'_{n+1}, i \neq j} d_{ij} x_{ij} \tag{5.1}$$

$$\sum_{j \in V'_{n+1}, i \neq j} x_{ij} = 1 \quad \forall i \in V \tag{5.2}$$

$$\sum_{j \in V'_{n+1}, i \neq j} x_{ij} \leq 1 \quad \forall i \in F' \tag{5.3}$$

$$\sum_{i \in V'_{n+1}, i \neq j} x_{ji} - \sum_{i \in V'_0, i \neq j} x_{ij} = 0 \quad \forall j \in V' \tag{5.4}$$

$$\tau_i + (t_{ij} + s_i) x_{ij} - l_0(1 - x_{ij}) \leq \tau_j \quad \forall i \in V_0, \forall j \in V'_{n+1}, i \neq j \tag{5.5}$$

$$\tau_i + t_{ij} x_{ij} + g(Q - y_i) - (l_0 + gQ)(1 - x_{ij}) \leq \tau_j \quad \forall i \in F', \forall j \in V'_{n+1}, i \neq j \tag{5.6}$$

$$e_j \leq \tau_j \leq l_j \quad \forall j \in V'_{0,n+1} \tag{5.7}$$

$$0 \leq u_j \leq u_i - q_i x_{ij} + C(1 - x_{ij}) \quad \forall i \in V'_0, \forall j \in V'_{n+1}, i \neq j \tag{5.8}$$

$$0 \leq u_0 \leq C \tag{5.9}$$

$$0 \leq y_j \leq y_i - (h \cdot d_{ij}) x_{ij} + Q(1 - x_{ij}) \quad \forall j \in V'_{n+1}, \forall i \in V, i \neq j \tag{5.10}$$

$$0 \leq y_j \leq Q - (h \cdot d_{ij}) x_{ij} \quad \forall j \in V'_{n+1}, \forall i \in F'_0, i \neq j \tag{5.11}$$

$$x_{ij} \in \{0, 1\} \quad \forall i \in V'_0, j \in V'_{n+1}, i \neq j \tag{5.12}$$

The objective of minimizing the traveled distance is defined in (5.1). In order to obtain the hierarchical objective function, the technique described in Section 2.1.1 can be used. Constraints (5.2) enforce the connectivity of costumers and Constraints (5.3) handle the connectivity of visits to recharging stations by restricting the number of outgoing arcs of a customer and a recharging visit vertex. Constraints (5.4) establish flow conservation by guaranteeing that at each vertex, the number of incoming arcs is equal to the number of outgoing arcs.

Constraints (5.5) guarantee time feasibility for arcs leaving customers and the depot. If arc $(i, j)$ is traveled, the arrival time $\tau_j$ at vertex $j$ must be at least the arrival time at vertex $i$ plus service time at vertex $i$ plus travel time from $i$ to $j$. If $(i, j)$ is not part of the solution, the condition is invalidated by subtracting

the latest feasible arrival time at the depot from the arrival time at vertex $i$. Thus, the arrival time at $j$ is only required to be greater than some negative value. Constraints (5.6) provide time feasibility for arcs leaving recharging visits, using a similar technique but considering recharging time instead of service time. Constraints (5.7) enforce that every vertex is visited within its time window and Constraints (5.5) - (5.7) further prevent the formation of subtours.

Constraints (5.8) and (5.9) guarantee demand fulfillment at all customers by assuring a nonnegative cargo load upon arrival at any vertex including the depot. This is achieved by subtracting the demand of a customer $i$ from the remaining load when leaving vertex $i$ in Constraints (5.8). Finally, Constraints (5.10) and (5.11) ensure that the battery charge never falls below zero. To this end, we define the battery to be completely charged at the depot and the recharging stations in Constraints (5.11) and reducing the charge level by the energy it takes to travel an arc in Constraints (5.10) and (5.11).

## 5.2 A Hybrid VNS/TS Solution Method for the E-VRPTW

As solution method for E-VRPTW, we use a combination of VNS and TS, a hybrid that has already proven its performance on routing and related problems (see, e.g., Melechovský et al. 2005, Tarantilis et al. 2008). As described in Section 2.1.2, VNS is an effective metaheuristic performing local search on increasingly larger neighborhoods in order to efficiently explore the solution space and to avoid getting stuck in local optima. It has successfully been applied to a variety of combinatorial optimization problems, among them routing problems like VRPTW with single or multiple depots (Bräysy 2003, Polacek et al. 2004, Tarantilis et al. 2008).

Figure 5.1 presents our solution method in pseudocode. After a preprocessing step removing infeasible arcs, we generate an initial solution $S$ with a given number of vehicles as described in Section 5.2.1. Infeasible solutions are allowed during the search and evaluated based on a penalizing cost function (see Section 5.2.2). Our VNS/TS is not designed to minimize the number of employed vehicles and the search is started from a given vehicle number $m$. To minimize the number of vehicles, the search is applied for various values of $m$ as described in Section 4.2.6. Starting from the given vehicle number, we first perform a feasibility

```
N_κ ← set of VNS neighborhood structures for κ = 1, ..., κ_max
S ← generateInitialSolution()
κ ← 1
i ← 0
feasibilityPhase ← true
while feasibilityPhase ∨ (¬feasibilityPhase ∧ i < η_dist) do
   S' ← generateRandomPoint(N_κ(S))
   S'' ← applyTabuSearch(S', η_tabu)
   if acceptSA(S'', S) then
      S ← S''
      κ ← 1
   else
      κ ← κ + 1  mod κ_max
   end if
   if feasibilityPhase then
      if ¬ feasible(S) then
         if i = η_feas then
            addVehicle(S)
            i ← −1
         end if
      else
         feasibilityPhase ← false
         i ← −1
      end if
   end if
   i ← i + 1
end while
```

**Figure 5.1:** Overview of the VNS/TS algorithm for solving E-VRPTW

phase during which $m$ is increased after no feasible solution has been found for a given number of $\eta_{feas}$ iterations. After a feasible solution is found, another $\eta_{dist}$ iterations are performed to improve traveled distance.

The search is guided by a VNS component described in Section 5.2.3. It uses the current VNS neighborhood $\mathcal{N}_\kappa$ to generate a random perturbation which serves as initial solution for $\eta_{tabu}$ iterations of the TS phase (Section 5.2.4). The acceptance criterion of the VNS is based on Simulated Annealing (SA).

### 5.2.1 Preprocessing and Generation of Initial Solution

We apply a series of preprocessing steps to remove infeasible arcs. Arc $(v, w)$ can be removed from the set of possible arcs if one of the following conditions holds:

$$v, w \in V \wedge q_v + q_w \geq C \tag{5.13}$$

$$v \in V_0', w \in V_{N+1}' \wedge e_v + s_v + t_{vw} \geq l_w \tag{5.14}$$

$$v \in V_0', w \in V' \wedge e_v + s_v + t_{vw} + s_w + t_{wN+1} \geq l_0 \tag{5.15}$$

$$v, w \in V \wedge \forall j \in F_0', i \in F_{N+1}' : h(d_{jv} + d_{vw} + d_{wi}) \geq Q \tag{5.16}$$

Equation (5.13) - (5.15) are well-known preprocessing steps that base on capacity and time window violations and have already been detailed in the context of driver-specific times in Section 4.2.1. Equation (5.16) is problem-specific and refers to violations of the battery capacity. If the charge consumption of traveling an arc and traveling to and from that arc to any station or the depot is higher than the battery capacity, this arc can be labeled infeasible. Numerical studies showed that this preprocessing step is able to perceptibly reduce the number of feasible arcs on our E-VRPTW test instances.

We construct an initial solution similar to the approach proposed in Cordeau et al. (2001). First, all customers are sorted in increasing order of the angle between the depot, a randomly chosen point and the customer. Then, customers are iteratively inserted into the active route at the position causing minimal increase in traveled distance until a violation of capacity or battery capacity occurs. If a violation occurs, we activate a new route until at most the predefined number of routes are opened, then the remaining customers are inserted into the last route. The battery capacity violation is determined under the assumption that no recharging possibility exists. To consider time window requirements, a customer $u$ is only allowed to be inserted between successive vertices $i$,$j$ if $e_i \leq e_u \leq e_j$. This rule helps to fulfill time windows but feasibility is only guaranteed concerning capacity and battery capacity for all routes but the last.

### 5.2.2 Generalized Cost Function

VNS/TS allows infeasible solutions during the search process. We evaluate a solution by means of the following generalized cost function:

$$f_{gen}(S) = f(S) + \alpha P_c(S) + \beta P_{tw}(S) + \gamma P_{batt}(S), \tag{5.17}$$

where $f(S)$ denotes the total traveled distance, $P_c(S)$ the total capacity violation, $P_{tw}(S)$ the time window violation, $P_{batt}(S)$ the battery capacity violation. The penalty factors $\alpha$, $\beta$ and $\gamma$ are dynamically updated between a given lower and upper bound as described in Section 3.1.1. The update is performed every $\eta_{penalty}$ consecutive iterations with a feasible/infeasible solution and the update factor is denoted as $\delta$.

In the following, we describe the efficient calculation of the constraint violations. Capacity violations are defined and determined as described in Section 2.2. In this way, we are able to calculate the change in capacity violation in constant time $\mathcal{O}(1)$ for all neighborhood operators of our TS method, which are introduced in Section 5.2.4.

To calculate battery capacity violations, we define the following two variables for each vertex of a route $r = \langle v_0, v_1, \ldots, v_n, v_{n+1} \rangle$: $\Upsilon^{\rightarrow}_{v_i}$ contains the battery charge that is needed to travel either from the last recharging station visit or from the depot to vertex $v_i$ and $\Upsilon^{\leftarrow}_{v_i}$ is the battery charge that is needed to travel from $v_i$ to either the next recharging station or the depot:

$$\Upsilon^{\rightarrow}_{v_i} = \begin{cases} h \cdot d_{v_{i-1}v_i} & \text{if } v_{i-1} \in F'_0 \\ \Upsilon^{\rightarrow}_{v_{i-1}} + h \cdot d_{v_{i-1}v_i} & \text{otherwise} \end{cases} \qquad i = 1, \ldots, n+1$$

$$\Upsilon^{\leftarrow}_{v_i} = \begin{cases} h \cdot d_{v_iv_{i+1}} & \text{if } v_{i+1} \in F'_{n+1} \\ \Upsilon^{\leftarrow}_{v_{i+1}} + h \cdot d_{v_iv_{i+1}} & \text{otherwise} \end{cases} \qquad i = 0, \ldots, n.$$

Using these variables, the battery capacity violation of a route $r$ can be defined as the sum of individual violations at every visit to a recharging station and on return to the depot:

$$P_{batt}(r) = \sum_{v_i \in Vert(r) \cap F'_{N+1}} \max\{\Upsilon^{\rightarrow}_{v_i} - Q, 0\}.$$

Thus, changes in battery capacity violation can be calculated in $\mathcal{O}(1)$ for all neighborhood operators described in Section 5.2.4 by calculating the change in battery capacity violation of the recharging stations immediately following the points of vertex insertion, removal or merging of both routes.

To calculate time window violations, we adapt the corrected time travel approach described in Section 2.2 to the E-VRPTW. For the VRPTW, by storing forward and backward time window penalty slacks, it is possible to calculate in constant time the time window penalties of a route $r_1 = \langle v_0, \ldots, u, w, \ldots, v_{n+1} \rangle$ that is constructed from two partial routes $\langle v_0, \ldots, u \rangle$ and $\langle w, \ldots, v_{n+1} \rangle$ or of a route $r_2 = \langle v_0, \ldots, u, v, w, \ldots, v_{n+1} \rangle$ that is constructed by inserting a vertex $v$ between two partial routes $\langle v_0, \ldots, u \rangle$ and $\langle w, \ldots, v_{n+1} \rangle$.

This is not always possible if recharging stations are present as the recharging time at a station depends on the battery charge, which itself depends on the traveled distance to the recharging station. If the partial route $\langle w, \ldots, v_{n+1} \rangle$ contains a recharging station $x$, i.e., $\langle w, \ldots, x, x+1, \ldots, v_{n+1} \rangle$, slack variables have to be recalculated by traversing the partial route $\langle w, \ldots, x+1 \rangle$ for $r_1$ and the partial route $\langle v, \ldots, x+1 \rangle$ for $r_2$. Note that a recharging station in the first partial route $\langle v_0, \ldots, u \rangle$ or the vertex to insert $v$ being a recharging station does not necessitate a recalculation.

### 5.2.3 The Variable Neighborhood Search Component

Within our hybrid VNS/TS heuristic, the VNS component is mainly used to diversify the search in a structured way. Our VNS applies a shaking phase as described in the standard VNS in Section 2.1.2 but the intensification phase and the acceptance criterion clearly differ from the standard VNS. In the following, we detail the shaking phase, the components of the VNS in our hybrid heuristic.

In every iteration, our VNS performs a random perturbation move according to the predefined neighborhood structure $\mathcal{N}_\kappa$. The neighborhood structures are all defined by means of the Cyclic-Exchange operator. In the Cyclic-Exchange, introduced by Thompson and Orlin (1989), customer sequences of arbitrary length are simultaneously transferred between routes. Figure 5.2 shows a Cyclic-Exchange involving three routes. Cyclic-Exchange, or a variant that is restricted to two routes, called Cross-Exchange, are commonly used in the perturbation phase of

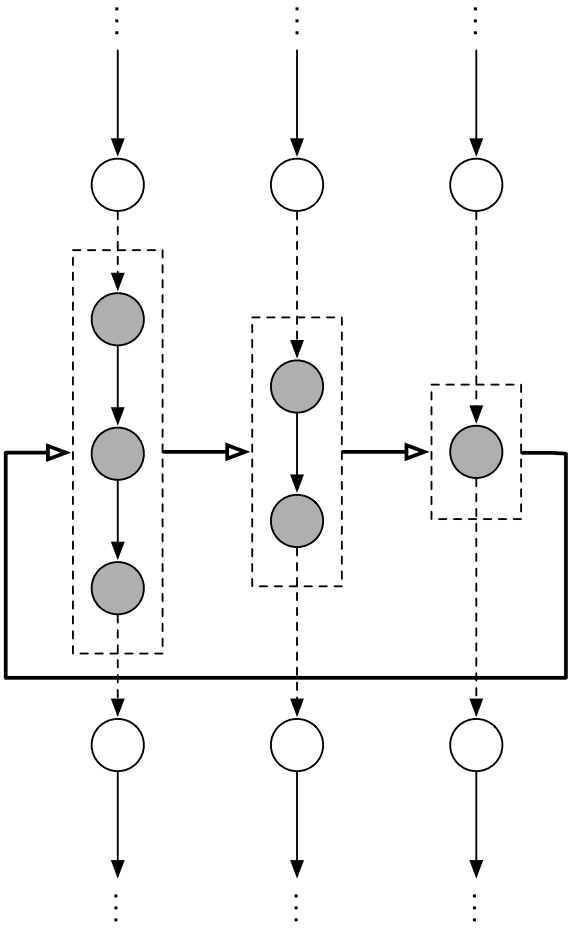


**Figure 5.2:** Example of a Cyclic-Exchange move involving three routes

VNS-algorithms due to their strong diversification capabilities (see, e.g, Polacek et al. 2004, Hemmelmayr et al. 2009).

Our $\kappa$ neighborhood structures, shown in Table 5.1, are defined according to two parameters. The number of routes that form the cycle is equal to $\#Rts$. In each route $r_k$, we randomly select the number of successive vertices that form the translocation chain in the interval $[0, \min\{\Lambda_{max}, n_k\}]$, where $n_k$ denotes the number of customers and stations contained in $r_k$ and $\Lambda_{max}$ the maximum number of translocated vertices. The initial vertex of the translocation chain is randomly chosen for each route.

Contrary to the local descent commonly used in VNS approaches, we use a TS to improve the randomly generated solution $S'$ in the intensification phase. The TS, which is detailed in Section 5.2.4, is run for $\eta_{tabu}$ iterations. Note that the

| $\kappa$ | $\#Rts$ | $\Lambda_{max}$ | $\kappa$ | $\#Rts$ | $\Lambda_{max}$ | $\kappa$ | $\#Rts$ | $\Lambda_{max}$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 1 | 6 | 3 | 1 | 11 | 4 | 1 |
| 2 | 2 | 2 | 7 | 3 | 2 | 12 | 4 | 2 |
| 3 | 2 | 3 | 8 | 3 | 3 | 13 | 4 | 3 |
| 4 | 2 | 4 | 9 | 3 | 4 | 14 | 4 | 4 |
| 5 | 2 | 5 | 10 | 3 | 5 | 15 | 4 | 5 |

**Table 5.1:** The $\kappa$-neighborhood structures used in the VNS defined by the number of involved route $\#Rts$ and the maximum number of translocated vertices $\Lambda_{max}$

perturbation move is added to the tabu list to prevent its reversal. Subsequently, we compare the best solution found during the TS $S''$ to the initial solution $S$. Instead of accepting only improving solutions, we use an acceptance criterion based on the metaheuristic SA (see Section 2.1.2). This method has been successfully applied in several VNS approaches, for example, in Hemmelmayr et al. (2009) and Stenger et al. (2011).

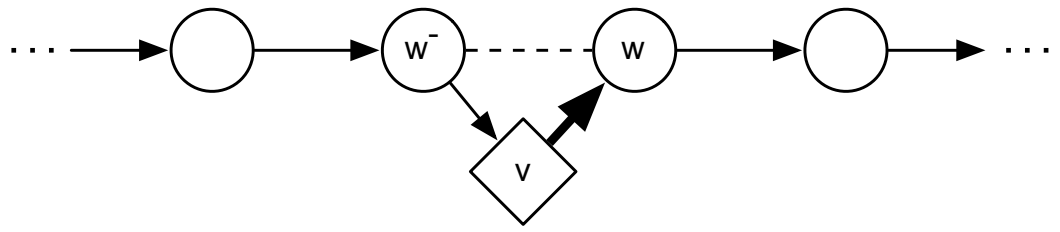
Like in SA, we always accept improving solutions, while deteriorating solutions are accepted according to the probability $e^{\frac{-(f(S'')-f(S))}{T}}$. At the beginning of the search, we initialize the temperature $T$ to $T_0$ in a way that a solution value $f(S'')$, which is $\Delta_{SA}$ worse than $f(S)$ is accepted with a probability of 50%. In this way, deteriorating solutions are often accepted, which helps to diversify the search. After every VNS iteration, the temperature is linearly decreased with a cooling factor that is chosen such that the temperature is below 0.0001 during the last 20% of iterations. By continuously decreasing the temperature during the search, an intensification is achieved and, finally, only improving solutions are accepted.

### 5.2.4 The Tabu Search Component

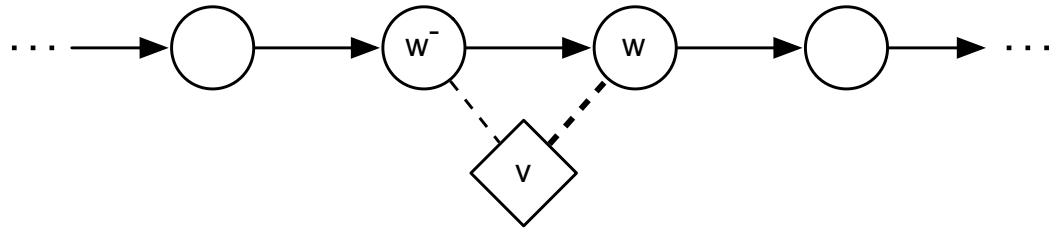
The TS phase starts from the solution $S'$ generated by the perturbation move of the VNS component. In each iteration, the composite neighborhood $\mathcal{N}(S)$ of TS is generated by applying the following neighborhood operators on every arc in the list of generator arcs (cp. Section 4.2.4): 2-opt*, Relocate, Exchange and a new, problem-specific operator called *stationInRe*. Each move is evaluated and the best non-tabu move is performed. A move is superior if it is able to reduce the number of employed vehicles or if it has a lower cost function value calculated with Equation (5.17). The 2-opt*, Relocate and Exchange operator and their realization by means of generator arcs are described in detail in Section 4.2.4.

Here, we apply 2-opt* for inter-route moves and define the operator for moving recharging stations, i.e., we allow the removal and insertion of arcs including recharging stations. The Relocate operator is also defined for recharging stations and applied as intra and inter-route operator. The Exchange operator is applied for inter-route and intra-route moves, but is not defined for recharging stations, i.e., we exclude the swapping of a recharging station with a customer or another station.

As the name suggests, the stationInRe operator performs insertions and removals of recharging stations. The operator is defined for all generator arcs $(v, w)$, where either $v$ or $w$ is a recharging station. Let $w^-$ denote the predecessor of vertex $w$. If the arc $(v, w)$ is not part of the current solution, stationInRe performs an insertion as depicted in Figure 5.3(a). If the arc is already present, a recharging station is removed as shown in Figure 5.3(b).




(a) Insertion




(b) Removal

**Figure 5.3:** Insertion and removal of a recharging station with the stationInRe operator. Generator arcs are shown in bold and removed arcs as dashed lines.

We set every arc $\xi$ that is deleted from the solution by the execution of a move tabu, i.e, we forbid the reinsertion of the arc into specific parts of the solution for a tabu tenure $\vartheta$ randomly drawn from the interval $[\vartheta_{min}, \vartheta_{max}]$. As station visits have a strong effect on charge levels and also on time windows due to the

recharging times incurred, we define the tabu attribute $(\xi, k, \Omega, \Phi)$. It prohibits the insertion of arc $\xi$ into route $r_k$ between $\Omega$ and $\Phi$, where $\Omega, \Phi \in F_{0,n+1}$ denote either a station or the depot. In this way, we allow the reinsertion of an arc into a different part of the route. The tabu status of a move is lifted if a feasible new best solution is generated.

To further diversify the search, we adapt the continuous diversification mechanism described in Section 3.1.1 to E-VRPTW. To this end, we define vertex-based attributes $(u, k, \Omega, \Phi)$ to describe that customer/station $u$ is positioned between stations/depot $\Omega$ and $\Phi$ in route $r_k$. In this way, each solution $S$ can be characterized by the attribute set $B(S) = \{(u, k, \Omega, \Phi)\}$. For each attribute, the frequency $\varrho_{uk\Omega\Phi}$ of its addition to a solution in previous moves is memorized and used to penalize solutions according to the frequency of their attributes. Thus, we guide the search to explore the possibilities of using different stations and different positions of customers and stations (relative to other stations or the depot) within a route. A solution $S'$ that deteriorates the current solution is penalized by:

$$P_{div}(S') = \lambda_{div} \cdot f(S') \cdot \sqrt{|V'|\, m(S')} \sum_{(u,k,\Omega,\Phi)\in B(S')} \varrho_{uk\Omega\Phi},$$

where $\lambda_{div}$ denotes the diversification factor, $|V'|$ the number of customers and utilized stations and $m(S')$ the number of vehicles in solution $S'$. The TS procedure stops after $\eta_{tabu}$ iterations.

## 5.3 Numerical Experiments

In this section, we present the extensive numerical testing conducted to evaluate the performance of our hybrid solution method. The first study evaluates the performance of VNS/TS on E-VRPTW instances. To be able to asses the solution quality, we use newly designed small instances which can be solved by means of the commercial solver IBM ILOG CPLEX Optimizer (CPLEX). In our second study, we analyze the efficiency of the algorithmic components of our hybrid heuristic, namely the VNS, TS and SA, on a set of medium-sized E-VRPTW instances, which we design based on classical Solomon VRPTW instances. Finally, we demonstrate the strong performance concerning solution

quality and run-time of our VNS/TS on available benchmark instances of the related problems MDVRPI and G-VRP.

The section is structured as follows. After a brief discussion of the chosen parameter setting in Section 5.3.1, we describe the tests performed on E-VRPTW benchmark instances in Section 5.3.2 and those performed on benchmark instances of related problems in Section 5.3.3.

### 5.3.1 Experimental Environment and Parameter Settings

All tests are performed on a desktop computer equipped with an Intel Core i5 750 processor clocked at 2.67 GHz with 4 GB RAM, running Windows 7 Professional. The VNS/TS is implemented as single-thread code in Java. The parameters we used to generate the final results are provided in Table 5.2. The presented values are the result of intensive studies we conducted to fine-tune our algorithm following the tuning strategy described in Section 3.2.

| VNS | | Penalties | | TS | |
|---|---|---|---|---|---|
| $\eta_{feas}$ | 500 | $\alpha_0, \beta_0, \gamma_0$ | 10 | $\vartheta_{min}$ | 15 |
| $\eta_{dist}$ | 200 | $\alpha_{min}, \beta_{min}, \gamma_{min}$ | 0.5 | $\vartheta_{max}$ | 30 |
| $\Delta_{SA}$ | 0.08 | $\alpha_{max}, \beta_{max}, \gamma_{max}$ | 5000 | $\lambda_{div}$ | 1.0 |
| | | $\delta$ | 1.2 | $\eta_{tabu}$ | 100 |
| | | $\eta_{penalty}$ | 2 | | |

**Table 5.2:** Overview of the parameter setting of VNS/TS chosen for the numerical studies

Concerning the feasibility phase, the tests showed that if the VNS/TS is not able to find a feasible solution with the given number of vehicles in $\eta_{feas} = 500$ VNS iterations, it is very unlikely that a feasible solution with this vehicle number is found in later iterations. The entire algorithm terminates after $\eta_{dist} = 200$ additional distance minimization iterations as this resulted in a good tradeoff between computing time and solution quality. At the beginning of the search, the SA acceptance criterion accepts a solution $S''$ whose objective function value $f(S'')$ is $\Delta_{SA} = 8\%$ worse than $f(S)$ with a probability of 50%.

The initial penalty factors $\alpha_0, \beta_0, \gamma_0$ are set to 10 as this proved to be a good compromise between diversification and intensification at the beginning of the search. Subsequently, the penalty factors are updated by multiplying or dividing

by factor $\delta = 1.2$ after every $\eta_{penalty} = 2$ iterations with a feasible/infeasible solution, while limiting the values to the interval $[0.5, 5000]$.

Furthermore, the length of the tabu list clearly affected the performance of our algorithm. However, we were not able to find a unique value that performed well on all instances of the different benchmark sets that we solved. Instead, we achieved the overall best results by randomly selecting the length from the interval $[15, 30]$ in each iteration. The TS is performed for $\eta_{tabu} = 100$ iterations using $\lambda_{div} = 1.0$ as diversification factor of the continuous diversification mechanism.

### 5.3.2 Experiments on E-VRPTW Instances

As we are the first to study E-VRPTW, no benchmark instances for assessing solution methods for this problem exist. We design two new sets of benchmark instances, which we describe in the following section. Subsequently, we present the results of our testing on the generated instances.

#### 5.3.2.1 Generation of E-VRPTW Benchmark Instances

We create two sets of benchmark instances for the E-VRPTW. A set of 56 large instances, each with 100 customers and 21 recharging stations, and a set of 36 small instances with 5, 10 and 15 customers per instance[1]. All instances are created based on the Solomon VRPTW benchmark described in Section 2.1.3. To generate E-VRPTW instances based on VRPTW instances, we have to 1) position the recharging stations, 2) set battery capacities, recharging and consumption rates and 3) adjust time windows in order to generate feasible instances.

**5.3.2.1.1 Location of Recharging Stations** We locate one recharging station at the depot because a recharging possibility at the depot seems to be a reasonable claim. The location of the remaining 20 stations is determined in a random manner. However, we limit the possible locations in order to generate feasible and meaningful instances, i.e., every customer can be reached from the depot using at most two different recharging stations. To this end, the recharging stations are placed within an area defined by three circles surrounding the depot. The circles are divided into equally spaced segments whose size depends on the number of

[1]The generated instances are available for download at `http://evrptw.wiwi.uni-frankfurt.de`

stations to place, as illustrated in Figure 5.4. Two recharging stations are then randomly positioned into each segment, one between the inner and middle circle and the other between the middle and outer circle.

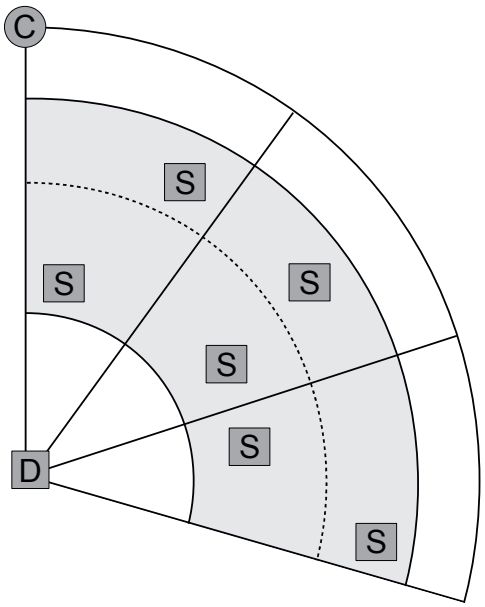



**Figure 5.4:** Positioning of recharging stations to generate E-VRPTW instances. C denotes the most remote customer in an instance, stations are denoted with S and D denotes the depot.

**5.3.2.1.2 Battery Capacity, Recharging and Consumption Rate** The battery capacity is set to the maximum of the following two values: 1) the charge needed to travel 60% of the average route length of the best known solution to the corresponding VRPTW instance and 2) twice the amount of battery charge required to travel the longest arc between a customer and a station. This procedure ensures that instances with geographically disperse and remote customers stay feasible. Furthermore, we thus guarantee that recharging stations have to be used. For the sake of simplicity, we set the consumption rate $h$ to 1.0. The recharging rate $g$ is set so that a complete recharge requires three times the average customer service time of the respective instance.

**5.3.2.1.3 Adjustment of Time Windows** The detours for visits to recharging stations and the recharging times incurred make it impossible to comply with the

customer time windows given in the original Solomon instances, i.e., some instances become infeasible because no possibility exists to reach certain customers within their original time window. Consequently, we have to generate new time windows to obtain feasible instances. The procedure used is very close to the original one described in Solomon (1987).

First, we determine for each customer the feasible time window range, i.e., the earliest time at which the customer is reachable from the depot (with potentially necessary station visits) and the latest time at which the customer must be left so that the depot can be reached in time. For instance sets R and RC, the time window center is randomly drawn from the determined interval. Like in Solomon (1987), the time window centers for set C are set corresponding to real arrival times, which we obtained by solving the instances without time windows using our VNS/TS. Next, the time window widths are chosen according to the respective original instances. If a thus generated customer time window is outside the feasible range, the violating range is cut and the time window is extended to the other side of the interval.

**5.3.2.1.4 Generation of Small-Sized Instances** To generate the set of small instances, we start with the 56 large instances described above. For each of the three sizes ($5, 10, 15$ customers), we randomly draw the respective number of customers from the large instances, thus generating 168 instances. The created instances are then solved with our VNS/TS heuristic and the solutions are inspected. For each problem group and instance size, we select the two instances whose solution uses the highest number of recharging stations. In this way, we create $6 \cdot 3 \cdot 2 = 36$ small test instances, which are denoted by the identifier of the underlying Solomon instance followed by the number of customers in the instance, e.g, R108-5.

### 5.3.2.2 Performance of VNS/TS on Small E-VRPTW Instances

We use the generated E-VRPTW test instances to analyze the performance of our VNS/TS heuristic on small E-VRPTW instances. To this end, we solve the instances with VNS/TS and compare the obtained results to the optimal (or near optimal) solution found by the commercial solver CPLEX 12.2 using the model described in Section 5.1 with the hierarchical objective function.

Table 5.3 provides an overview of the results. For both, CPLEX and our heuristic, we provide the computing time in seconds in column $t(s)$. For the solutions obtained with CPLEX, the vehicle number and traveled distance given in columns $m$ and $f$ correspond to the optimal solution, or the best upper bound found within 7200 seconds. For VNS/TS, columns $m$ and $f$ provide the best solution found in 10 runs and column $\Delta f$ denotes the gap to the traveled distance found by CPLEX. No gap for the number of employed vehicles needs to be reported as the numbers obtained with CPLEX and VNS/TS are identical for all instances.

The results clearly show the ability of our VNS/TS heuristic to solve small E-VRPTW instances to optimality in only a few seconds. Independent of the instance structure or size, we always obtain the optimal solution, if CPLEX finds an optimum within 7200 seconds. For most of the 15-customer instances and one 10-customer instance, CPLEX is not able to provide the optimal solution. On those instances, we either find a solution equal to the upper bound provided by CPLEX or a better solution in one case.

#### 5.3.2.3 Analyzing the Effect of the VNS/TS components

This section aims at demonstrating the positive effect achieved by the hybridization of VNS and TS. To this end, we compare the results obtained by our VNS/TS heuristic on the 100-customer E-VRPTW instances to the solutions found with 1) a VNS/TS heuristic that accepts only improving solutions after the TS phase instead of using an SA-based criterion (VNS/TS w/o SA) and 2) the pure TS heuristic.

An overview of the results is given in Table 5.4. For each heuristic, we provide the best solution found in 10 runs in columns $m$ and $f$. Furthermore, we determine gaps to the best-known solution (BKS) found during the overall testing for both the number of vehicles ($\Delta m$) and the traveled distance ($\Delta f$). Finally, at the bottom of the table, the average computing time in minutes is reported in row $t(\text{min})$.

The results show that the VNS/TS heuristic performs best with a vehicle gap of 0 and an average gap of the traveled distance of 0.35% to the BKS. A comparison to the results obtained with VNS/TS w/o SA allows to quantify the impact of the SA-based acceptance criterion. Using SA instead of simply accepting

| | **CPLEX** | | | **VNS/TS** | | | |
|---|---|---|---|---|---|---|---|
| Inst. | $m$ | $f$ | $t(s)$ | $m$ | $f$ | $\Delta f(\%)$ | $t(s)$ |
| C101-5 | 2 | 257.75 | 81 | 2 | 257.75 | 0.00 | 0.21 |
| C103-5 | 1 | 176.05 | 5 | 1 | 176.05 | 0.00 | 0.12 |
| C206-5 | 1 | 242.55 | 518 | 1 | 242.55 | 0.00 | 0.14 |
| C208-5 | 1 | 158.48 | 15 | 1 | 158.48 | 0.00 | 0.11 |
| R104-5 | 2 | 136.69 | 1 | 2 | 136.69 | 0.00 | 0.13 |
| R105-5 | 2 | 156.08 | 3 | 2 | 156.08 | 0.00 | 0.11 |
| R202-5 | 1 | 128.78 | 1 | 1 | 128.78 | 0.00 | 0.11 |
| R203-5 | 1 | 179.06 | 5 | 1 | 179.06 | 0.00 | 0.15 |
| RC105-5 | 2 | 241.30 | 764 | 2 | 241.3 | 0.00 | 0.14 |
| RC108-5 | 1 | 253.93 | 311 | 1 | 253.93 | 0.00 | 0.17 |
| RC204-5 | 1 | 176.39 | 54 | 1 | 176.39 | 0.00 | 0.15 |
| RC208-5 | 1 | 167.98 | 21 | 1 | 167.98 | 0.00 | 0.13 |
| C101-10 | 3 | 393.76 | 171 | 3 | 393.76 | 0.00 | 0.77 |
| C104-10 | 2 | 273.93 | 360 | 2 | 273.93 | 0.00 | 0.95 |
| C202-10 | 1 | 304.06 | 300 | 1 | 304.06 | 0.00 | 0.71 |
| C205-10 | 2 | 228.28 | 4 | 2 | 228.28 | 0.00 | 0.49 |
| R102-10 | 3 | 249.19 | 389 | 3 | 249.19 | 0.00 | 0.65 |
| R103-10 | 2 | 207.05 | 119 | 2 | 207.05 | 0.00 | 0.72 |
| R201-10 | 1 | 241.51 | 177 | 1 | 241.51 | 0.00 | 0.78 |
| R203-10 | 1 | 218.21 | 573 | 1 | 218.21 | 0.00 | 0.71 |
| RC102-10 | 4 | 423.51 | 810 | 4 | 423.51 | 0.00 | 0.69 |
| RC108-10 | 3 | 345.93 | 39 | 3 | 345.93 | 0.00 | 0.9 |
| RC201-10 | 1 | 412.86 | 7200 | 1 | 412.86 | 0.00 | 0.9 |
| RC205-10 | 2 | 325.98 | 399 | 2 | 325.98 | 0.00 | 0.81 |
| C103-15 | 3 | 384.29 | 7200 | 3 | 384.29 | 0.00 | 15.37 |
| C106-15 | 3 | 275.13 | 17 | 3 | 275.13 | 0.00 | 14.94 |
| C202-15 | 2 | 383.62 | 7200 | 2 | 383.61 | 0.00 | 13.41 |
| C208-15 | 2 | 300.55 | 5060 | 2 | 300.55 | 0.00 | 11.08 |
| R102-15 | 5 | 413.93 | 7200 | 5 | 413.93 | 0.00 | 19.55 |
| R105-15 | 4 | 336.15 | 7200 | 4 | 336.15 | 0.00 | 13.35 |
| R202-15 | 2 | 358.00 | 7200 | 2 | 358.00 | 0.00 | 13.17 |
| R209-15 | 1 | 313.24 | 7200 | 1 | 313.24 | 0.00 | 13.73 |
| RC103-15 | 4 | 397.67 | 7200 | 4 | 397.67 | 0.00 | 14.62 |
| RC108-15 | 3 | 370.25 | 7200 | 3 | 370.25 | 0.00 | 12.92 |
| RC202-15 | 2 | 394.39 | 7200 | 2 | 394.39 | 0.00 | 12.74 |
| RC204-15 | 1 | 407.45 | 7200 | 1 | 384.86 | -5.87 | 15.57 |
| **Avg.** | | | 2483.25 | | | -0.16 | 5.03 |

**Table 5.3:** Comparison of results obtained with CPLEX and VNS/TS on the small-sized instances. $m$ denotes the vehicle number and $f$ the traveled distance. $t(s)$ denotes the total run-time in seconds. The maximum duration for CPLEX was set to two hours, so optimality is not guaranteed for CPLEX results which used the full time.

| | BKS | | VNS/TS | | | | VNS/TS w/o SA | | | | TS | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Inst. | $m$ | $f$ | $m$ | $f$ | $\Delta_m$ | $\Delta_f(\%)$ | $m$ | $f$ | $\Delta_m$ | $\Delta_f(\%)$ | $m$ | $f$ | $\Delta_m$ | $\Delta_f(\%)$ |
| c101 | 12 | **1053.83** | 12 | **1053.83** | 0 | 0.00 | 12 | **1053.83** | 0 | 0.00 | 12 | **1053.83** | 0 | 0.00 |
| c102 | 11 | **1056.47** | 11 | 1057.16 | 0 | 0.07 | 11 | **1056.47** | 0 | 0.00 | 11 | 1069.35 | 0 | 1.22 |
| c103 | 10 | **1041.55** | 10 | 1041.55 | 0 | 0.00 | 11 | 1002.03 | 1 | -3.79 | 10 | 1134.36 | 0 | 8.91 |
| c104 | 10 | **979.51** | 10 | 980.82 | 0 | 0.13 | 10 | 988.77 | 0 | 0.95 | 10 | 979.63 | 0 | 0.01 |
| c105 | 11 | **1075.37** | 11 | **1075.37** | 0 | 0.00 | 11 | **1075.37** | 0 | 0.00 | 11 | 1079.69 | 0 | 0.40 |
| c106 | 11 | **1057.87** | 11 | **1057.87** | 0 | 0.00 | 11 | **1057.87** | 0 | 0.00 | 11 | **1057.87** | 0 | 0.00 |
| c107 | 11 | **1031.56** | 11 | **1031.56** | 0 | 0.00 | 11 | **1031.56** | 0 | 0.00 | 11 | 1033.08 | 0 | 0.15 |
| c108 | 10 | **1100.32** | 10 | **1100.32** | 0 | 0.00 | 11 | 1015.73 | 1 | -7.69 | 11 | 1015.73 | 1 | -7.69 |
| c109 | 10 | **1036.64** | 10 | 1051.84 | 0 | 1.47 | 10 | **1036.64** | 0 | 0.00 | 10 | 1051.36 | 0 | 1.42 |
| c201 | 4 | **645.16** | 4 | **645.16** | 0 | 0.00 | 4 | **645.16** | 0 | 0.00 | 4 | **645.16** | 0 | 0.00 |
| c202 | 4 | **645.16** | 4 | **645.16** | 0 | 0.00 | 4 | **645.16** | 0 | 0.00 | 4 | **645.16** | 0 | 0.00 |
| c203 | 4 | **644.98** | 4 | **644.98** | 0 | 0.00 | 4 | **644.98** | 0 | 0.00 | 4 | **644.98** | 0 | 0.00 |
| c204 | 4 | **636.43** | 4 | **636.43** | 0 | 0.00 | 4 | **636.43** | 0 | 0.00 | 4 | **636.43** | 0 | 0.00 |
| c205 | 4 | **641.13** | 4 | **641.13** | 0 | 0.00 | 4 | **641.13** | 0 | 0.00 | 4 | **641.13** | 0 | 0.00 |
| c206 | 4 | **638.17** | 4 | **638.17** | 0 | 0.00 | 4 | **638.17** | 0 | 0.00 | 4 | **638.17** | 0 | 0.00 |
| c207 | 4 | **638.17** | 4 | **638.17** | 0 | 0.00 | 4 | **638.17** | 0 | 0.00 | 4 | **638.17** | 0 | 0.00 |
| c208 | 4 | **638.17** | 4 | **638.17** | 0 | 0.00 | 4 | **638.17** | 0 | 0.00 | 4 | **638.17** | 0 | 0.00 |
| r101 | 18 | **1670.8** | 18 | 1672.55 | 0 | 0.10 | 18 | 1673.12 | 0 | 0.14 | 18 | **1670.8** | 0 | 0.00 |
| r102 | 16 | **1495.31** | 16 | 1535.81 | 0 | 2.71 | 16 | 1522.84 | 0 | 1.84 | 16 | **1495.31** | 0 | 0.00 |
| r103 | 13 | **1299.17** | 13 | 1299.64 | 0 | 0.04 | 13 | **1299.17** | 0 | 0.00 | 13 | 1348.25 | 0 | 3.78 |
| r104 | 11 | **1088.43** | 11 | **1088.43** | 0 | 0.00 | 11 | 1143.69 | 0 | 5.08 | 11 | 1097.09 | 0 | 0.80 |
| r105 | 14 | **1461.25** | 14 | 1473.59 | 0 | 0.84 | 15 | 1401.24 | 1 | -4.11 | 14 | 1514.36 | 0 | 3.63 |
| r106 | 13 | **1344.66** | 13 | **1344.66** | 0 | 0.00 | 13 | 1395.18 | 0 | 3.76 | 13 | 1369.55 | 0 | 1.85 |
| r107 | 12 | **1154.52** | 12 | **1154.52** | 0 | 0.00 | 12 | 1158.13 | 0 | 0.31 | 12 | 1162.9 | 0 | 0.73 |
| r108 | 11 | **1050.04** | 11 | 1065.89 | 0 | 1.51 | 11 | 1061.91 | 0 | 1.13 | 11 | 1056.84 | 0 | 0.65 |
| r109 | 12 | **1294.05** | 12 | **1294.05** | 0 | 0.00 | 12 | 1341.01 | 0 | 3.63 | 12 | 1308.62 | 0 | 1.13 |
| r110 | 11 | **1126.74** | 11 | 1143.52 | 0 | 1.49 | 11 | 1141.9 | 0 | 1.35 | 11 | **1126.74** | 0 | 0.00 |
| r111 | 12 | **1106.19** | 12 | 1124.06 | 0 | 1.62 | 12 | 1107.52 | 0 | 0.12 | 12 | 1123.96 | 0 | 1.61 |
| r112 | 11 | **1026.52** | 11 | **1026.52** | 0 | 0.00 | 11 | 1033.97 | 0 | 0.73 | 11 | 1047.92 | 0 | 2.08 |
| r201 | 3 | **1264.82** | 3 | **1264.82** | 0 | 0.00 | 3 | **1264.82** | 0 | 0.00 | 3 | 1266.26 | 0 | 0.11 |
| r202 | 3 | **1052.32** | 3 | **1052.32** | 0 | 0.00 | 3 | 1053.11 | 0 | 0.08 | 3 | 1052.65 | 0 | 0.03 |
| r203 | 3 | **895.91** | 3 | 912.86 | 0 | 1.89 | 3 | 914.68 | 0 | 2.10 | 3 | 914.1 | 0 | 2.03 |
| r204 | 2 | **790.57** | 2 | **790.57** | 0 | 0.00 | 2 | 801.56 | 0 | 1.39 | 2 | 790.68 | 0 | 0.01 |
| r205 | 3 | **988.67** | 3 | **988.67** | 0 | 0.00 | 3 | 1000.96 | 0 | 1.24 | 3 | 997.15 | 0 | 0.86 |
| r206 | 3 | **925.2** | 3 | **925.2** | 0 | 0.00 | 3 | 926.94 | 0 | 0.19 | 3 | 928.26 | 0 | 0.33 |
| r207 | 2 | **848.53** | 2 | 852.73 | 0 | 0.49 | 2 | **848.53** | 0 | 0.00 | 2 | 855.99 | 0 | 0.88 |
| r208 | 2 | **736.6** | 2 | **736.6** | 0 | 0.00 | 2 | 737.05 | 0 | 0.06 | 2 | 741.44 | 0 | 0.66 |
| r209 | 3 | **872.36** | 3 | **872.36** | 0 | 0.00 | 3 | 877.4 | 0 | 0.58 | 3 | 874.74 | 0 | 0.27 |
| r210 | 3 | **847.06** | 3 | **847.06** | 0 | 0.00 | 3 | 850.41 | 0 | 0.39 | 3 | 848.44 | 0 | 0.16 |
| r211 | 2 | **847.45** | 2 | 866.21 | 0 | 2.21 | 2 | 860.32 | 0 | 1.52 | 2 | 861.17 | 0 | 1.62 |
| rc101 | 16 | **1731.07** | 16 | **1731.07** | 0 | 0.00 | 16 | 1766.44 | 0 | 2.04 | 16 | 1753.35 | 0 | 1.29 |
| rc102 | 15 | **1554.61** | 15 | **1554.61** | 0 | 0.00 | 15 | 1556.08 | 0 | 0.09 | 15 | 1559.95 | 0 | 0.34 |
| rc103 | 13 | **1351.15** | 13 | 1353.55 | 0 | 0.18 | 13 | **1351.15** | 0 | 0.00 | 13 | 1355.36 | 0 | 0.31 |
| rc104 | 11 | **1238.56** | 11 | 1249.23 | 0 | 0.86 | 11 | 1267.55 | 0 | 2.34 | 11 | 1280.82 | 0 | 3.41 |
| rc105 | 14 | **1475.31** | 14 | 1483.38 | 0 | 0.55 | 14 | **1475.31** | 0 | 0.00 | 14 | 1479.56 | 0 | 0.29 |
| rc106 | 13 | **1437.96** | 13 | 1440.19 | 0 | 0.15 | 13 | 1469.99 | 0 | 2.23 | 13 | **1437.96** | 0 | 0.00 |
| rc107 | 12 | **1279.08** | 12 | 1275.89 | 0 | 0.00 | 12 | 1280.44 | 0 | 0.36 | 12 | 1284.47 | 0 | 0.67 |
| rc108 | 11 | **1209.61** | 11 | 1238.81 | 0 | 2.41 | 11 | 1227.88 | 0 | 1.51 | 11 | **1209.61** | 0 | 0.00 |
| rc201 | 4 | **1444.94** | 4 | 1447.2 | 0 | 0.16 | 4 | **1444.94** | 0 | 0.00 | 4 | 1446.03 | 0 | 0.08 |
| rc202 | 3 | **1418.79** | 3 | 1412.91 | 0 | 0.00 | 3 | **1418.79** | 0 | 0.42 | 3 | 1425.17 | 0 | 0.87 |
| rc203 | 3 | **1073.98** | 3 | 1078.28 | 0 | 0.40 | 3 | 1077.16 | 0 | 0.30 | 3 | 1084.66 | 0 | 0.99 |
| rc204 | 3 | **885.35** | 3 | 889.22 | 0 | 0.44 | 3 | 886.03 | 0 | 0.08 | 3 | 889.22 | 0 | 0.44 |
| rc205 | 3 | **1330.53** | 3 | 1321.75 | 0 | 0.00 | 3 | 1353.54 | 0 | 2.41 | 3 | 1360.39 | 0 | 2.92 |
| rc206 | 3 | **1190.75** | 3 | 1191.13 | 0 | 0.03 | 3 | 1204.93 | 0 | 1.19 | 3 | 1207.77 | 0 | 1.43 |
| rc207 | 3 | **1004.38** | 3 | 995.52 | 0 | 0.00 | 3 | 1015.6 | 0 | 2.02 | 3 | 1010.66 | 0 | 1.52 |
| rc208 | 3 | **837.82** | 3 | 838.03 | 0 | 0.03 | 3 | 838.41 | 0 | 0.07 | 3 | 838.03 | 0 | 0.03 |
| **Sum/Avg.** | | | | | 0 | 0.35 | | | 3 | 0.46 | | | 1 | 0.75 |
| $t$(min) | | | | 15.34 | | | | 16.22 | | | | 16.01 | | |

**Table 5.4:** Comparison of the effect of different heuristic components: VNS/TS denotes the standard setting of a hybrid VNS with an SA acceptance criterion. VNS/TS w/o SA denotes a combination of TS and a VNS only accepting improving solutions. TS denotes a pure TS without VNS. Gaps are calculated to the best known (BKS) solution found during the overall testing of the considered methods.

improving solutions reduces the cumulated vehicle number (CNV) by 3 and yields a reduction of the traveled distance of 0.1% on average. Comparing the results of VNS/TS to those of the pure TS, we can see that the hybridization of VNS and TS is able to reduce the CNV by one. Concerning the traveled distance, VNS/TS reduces the gap to the best known solution by more than half. Overall, the results show the positive effect of the hybridization of VNS, TS and SA. The solution quality is improved by every component incorporated into our heuristic, while computing times can even be slightly reduced.

### 5.3.3 Performance on Benchmark Instances of Related Problems

The E-VRPTW is closely related to the MDVRPI and the G-VRP (see Section 2.4). For both problems, sets of benchmark instances exist. To demonstrate the performance of our VNS/TS heuristic on large problem sets, we solve all benchmark instances available for the related problems and compare the results obtained to those reported for the competing algorithms, which were specifically designed for MDVRPI and G-VRP.

#### 5.3.3.1 Multi-Depot VRP with Inter-Depot Routes

For the MDVRPI (respectively the VRP with Intermediate Replenishment Facilities (VRPIRF), see Section 2.4), two benchmark sets with a total of 76 instances are available from the literature. The first set of benchmark instances was proposed by Crevier et al. (2007) and includes 22 instances. The instances consist of 48-216 customers, 3-6 depots and 4-6 vehicles. Depots are centered and customers are located in clusters. The second set was designed by Tarantilis et al. (2008) and involves 54 instances. The set consists of 18 depot-customer combinations, which were created following the design described in Crevier et al. (2007) and compromise 50-175 customers and 3-8 depots. From each of these 18 depot-customer combinations, three instances were created differing in the number of vehicles available.

In Table 5.5, we compare the results obtained with our VNS/TS on the instance set of Crevier et al. (2007) to the solutions of the heuristics of Tarantilis, Zachariadis and Kiranoudis (2008) (TZK) and Crevier, Cordeau and Laporte (2007) (CCL). For CCL and our VNS/TS, we provide the best solution found in 10 runs ($f_{best}$) and the computing time in minutes ($t$(min)). By contrast, the

value given in column $f_{best}$ for TZK corresponds to the best solution ever found with the final parameter setting. We further provide the gap of the best ($\Delta_{best}$) and average solution ($\Delta_{avg}$) to the best known solutions (BKS), which do not include the new best solutions that we found with our VNS/TS heuristic during the overall testing. They are shown in the last column ($\overline{\text{VNS/TS}}$) together with the percentage improvement compared to the formerly best known solutions.

Considering the complete set, our VNS/TS heuristic clearly outperforms the CCL approach in terms of solution quality and speed. We obtain an average gap to the best solution of 0.18% in about 27 minutes on average, while CCL achieves a 0.66% gap requiring more than double our computing time. In addition, we found 10 new overall best solutions during the testing. Tarantilis et al. (2008) solved only the first subset of instances with their TZK approach. Compared to their results, we are on average 0.48% worse, however, a direct comparison is not adequate, since the TZK results correspond to the best solution they ever found during their testing.

Table 5.6 compares the results of our VNS/TS heuristic with those of TZK on the instance set of Tarantilis et al. (2008). On those instances, our VNS/TS heuristic shows a really strong performance. Our results are on average 0.05% better than those of TZK. This is even more impressive when considering the fact that they provide only the best solution ever found. The gap of the average solution found by our VNS/TS is 1.44% and is hence also lower than the 1.6% gap of TZK. During our overall testing activities, we additionally obtained new best solutions for the majority of instances that improve the former ones by 0.45% on average.

#### 5.3.3.2 Green VRP

The benchmark instances for the G-VRP were proposed in Erdogan and Miller-Hooks (2012) and consist of four sets, each involving ten instances with 20 customers each. The instances differ in the customer distribution (random or clustered) and the number of available alternative fuel stations (AFS) (2 to 10). Furthermore, Erdogan and Miller-Hooks (2012) present a case study with 12 instances incorporating up to 500 customers.

Note that some customers contained in the small instances are infeasible, i.e., they cannot be served under the given restriction that each customer has to be

| | | CCL | | | | TZK | | | | VNS/TS | | | | $\overline{\text{VNS/TS}}$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Inst. | BKS | $f_{best}$ | $\Delta_{best}(\%)$ | $\Delta_{avg}(\%)$ | $t$(min) | $f_{best}$* | $\Delta_{best}$*$(\%)$ | $\Delta_{avg}(\%)$ | $t$(min) | $f_{best}$ | $\Delta_{best}(\%)$ | $\Delta_{avg}(\%)$ | $t$(min) | $f$ | $\Delta f(\%)$ |
| a1 | **1179.79** | 1203.39 | 2.00 | 2.67 | 4.58 | **1179.79** | 0.00 | 0.84 | 3.4 | **1179.79** | 0.00 | 1.51 | 1.82 | **1179.79** | 0.00 |
| b1 | **1217.07** | 1217.07 | 0.00 | 1.28 | 9.17 | **1217.07** | 0.00 | 0.66 | 7.8 | **1217.07** | 0.00 | 0.80 | 7.14 | **1217.07** | 0.00 |
| c1 | **1883.05** | 1888.22 | 0.27 | 0.53 | 36.22 | **1883.05** | 0.00 | 0.84 | 34.2 | 1897.3 | 0.76 | 2.24 | 33.93 | 1897.3 | 0.76 |
| d1 | **1059.43** | 1059.43 | 0.00 | 1.59 | 8.55 | **1059.43** | 0.00 | 0.46 | 5.9 | 1060.1 | 0.06 | 0.34 | 1.82 | **1059.43** | 0.00 |
| e1 | **1309.12** | **1309.12** | 0.00 | 0.19 | 13.52 | **1309.12** | 0.00 | 0.00 | 8.7 | **1309.12** | 0.00 | 2.66 | 7.29 | **1309.12** | 0.00 |
| f1 | **1572.17** | 1592.25 | 1.28 | 1.87 | 41.41 | **1572.17** | 0.00 | 0.87 | 38.8 | 1584.06 | 0.76 | 3.03 | 34.61 | 1575.57 | 0.22 |
| g1 | **1181.13** | 1190.93 | 0.83 | 1.77 | 55.22 | **1181.13** | 0.00 | 0.77 | 5.8 | 1181.99 | 0.07 | 0.81 | 4.21 | **1181.13** | 0.00 |
| h1 | **1547.25** | 1566.75 | 1.26 | 3.31 | 32.07 | **1547.24** | 0.00 | 1.96 | 11.1 | 1566.19 | 1.22 | 2.27 | 18.03 | 1562.56 | 0.99 |
| i1 | **1925.99** | 1945.73 | 1.02 | 2.60 | 51.01 | **1925.99** | 0.00 | 1.57 | 42.5 | 1953.39 | 1.42 | 4.07 | 45.62 | 1933.05 | 0.37 |
| j1 | 1117.20 | 1144.41 | 2.44 | 3.99 | 58.90 | 1117.20 | 0.00 | 1.04 | 5.5 | **1115.78** | -0.13 | 0.31 | 4.24 | **1115.78** | -0.13 |
| k1 | **1580.39** | 1586.92 | 0.41 | 2.41 | 64.61 | **1580.39** | 0.00 | 0.72 | 12.1 | 1586.64 | 0.40 | 1.35 | 18.11 | 1580.92 | 0.03 |
| l1 | **1880.60** | 1897.74 | 0.91 | 1.94 | 104.27 | **1880.60** | 0.00 | 1.27 | 51.4 | 1902.72 | 1.18 | 2.78 | 46.14 | 1894.05 | 0.72 |
| **Avg.** | | | 0.87 | 2.01 | 39.96 | | 0.00 | 0.92 | 18.92 | | 0.48 | 1.85 | 18.58 | | |
| a2 | **997.94** | 1000.24 | 0.23 | 0.72 | 6.4 | | | | | **997.94** | 0.00 | 0.47 | 1.8 | **997.94** | 0.00 |
| b2 | 1307.28 | 1307.28 | 0.00 | 1.98 | 14.7 | | | | | 1301.21 | -0.46 | 1.34 | 7.35 | **1291.19** | -1.23 |
| c2 | 1747.61 | 1751.45 | 0.22 | 2.57 | 61.7 | | | | | 1732.19 | -0.88 | 0.76 | 18.05 | **1719.47** | -1.61 |
| d2 | 1871.42 | 1877.03 | 0.30 | 1.43 | 40.5 | | | | | 1892.62 | 1.13 | 1.97 | 35.1 | **1866.97** | -0.24 |
| e2 | 1942.85 | 1974.13 | 1.61 | 2.72 | 73.8 | | | | | 1940.52 | -0.12 | 2.61 | 59.12 | **1928.06** | -0.76 |
| f2 | 2284.35 | 2298.51 | 0.62 | 1.22 | 162.2 | | | | | 2292.4 | 0.35 | 1.79 | 89.86 | **2275.28** | -0.40 |
| g2 | 1162.58 | 1162.58 | 0.00 | 2.01 | 29.5 | | | | | 1158.21 | -0.38 | -0.09 | 4.14 | **1152.92** | -0.83 |
| h2 | 1587.37 | 1593.40 | 0.38 | 1.54 | 160.8 | | | | | 1597.41 | 0.63 | 1.46 | 18.35 | **1580.55** | -0.43 |
| i2 | 1972.00 | 1978.70 | 0.34 | 1.33 | 322.4 | | | | | 1934.09 | -1.92 | -0.10 | 47.58 | **1925.52** | -2.36 |
| j2 | 2294.06 | 2303.01 | 0.39 | 1.36 | 256.9 | | | | | 2293.4 | -0.03 | 1.58 | 91.3 | **2276.52** | -0.76 |
| **Avg.** | | | 0.41 | 1.69 | 112.89 | | | | | | -0.17 | 1.18 | 37.27 | | |
| **Tot. Avg.** | | | 0.66 | 1.86 | 73.11 | | | | | | 0.18 | 1.54 | 27.07 | | |

* Note that this value corresponds to the best solution ever obtained with the final parameter setting

**Table 5.5:** Comparison of the solutions obtained by VNS/TS on the MDVRPI instances, proposed by Crevier et al. (2007), to those of TZK and CCL. BKS denotes the previously best known solution. Gaps are calculated in dependence of BKS. Additionally, we provide the best solutions in $\overline{\text{VNS/TS}}$ that we ever obtained on the instances during our testing activities. $t(min)$ denotes the average run-time for each run.

| | | **TZK** | | | | **VNS/TS** | | | | $\overline{\textbf{VNS/TS}}$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Instance | BKS | $f_{best}$* | $\Delta_{best}(\%)$* | $\Delta_{avg}(\%)$ | $t(min)$ | $f_{best}$ | $\Delta_{best}(\%)$ | $\Delta_{avg}(\%)$ | $t(min)$ | $f_{best}$ | $\Delta_{best}(\%)$ |
| 50c3d2v | **2209.83** | **2209.83** | 0.00 | 2.27 | 2.85 | **2209.83** | 0.00 | 0.10 | 1.82 | **2209.83** | 0.00 |
| 50c3d4v | **2368.33** | **2368.33** | 0.00 | 2.18 | 2.23 | **2368.33** | 0.00 | 0.89 | 1.84 | **2368.33** | 0.00 |
| 50c3d6v | 3000.88 | 3000.88 | 0.00 | 2.13 | 2.74 | **2999.29** | -0.05 | 0.75 | 1.88 | **2999.29** | -0.05 |
| 50c5d2v | **2608.25** | **2608.25** | 0.00 | 2.87 | 1.54 | **2608.25** | 0.00 | 1.01 | 2 | **2608.25** | 0.00 |
| 50c5d4v | **3086.58** | **3086.58** | 0.00 | 1.23 | 2.07 | **3086.58** | 0.00 | 0.00 | 1.98 | **3086.58** | 0.00 |
| 50c5d6v | 3552 | 3552 | 0.00 | 0.90 | 3.04 | 3552 | 0.00 | 0.28 | 2.02 | **3548.88** | -0.09 |
| 50c7d2v | **3353.08** | **3353.08** | 0.00 | 2.37 | 3.16 | 3353.83 | 0.02 | 3.10 | 2.38 | **3353.08** | 0.00 |
| 50c7d4v | 3381.57 | 3381.57 | 0.00 | 2.63 | 3.36 | **3380.27** | -0.04 | 0.54 | 2.1 | **3380.27** | -0.04 |
| 50c7d6v | 4097.8 | 4097.8 | 0.00 | 0.25 | 3.42 | **4074.44** | -0.57 | 0.37 | 2.07 | **4074.44** | -0.57 |
| 75c3d2v | **2678.8** | **2678.8** | 0.00 | 0.57 | 4.5 | 2692.76 | 0.52 | 1.68 | 4.67 | **2678.8** | 0.00 |
| 75c3d4v | **2746.74** | **2746.74** | 0.00 | 1.95 | 3.38 | 2746.74 | 0.00 | 0.17 | 4.33 | **2746.74** | 0.00 |
| 75c3d6v | 3454.71 | 3454.71 | 0.00 | 1.30 | 4.89 | 3448.64 | -0.18 | 0.41 | 4.34 | **3404.34** | -1.46 |
| 75c5d2v | **3373.69** | **3373.69** | 0.00 | 2.98 | 3.29 | 3386.64 | 0.38 | 2.34 | 5.09 | **3373.69** | 0.00 |
| 75c5d4v | 3568.35 | 3568.35 | 0.00 | 2.43 | 3.54 | 3569.82 | 0.04 | 0.63 | 4.42 | **3553.46** | -0.42 |
| 75c5d6v | 4198.61 | 4198.61 | 0.00 | 1.66 | 4.18 | 4215.3 | 0.40 | 2.04 | 4.55 | **4193.86** | -0.11 |
| 75c7d2v | **3569.02** | **3569.02** | 0.00 | 2.41 | 5.38 | 3581.32 | 0.34 | 1.63 | 5.06 | **3569.02** | 0.00 |
| 75c7d4v | 3830.43 | 3830.43 | 0.00 | 2.13 | 5.51 | 3830.43 | 0.00 | 1.70 | 4.61 | **3825.37** | -0.13 |
| 75c7d6v | **4239.76** | **4239.76** | 0.00 | 2.02 | 4.29 | 4244.35 | 0.11 | 0.75 | 4.8 | 4242.08 | 0.05 |
| 100c3d3v | **3123.51** | **3123.51** | 0.00 | 1.10 | 7.01 | 3127.65 | 0.13 | 2.33 | 7.94 | 3126.55 | 0.10 |
| 100c3d5v | 3552.5 | 3552.5 | 0.00 | 2.37 | 7.31 | **3548.75** | -0.11 | 0.18 | 7.62 | **3548.44** | -0.11 |
| 100c3d7v | 4239.83 | 4239.83 | 0.00 | 0.83 | 6.62 | 4268.34 | 0.67 | 2.34 | 7.92 | **4239.5** | -0.01 |
| 100c5d3v | **4053.95** | **4053.95** | 0.00 | 1.06 | 7.88 | **4053.95** | 0.00 | 1.58 | 8.49 | **4053.95** | 0.00 |
| 100c5d5v | **4413.17** | **4413.17** | 0.00 | 2.69 | 7.2 | 4424.81 | 0.26 | 5.52 | 7.7 | 4415.48 | 0.05 |
| 100c5d7v | 5148.98 | 5148.98 | 0.00 | 0.56 | 7.72 | **5142.52** | -0.13 | 0.15 | 7.93 | **5142.52** | -0.13 |
| 100c7d3v | **4216.47** | **4216.47** | 0.00 | 0.61 | 8.53 | 4242.38 | 0.61 | 1.62 | 8.87 | **4216.47** | 0.00 |
| 100c7d5v | 4462.51 | 4462.51 | 0.00 | 1.36 | 8.79 | 4448.15 | -0.32 | 0.67 | 8 | **4439.72** | -0.51 |
| 100c7d7v | 4897.47 | 4897.47 | 0.00 | 1.55 | 8.35 | 4916.62 | 0.39 | 3.83 | 8.1 | **4869.66** | -0.57 |
| 125c4d3v | 3920.05 | 3920.05 | 0.00 | 1.18 | 8.73 | 3966.61 | 1.19 | 3.62 | 13.23 | **3916.02** | -0.10 |
| 125c4d5v | 4315.68 | 4315.68 | 0.00 | 1.30 | 9 | **4308.44** | -0.17 | 0.28 | 12.33 | **4308.44** | -0.17 |
| 125c4d7v | 4763.49 | 4763.49 | 0.00 | 1.48 | 8.4 | 4694.32 | -1.45 | 0.57 | 12.54 | **4668.77** | -1.99 |
| 125c6d3v | **4064.2** | **4064.2** | 0.00 | 0.78 | 9.19 | 4117.41 | 1.31 | 3.40 | 13.56 | 4076.04 | 0.29 |
| 125c6d5v | 4826.71 | 4826.71 | 0.00 | 2.70 | 8.33 | 4786.74 | -0.83 | 0.22 | 13.09 | **4765.97** | -1.26 |
| 125c6d7v | 5325.28 | 5325.28 | 0.00 | 2.65 | 9.18 | 5221.52 | -1.95 | -0.55 | 12.89 | **5164.18** | -3.03 |
| 125c8d3v | 4553.28 | 4553.28 | 0.00 | 2.73 | 10.23 | 4574.82 | 0.47 | 1.51 | 14.98 | **4545.44** | -0.17 |
| 125c8d5v | 5045.65 | 5045.65 | 0.00 | 1.38 | 9.64 | **4958.26** | -1.73 | 1.85 | 13.38 | **4958.26** | -1.73 |
| 125c8d7v | 5416.96 | 5416.96 | 0.00 | 0.63 | 9.34 | 5397.86 | -0.35 | 1.05 | 13.38 | **5347.1** | -1.29 |
| 150c4d3v | **4049.48** | **4049.48** | 0.00 | 0.01 | 9.71 | 4072.95 | 0.58 | 3.05 | 21.84 | 4069.72 | 0.50 |
| 150c4d5v | 4638.72 | 4638.72 | 0.00 | 1.44 | 8.19 | **4622.77** | -0.34 | 0.61 | 19.11 | **4622.77** | -0.34 |
| 150c4d7v | 5176.5 | 5176.5 | 0.00 | 1.30 | 8 | 5163.02 | -0.26 | 0.56 | 19.06 | **5137.69** | -0.75 |
| 150c6d3v | **4057.09** | **4057.09** | 0.00 | 0.15 | 9.96 | 4066.71 | 0.24 | 1.45 | 22.07 | 4062.53 | 0.13 |
| 150c6d5v | **4872.08** | **4872.08** | 0.00 | 0.54 | 10.23 | 4931.13 | 1.21 | 2.42 | 21.16 | 4876.91 | 0.10 |
| 150c6d7v | 5768.29 | 5768.29 | 0.00 | 2.58 | 10.73 | 5840.52 | 1.25 | 2.00 | 20.4 | **5712.01** | -0.98 |
| 150c8d3v | 4653.9 | 4653.9 | 0.00 | 1.79 | 10.18 | 4689.13 | 0.76 | 3.65 | 22.67 | 4667.5 | 0.29 |
| 150c8d5v | 5113.77 | 5113.77 | 0.00 | 1.10 | 11.62 | 5116.55 | 0.05 | 1.69 | 19.6 | **5073.8** | -0.78 |
| 150c8d7v | 5665.23 | 5665.23 | 0.00 | 0.00 | 12.01 | 5648.32 | -0.30 | 0.49 | 19.67 | **5612.02** | -0.94 |
| 175c4d4v | **4706.76** | **4706.76** | 0.00 | 1.60 | 21.74 | 4720.36 | 0.29 | 1.60 | 28.69 | 4708.66 | 0.04 |
| 175c4d6v | **4835.64** | **4835.64** | 0.00 | 2.58 | 23.01 | 4863.88 | 0.58 | 2.50 | 26.71 | 4841.51 | 0.12 |
| 175c4d8v | 5943.28 | 5943.28 | 0.00 | 1.53 | 18.4 | 5853.9 | -1.50 | -0.15 | 27.35 | **5832.26** | -1.87 |
| 175c6d4v | 5025.51 | 5025.51 | 0.00 | 1.64 | 21.51 | 5011.01 | -0.29 | 1.90 | 29.28 | **5020.01** | -0.11 |
| 175c6d6v | 5431.34 | 5431.34 | 0.00 | 0.11 | 22.54 | 5382.57 | -0.90 | 0.96 | 27.43 | **5360.35** | -1.31 |
| 175c6d8v | 6090.01 | 6090.01 | 0.00 | 1.27 | 25.81 | 6066.1 | -0.39 | 1.08 | 27.97 | **6043.43** | -0.76 |
| 175c8d4v | 5878.58 | 5878.58 | 0.00 | 2.59 | 24.9 | 5840.25 | -0.65 | 1.30 | 29.83 | **5822.55** | -0.95 |
| 175c8d6v | 5989.63 | 5989.63 | 0.00 | 2.80 | 25.21 | 5968.99 | -0.34 | 2.24 | 27.78 | **5953.54** | -0.60 |
| 175c8d8v | 6943.63 | 6943.63 | 0.00 | 1.90 | 26.7 | 6840.04 | -1.49 | 1.60 | 27.98 | **6775.68** | -2.42 |
| **Avg.** | | | 0.00 | 1.60 | 9.54 | | -0.05 | 1.44 | 12.79 | | -0.45 |

* Note that this value corresponds to the best solution ever obtained with the final parameter setting

**Table 5.6:** Comparison of the performance of our VNS/TS heuristic on the MDVRPI instances proposed by Tarantilis et al. (2008) with the solutions of TZK. BKS denotes the previously best known solution. Gaps are calculated in dependence of BKS. Additionally, we provide the best solutions in $\overline{\text{VNS/TS}}$ that we ever obtained on the instances during our testing activities. $t(min)$ denotes the average run-time for each run.

reached in time with at most one halt at an AFS for refueling. Thus, these customers have to be identified and removed in a preprocessing step. Erdogan and Miller-Hooks (2012) report solutions found by their two heuristics, a Modified Clarke and Wright Savings (MCWS) algorithm and a Density-Based Clustering Algorithm (DBCA), as well as solutions determined with the commercial solver CPLEX. The CPLEX solution is, however, not the optimal solution to the instance. In their mathematical formulation, they fixed the number of vehicles to the value obtained with the best heuristic in order to get solutions that are comparable, i.e., to determine the best solution with a given number of vehicles.

We compiled an improved version of their G-VRP model (see Appendix C) and use it to solve the set of small instances with CPLEX 12.2. Instead of the hierarchical objective considered before, we minimize traveled distance for all tests on the G-VRP instances. In Table 5.7, we report the best upper bound for the traveled distance found by CPLEX in at most 3 hours of computing time in column $f$. In four cases, CPLEX was not able to determine any feasible solution. Furthermore, we solved all instances 10 times by means of our VNS/TS heuristic and report the best traveled distance $f$ and the average computing time in minutes ($t$(min)).

We compare our results to those obtained with the MCWS and the DBCA heuristic, for which the best solution obtained in multiple runs is reported (no exact number of runs is given in the paper). Unfortunately, no computing times are available for these heuristics. In the table, we further report the gap of the best solution found by each of the heuristic methods to the CPLEX solution ($\Delta f$). Column $n$ provides the number of feasible customers and $m$ the number of vehicles required in the respective solutions.

Our VNS/TS heuristic clearly outperforms both heuristic methods proposed by Erdogan and Miller-Hooks (2012), which both achieve an average gap to the CPLEX solution of about 8%. On all instances, VNS/TS obtains the best solution found by CPLEX or even a solution that improves on the upper bound, resulting in an average gap of $-0.09\%$. It is also worth mentioning that the VNS/TS solutions reduce the number of vehicles in almost half of the instances, while requiring less than 40 seconds of computing time on average.

We additionally solve all large instances of the case study presented by Erdogan and Miller-Hooks (2012). In Table 5.8, we compare the results obtained with our

| | CPLEX | | | MCWS | | | | DBCA | | VNS/TS | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $m$ | $n$ | $f$ | $m$ | $n$ | $f$ | $\Delta f$(%) | $f$ | $\Delta f$(%) | $m$ | $n$ | $f$ | $t$(min) | $\Delta f$(%) |
| 20c3sU1 | **6** | 20 | **1797.49** | **6** | 20 | 1818.35 | 1.16 | 1797.51 | 0.00 | **6** | 20 | **1797.49** | 0.69 | 0.00 |
| 20c3sU2 | **6** | 20 | **1574.77** | **6** | 20 | 1614.15 | 2.50 | 1613.53 | 2.46 | **6** | 20 | **1574.77** | 0.64 | 0.00 |
| 20c3sU3 | **6** | 20 | **1704.48** | 7 | 20 | 1969.64 | 15.56 | 1964.57 | 15.26 | **6** | 20 | **1704.48** | 0.64 | 0.00 |
| 20c3sU4 | **5** | 20 | **1482** | 6 | 20 | 1508.41 | 1.78 | 1487.15 | 0.35 | **5** | 20 | **1482** | 0.65 | 0.00 |
| 20c3sU5 | 6 | 20 | **1689.37** | **5** | 20 | 1752.73 | 3.75 | 1752.73 | 3.75 | 6 | 20 | **1689.37** | 0.67 | 0.00 |
| 20c3sU6 | **6** | 20 | **1618.65** | **6** | 20 | 1668.16 | 3.06 | 1668.16 | 3.06 | **6** | 20 | **1618.65** | 0.67 | 0.00 |
| 20c3sU7 | **6** | 20 | **1713.66** | **6** | 20 | 1730.45 | 0.98 | 1730.45 | 0.98 | **6** | 20 | **1713.66** | 0.64 | 0.00 |
| 20c3sU8 | **6** | 20 | **1706.5** | **6** | 20 | 1718.67 | 0.71 | 1718.67 | 0.71 | **6** | 20 | **1706.5** | 0.67 | 0.00 |
| 20c3sU9 | **6** | 20 | **1708.81** | **6** | 20 | 1714.43 | 0.33 | 1714.43 | 0.33 | **6** | 20 | **1708.81** | 0.66 | 0.00 |
| 20c3sU10 | **4** | 20 | **1181.31** | 5 | 20 | 1309.52 | 10.85 | 1309.52 | 10.85 | **4** | 20 | **1181.31** | 0.64 | 0.00 |
| 20c3sC1 | **4** | 20 | **1173.57** | 5 | 20 | 1300.62 | 10.83 | 1300.62 | 10.83 | **4** | 20 | **1173.57** | 0.62 | 0.00 |
| 20c3sC2 | **5** | 19 | **1539.97** | **5** | 19 | 1553.53 | 0.88 | 1553.53 | 0.88 | 5 | 19 | **1539.97** | 0.58 | 0.00 |
| 20c3sC3 | **3** | 12 | **880.2** | 4 | 12 | 1083.12 | 23.05 | 1083.12 | 23.05 | **3** | 12 | **880.2** | 0.25 | 0.00 |
| 20c3sC4 | **4** | 18 | **1059.35** | 5 | 18 | 1135.9 | 7.23 | 1091.78 | 3.06 | **4** | 18 | **1059.35** | 0.53 | 0.00 |
| 20c3sC5 | 7 | 19 | - | **7** | 19 | 2190.68 | | 2190.68 | | 7 | 19 | **2156.01** | 0.6 | |
| 20c3sC6 | **8** | 17 | **2758.17** | 9 | 17 | 2883.71 | 4.55 | 2883.71 | 4.55 | **8** | 17 | **2758.17** | 0.71 | 0.00 |
| 20c3sC7 | **4** | 6 | **1393.99** | 5 | 6 | 1701.4 | 22.05 | 1701.4 | 22.05 | **4** | 6 | **1393.99** | 0.18 | 0.00 |
| 20c3sC8 | **9** | 18 | **3139.72** | 10 | 18 | 3319.74 | 5.73 | 3319.74 | 5.73 | **9** | 18 | **3139.72** | 0.62 | 0.00 |
| 20c3sC9 | **6** | 19 | **1799.94** | **6** | 19 | 1811.05 | 0.62 | 1811.05 | 0.62 | 6 | 19 | **1799.94** | 0.6 | 0.00 |
| 20c3sC10 | **8** | 15 | - | **8** | 15 | 2648.84 | | 2644.11 | | 8 | 15 | **2583.42** | 0.45 | |
| S1_2i6s | **6** | 20 | **1578.12** | **6** | 20 | 1614.15 | 2.28 | 1614.15 | 2.28 | 6 | 20 | **1578.12** | 0.71 | 0.00 |
| S1_4i6s | **5** | 20 | 1413.96 | **5** | 20 | 1561.3 | 10.42 | 1541.46 | 9.02 | 5 | 20 | **1397.27** | 0.75 | -1.18 |
| S1_6i6s | **5** | 20 | **1560.49** | 6 | 20 | 1616.2 | 3.57 | 1616.2 | 3.57 | 5 | 20 | **1560.49** | 0.73 | 0.00 |
| S1_8i6s | **6** | 20 | **1692.32** | **6** | 20 | 1902.51 | 12.42 | 1882.54 | 11.24 | 6 | 20 | **1692.32** | 0.74 | 0.00 |
| S1_10i6s | **4** | 20 | **1173.48** | 5 | 20 | 1309.52 | 11.59 | 1309.52 | 11.59 | **4** | 20 | **1173.48** | 0.71 | 0.00 |
| S2_2i6s | **6** | 20 | **1633.1** | **6** | 20 | 1645.8 | 0.78 | 1645.8 | 0.78 | 6 | 20 | **1633.1** | 0.75 | 0.00 |
| S2_4i6s | **5** | 19 | 1555.2 | 6 | 19 | **1505.06** | -3.22 | **1505.06** | -3.22 | 5 | 19 | 1532.96 | 0.88 | -1.43 |
| S2_6i6s | **7** | 20 | - | 10 | 20 | 3115.1 | | 3115.1 | | **7** | 20 | **2431.33** | 0.78 | |
| S2_8i6s | **7** | 16 | **2158.35** | 9 | 16 | 2722.55 | 26.14 | 2722.55 | 26.14 | **7** | 16 | **2158.35** | 0.57 | 0.00 |
| S2_10i6s | **6** | *17* | - | **6** | 16 | 1995.62 | | 1995.62 | | 6 | *17* | **1958.46** | 0.61 | |
| S1_4i2s | 6 | 20 | **1582.21** | **6** | 20 | **1582.2** | 0.00 | **1582.2** | 0.00 | 6 | 20 | **1582.21** | 0.63 | 0.00 |
| S1_4i4s | **5** | 20 | **1460.09** | 6 | 20 | 1580.52 | 8.25 | 1580.52 | 8.25 | **5** | 20 | **1460.09** | 0.68 | 0.00 |
| S1_4i6s | 5 | 20 | **1397.27** | **5** | 20 | 1561.29 | 11.74 | 1541.46 | 10.32 | 5 | 20 | **1397.27** | 0.75 | 0.00 |
| S1_4i8s | 6 | 20 | 1403.57 | **6** | 20 | 1561.29 | 11.24 | 1561.29 | 11.24 | 6 | 20 | **1397.27** | 0.82 | -0.45 |
| S1_4i10s | 5 | 20 | 1397.27 | **5** | 20 | 1536.04 | 9.93 | 1529.73 | 9.48 | 5 | 20 | **1396.02** | 0.85 | -0.09 |
| S2_4i2s | **4** | 18 | **1059.35** | 5 | 18 | 1135.89 | 7.23 | 1117.32 | 5.47 | **4** | 18 | **1059.35** | 0.51 | 0.00 |
| S2_4i4s | **5** | 19 | **1446.08** | 6 | 19 | 1522.72 | 5.30 | 1522.72 | 5.30 | **5** | 19 | **1446.08** | 0.6 | 0.00 |
| S2_4i6s | **5** | 20 | **1434.14** | 6 | 20 | 1786.21 | 24.55 | 1730.47 | 20.66 | **5** | 20 | **1434.14** | 0.69 | 0.00 |
| S2_4i8s | **5** | 20 | **1434.14** | 6 | 20 | 1786.21 | 24.55 | 1786.21 | 24.55 | **5** | 20 | **1434.14** | 0.75 | 0.00 |
| S2_4i10s | **5** | 20 | **1434.13** | 6 | 20 | 1783.63 | 24.37 | 1729.51 | 20.60 | **5** | 20 | **1434.13** | 0.78 | 0.00 |
| **Avg.** | 5.58 | 18.8 | 1575.98 | 6.13 | 18.78 | 1781.42 | 8.52 | 1774.15 | 7.94 | 5.58 | 18.8 | 1645.45 | 0.65 | -0.09 |

**Table 5.7:** Results of VNS/TS on the small-sized G-VRP instances. Comparison of the solutions obtained by the MCWS and DBCA heuristics, the solutions determined by our CPLEX implementation and those of our VNS/TS. $f$ denotes the traveled distance of the best solution found by the respective method, and $\Delta f$ the gap to the CPLEX solution. $t$(min) reports the average computing time in minutes. We terminate CPLEX after 3 hours, so optimality is not guaranteed for any of the reported CPLEX solutions. Numbers in bold indicate the best solution found. Note that in one case, our preprocessing identified a higher number of feasible customers (numbers in italic) than Erdogan and Miller-Hooks (2012).

heuristic to those of MCWS and DBCA. As VNS/TS provides the best solution for all instances, the value given in column $\Delta f$ denotes the gap of to best solution found by VNS/TS in 10 runs. The results obtained by our heuristic on the large instance set is quite impressive. We reduce the traveled distance of the solutions of MCWS and DBCA by almost 15%. In addition, we require significantly fewer vehicles on average.

| | MCWS | | | | DBCA | | VNS/TS | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | $m$ | $n$ | $f$ | $\Delta f(\%)$ | $f$ | $\Delta f(\%)$ | $m$ | $n$ | $f$ | $t$(min) |
| 111c_21 | 20 | 109 | 5626.64 | 17.29 | 5626.64 | 17.29 | **17** | 109 | **4797.15** | 21.76 |
| 111c_22 | 20 | 109 | 5610.57 | 16.83 | | | **17** | 109 | **4802.16** | 23.56 |
| 111c_24 | 20 | 109 | 5412.48 | 13.07 | | | **17** | 109 | **4786.96** | 21.9 |
| 111c_26 | 20 | 109 | 5408.38 | 13.18 | | | **17** | 109 | **4778.62** | 25.12 |
| 111c_28 | 20 | 109 | 5331.93 | 11.10 | | | **17** | 109 | **4799.15** | 24.17 |
| 200c | 35 | 190 | 10428.59 | 16.35 | 10413.59 | 16.18 | **35** | *192* | **8963.46** | 76.65 |
| 250c | 41 | 235 | 11886.61 | 10.06 | 11886.61 | 10.06 | **39** | *237* | **10800.18** | 120.9 |
| 300c | 49 | 281 | 14242.56 | 13.08 | 14229.92 | 12.98 | **46** | *283* | **12594.77** | 182.23 |
| 350c | 57 | 329 | 16471.10 | 15.00 | 16460.30 | 14.92 | **51** | 329 | **14323.02** | 232.03 |
| 400c | 67 | 378 | 19472.10 | 15.56 | 19099.04 | 13.35 | **61** | 378 | **16850.21** | 305.12 |
| 450c | 75 | 424 | 21854.17 | 18.00 | 21854.19 | 18.00 | **68** | 424 | **18521.23** | 525.52 |
| 500c | 84 | 471 | 24527.46 | 15.85 | 24517.08 | 15.81 | **76** | 471 | **21170.9** | 356.01 |
| Average | 42.33 | 237.75 | 12189.38 | 14.61 | 15510.92 | 14.82 | 38.42 | 238.25 | 10598.98 | 159.58 |

**Table 5.8:** Results on the large-scale G-VRP instances. Comparison of the solutions obtained by the MCWS and DBCA heuristics and those of our VNS/TS. For VNS/TS, $f$ denotes the best solution found in 10 runs and $t$(min) reports the average computing time in minutes. Numbers in bold indicate the best solution found. As these are all provided by VNS/TS, we give the percentage gap to the VNS/TS solution for MCWS and DBCA in column $\Delta f$. Note that in some cases, our preprocessing identified a higher number of feasible customers (numbers in italic) than Erdogan and Miller-Hooks (2012).

To conclude, although our approach is not specifically tailored to the MDVRPI or G-VRP, we are able to outperform the state-of-the-art heuristics on the G-VRP and the second MDVRPI benchmark set. On the first MDVRPI set, we obtain competitive results while requiring moderate computing times.

## 5.4 Summary and Conclusion

In this chapter, we present a new vehicle routing problem for determining cost-optimal routes for electric vehicles. The E-VRPTW considers a limited vehicle and battery capacity and traveling along arcs consumes battery charge according to a constant consumption factor. Vehicles have the possibility of visiting

recharging stations along the route. The recharging time depends on the current battery charge on arrival at the station. Furthermore, customer time windows are incorporated into the E-VRPTW model in order to represent real-world requirements.

We develop a hybrid VNS/TS heuristic, which makes use of the strong diversification effect of VNS and involves a TS heuristic to efficiently search the solution space from a randomly generated solution of the VNS component. Furthermore, we increase the diversification abilities of our method by implementing an acceptance criterion based on the Metropolis probability. In numerical studies performed on newly designed E-VRPTW benchmark instances, we demonstrate the strong performance of our metaheuristic and the positive effect of combining the two metaheuristics VNS and TS. Moreover, we solve benchmark instances of the related problems MDVRPI and G-VRP. Although our VNS/TS algorithm is not specifically tailored to solve those problems, it outperforms all competing algorithms on both G-VRP instance sets as well as on the large MDVRPI instance set. It is also worth mentioning that we found new best solutions for a large number of benchmark instances available for MDVRPI and G-VRP.

The content of this chapter is available in similar form as technical report (Schneider, Stenger and Goeke 2012) and has been submitted to an international journal.

# Chapter 6

# Summary, Conclusion and Outlook

This thesis investigates new problems faced by small package shipping (SPS) companies with time definite service requirements from an operations research (OR) perspective. We develop OR models and solution methods to study the performance of a fixed-area-based routing approach (FABRA) under time window constraints and to provide decision support for route planning operations that have to consider 1) available driver knowledge and 2) the specifics of last-mile delivery by means of electric vehicles. This chapter summarizes the contents of this thesis, lists its major contributions and discusses possibilities for future research.

## 6.1 Summary

In Chapter 2, we introduce the Vehicle Routing Problem with Time Windows (VRPTW), which provides the foundation for all other problems addressed in this work. We present an overview of the most successful metaheuristics for the VRPTW and detail Tabu Search (TS), Variable Neighborhood Search (VNS) and Simulated Annealing (SA) as the main components of our solution methods. The Solomon VRPTW benchmark is introduced and the characteristics of the different problem groups are explained.

Subsequently, we discuss how changes in capacity violations can be calculated in constant time by means of capacity slack variables if the conventional neighborhood operators 2-opt*, Or-Opt, Relocate or Exchange are used. Moreover, we present a new approach proposed by Nagata et al. (2010), which allows the calculation of time window violations in constant time for inter-route moves with the aforementioned operators. Finally, the relevant literature for the integration of driver learning aspects and the employment of electric vehicles into route planning operations is presented.

In Chapter 3, we develop Semi-Fixed Service Territory Routing (SFSTR), a two-phase FABRA that is able to handle customer time windows. In the districting phase, service territories are constructed based on the TS solutions obtained for a series of VRPTW sample days that represent historical demand data. The characteristics of the solutions and spatial aspects are used to select a set of seed customers and to determine further customers who are iteratively added to the seed customers until territories of the desired size are created. In the subsequent routing phase, the TS conducts the daily routing based on the generated territories.

We develop a set of benchmark instances that comprise 100 days of VRPTW problems with customers selected from a common base set, for which we used a selection of the 1000-customer Gehring and Homberger instances. To assess the performance of SFSTR, we analyze efficiency measures (number of routes, traveled distance and number of outsourced customers) and consistency measures (customer familiarity and driver diversity) compared to a route reoptimization (RR) strategy. We further investigate the effect of different territory sizes and different variabilities in the number of customers requiring service on each day.

In Chapter 4, we study the Vehicle Routing Problem with Time Windows and Driver-Specific Times (VRPTWDST), a routing model to provide decision support for delivery operations that consider different levels of driver knowledge by means of driver-specific travel and service times. As solution method, we present a TS, called TS-DST, which is able to yield high-quality solutions in fast time. It generates an initial solution with a modified Solomon I1 heuristic and uses moves generated by the operators 2-opt*, Exchange and Relocate to improve the solution. Infeasible solutions are allowed during the search and are evaluated based on a penalizing cost function. The time window handling approach of Nagata et al. (2010) is adapted to the problem by defining separate time window handling related variables for each driver. Finally, we use a continuous diversification mechanism, probabilistic phases and sequences of random moves to provide a thorough exploration of the search space.

We generate a comprehensive set of VRPTWDST benchmark instances by means of the *Random* and *Cluster* distribution mechanisms, which determine the learned arcs and customers of each driver. These are subsequently assigned a learning factor representing the knowledge of the driver concerning the

arc/customer. TS-DST is applied to the generated instances and the influence of different distributions of learned customers and different magnitudes of the learning factors is investigated. Finally, TS-DST is run on the Solomon VRPTW benchmark and its performance in comparison to the best-performing VRPTW methods is assessed. Among the VRPTW methods achieving a cumulated number of vehicles (CNV) of 405, TS-DST ranks $9^{\text{th}}$ concerning solution quality and third concerning run-time. Moreover, it outperforms all previously proposed TS approaches for the VRPTW.

Chapter 5 introduces the Electric Vehicle Routing Problem with Time Windows and Recharging Stations (E-VRPTW). It integrates the possibility of en route recharging at one of the available stations with a recharging time that depends on the battery level on arrival at the station. To solve the problem, we develop a hybrid metaheuristic using 1) a VNS based on Cyclic-Exchange neighborhoods to diversify the search in a structured manner and 2) a TS method for the descent phase of the VNS. An SA-based acceptance decision is used to further diversify the search.

We generate a set of small and a set of medium-sized test instances based on the procedure used for generating the Solomon instances but extended by E-VRPTW specifics like recharging stations, battery capacities and recharging and consumption rates. The small instances are used to assess the quality of our VNS/TS in comparison to CPLEX solutions and the larger instances to evaluate the contribution of the heuristic components VNS, TS and SA. Moreover, we test VNS/TS on benchmark instances of the related problems Multi-Depot Vehicle Routing Problem with Inter-Depot Routes (MDVRPI) and Green Vehicle Routing Problem (G-VRP). On these instances, we are able to match or surpass the performance of solution methods specifically developed for the respective problem, and VNS/TS is able to provide a large number of new best solutions.

## 6.2 Conclusion

This work contributes by integrating 1) service consistency and driver knowledge aspects and 2) the utilization of electric vehicles into route planning models for small package shippers.

First, we develop a routing approach based on partially fixed territories that is capable of handling time-definite delivery requirements. We find that even with high variations of the number of customers requiring service, an unfavorable uniform distribution of customers and high time window densities, SFSTR shows only moderate efficiency forfeits while achieving significantly higher routing consistency compared to the RR strategy. Moreover, an increase of the customer percentage fixedly assigned to drivers leads to approximately proportional changes in the efficiency and consistency measures, which suggests that the size of the service territories can be utilized to control the achieved tradeoff between driver consistency and route efficiency.

Further results show that increasing the variance of the number of customers requiring service on each day does not have the detrimental effect that one could expect. Independent of the territory size, efficiency measures only deteriorate slightly compared to an RR approach while consistency measures remain stable on a high level. Such routing consistency is highly valuated by SPS companies. If the real-world possibility of subcontracting customers is taken into account, the problem of invalid vehicle routes obtained on some days can be solved without difficulties in practice.

Second, we are the first to carry out a systematic investigation of the VRPTWDST based on the generated benchmark instances, which cover different learning levels and customer arrangements. We find that consideration of driver knowledge in the route planning clearly increases the efficiency of vehicle routes, an effect that intensifies for higher learning levels. Moreover, increased learning benefits are produced if the customers that drivers are familiar with are geographically contiguous. The presented TS-DST shows how a simple metaheuristic method can be enhanced with diversification methods to accomplish a top ten rank among metaheuristics for the intensively studied VRPTW, competing with sophisticated approaches of which some might even be considered as "over-engineered" (Laporte 2009). TS-DST can consequently provide well-founded decision support for any logistics company whose routing operations have to take driver learning effects into account.

Third, we present the first routing model that integrates the specifics of electric vehicles and propose VNS/TS as solution method. In numerical tests, we show that each of the heuristic components VNS, TS and SA contributes to the

success of the metaheuristic hybrid. The comparison with CPLEX on small-sized E-VRPTW instances shows that our VNS/TS is capable of determining highly efficient vehicle routes making use of the available recharging stations. The quality of our method is further substantiated by the fact that we are able to compete or outperform specialized solution methods on benchmark instances of related problems. Thus, our method seems able to successfully assist routing decisions for electric vehicles employed in last-mile delivery operations.

Last, we contribute by identifying and correcting an error in the time travel approach for handling time window violations presented by Nagata et al. (2010). The error leads to a systematic undervaluation of time window violations and can thus even lead to situations where infeasible solutions are not identified as such. More than 50% of move evaluations in our TS test method cover cases for which the uncorrected formula yields wrong results. This has a significant negative impact on the solution quality of the TS method. We provide the corrected rule and show how the time travel approach can be successfully adapted to VRPTW variants like the VRPTWDST and the E-VRPTW.

## 6.3 Outlook

Several interesting topics for future research come into mind with regard to SFSTR. First, a more extensive evaluation of the impact of different time window characteristics like time window width and the geographic distribution of the customers with time-definite deliveries could provide further insights. Second, we assume that each customer has the same probability of requiring service on a given day in order to generate the worst case scenario for a FABRA. However, the performance of SFSTR shall also be investigated on benchmark instances with more realistic customer distributions, where a set of core customers have higher order volumes and occurrence frequencies than standard customers. Third, the application of SFSTR on real-life data of an SPS company and comparison to their route planning method would allow valuable conclusions.

TS-DST is a simple but high-quality method for generating feasible solutions with a given vehicle number. From an algorithmic viewpoint, it would be interesting to study if significant improvements of the solution quality can be achieved by integrating a VNS component as done for the E-VRPTW. Furthermore, we shall

investigate the suitability of using a series of VRPTWDST problems to promote driver learning and thus provide an alternative to FABRAs. This is achieved by linking day problems by means of a learning function that determines the driver-specific times for day $\tau + 1$ based on the times on day $\tau$ and the route performed on day $\tau$ (which is determined by TS-DST). First tests indicate that this approach could be successfully applied to obtain consistency benefits while maintaining routing flexibility in the presence of time windows.

In the E-VRPTW, lifting the constraint that batteries have to be completely recharged on visit of a recharging stations increases the route planning flexibility. Even a higher number of shorter recharging visits might sometimes be profitable in order to fulfill time window requirements. Therefore, this relaxation shall be integrated into the E-VRPTW model and our VNS/TS solution method. Moreover, we shall integrate the VRPTWDST and the E-VRPTW and adapt the VNS/TS to solve the resulting problem. Finally, future research should also consider decision support for planning the locations of recharging stations, e.g., by means of adapted location routing models and associated solution methods.

# Bibliography


Artmeier, A., Haselmayr, J., Leucker, M. and Sachenbacher, M. (2010). The shortest path problem revisited: Optimal routing for electric vehicles, *in* R. Dillmann, J. Beyerer, U. Hanebeck and T. Schultz (eds), *KI 2010: Advances in Artificial Intelligence*, Vol. 6359 of *Lecture Notes in Computer Science*, Springer, pp. 309–316.

Badeau, P., Guertin, F., Gendreau, M., Potvin, J.-Y. and Taillard, E. (1997). A parallel tabu search heuristic for the vehicle routing problem with time windows, *Transportation Research Part C: Emerging Technologies* **5**(2): 109–122.

Baldacci, R., Battarra, M. and Vigo, D. (2008). Routing a heterogeneous fleet of vehicles, *in* E. Wasil, S. Raghavan and B. L. Golden (eds), *The Vehicle Routing Problem: Latest Advances and New Challenges*, Vol. 43 of *Operations Research/Computer Science Interfaces*, Springer, pp. 3–27.

Baldacci, R., Mingozzi, A. and Roberti, R. (2012). Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints, *European Journal of Operational Research* **218**(1): 1–6.

Beasley, J. E. (1984). Fixed routes, *Journal of the Operational Research Society* **35**(1): 49–55.

Bent, R. and Van Hentenryck, P. (2004). A two-stage hybrid local search for the vehicle routing problem with time windows, *Transportation Science* **38**(4): 515–530.

Berger, J. and Barkaoui, M. (2004). A parallel hybrid genetic algorithm for the vehicle routing problem with time windows, *Computers & Operations Research* **31**(12): 2037–2053.

Blum, C. and Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison, *ACM Computing Surveys* **35**(3): 268–308.

Boostani, A., Ghodsi, R. and Miab, A. K. (2010). Optimal location of compressed natural gas (CNG) refueling station using the arc demand coverage model, *Proceedings of the 2010 Fourth Asia International Conference on Mathematical/Analytical Modelling and Computer Simulation*, IEEE Computer Society, pp. 193–198.

Bouthillier, A. and Crainic, T. (2005). A cooperative parallel meta-heuristic for the

vehicle routing problem with time windows, *Computers & Operations Research* **32**(7): 1685–1708.

Bräysy, O. (2003). A reactive variable neighborhood search for the vehicle routing problem with time windows, *INFORMS Journal on Computing* **15**(4): 347–368.

Bräysy, O. and Gendreau, M. (2002). Tabu search heuristics for the vehicle routing problem with time windows, *TOP: An Official Journal of the Spanish Society of Statistics and Operations Research* **10**(2): 211–237.

Bräysy, O. and Gendreau, M. (2005a). Vehicle routing problem with time windows, Part I: Route construction and local search algorithms, *Transportation Science* **39**(1): 104–118.

Bräysy, O. and Gendreau, M. (2005b). Vehicle routing problem with time windows, Part II: Metaheuristics, *Transportation Science* **39**(1): 119–139.

Campbell, A. M. and Thomas, B. W. (2008). Challenges and advances in a priori routing, *in* B. Golden, S. Raghavan and E. Wasil (eds), *The Vehicle Routing Problem: Latest Advances and New Challenges*, Vol. 43 of *Operations Research/Computer Science Interfaces Series*, Springer, pp. 123–142.

Campbell, A. M. and Thomas, B. W. (2009). Runtime reduction techniques for the probabilistic traveling salesman problem with deadlines, *Computers & Operations Research* **36**(4): 1231–1248.

Carlsson, J. G. (2011). Dividing a territory among several vehicles. Forthcoming in *INFORMS Journal on Computing*. doi:10.1287/ijoc.1110.0479.

Christofides, N. (1971). Fixed routes and areas for delivery operations, *International Journal of Physical Distribution & Logistics Management* **1**(2): 87–92.

Coelho, L. C., Cordeau, J.-F. and Laporte, G. (2011). Consistency in multi-vehicle inventory-routing, *Technical Report 2011-66*, CIRRELT, Canada.

Cordeau, J.-F., Desaulniers, G., Desrosiers, J., Solomon, M. and Soumis, F. (2002). VRP with time windows, *in* P. Toth and D. Vigo (eds), *The Vehicle Routing Problem*, Monographs on Discrete Mathematics and Applications, SIAM, pp. 157–193.

Cordeau, J.-F., Gendreau, M. and Laporte, G. (1997). A tabu search heuristic for periodic and multi-depot vehicle routing problems, *Networks* **30**(2): 105–119.

Cordeau, J.-F. and Laporte, G. (2005). Tabu search heuristics for the vehicle routing problem, *in* R. Sharda, S. Voß, C. Rego and B. Alidaee (eds), *Metaheuristic Optimization via Memory and Evolution*, Vol. 30 of *Operations Research/Computer Science Interfaces Series*, Springer, pp. 145–163.

Cordeau, J.-F., Laporte, G. and Mercier, A. (2001). A unified tabu search heuristic for vehicle routing problems with time windows, *The Journal of the Operational Research Society* **52**(8): 928–936.

Corne, D., Dorigo, M., Glover, F., Dasgupta, D., Moscato, P., Poli, R. and Price, K. V. (eds) (1999). *New Ideas in Optimization*, McGraw-Hill.

Crainic, T. G., Vidal, T., Gendreau, M., Lahrichi, N. and Rei, W. (2012). A hybrid genetic algorithm for multi-depot and periodic vehicle routing problems. Forthcoming in *Operations Research*.

Crevier, B., Cordeau, J.-F. and Laporte, G. (2007). The multi-depot vehicle routing problem with inter-depot routes, *European Journal of Operational Research* **176**(2): 756–773.

Czech, Z. J. and Czarnas, P. (2002). Parallel simulated annealing for the vehicle routing problem with time windows, *10th Euromicro Workshop on Parallel, Distributed and Network-Based Processing*, pp. 376–383.

Daganzo, C. and Erera, A. (1999). On planning and design of logistics systems for uncertain environments, *in* M. Speranza and P. Stahly (eds), *New Trends in Distribution Logistics*, Vol. 480 of *Lecture Notes in Economics and Mathematical Systems*, Springer, pp. 3–21.

Dantzig, G. B. and Ramser, J. H. (1959). The truck dispatching problem, *Management Science* **6**(1): 80–91.

Debudaj-Grabysz, A. and Czech, Z. (2005). A concurrent implementation of simulated annealing and its application to the VRPTW optimization problem, *in* Z. Juhász, P. Kacsuk and D. Kranzlmüller (eds), *Distributed and Parallel Systems*, Vol. 777 of *The Kluwer International Series in Engineering and Computer Science*, Springer, pp. 201–209.

Dondo, R. and Cerdá, J. (2006). An MILP framework for dynamic vehicle routing problems with time windows, *Latin American Applied Research* **36**: 255 – 261.

Dondo, R. and Cerdá, J. (2007). A cluster-based optimization approach for the multi-depot heterogeneous fleet vehicle routing problem with time windows, *European Journal of Operational Research* **176**(3): 1478 – 1507.

Dongarra, J. J. (2011). Performance of various computers using standard linear equations software, *Technical Report CS-89-85*, Electrical Engineering and Computer Science Department, University of Tennessee.

Erdogan, S. and Miller-Hooks, E. (2012). A green vehicle routing problem, *Transportation Research Part E: Logistics and Transportation Review* **48**(1): 100–114.

Erera, A. (2000). *Design of large-scale logistics systems for uncertain environments*, PhD thesis, University of California, Berkeley, USA.

Esser, K. and Kurte, J. (2011). Bundesverband Internationaler Express- und Kurierdienste e.V., KEP-STUDIE 2011 - Wirtschaftliche Bedeutung der Kurier-, Express- und Paketbranche.
**URL:** *http://www.biek.de/download/gutachten/kep_studie_2011.pdf*

European Commission (2011). White Paper - Roadmap to a Single European Transport Area – Towards a competitive and resource efficient transport system.
**URL:** *http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:52011DC0144:EN:NOT*

European Parliament and European Council (2011). Regulation (EU) No 510/2011 - setting emission performance standards for new light commercial vehicles as part of the Union's integrated approach to reduce $CO_2$ emissions from light-duty vehicles, *Official Journal of the European Union* **544**(145): 1–18.

Ferland, J. A. and Michelon, P. (1988). The vehicle scheduling problem with multiple vehicle types, *The Journal of the Operational Research Society* **39**(6): 577–583.

Figliozzi, M. A. (2010). An iterative route construction and improvement algorithm for the vehicle routing problem with soft time windows, *Transportation Research Part C: Emerging Technologies* **18**(5): 668 – 679.

Figliozzi, M. A., Mahmassani, H. S. and Jaillet, P. (2007). Pricing in dynamic vehicle routing problems, *Transportation Science* **41**(3): 302–318.

Francis, P., Smilowitz, K. and Tzur, M. (2007). Flexibility and complexity in periodic distribution problems, *Naval Research Logistics (NRL)* **54**(2): 136–150.

Gambardella, L. M., Taillard, E. D. and Agazzi, G. (1999). MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows, *in* D. Corne, M. Dorigo, F. Glover, D. Dasgupta, P. Moscato, R. Poli and K. V. Price (eds), *New Ideas in Optimization*, McGraw-Hill, pp. 63–76.

Garcia, B.-L., Potvin, J.-Y. and Rousseau, J.-M. (1994). A parallel implementation of the tabu search heuristic for vehicle routing problems with time window constraints, *Computers & Operations Research* **21**(9): 1025–1033.

Garey, M. and Johnson, D. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman and Co.

Gehring, H. and Homberger, J. (1999). A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows, *Proceedings of EUROGEN99*, pp. 57–64.

Gendreau, M. (2001). Recent advances in tabu search, *in* C. C. Ribeiro and P. Hansen (eds), *Essays and Surveys in Metaheuristics*, Vol. 15 of *Operations Research/Computer Science Interfaces Series*, Kluwer Academic Publishers, pp. 369–377.

Gendreau, M. (2003). An introduction to tabu search, *in* F. Glover and G. Kochenberger (eds), *Handbook of Metaheuristics*, Vol. 57 of *International Series in Operations Research & Management Science*, Springer, pp. 37–54.

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

Gendreau, M. and Potvin, J.-Y. (2005). Metaheuristics in combinatorial optimization, *Annals of Operations Research* **140**(1): 189–213.

Gendreau, M. and Potvin, J.-Y. (2010a). Tabu search, *in* M. Gendreau and J.-Y. Potvin (eds), *Handbook of Metaheuristics*, Vol. 146 of *International Series in Operations Research & Management Science*, Springer, pp. 41–59.

Gendreau, M., Potvin, J.-Y., Bräysy, O., Hasle, G. and Løkketangen, A. (2008). Metaheuristics for the vehicle routing problem and its extensions: A categorized bibliography, *in* B. Golden, S. Raghavan and E. Wasil (eds), *The Vehicle Routing Problem: Latest Advances and New Challenges*, Vol. 43 of *Operations Research/Computer Science Interfaces Series*, Springer, pp. 143–169.

Gendreau, M. and Potvin, J.-Y. (eds) (2010b). *Handbook of Metaheuristics*, Vol. 146 of *International Series in Operations Research & Management Science*, Springer.

Gendreau, M. and Tarantilis, C. D. (2010). Solving large-scale vehicle routing problems with time windows: The state-of-the-art, *Technical Report 2010-04*, CIRRELT, Canada.

Gillett, B. E. and Miller, L. R. (1974). A heuristic algorithm for the vehicle-dispatch problem, *Operations Research* **22**: 340–349.

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence, *Computers & Operations Research* **13**(5): 533–549.

Glover, F. (1989). Tabu search - Part I, *Informs Journal on Computing* **1**(3): 190–206.

Glover, F. (1990). Tabu search - Part II, *Informs Journal on Computing* **2**(1): 4–32.

Glover, F. and Kochenberger, G. A. (eds) (2003). *Handbook of Metaheuristics*, Vol. 57 of *International Series in Operations Research & Management Science*, Springer.

Glover, F. and Laguna, M. (1997). *Tabu Search*, Springer.

Glover, F. and Laguna, M. (2002). Tabu search, *in* P. M. Pardalos and M. G. C. Resende (eds), *Handbook of Applied Optimization*, Oxford University Press, pp. 194–208.

Gonçalves, F., Cardoso, S. R., Relvas, S. and Barbosa-Póvoa, A. P. F. D. (2011). Optimization of a distribution network using electric vehicles: A VRP problem, *IO2011 - 15° Congresso da associação Portuguesa de Investigação Operacional*.

Groër, C., Golden, B. and Wasil, E. (2009). The consistent vehicle routing problem, *Manufacturing & Service Operations Management* **11**(4): 630–643.

Gutin, G. and Punnen, A. (2002). *The Traveling Salesman Problem and Its Variations*, Kluwer Academic Publishers.

Haase, K. and Hoppe, M. (2008). Transportnetzgestaltung für Paketdienstleister, *Zeitschrift für Betriebswirtschaft* **78**: 857–874.

Hansen, P. and Mladenović, N. (1999). An introduction to variable neighborhood search, *in* S. Voss, I. H. Osman and C. Roucairol (eds), *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, Kluwer Academic Publishers, pp. 433–458.

Hansen, P. and Mladenović, N. (2001a). Developments of variable neighborhood search, *in* C. C. Ribeiro and P. Hansen (eds), *Essays and Surveys in Metaheuristics*, Vol. 15 of *Operations Research/Computer Science Interfaces Series*, Kluwer Academic Publishers, pp. 415–439.

Hansen, P. and Mladenović, N. (2001b). Variable neighborhood search: Principles and applications, *European Journal of Operational Research* **130**(3): 449–467.

Hansen, P. and Mladenović, N. (2002). Variable neighborhood search, *in* M. G. C. Resende and P. M. Pardalos (eds), *Handbook of Applied Optimization*, Oxford University Press, pp. 221–234.

Hansen, P. and Mladenović, N. (2003). Variable neighborhood search, *in* F. Glover and G. A. Kochenberger (eds), *Handbook of Metaheuristics*, Vol. 57 of *International Series in Operations Research & Management Science*, Springer, pp. 145–184.

Hansen, P., Mladenović, N., Brimberg, J. and Moreno Pérez, J. A. (2010). Variable neighborhood search, *in* M. Gendreau and J.-Y. Potvin (eds), *Handbook of Metaheuristics*, Vol. 146 of *International Series in Operations Research & Management Science*, Springer.

Hashimoto, H., Yagiura, M. and Ibaraki, T. (2008). An iterated local search algorithm for the time-dependent vehicle routing problem with time windows, *Discrete Optimization* **5**(2): 434–456.

Haughton, M. A. (2007). Assigning delivery routes to drivers under variable customer demands, *Transportation Research Part E: Logistics and Transportation Review* **43**(2): 157–172.

Haughton, M. A. (2008). The efficacy of exclusive territory assignments to delivery vehicle drivers, *European Journal of Operational Research* **184**(1): 24–38.

Haugland, D., Ho, S. C. and Laporte, G. (2007). Designing delivery districts for the vehicle routing problem with stochastic demands, *European Journal of Operational Research* **180**(3): 997–1010.

Hemmelmayr, V. C., Doerner, K. F. and Hartl, R. F. (2009). A variable neighborhood search heuristic for periodic routing problems, *European Journal of Operational Research* **195**(2): 791–802.

Homberger, J. (2000). *Verteilt-parallele Metaheuristiken zur Tourenplanung: Lösungsverfahren für das Standardproblem mit Zeitfensterrestriktionen*, Deutscher Universitäts-Verlag.

Homberger, J. and Gehring, H. (2005). A two-phase hybrid metaheuristic for the vehicle routing problem with time windows, *European Journal of Operational Research* **162**(1): 220–238.

Ibaraki, T., Imahori, S., Kubo, M., Masuda, T., Uno, T. and Yagiura, M. (2005). Effective local search algorithms for routing and scheduling problems with general time-window constraints, *Transportation Science* **39**(2): 206–232.

Ibaraki, T., Imahori, S., Nonobe, K., Sobue, K., Uno, T. and Yagiura, M. (2008). An iterated local search algorithm for the vehicle routing problem with convex time penalty functions, *Discrete Applied Mathematics* **156**(11): 2050–2069.

Irnich, S. (2008). A unified modeling and solution framework for vehicle routing and local search-based metaheuristics, *Informs Journal on Computing* **20**(2): 270–287.

Irnich, S., Funke, B. and Grünert, T. (2006). Sequential search and its application to vehicle-routing problems, *Computers & Operations Research* **33**(8): 2405–2429.

Kallehauge, B. (2008). Formulations and exact algorithms for the vehicle routing problem with time windows, *Computers & Operations Research* **35**(7): 2307–2330.

Kim, B.-I., Kim, S. and Sahoo, S. (2006). Waste collection vehicle routing problem with time windows, *Computers & Operations Research* **33**(12): 3624–3642.

Kindervater, G. and Savelsbergh, M. (1997). Vehicle routing: Handling edge exchanges, *in* E. Aarts and J. Lenstra (eds), *Local Search in Combinatorial Optimization*, John Wiley & Sons, pp. 337–360.

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

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

Lau, H., Sim, M. and Teo, K. (2003). Vehicle routing problem with time windows and a limited number of vehicles, *European Journal of Operational Research* **148**(3): 559–569.

Li, H. and Lim, A. (2003). Local search with annealing-like restarts to solve the VRPTW, *European Journal of Operational Research* **150**(1): 115–127.

Lim, A. and Zhang, X. (2007). A two-stage heuristic with ejection pools and generalized ejection chains for the vehicle routing problem with time windows, *INFORMS Journal on Computing* **19**(3): 443–457.

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

Malandraki, C., Zaret, D., Perez, J. R. and Holland, C. (2001). Industrial engineering applications in transportation, *Handbook of Industrial Engineering: Technology and Operations Management, Third Edition*, John Wiley & Sons, pp. 787–824.

Mehrez, A. and Stern, H. I. (1985). Optimal refueling strategies for a mixed-vehicle fleet, *Naval Research Logistics Quarterly* **32**(2): 315–328.

Melechovský, J., Prins, C. and Calvo, R. (2005). A metaheuristic to solve a location-routing problem with non-linear costs, *Journal of Heuristics* **11**(5-6): 375–391.

Melkman, A. A., Stern, H. I. and Mehrez, A. (1986). Optimal refueling sequence for a mixed fleet with limited refuelings, *Naval Research Logistics Quarterly* **33**(4): 759–762.

Mester, D. and Bräysy, O. (2005). Active guided evolution strategies for large-scale vehicle routing problems with time windows, *Computers & Operations Research* **32**(6): 1593–1614.

Mester, D., Bräysy, O. and Dullaert, W. (2007). A multi-parametric evolution strategies algorithm for vehicle routing problems, *Expert Systems with Applications* **32**(2): 508–517.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. and Teller, E. (1953). Equation of state calculations by fast computing machines, *The Journal of Chemical Physics* **21**(6): 1087–1092.

Mladenović, N. and Hansen, P. (1997). Variable neighborhood search, *Computers & Operations Research* **24**(11): 1097–1100.

Nagata, Y. (2007). Efficient evolutionary algorithm for the vehicle routing problem with time windows: Edge assembly crossover for the VRPTW, *IEEE Congress on Evolutionary Computation*, pp. 1175–1182.

Nagata, Y. and Bräysy, O. (2009). A powerful route minimization heuristic for the vehicle routing problem with time windows, *Operations Research Letters* **37**(5): 333–338.

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

Nikolaev, A. G. and Jacobson, S. H. (2010). Simulated annealing, *in* M. Gendreau and J.-Y. Potvin (eds), *Handbook of Metaheuristics*, Vol. 146 of *International Series in Operations Research & Management Science*, Springer, pp. 1–39.

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

Osman, I. H. and Laporte, G. (1996). Metaheuristics: A bibliography, *Annals of Operations Research* **63**: 511–623.

Ouyang, Y. (2007). Design of vehicle routing zones for large-scale distribution systems, *Transportation Research Part B* **41**(10): 1079–1093.

Pisinger, D. and Ropke, S. (2007). A general heuristic for vehicle routing problems, *Computers & Operations Research* **34**(8): 2403–2435.

Polacek, M., Hartl, R. F., Doerner, K. and Reimann, M. (2004). A variable neighborhood search for the multi depot vehicle routing problem with time windows, *Journal of Heuristics* **10**(6): 613–627.

Potvin, J.-Y. (1996). Genetic algorithms for the traveling salesman problem, *Annals of Operations Research* **63**: 339–370.

Potvin, J.-Y. and Rousseau, J.-M. (1995). An exchange heuristic for routeing problems with time windows, *Journal of the Operational Research Society* **46**(12): 1433–1446.

Prescott-Gagnon, E., Desaulniers, G. and Rousseau, L.-M. (2009). A branch-and-price-based large neighborhood search algorithm for the vehicle routing problem with time windows, *Networks* **54**(4): 190–204.

Psaraftis, H. N. (1983). k-interchange procedures for local search in a precedence-constrained routing problem, *European Journal of Operational Research* **13**(4): 391–402.

Qiu, Y., Liu, H., Wang, D. and Liu, X. (2011). Intelligent strategy on coordinated charging of PHEV with TOU price, *Asia-Pacific Power and Energy Engineering Conference (APPEEC)*, pp. 1–5.

Reeves, C. R. (ed.) (1993). *Modern heuristic techniques for combinatorial problems*, John Wiley & Sons.

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

Resende, M. G. C. and Pardalos, P. M. (eds) (2002). *Handbook of Applied Optimization*, Oxford University Press.

Ribeiro, C. C. and Hansen, P. (eds) (2001). *Essays and Surveys in Metaheuristics*, Vol. 15 of *Operations Research/Computer Science Interfaces Series*, Kluwer Academic Publishers.

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

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

Rousseau, L.-M., Gendreau, M. and Pesant, G. (2002). Using constraint-based operators to solve the vehicle routing problem with time windows, *Journal of Heuristics* **8**(1): 43–58.

Sand, B., Schneider, M., Wendt, O. and Schwind, M. (2011). Dezentrale Allokation von Transportaufträgen: Reduktion des „Price of Anarchy“ durch lernende Agenten, *in* H. Corsten and R. Gössinger (eds), *Dezentrale Koordination ökonomischer Aktivitäten*, Erich Schmidt Verlag, pp. 153–180.

Savelsbergh, M. (1985). Local search in routing problems with time windows, *Annals of Operations Research* **4**(1): 285–305.

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

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

Schneider, M., Doppstadt, C., Sand, B., Stenger, A. and Schwind, M. (2010). A vehicle routing problem with time windows and driver familiarity, *Proceedings of the Seventh Triennial Symposium on Transportation Analysis*, pp. 677–680.

Schneider, M., Doppstadt, C., Stenger, A. and Schwind, M. (2010). Ant colony optimization for a stochastic vehicle routing problem with driver learning, *IEEE Congress on Evolutionary Computation*, pp. 1–8.

Schneider, M., Sand, B. and Stenger, A. (2012). A note on the time travel approach

for handling time windows in vehicle routing problems. Working Paper, BISOR, University of Kaiserslautern, Germany.

Schneider, M., Stenger, A. and Goeke, D. (2012). The electric vehicle routing problem with time windows and recharging stations, *Technical Report 2/2012*, BISOR, University of Kaiserlautern, Germany.

Schneider, M., Stenger, A., Lagemann, H. and Vigo, D. (2012). On fixed-area-based vehicle routing in the presence of time window constraints, *Technical Report 1/2012*, BISOR, University of Kaiserlautern, Germany.

Schneider, M., Stenger, A., Schwahn, F. and Vigo, D. (2012). Sparsification methods for the vehicle routing problem with time windows. Working paper, University of Kaiserslautern, Germany.

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

Shaw, P. (1997). A new local search algorithm providing high quality solutions to vehicle routing problems. Working Paper, University of Strathclyde, Glasgow, Scotland.

**URL:** *http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.51.1273\&rep=rep1\&type=pdf*

Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems, *in* M. Maher and J.-F. Puget (eds), *Principles and Practice of Constraint Programming - CP98*, Vol. 1520 of *Lecture Notes in Computer Science*, Springer, pp. 417–431.

Smilowitz, K., Nowak, M. and Jiang, T. (2012). Workforce management in periodic delivery operations. Forthcoming in *Transportation Science*. doi:10.1287/trsc.1120.0407.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints, *Operations Research* **35**(2): 254–265.

Soriano, P. and Gendreau, M. (1996). Diversification strategies in tabu search algorithms for the maximum clique problem, *Annals of Operations Research* **63**: 189–207.

Stenger, A., Vigo, D., Enz, S. and Schwind, M. (2011). An adaptive variable neighborhood search algorithm for a vehicle routing problem arising in small package shipping. Forthcoming in *Transportation Science*. doi:10.1287/trsc.1110.0396.

Sungur, I., Ren, Y., Ordóñez, F., Dessouky, M. and Zhong, H. (2010). A model and

algorithm for the courier delivery problem with uncertainty, *Transportation Science* **44**(2): 193–205.

Taillard, É., Badeau, P., Gendreau, M., Guertin, F. and Potvin, J.-Y. (1997). A tabu search heuristic for the vehicle routing problem with soft time windows, *Transportation Science* **31**(2): 170–186.

Talbi, E.-G. (2009). *Metaheuristics: From design to implementation*, John Wiley & Sons.

Tan, K., Lee, L., Zhu, Q. and Ou, K. (2001). Heuristic methods for vehicle routing problem with time windows, *Artificial Intelligence in Engineering* **15**(3): 281 – 295.

Tarantilis, C. D., Zachariadis, E. E. and Kiranoudis, C. T. (2008). A hybrid guided local search for the vehicle-routing problem with intermediate replenishment facilities, *INFORMS Journal on Computing* **20**(1): 154–168.

Thompson, P. M. and Orlin, J. B. (1989). Theory of cyclic transfers, Working Paper, Operations Research Center, MIT, USA.

Toth, P. and Vigo, D. (2003). The granular tabu search and its application to the vehicle-routing problem, *INFORMS Journal on Computing* **15**(4): 333–346.

Toth, P. and Vigo, D. (eds) (2002). *The Vehicle Routing Problem*, Monographs on Discrete Mathematics and Applications, SIAM.

Voss, S., Osman, I. H. and Roucairol, C. (eds) (1999). *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, Kluwer Academic Publishers.

Wang, H. and Shen, J. (2007). Heuristic approaches for solving transit vehicle scheduling problem with route and fueling time constraints, *Applied Mathematics and Computation* **190**(2): 1237–1249.

Wang, Y.-W. and Lin, C.-C. (2009). Locating road-vehicle refueling stations, *Transportation Research Part E: Logistics and Transportation Review* **45**(5): 821–829.

Wang, Y.-W. and Wang, C.-R. (2010). Locating passenger vehicle refueling stations, *Transportation Research Part E: Logistics and Transportation Review* **46**(5): 791–801.

Wasner, M. and Zäpfel, G. (2004). An integrated multi-depot hub-location vehicle routing model for network planning of parcel service, *International Journal of Production Economics* **90**(3): 403–419.

Woch, M. and Łebkowski, P. (2009). Sequential simulated annealing for the vehicle routing problem with time windows, *Decision Making in Manufacturing and Services* **3**(1–2): 87–100.

Wong, K. F. and Beasley, J. E. (1984). Vehicle routing using fixed delivery areas, *Omega* **12**(6): 591–600.

Wong, R. T. (2008). Vehicle routing for small package delivery and pickup services, *in* B. Golden, S. Raghavan and E. Wasil (eds), *The Vehicle Routing Problem: Latest Advances and New Challenges*, Vol. 43 of *Operations Research/Computer Science Interfaces Series*, Springer, pp. 475–485.

Zhong, H., Hall, R. W. and Dessouky, M. (2007). Territory planning and vehicle dispatching with driver learning, *Transportation Science* **41**(1): 74–89.

# Appendix A

# Influence of the Incorrect Equation in the Time Travel Approach

We have identified two cases in which the formula proposed by Nagata et al. (2010) yields incorrect results in Section 2.2.2. However, it is not clear how often these cases occur and how strong the incorrect calculation affects a local-search-based heuristic designed for the Vehicle Routing Problem with Time Windows (VRPTW). To answer these questions, we perform numerical experiments with a Tabu Search (TS) heuristic on the Solomon VRPTW benchmark instances and thus study the impact of the incorrect time window handling on the solution process of a classical metaheuristic. Section A.1 gives a short description of the TS. In Section A.2, we present the results of our studies.

## A.1 Tabu Search Heuristic

The TS heuristic used in the numerical studies is detailed in Schneider, Stenger, Schwahn and Vigo (2012) and is very similar to the one described in Section 4.2. The parameter settings reported here were again found by pretests using the procedure described in Ropke and Pisinger (2006) (see Section 3.2.2). To generate an initial solution, we use the earliest ready time heuristic put forward in Lau et al. (2003). Relocate, Exchange and 2-opt* are used as neighborhood operators and are implemented using the generator arc technique introduced in Toth and Vigo (2003) (see Section 4.2.4). Infeasible solutions during the search are handled as described in Section 3.1.1. The penalty factors $\alpha$ and $\beta$ are initialized to $\alpha_0 = \beta_0 = 1$ and are scaled with $\delta = 1.2$ restricted to the interval $[1, 6400]$.

In each iteration, the best non-tabu move is performed. The tabu status is defined by forbidding to reinsert recently removed edges into the solution for $\vartheta$

iterations. The tabu tenure $\vartheta$ is randomly drawn from the interval $[20, 40]$. Finding a new best overall solution is used as aspiration criterion. As diversification techniques, we use the continuous diversification mechanism described in Section 4.2.5 using $\lambda_{div} = 1$ as diversification factor. To further diversify the search, we perform 100 random moves after every 1000 iterations without improving the best solution. If the best solution has not improved for 2500 iterations, we reset the search to the best solution found. Last, we select not the best move in each iteration but a random one among the 250 best moves if the amount of time window violation of the best solution identified in the last 100 iterations has not changed.

For the studies, we implemented the following three variants of the TS:

**Variant 1** The original variant as described above, using the corrected time window handling to determine time window penalties. This variant obtains quite competitive results on the Solomon benchmark instances. As commonly done in the VRPTW literature, the search is started with the best-known vehicle number for each instance (see Section 4.2.6). The search runs for a maximum of 50000 iterations in order to find a feasible solution with the given vehicle number. If a feasible solution is found, an additional 5000 iterations are spent on minimizing the distance.

Conducting 10 runs on each instance and using the best solution found in the runs, we achieved a cumulated number of vehicles (CNV) of 405 and a cumulated traveled distance (CTD) of 57702.62. This corresponds to a gap of 0% to the best-known CNV and a 0.91% gap to the best known CTD as reported in Section 2.1.3. The run-times on a desktop computer with an Intel Core 2 Quad Processor at 2.83 GHz using a 4 GB memory were 140 seconds on average.

**Variant 2** In Variant 2, time window penalties are determined based on the incorrect rule for determining time window penalties. This inaccurate solution assessment is used to rank the moves in each iteration and thus to select the best move. Furthermore, the overall best solution of the search is updated based on the incorrect evaluation. Only afterwards, a correct computation of the time window violation and the objective function value takes place.

**Variant 3** Variant 3 differs from Variant 2 in that the update of the overall best solution is already based on a correct evaluation of time window violations and objective function value. This is inspired by the fact that after carrying out a move, the newly generated routes have to be traversed to update the helper variables of the time travel approach and thus the update of the overall best solution is based on correct values if it is performed afterwards.

## A.2 Impact of the Incorrect Time Window Handling

Tests are performed on the 56 Solomon VRPTW instances. For all three TS variants, every instance is solved 10 times. The objective function commonly used for the VRPTW is hierarchical and the first goal is minimizing the number of vehicles. Therefore, to assess the impact of the incorrect time window handling on the solution quality, we compare the three TS variants concerning their ability to find a feasible solution with the number of vehicles that corresponds to the best-known solution of the respective instance. To this end, Variants 2 and 3 are started on each instance with the best-known vehicle number and are run for 50000 iterations with the goal of finding a feasible solution. As mentioned above, considering the best of 10 runs, Variant 1 is able to find a feasible solution for all Solomon instances and can therefore be used as 100% level.

Our first test assesses how often the two cases for which the original formula provides incorrect results occur when using a standard metaheuristic procedure like our TS. To this end, we measure the percentage of move evaluations for which the original formula provides incorrect results during all runs of Variant 2 on all Solomon instances. We provide overall averages and separate results for the relocate and the exchange operator. Moreover, we differentiate between two types of errors: Any deviation between the time window violation found by the original and the corrected formula results in a *general error*. A *feasibility error* denotes the situation where a solution is deemed feasible by the original formula although time windows are violated. Note that feasibility errors are included in the standard errors.

Table A.1 reports the results as averages over the 6 groups of Solomon instances. The percentage of moves in which the original formula produces a general error is given in column $\text{Err}_{gen}$ and the percentage of moves in which a

| | Total | | | | | Relocate | | Exchange | |
|---|---|---|---|---|---|---|---|---|---|
| Inst. | $\text{Err}_{gen}(\%)$ | $\text{Err}_{feas}(\%)$ | $\Delta_{avg}$ | $\Delta_{max}$ | $\Delta_{min}$ | $\text{Err}_{gen}(\%)$ | $\text{Err}_{feas}(\%)$ | $\text{Err}_{gen}(\%)$ | $\text{Err}_{feas}(\%)$ |
| C1 | 71.9 | 18.6 | 253.96 | 2099.84 | 0 | 55.7 | 15.2 | 73.7 | 19.1 |
| C2 | 64.6 | 12.0 | 851.07 | 6222.14 | 0 | 59.0 | 11.7 | 75.6 | 13.8 |
| R1 | 46.6 | 5.1 | 27.90 | 365.14 | 0 | 38.5 | 5.3 | 60.5 | 7.0 |
| R2 | 48.6 | 11.2 | 131.88 | 1657.56 | 0 | 38.4 | 10.9 | 58.5 | 11.3 |
| RC1 | 46.3 | 4.2 | 22.70 | 372.62 | 0 | 33.6 | 3.5 | 54.8 | 3.9 |
| RC2 | 50.7 | 10.5 | 135.42 | 1712.00 | 0 | 41.7 | 10.6 | 61.7 | 10.7 |
| **Avg.** | 54.8 | 10.3 | 237.15 | | | 44.5 | 9.5 | 64.1 | 11.0 |

**Table A.1:** Evaluation of the frequency and magnitude of the errors produced by the incorrect time window handling. The table reports the percentage of general errors, of feasibility errors and the average, maximal and minimal deviation between the original and the corrected formula in 10 runs of the TS. Results are given as averages over the instance groups, as total averages and separately for the relocate and exchange operator.

feasibility error occurs in $\text{Err}_{feas}$. Additionally, we provide the minimal ($\Delta_{min}$), maximal ($\Delta_{max}$) and average ($\Delta_{avg}$) deviation of the time window violations, measured as the correct value minus the result of the original formula.

The results confirm a high number of erroneous evaluations due to the incorrect formula. In total, the time window violations of almost 55% of all move evaluations are calculated incorrectly by the original formula. In more than 10% of evaluations, the formula even rates a move feasible although time windows are violated. Moreover, one can see that the deviation between corrected and original formula is not negligible. Note that the minimal deviation does not fall below 0 as the original formula consistently underestimates the real time window violation. As could be expected, the percentage of erroneous evaluations is higher for the exchange move as the incorrect rule is applied twice.

The second study investigates the effect of the incorrect time window handling on the search process and the obtained solution quality. To this end, we evaluate for Variants 2 and 3 the percentage of iterations in which the algorithm selects an incorrect move due to the wrong formula, i.e., not the best move as would have been selected with a correct time window handling. The values are reported in Table A.2 in column $\text{Err}_{move}$. Moreover, we give the percentage of instances in which the TS ends up with a feasible VRPTW solution in at least one of the 10 test runs (Feas.). For Variant 2, we additionally report the percentage of runs

| | Variant 2 | | | Variant 3 | |
|---|---|---|---|---|---|
| | Cons. Feas.(%) | Feas.(%) | $\text{Err}_{move}$(%) | Feas.(%) | $\text{Err}_{move}$(%) |
| C1 | 100.0 | 22.2 | 88.5 | 44.4 | 86.4 |
| C2 | 100.0 | 100.0 | 78.8 | 100.0 | 78.0 |
| R1 | 41.7 | 8.3 | 86.1 | 8.3 | 82.3 |
| R2 | 100.0 | 0.0 | 77.6 | 72.7 | 77.2 |
| RC1 | 50.0 | 0.0 | 83.6 | 25.0 | 78.6 |
| RC2 | 100.0 | 12.5 | 81.6 | 37.5 | 80.7 |
| Avg. | 81.9 | 23.8 | 82.7 | 48.0 | 80.7 |

**Table A.2:** Impact of incorrect time window handling on search process of the TS method. We report the percentage of iterations selecting the wrong move ($\text{Err}_{move}$) and the percentage of instances with a feasible solution in 10 runs (Feas.). Additionally, for Variant 2, the percentage of runs in which the solution found is considered feasible using the original, incorrect time window handling is given (Cons. Feas.).

in which the solution found is considered feasible using the original, incorrect time window handling (Cons. Feas.).

The results show that the algorithm performs another than the best local search move in the majority of iterations ($\approx$ 80%) due to the underestimated time window violation. This is true for all groups of instances and shows the significant impact of the wrong time window handling on the search procedure. Note that the values for Variants 2 and 3 are approximately equal since the two methods do not differ regarding the move selection.

When basing the feasibility of the obtained solution on the original time window handling, Variant 2 finds a seemingly feasible solution with the given vehicle number for 81.9% of the Solomon instances. Evaluating the obtained solutions with the corrected formula reveals that the solution found is actually only feasible for 23.8% of the instances. This value significantly differs among the groups of instances. Only for the C2 instances, Variant 2 is able to obtain valid results for all instances in at least one of the 10 runs per instance, which can be explained by the fact that time windows for this group are not very restrictive. For all other groups, the number of instances solved to feasibility is clearly below 25% and even zero or close to zero for R1, R2 and RC1. The deviation between the percentage of solutions considered feasible and the percentage actually being feasible is highest for the set R2. Here, a feasible solution is found for all

instances according to the original time window handling, although none of the solutions obtained in 10 runs for any of the instances is actually feasible.

Basing the update of the overall best solution on the corrected formula as done in Variant 3 significantly increases the percentage of feasible solutions. However, with only 48% of feasible solutions compared to a 100% feasibility obtained with Variant 1, this result further substantiates the strong negative influence of the incorrect time windows handling on the solution process and the results of a metaheuristic method like TS. Apart from the fact that wrong penalties are added to solutions and thus the selection of the best move in each iteration is erroneous, the incorrect time window handling influences the penalty factor update mechanism. It may occur that penalty factors are reduced although time windows are violated and thus the search is further misguided.

Summing up, we have presented a correct formula to implement the time travel approach of Nagata et al. (2010) in Section 2.2.2. We have shown that the cases for which the originally proposed formula produces wrong results cover more than 50% of move evaluations in our TS metaheuristic. Moreover, our experiments demonstrate that the incorrect time window handling has a significant negative influence on the solution procedure and quality of a metaheuristic method like TS.

# Appendix B

# Additional Computational Results for the VRPTWDST

## B.1 Effect of Preprocessing on Solomon Instances

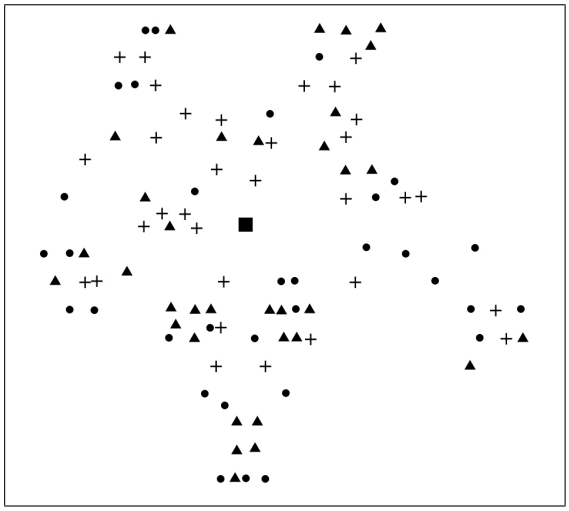
We show the results of applying the preprocessing steps introduced in Section 4.2.1 on the Solomon VRPTW set. For each instance, Table B.1 provides the percentage of arcs that could be removed due to the rules, which lies between 0.09% (9 arcs in absolute numbers) for instance RC208 and 67.99% (6867 arcs) for instance R101. If considered in combination with the characteristics of the Solomon instances as shown in Table 2.1, the results show the strong influence of the time window violation related preprocessing rules. The percentage of infeasible arcs increases significantly for instances with a higher time window density (TWD), namely it is on average 33.55% for instances with 100% TWD compared to 30.21% for 75% TWD, 15.08% for 50% TWD and 4.84% for 25% TWD. Moreover, it can be noted that the percentage of infeasible arcs decreases with a higher time window width.

| | C1 | C2 | R1 | R2 | RC1 | RC2 |
|---|---|---|---|---|---|---|
| 1 | 55.33 | 48.32 | **67.99** | 41.42 | 63.99 | 41.45 |
| 2 | 33.84 | 27.11 | 42.56 | 23.58 | 40.64 | 23.22 |
| 3 | 15.62 | 11.95 | 23.77 | 10.40 | 22.62 | 10.31 |
| 4 | 5.07 | 2.92 | 9.08 | 2.52 | 8.23 | 2.44 |
| 5 | 50.00 | 43.58 | 57.89 | 27.47 | 48.15 | 28.99 |
| 6 | 46.35 | 38.50 | 35.11 | 15.64 | 46.76 | 27.08 |
| 7 | 44.39 | 34.87 | 18.80 | 7.16 | 25.40 | 14.67 |
| 8 | 37.22 | 34.02 | 6.87 | 1.62 | 7.87 | **0.09** |
| 9 | 26.20 | | 40.36 | 15.68 | | |
| 10 | | | 21.19 | 15.20 | | |
| 11 | | | 21.25 | 1.03 | | |
| 12 | | | 0.88 | | | |

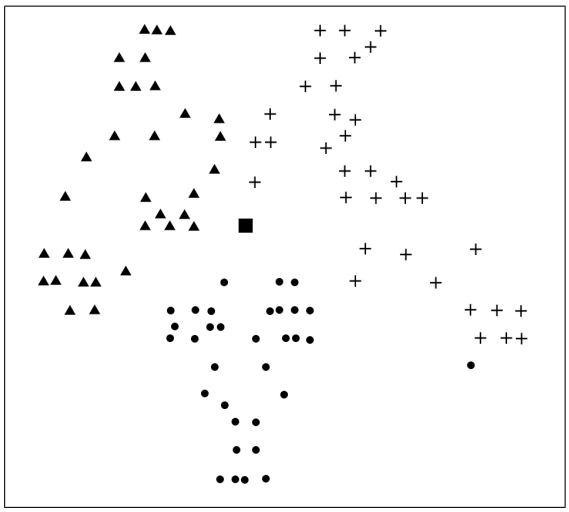
**Table B.1:** Percentage of arcs removed by the preprocessing step described in Section 4.2.1 on the Solomon VRPTW instances

## B.2 Visualization of the Distribution Mechanisms *Cluster* and *Random*

We present the results of applying the two distribution mechanisms *Cluster* and *Random* described in Section 2.1.3 on three Solomon instances selected to cover the geographical distributions C2, R and RC in Figures B.1 - B.2 (C1 is covered in Section 4.3.2). To depict the influence of different scheduling horizons, we choose instances R202 and RC101.

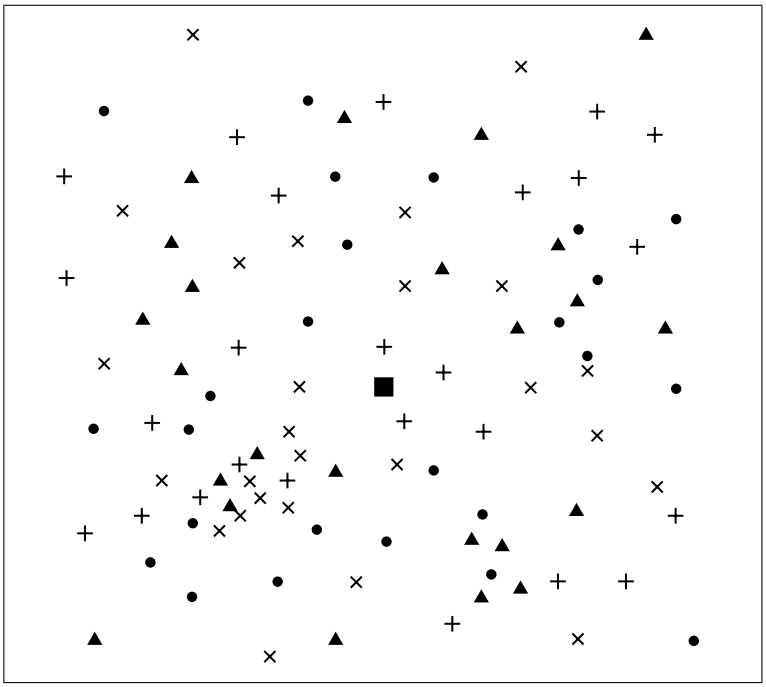


(a) *Random* distribution

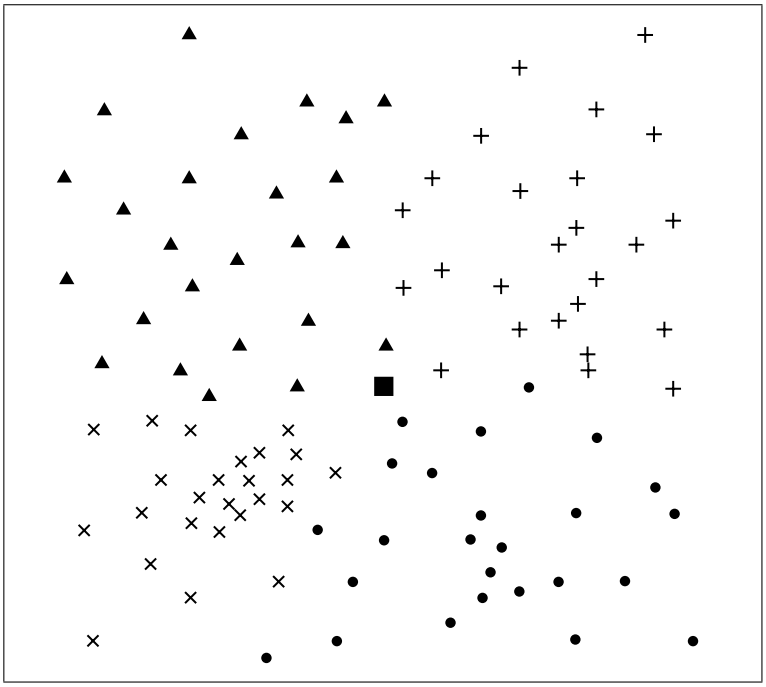


(b) *Cluster* distribution

**Figure B.1:** Distribution of learned customers for Solomon instance C203 using the *Random* and *Cluster* mechanism

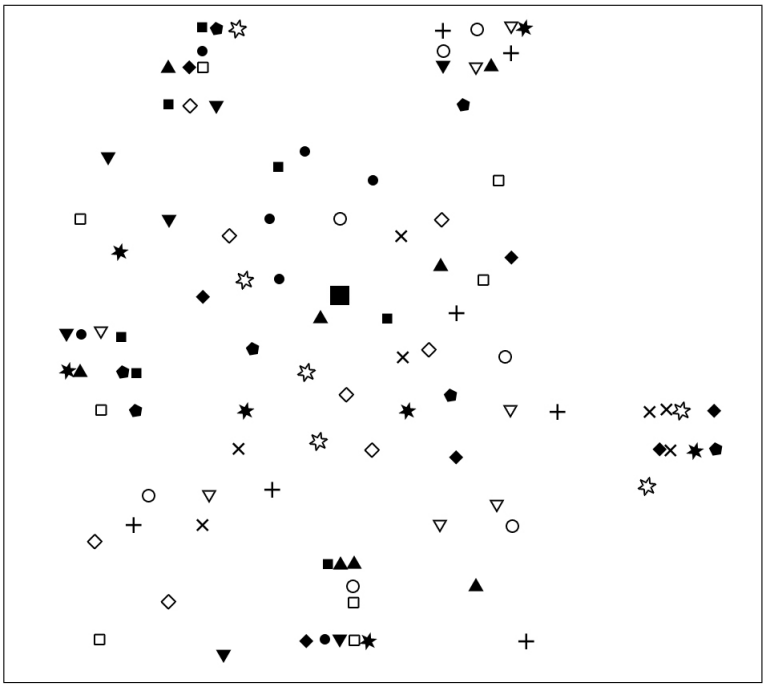


(a) *Random* distribution for R201

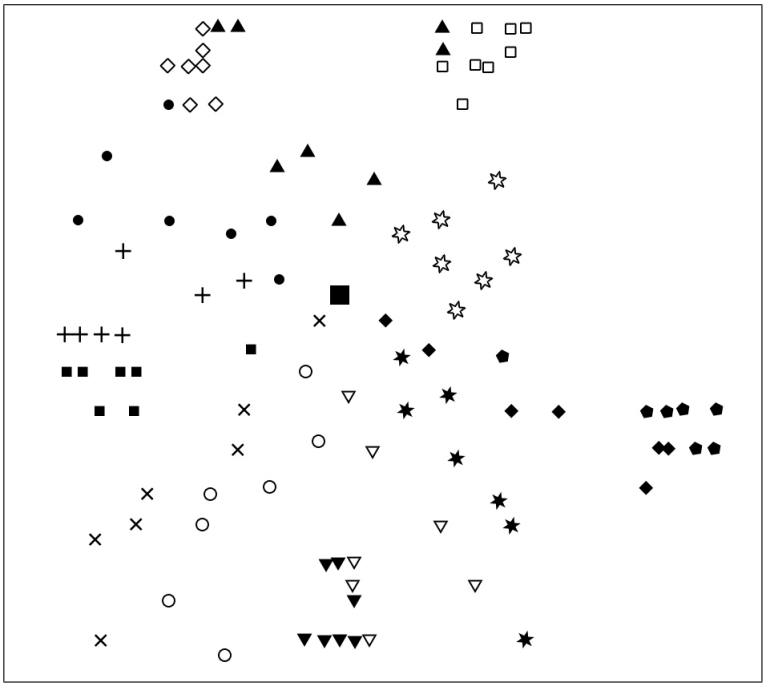


(b) *Cluster* distribution for R201

**Figure B.2:** Distribution of learned customers for Solomon instance R201 using the *Random* and *Cluster* mechanism



(a) *Random* distribution



(b) *Cluster* distribution

**Figure B.3:** Distribution of learned customers for Solomon instance RC101 using the *Random* and *Cluster* mechanism

## B.3 Detailed Results of TS-DST on Solomon Instances

In Table B.2, we present the results and run-times of TS-DST applied to the Solomon VRPTW benchmark on an instance basis. TS-DST is run 10 times for each problem instance, starting from the best-known vehicle number reported in the literature for the respective instance as described in Section 4.2.6. We report the best solution found in the 10 runs, the average solution and the average run-time. The solution quality is given in terms of vehicle number (#Rts.) and traveled distance (TD). Moreover, we follow the common procedure and give averages over the problem classes C, R and RC and the CNV and CTD. For comparison reasons, the best-known solution (BKS) for each instance is reported. If the BKS is reached by TS-DST, the respective solutions are also printed in bold.

## B.4 Detailed Results of TS-DST on VRPTWDST Instances

We present detailed results of the testing of TS-DST on the newly generated VRPTWDST benchmark instances described in Section 4.3.2. TS-DST is run 10 times for each problem instance and we report the number of vehicles (#Rts.) and the value of the secondary objective function (traveled distance TD, working time WT and working duration WD) of the best solution found. Furthermore, averages over the problem classes C, R and RC and the CNV and the cumulated secondary objective function values (CTD, CWT and CWD) are given.

For the secondary objective of minimizing the traveled distance, the results can be found in Table B.3 for the instance sets generated with the *Random* mechanism and in Table B.4 for the sets generated with the *Cluster* mechanism. Here, we also report CPU times as the tests were conducted on our desktop PC. For the secondary objectives of minimizing working time and working duration, the results are given in Tables B.5 – B.8. No run-times are reported as the tests were conducted on the computing cluster Elwetritsch.

| | BKS | | TS-DST(Best) | | TS-DST(Avg.) | | |
|---|---|---|---|---|---|---|---|
| | #Rts | TD | #Rts. | TD | #Rts. | TD | CPU (s) |
| c101 | **10** | **828.94** | **10** | **828.94** | **10.00** | **828.94** | 0.06 |
| c102 | **10** | **828.94** | **10** | **828.94** | **10.00** | **828.94** | 0.11 |
| c103 | **10** | **828.06** | **10** | **828.06** | **10.00** | **828.06** | 13.60 |
| c104 | **10** | **824.78** | **10** | **824.78** | 10.00 | 825.19 | 31.78 |
| c105 | **10** | **828.94** | **10** | **828.94** | **10.00** | **828.94** | 0.04 |
| c106 | **10** | **828.94** | **10** | **828.94** | **10.00** | **828.94** | 0.06 |
| c107 | **10** | **828.94** | **10** | **828.94** | **10.00** | **828.94** | 0.07 |
| c108 | **10** | **828.94** | **10** | **828.94** | **10.00** | **828.94** | 0.23 |
| c109 | **10** | **828.94** | **10** | **828.94** | **10.00** | **828.94** | 0.24 |
| **Avg. C1** | **10.00** | **828.38** | **10.00** | **828.38** | 10.00 | 828.43 | 5.13 |
| c201 | **3** | **591.56** | **3** | **591.56** | **3.00** | **591.56** | 0.05 |
| c202 | **3** | **591.56** | **3** | **591.56** | **3.00** | **591.56** | 0.19 |
| c203 | **3** | **591.17** | **3** | **591.17** | **3.00** | **591.17** | 16.11 |
| c204 | **3** | **590.60** | **3** | **590.60** | 3.00 | 593.01 | 45.48 |
| c205 | **3** | **588.88** | **3** | **588.88** | **3.00** | **588.88** | 0.24 |
| c206 | **3** | **588.49** | **3** | **588.49** | **3.00** | **588.49** | 1.49 |
| c207 | **3** | **588.29** | **3** | **588.29** | **3.00** | **588.29** | 2.87 |
| c208 | **3** | **588.32** | **3** | **588.32** | **3.00** | **588.32** | 6.45 |
| **Avg. C2** | **3.00** | **589.86** | **3.00** | **589.86** | 3.00 | 590.16 | 9.11 |
| r101 | **19** | **1.645.79** | 19 | 1.650.80 | 19.00 | 1.654.08 | 27.12 |
| r102 | **17** | **1.486.12** | **17** | **1.486.12** | 17.20 | 1.488.30 | 100.71 |
| r103 | **13** | **1.292.68** | **13** | **1.292.68** | 13.40 | 1.265.06 | 250.28 |
| r104 | **9** | **1.007.24** | 9 | 1.018.07 | 9.90 | 999.90 | 281.63 |
| r105 | **14** | **1.377.11** | **14** | **1.377.11** | 14.00 | 1.394.79 | 25.99 |
| r106 | **12** | **1.251.98** | 12 | 1.257.96 | 12.00 | 1.267.18 | 34.23 |
| r107 | **10** | **1.104.66** | 10 | 1.115.14 | 10.00 | 1.129.10 | 100.38 |
| r108 | **9** | **960.88** | 9 | 967.35 | 9.00 | 981.13 | 81.85 |
| r109 | **11** | **1.194.73** | 11 | 1.203.74 | 11.00 | 1.243.31 | 43.95 |
| r110 | **10** | **1.118.59** | 10 | 1.118.84 | 10.00 | 1.152.42 | 152.41 |
| r111 | **10** | **1.096.72** | 10 | 1.106.25 | 10.10 | 1.140.64 | 68.85 |
| r112 | **9** | **982.14** | 9 | 1.024.82 | 9.80 | 983.47 | 294.33 |
| **Avg. R1** | **11.92** | **1.209.89** | 11.92 | 1.218.24 | 12.12 | 1.224.95 | 121.81 |
| r201 | **4** | **1.252.37** | 4 | 1.254.57 | 4.00 | 1.261.33 | 29.11 |
| r202 | **3** | **1.191.70** | 3 | 1.198.45 | 3.00 | 1.210.73 | 68.86 |
| r203 | **3** | **939.50** | 3 | 945.35 | 3.00 | 952.16 | 72.77 |
| r204 | **2** | **825.52** | 2 | 833.65 | 2.00 | 842.06 | 84.71 |
| r205 | **3** | **994.42** | 3 | 1.005.94 | 3.00 | 1.023.24 | 43.60 |
| r206 | **3** | **906.14** | 3 | 915.06 | 3.00 | 929.79 | 72.12 |
| r207 | **2** | **890.61** | 2 | 900.00 | 2.00 | 914.97 | 73.10 |
| r208 | **2** | **726.75** | 2 | 731.06 | 2.00 | 742.73 | 95.60 |
| r209 | **3** | **909.16** | 3 | 915.58 | 3.00 | 931.17 | 52.16 |
| r210 | **3** | **939.34** | 3 | 952.88 | 3.00 | 963.75 | 58.35 |
| r211 | **2** | **885.71** | 2 | 934.47 | 2.00 | 955.83 | 99.17 |
| **Avg. R2** | **2.73** | **951.02** | 2.73 | 962.46 | 2.73 | 975.25 | 68.14 |
| rc101 | **14** | **1.696.94** | 14 | 1.696.95 | 14.80 | 1.656.61 | 105.29 |
| rc102 | **12** | **1.554.75** | 12 | 1.562.96 | 12.70 | 1.533.36 | 200.56 |
| rc103 | **11** | **1.261.67** | 11 | 1.263.38 | 11.00 | 1.269.48 | 51.30 |
| rc104 | **10** | **1.135.48** | 10 | 1.143.74 | 10.00 | 1.156.83 | 47.44 |
| rc105 | **13** | **1.629.44** | 13 | 1.637.71 | 13.70 | 1.587.82 | 200.91 |
| rc106 | **11** | **1.424.73** | 11 | 1.427.13 | 11.80 | 1.408.32 | 179.47 |
| rc107 | **11** | **1.230.48** | 11 | 1.232.26 | 11.00 | 1.239.36 | 30.57 |
| rc108 | **10** | **1.139.82** | 10 | 1.156.99 | 10.00 | 1.188.09 | 78.99 |
| **Avg. RC1** | **11.50** | **1.384.16** | 11.50 | 1.390.14 | 11.88 | 1.379.98 | 111.82 |
| rc201 | **4** | **1.406.91** | 4 | 1.418.49 | 4.00 | 1.432.03 | 37.73 |
| rc202 | **3** | **1.365.65** | 3 | 1.369.18 | 3.00 | 1.414.93 | 49.30 |
| rc203 | **3** | **1.049.62** | 3 | 1.079.53 | 3.00 | 1.092.97 | 59.99 |
| rc204 | **3** | **798.41** | 3 | 800.24 | 3.00 | 819.03 | 78.86 |
| rc205 | **4** | **1.297.19** | 4 | 1.300.25 | 4.00 | 1.309.65 | 40.85 |
| rc206 | **3** | **1.146.32** | 3 | 1.156.45 | 3.00 | 1.181.49 | 42.92 |
| rc207 | **3** | **1.061.14** | 3 | 1.070.02 | 3.00 | 1.100.67 | 67.95 |
| rc208 | **3** | **828.14** | 3 | 829.69 | 3.00 | 851.95 | 75.23 |
| RC2 | **3.25** | **1.119.17** | 3.25 | 1.127.98 | 3.25 | 1.150.34 | 56.60 |
| **CNV/CTD** | **405** | **57180.84** | **405** | **57525.15** | **410.40** | **57846.82** | **65.67** |

**Table B.2:** Detailed results of TS-DST on Solomon VRPTW instances. Best and average results of 10 TS-DST runs are reported compared to the best-known solution (BKS). CPU time is given as average over the runs.

| | $\Gamma = 0.9$ | | | $\Gamma = 0.7$ | | | $\Gamma = 0.5$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | **#Rts.** | **TD** | **CPU (s)** | **#Rts.** | **TD** | **CPU (s)** | **#Rts.** | **TD** | **CPU (s)** |
| c101 | 10 | 828.94 | 17.03 | 10 | 828.94 | 17.03 | 10 | 828.94 | 17.64 |
| c102 | 10 | 828.94 | 23.44 | 10 | 828.94 | 23.61 | 10 | 828.94 | 23.83 |
| c103 | 10 | 828.06 | 36.86 | 10 | 828.06 | 40.09 | 10 | 828.06 | 37.63 |
| c104 | 10 | 824.78 | 32.22 | 10 | 822.79 | 31.95 | 10 | 822.79 | 41.40 |
| c105 | 10 | 828.94 | 18.37 | 10 | 828.94 | 18.70 | 10 | 828.94 | 19.36 |
| c106 | 10 | 828.94 | 19.29 | 10 | 828.94 | 19.65 | 10 | 828.94 | 20.52 |
| c107 | 10 | 828.94 | 20.10 | 10 | 828.94 | 20.79 | 10 | 828.94 | 21.05 |
| c108 | 10 | 828.94 | 22.09 | 10 | 828.94 | 22.75 | 10 | 828.94 | 23.11 |
| c109 | 10 | 828.94 | 25.16 | 10 | 828.94 | 25.80 | 10 | 828.94 | 26.07 |
| **Avg. C1** | 10.00 | 828.38 | 23.84 | 10.00 | 828.16 | 24.49 | 10.00 | 828.16 | 25.62 |
| c201 | 3 | 591.56 | 27.12 | 3 | 591.56 | 27.10 | 3 | 588.88 | 27.03 |
| c202 | 3 | 591.56 | 38.04 | 3 | 591.56 | 37.67 | 3 | 588.88 | 40.07 |
| c203 | 3 | 588.49 | 51.24 | 3 | 585.46 | 78.12 | 3 | 585.27 | 70.94 |
| c204 | 3 | 587.71 | 48.89 | 3 | 584.68 | 58.39 | 3 | 584.49 | 62.07 |
| c205 | 3 | 588.49 | 28.35 | 3 | 588.49 | 27.91 | 3 | 588.49 | 28.83 |
| c206 | 3 | 588.49 | 31.92 | 3 | 588.49 | 32.50 | 3 | 588.49 | 34.47 |
| c207 | 3 | 588.29 | 46.82 | 3 | 588.29 | 37.76 | 3 | 588.29 | 32.13 |
| c208 | 3 | 588.32 | 36.85 | 3 | 588.29 | 36.27 | 3 | 588.29 | 34.49 |
| **Avg. C2** | 3.00 | 589.11 | 38.65 | 3.00 | 588.35 | 41.97 | 3.00 | 587.64 | 41.25 |
| r101 | 18 | 1619.42 | 39.30 | 18 | 1604.37 | 32.16 | 17 | 1624.28 | 64.01 |
| r102 | 17 | 1450.43 | 57.34 | 16 | 1445.26 | 113.00 | 15 | 1422.61 | 122.80 |
| r103 | 13 | 1246.99 | 124.57 | 13 | 1230.25 | 162.23 | 12 | 1189.59 | 78.82 |
| r104 | 9 | 1014.08 | 233.32 | 9 | 994.30 | 93.75 | 9 | 977.04 | 97.07 |
| r105 | 14 | 1369.26 | 26.69 | 13 | 1458.44 | 177.86 | 13 | 1383.54 | 46.80 |
| r106 | 12 | 1245.86 | 28.27 | 11 | 1285.52 | 213.03 | 11 | 1245.63 | 66.20 |
| r107 | 10 | 1082.08 | 52.63 | 10 | 1074.40 | 52.94 | 9 | 1127.94 | 40.43 |
| r108 | 9 | 965.17 | 44.27 | 9 | 947.11 | 41.94 | 8 | 976.76 | 157.68 |
| r109 | 11 | 1203.75 | 31.12 | 11 | 1181.80 | 32.35 | 10 | 1338.81 | 189.80 |
| r110 | 10 | 1138.26 | 44.42 | 10 | 1109.75 | 36.65 | 9 | 1359.07 | 336.90 |
| r111 | 10 | 1095.53 | 35.51 | 10 | 1079.36 | 32.39 | 10 | 1041.87 | 50.00 |
| r112 | 9 | 1001.84 | 131.92 | 9 | 963.34 | 51.00 | 9 | 946.82 | 49.09 |
| **Avg. R1** | 11.83 | 1202.72 | 70.78 | 11.58 | 1197.83 | 86.61 | 11.00 | 1219.50 | 108.30 |
| r201 | 4 | 1253.26 | 36.85 | 3 | 1529.65 | 233.29 | 3 | 1435.85 | 33.35 |
| r202 | 3 | 1186.99 | 68.19 | 3 | 1151.97 | 87.03 | 3 | 1132.16 | 63.37 |
| r203 | 3 | 939.10 | 84.32 | 2 | 1098.85 | 597.72 | 2 | 1064.10 | 103.04 |
| r204 | 2 | 824.15 | 77.52 | 2 | 813.90 | 100.48 | 2 | 781.89 | 100.85 |
| r205 | 3 | 994.43 | 46.16 | 3 | 1006.34 | 66.61 | 3 | 999.15 | 51.25 |
| r206 | 3 | 912.85 | 52.90 | 2 | 1032.28 | 119.35 | 2 | 1011.96 | 63.53 |
| r207 | 2 | 891.57 | 111.44 | 2 | 871.59 | 77.17 | 2 | 846.39 | 63.65 |
| r208 | 2 | 727.34 | 107.15 | 2 | 728.14 | 78.84 | 2 | 724.68 | 84.47 |
| r209 | 3 | 908.38 | 66.45 | 3 | 904.92 | 90.68 | 2 | 1091.77 | 68.07 |
| r210 | 3 | 945.52 | 78.31 | 2 | 1102.86 | 439.54 | 2 | 1065.04 | 107.82 |
| r211 | 2 | 897.11 | 73.14 | 2 | 852.50 | 100.32 | 2 | 817.82 | 89.61 |
| **Avg. R2** | 2.73 | 952.79 | 72.95 | 2.36 | 1008.45 | 181.00 | 2.27 | 997.35 | 75.36 |
| rc101 | 14 | 1689.43 | 91.27 | 14 | 1623.48 | 48.58 | 13 | 1712.81 | 109.01 |
| rc102 | 12 | 1494.52 | 178.77 | 12 | 1465.11 | 37.19 | 11 | 1506.54 | 256.07 |
| rc103 | 10 | 1336.10 | 211.83 | 10 | 1282.68 | 35.33 | 10 | 1231.24 | 29.99 |
| rc104 | 10 | 1139.89 | 44.07 | 9 | 1223.58 | 224.23 | 9 | 1160.92 | 36.11 |
| rc105 | 13 | 1632.79 | 214.06 | 13 | 1496.91 | 37.49 | 12 | 1504.09 | 221.32 |
| rc106 | 11 | 1449.21 | 203.22 | 11 | 1365.96 | 61.57 | 11 | 1320.88 | 32.71 |
| rc107 | 10 | 1325.93 | 300.59 | 10 | 1262.03 | 173.69 | 10 | 1249.10 | 32.66 |
| rc108 | 10 | 1145.49 | 45.12 | 10 | 1112.53 | 45.52 | 9 | 1197.78 | 45.91 |
| **Avg. RC1** | 11.25 | 1401.67 | 161.12 | 11.13 | 1354.04 | 82.95 | 10.63 | 1360.42 | 95.47 |
| rc201 | 4 | 1410.27 | 42.77 | 4 | 1383.47 | 36.23 | 3 | 1633.01 | 332.30 |
| rc202 | 3 | 1307.04 | 58.69 | 3 | 1287.43 | 82.26 | 3 | 1252.71 | 60.00 |
| rc203 | 3 | 1051.71 | 61.38 | 3 | 1030.78 | 59.96 | 2 | 1303.04 | 601.65 |
| rc204 | 3 | 796.71 | 75.54 | 2 | 959.58 | 176.73 | 2 | 896.05 | 103.11 |
| rc205 | 4 | 1297.65 | 40.41 | 3 | 1499.25 | 48.59 | 3 | 1502.57 | 73.92 |
| rc206 | 3 | 1137.48 | 56.61 | 3 | 1123.17 | 46.12 | 3 | 1122.25 | 58.44 |
| rc207 | 3 | 1062.00 | 81.43 | 3 | 1037.97 | 63.91 | 3 | 1030.06 | 66.39 |
| rc208 | 3 | 833.29 | 64.79 | 2 | 1001.39 | 178.59 | 2 | 992.17 | 72.79 |
| **Avg. RC2** | 3.25 | 1112.02 | 60.20 | 2.88 | 1165.38 | 86.55 | 2.63 | 1216.48 | 171.08 |
| **CNV/CTD** | 402 | 57191.21 | 70.47 | 391.00 | 57782.47 | 88.26 | 377 | 58374.50 | 86.10 |

**Table B.3:** Results of TS-DST for the secondary objective of minimizing traveled distance on the VRPTWDST benchmark sets generated with the *Random* mechanism. The best solution found in 10 runs is reported in columns #Rts. and TD and the average run-time in column CPU.

| | $\Gamma = 0.9$ | | | $\Gamma = 0.7$ | | | $\Gamma = 0.5$ | | |
|---|---|---|---|---|---|---|---|---|---|
| | **#Rts.** | **TD** | **CPU (s)** | **#Rts.** | **TD** | **CPU (s)** | **#Rts.** | **TD** | **CPU (s)** |
| c101 | 10 | 828.94 | 16.62 | 10 | 828.94 | 17.57 | 10 | 828.94 | 18.13 |
| c102 | 10 | 828.94 | 23.56 | 10 | 828.94 | 23.77 | 10 | 828.94 | 25.14 |
| c103 | 10 | 828.06 | 41.72 | 10 | 828.06 | 31.58 | 10 | 828.06 | 36.85 |
| c104 | 10 | 824.78 | 47.28 | 10 | 822.79 | 49.98 | 10 | 822.79 | 45.82 |
| c105 | 10 | 828.94 | 18.45 | 10 | 828.94 | 18.97 | 10 | 828.94 | 20.01 |
| c106 | 10 | 828.94 | 19.58 | 10 | 828.94 | 20.60 | 10 | 828.94 | 20.96 |
| c107 | 10 | 828.94 | 20.26 | 10 | 828.94 | 20.90 | 10 | 828.94 | 21.64 |
| c108 | 10 | 828.94 | 22.12 | 10 | 828.94 | 22.84 | 10 | 828.94 | 23.34 |
| c109 | 10 | 828.94 | 25.00 | 10 | 827.38 | 27.48 | 10 | 825.65 | 27.97 |
| **Avg. C1** | 10.00 | 828.38 | 26.06 | 10.00 | 827.99 | 25.97 | 10.00 | 827.79 | 26.65 |
| c201 | 3 | 591.56 | 26.86 | 3 | 591.56 | 26.98 | 3 | 588.88 | 26.97 |
| c202 | 3 | 591.56 | 36.63 | 3 | 591.56 | 39.30 | 3 | 588.88 | 38.77 |
| c203 | 3 | 585.27 | 52.20 | 3 | 585.27 | 70.43 | 3 | 585.27 | 47.16 |
| c204 | 3 | 584.49 | 61.99 | 3 | 584.49 | 61.96 | 3 | 584.49 | 58.38 |
| c205 | 3 | 588.49 | 28.14 | 3 | 588.49 | 27.70 | 3 | 588.49 | 30.75 |
| c206 | 3 | 588.49 | 31.67 | 3 | 588.49 | 30.89 | 3 | 588.49 | 33.00 |
| c207 | 3 | 588.29 | 34.67 | 3 | 588.29 | 33.17 | 3 | 587.89 | 33.97 |
| c208 | 3 | 588.32 | 40.71 | 3 | 588.29 | 44.14 | 3 | 588.29 | 50.50 |
| **Avg. C2** | 3.00 | 588.31 | 39.11 | 3.00 | 588.31 | 41.82 | 3.00 | 587.59 | 39.94 |
| r101 | 18 | 1610.94 | 26.27 | 17 | 1562.26 | 23.97 | 15 | 1552.30 | 165.48 |
| r102 | 17 | 1440.65 | 49.46 | 15 | 1369.68 | 67.12 | 14 | 1367.88 | 217.82 |
| r103 | 13 | 1259.19 | 190.35 | 13 | 1157.22 | 95.48 | 11 | 1104.37 | 170.63 |
| r104 | 9 | 975.57 | 52.40 | 8 | 975.30 | 359.97 | 8 | 949.61 | 222.41 |
| r105 | 13 | 1426.94 | 162.04 | 12 | 1356.64 | 36.59 | 11 | 1306.69 | 54.55 |
| r106 | 11 | 1325.28 | 263.76 | 10 | 1222.31 | 155.93 | 9 | 1156.73 | 172.68 |
| r107 | 10 | 1070.79 | 59.88 | 9 | 1014.54 | 45.53 | 8 | 1066.29 | 109.63 |
| r108 | 9 | 936.98 | 76.61 | 8 | 917.59 | 40.41 | 8 | 875.50 | 65.09 |
| r109 | 11 | 1168.25 | 33.80 | 9 | 1241.32 | 254.46 | 9 | 1063.02 | 45.25 |
| r110 | 10 | 1088.36 | 43.15 | 9 | 1056.70 | 48.12 | 8 | 1064.62 | 63.12 |
| r111 | 10 | 1063.91 | 51.94 | 9 | 1005.63 | 52.88 | 8 | 1000.50 | 52.76 |
| r112 | 9 | 966.34 | 45.48 | 8 | 932.54 | 38.93 | 8 | 874.12 | 48.03 |
| **Avg. R1** | 11.67 | 1194.43 | 87.93 | 10.58 | 1150.98 | 101.62 | 9.75 | 1115.14 | 115.62 |
| r201 | 4 | 1245.96 | 35.70 | 3 | 1391.61 | 32.83 | 3 | 1292.95 | 39.04 |
| r202 | 3 | 1151.15 | 56.06 | 3 | 1107.64 | 72.72 | 3 | 1078.52 | 59.14 |
| r203 | 3 | 936.52 | 84.44 | 2 | 1122.75 | 99.14 | 2 | 1057.62 | 129.93 |
| r204 | 2 | 816.19 | 81.86 | 2 | 794.07 | 94.70 | 2 | 766.30 | 109.29 |
| r205 | 3 | 1001.71 | 63.54 | 2 | 1157.96 | 468.98 | 2 | 1095.10 | 65.93 |
| r206 | 2 | 1074.00 | 514.09 | 2 | 977.66 | 69.74 | 2 | 955.16 | 94.46 |
| r207 | 2 | 881.21 | 66.40 | 2 | 855.19 | 88.69 | 2 | 832.98 | 86.43 |
| r208 | 2 | 727.07 | 111.79 | 2 | 716.94 | 94.13 | 2 | 721.27 | 113.69 |
| r209 | 3 | 909.75 | 70.25 | 2 | 1064.81 | 85.48 | 2 | 944.84 | 90.24 |
| r210 | 3 | 946.89 | 72.64 | 2 | 1056.77 | 138.89 | 2 | 1019.48 | 78.22 |
| r211 | 2 | 872.72 | 76.98 | 2 | 804.31 | 77.98 | 2 | 771.77 | 76.35 |
| **Avg. R2** | 2.64 | 960.29 | 112.16 | 2.18 | 1004.52 | 120.30 | 2.18 | 957.82 | 85.70 |
| rc101 | 14 | 1624.72 | 25.06 | 13 | 1507.44 | 19.22 | 11 | 1517.08 | 125.44 |
| rc102 | 12 | 1476.36 | 35.83 | 11 | 1380.37 | 50.59 | 10 | 1342.22 | 69.70 |
| rc103 | 10 | 1273.25 | 138.61 | 9 | 1279.57 | 302.23 | 9 | 1184.77 | 50.04 |
| rc104 | 10 | 1124.86 | 44.59 | 9 | 1094.47 | 43.81 | 9 | 1067.40 | 47.73 |
| rc105 | 13 | 1491.69 | 91.02 | 11 | 1454.70 | 37.27 | 10 | 1380.15 | 215.27 |
| rc106 | 11 | 1353.91 | 76.29 | 10 | 1313.35 | 47.70 | 9 | 1287.52 | 219.38 |
| rc107 | 10 | 1281.59 | 118.80 | 9 | 1314.24 | 298.10 | 9 | 1169.18 | 36.44 |
| rc108 | 10 | 1112.36 | 55.11 | 9 | 1106.37 | 48.51 | 9 | 1058.32 | 34.85 |
| **Avg. RC1** | 11.25 | 1342.34 | 73.16 | 10.13 | 1306.31 | 105.93 | 9.50 | 1250.83 | 99.85 |
| rc201 | 4 | 1376.69 | 53.88 | 3 | 1574.45 | 227.99 | 3 | 1500.80 | 52.63 |
| rc202 | 3 | 1314.05 | 71.39 | 3 | 1274.34 | 75.32 | 3 | 1237.71 | 57.53 |
| rc203 | 3 | 1049.19 | 76.44 | 2 | 1225.63 | 104.75 | 2 | 1184.25 | 116.57 |
| rc204 | 2 | 961.14 | 197.80 | 2 | 901.82 | 88.90 | 2 | 821.96 | 73.34 |
| rc205 | 3 | 1606.68 | 350.12 | 3 | 1461.57 | 56.08 | 3 | 1364.09 | 46.30 |
| rc206 | 3 | 1124.93 | 49.98 | 3 | 1112.98 | 46.94 | 2 | 1279.81 | 110.25 |
| rc207 | 3 | 1065.59 | 68.64 | 3 | 1013.99 | 77.89 | 2 | 1178.83 | 71.63 |
| rc208 | 2 | 995.77 | 703.09 | 2 | 919.70 | 96.08 | 2 | 851.47 | 85.23 |
| **Avg. RC2** | 2.88 | 1186.76 | 196.42 | 2.63 | 1185.56 | 96.74 | 2.38 | 1177.37 | 76.68 |
| **CNV/CTD** | 396 | 57291.04 | 89.16 | 367.00 | 56954.74 | 84.51 | 350 | 55494.00 | 76.82 |

**Table B.4:** Results of TS-DST for the secondary objective of minimizing traveled distance on the VRPTWDST benchmark sets generated with the *Cluster* mechanism. The best solution found in 10 runs is reported in columns #Rts. and TD and the average run-time in column CPU.

| | $\Gamma = 0.9$ | | $\Gamma = 0.7$ | | $\Gamma = 0.5$ | |
|---|---|---|---|---|---|---|
| | **#Rts.** | **WT** | **#Rts.** | **WT** | **#Rts.** | **WT** |
| c101 | 10 | 9601.59 | 10 | 9060.92 | 10 | 8236.12 |
| c102 | 10 | 9598.74 | 10 | 9058.99 | 10 | 8138.03 |
| c103 | 10 | 9598.74 | 10 | 9038.91 | 10 | 8214.57 |
| c104 | 10 | 9590.14 | 10 | 8950.39 | 10 | 8176.59 |
| c105 | 10 | 9601.59 | 10 | 9065.68 | 10 | 7756.24 |
| c106 | 10 | 9601.59 | 10 | 8969.81 | 10 | 8084.53 |
| c107 | 10 | 9590.59 | 10 | 8958.20 | 10 | 7666.05 |
| c108 | 10 | 9590.59 | 10 | 8902.20 | 10 | 8022.50 |
| c109 | 10 | 9589.58 | 10 | 9037.92 | 10 | 8352.61 |
| **Avg. C1** | 10.00 | 9595.91 | 10.00 | 9004.78 | 10.00 | 8071.92 |
| c201 | 3 | 9276.15 | 3 | 8636.69 | 3 | 7561.71 |
| c202 | 3 | 9276.15 | 3 | 8596.14 | 3 | 7922.81 |
| c203 | 3 | 9273.39 | 3 | 8642.27 | 3 | 8080.94 |
| c204 | 3 | 9271.96 | 3 | 8607.33 | 3 | 7803.34 |
| c205 | 3 | 9273.39 | 3 | 8631.19 | 3 | 7949.20 |
| c206 | 3 | 9273.39 | 3 | 8643.18 | 3 | 7949.20 |
| c207 | 3 | 9272.20 | 3 | 8640.02 | 3 | 7976.25 |
| c208 | 3 | 9273.22 | 3 | 8640.02 | 3 | 7959.85 |
| **Avg. C2** | 3.00 | 9273.73 | 3.00 | 8629.61 | 3.00 | 7900.41 |
| r101 | 18 | 2568.15 | 18 | 2441.12 | 17 | 2305.48 |
| r102 | 17 | 2402.74 | 16 | 2283.75 | 15 | 2163.07 |
| r103 | 13 | 2210.44 | 13 | 2092.74 | 12 | 1955.88 |
| r104 | 9 | 1998.05 | 9 | 1892.25 | 9 | 1794.57 |
| r105 | 14 | 2324.84 | 13 | 2289.76 | 13 | 2088.02 |
| r106 | 12 | 2210.46 | 11 | 2159.62 | 11 | 2029.29 |
| r107 | 10 | 2051.81 | 10 | 1970.96 | 9 | 1888.46 |
| r108 | 9 | 1931.38 | 9 | 1825.53 | 8 | 1709.74 |
| r109 | 11 | 2163.20 | 11 | 2072.37 | 10 | 2033.48 |
| r110 | 10 | 2104.74 | 10 | 1994.76 | 9 | 2004.34 |
| r111 | 10 | 2057.23 | 10 | 1952.69 | 10 | 1858.33 |
| r112 | 9 | 1961.24 | 9 | 1844.41 | 9 | 1747.88 |
| **Avg. R1** | 11.83 | 2165.36 | 11.58 | 2068.33 | 11.00 | 1964.88 |
| r201 | 4 | 2209.33 | 3 | 2332.45 | 3 | 2153.04 |
| r202 | 3 | 2136.04 | 3 | 1999.81 | 3 | 1838.19 |
| r203 | 3 | 1893.06 | 2 | 1889.65 | 2 | 1728.45 |
| r204 | 2 | 1761.94 | 2 | 1626.54 | 2 | 1451.76 |
| r205 | 3 | 1944.37 | 3 | 1860.83 | 3 | 1727.22 |
| r206 | 3 | 1859.76 | 2 | 1887.47 | 2 | 1755.93 |
| r207 | 2 | 1827.68 | 2 | 1669.02 | 2 | 1499.55 |
| r208 | 2 | 1671.25 | 2 | 1552.69 | 2 | 1427.23 |
| r209 | 3 | 1864.71 | 3 | 1758.33 | 2 | 1743.37 |
| r210 | 3 | 1903.16 | 2 | 1949.65 | 2 | 1750.80 |
| r211 | 2 | 1834.42 | 2 | 1675.48 | 2 | 1488.69 |
| **Avg. R2** | 2.73 | 1900.52 | 2.36 | 1836.54 | 2.27 | 1687.66 |
| rc101 | 14 | 2626.78 | 14 | 2454.57 | 13 | 2395.99 |
| rc102 | 12 | 2438.45 | 12 | 2297.30 | 11 | 2199.59 |
| rc103 | 10 | 2286.91 | 10 | 2148.68 | 10 | 2031.80 |
| rc104 | 10 | 2097.76 | 9 | 2052.89 | 9 | 1925.60 |
| rc105 | 13 | 2573.91 | 13 | 2331.55 | 12 | 2210.13 |
| rc106 | 11 | 2404.84 | 11 | 2228.17 | 11 | 2054.26 |
| rc107 | 10 | 2289.35 | 10 | 2123.89 | 10 | 1990.41 |
| rc108 | 10 | 2098.65 | 10 | 1971.69 | 9 | 1922.01 |
| **Avg. RC1** | 11.25 | 2352.08 | 11.13 | 2201.09 | 10.63 | 2091.22 |
| rc201 | 4 | 2366.18 | 4 | 2257.28 | 3 | 2309.29 |
| rc202 | 3 | 2255.63 | 3 | 2100.00 | 3 | 1911.47 |
| rc203 | 3 | 1998.35 | 3 | 1880.04 | 2 | 1854.65 |
| rc204 | 3 | 1754.03 | 2 | 1780.47 | 2 | 1580.25 |
| rc205 | 4 | 2242.27 | 3 | 2347.04 | 3 | 2210.01 |
| rc206 | 3 | 2083.03 | 3 | 1976.68 | 3 | 1859.66 |
| rc207 | 3 | 2018.35 | 3 | 1897.50 | 3 | 1769.38 |
| rc208 | 3 | 1790.19 | 2 | 1825.26 | 2 | 1695.08 |
| **Avg. RC2** | 3.25 | 2063.50 | 2.88 | 2008.03 | 2.63 | 1898.72 |
| **CNV/CWT** | 402 | 242767.68 | 391 | 228774.75 | 377 | 209912.89 |

**Table B.5:** Results of TS-DST for the secondary objective of minimizing working time on the VRPTWDST benchmark sets generated with the *Random* mechanism. The best solution found in 10 runs is reported in columns #Rts. and WT.

| | $\Gamma = 0.9$ | | $\Gamma = 0.7$ | | $\Gamma = 0.5$ | |
|---|---|---|---|---|---|---|
| | **#Rts.** | **WT** | **#Rts.** | **WT** | **#Rts.** | **WT** |
| c101 | 10 | 8907.58 | 10 | 7058.25 | 10 | 5197.93 |
| c102 | 10 | 8907.58 | 10 | 7058.25 | 10 | 5208.91 |
| c103 | 10 | 8903.72 | 10 | 7055.02 | 10 | 5206.32 |
| c104 | 10 | 8891.10 | 10 | 7057.94 | 10 | 5156.38 |
| c105 | 10 | 8907.58 | 10 | 7058.25 | 10 | 5197.87 |
| c106 | 10 | 8907.58 | 10 | 7063.85 | 10 | 5178.84 |
| c107 | 10 | 8907.58 | 10 | 7058.25 | 10 | 5160.64 |
| c108 | 10 | 8907.58 | 10 | 7050.99 | 10 | 5167.89 |
| c109 | 10 | 8904.90 | 10 | 7073.37 | 10 | 5195.93 |
| **Avg. C1** | 10.00 | 8905.02 | 10.00 | 7059.35 | 10.00 | 5185.63 |
| c201 | 3 | 8642.80 | 3 | 6745.29 | 3 | 4840.41 |
| c202 | 3 | 8642.80 | 3 | 6745.29 | 3 | 4846.43 |
| c203 | 3 | 8648.75 | 3 | 6775.73 | 3 | 4902.70 |
| c204 | 3 | 8639.34 | 3 | 6775.18 | 3 | 4846.29 |
| c205 | 3 | 8640.04 | 3 | 6743.14 | 3 | 4846.24 |
| c206 | 3 | 8640.04 | 3 | 6743.14 | 3 | 4846.24 |
| c207 | 3 | 8639.86 | 3 | 6743.00 | 3 | 4845.94 |
| c208 | 3 | 8639.89 | 3 | 6743.00 | 3 | 4846.14 |
| **Avg. C2** | 3.00 | 8641.69 | 3.00 | 6751.72 | 3.00 | 4852.55 |
| r101 | 18 | 2507.18 | 17 | 2173.09 | 15 | 1872.55 |
| r102 | 17 | 2328.55 | 15 | 1946.99 | 14 | 1691.21 |
| r103 | 13 | 2170.86 | 13 | 1820.94 | 11 | 1518.61 |
| r104 | 9 | 1860.58 | 8 | 1580.66 | 8 | 1336.94 |
| r105 | 13 | 2313.84 | 12 | 1963.47 | 11 | 1618.78 |
| r106 | 11 | 2226.27 | 10 | 1832.76 | 9 | 1618.11 |
| r107 | 10 | 1946.09 | 9 | 1580.40 | 8 | 1457.94 |
| r108 | 9 | 1790.99 | 8 | 1545.22 | 8 | 1301.87 |
| r109 | 11 | 2024.13 | 9 | 1927.77 | 9 | 1459.54 |
| r110 | 10 | 1963.10 | 9 | 1702.05 | 8 | 1481.74 |
| r111 | 10 | 1905.66 | 9 | 1635.45 | 8 | 1415.83 |
| r112 | 9 | 1839.47 | 8 | 1554.15 | 8 | 1251.82 |
| **Avg. R1** | 11.67 | 2073.06 | 10.58 | 1771.91 | 9.75 | 1502.08 |
| r201 | 4 | 2081.98 | 3 | 2044.80 | 3 | 1679.00 |
| r202 | 3 | 1999.38 | 3 | 1630.12 | 3 | 1227.19 |
| r203 | 3 | 1793.00 | 2 | 1754.65 | 2 | 1509.86 |
| r204 | 2 | 1666.34 | 2 | 1311.66 | 2 | 988.01 |
| r205 | 3 | 1876.23 | 2 | 1783.94 | 2 | 1528.07 |
| r206 | 2 | 1952.42 | 2 | 1642.90 | 2 | 1357.53 |
| r207 | 2 | 1744.13 | 2 | 1448.83 | 2 | 1091.50 |
| r208 | 2 | 1584.59 | 2 | 1285.16 | 2 | 1010.63 |
| r209 | 3 | 1756.90 | 2 | 1697.24 | 2 | 1415.82 |
| r210 | 3 | 1811.65 | 2 | 1704.30 | 2 | 1486.92 |
| r211 | 2 | 1722.63 | 2 | 1341.89 | 2 | 973.97 |
| **Avg. R2** | 2.64 | 1817.20 | 2.18 | 1604.14 | 2.18 | 1297.14 |
| rc101 | 14 | 2483.87 | 13 | 2058.38 | 11 | 1842.51 |
| rc102 | 12 | 2343.06 | 11 | 1946.23 | 10 | 1621.36 |
| rc103 | 10 | 2144.40 | 9 | 1933.09 | 9 | 1621.45 |
| rc104 | 10 | 2000.70 | 9 | 1699.34 | 9 | 1340.66 |
| rc105 | 13 | 2377.01 | 11 | 2108.21 | 10 | 1749.20 |
| rc106 | 11 | 2222.87 | 10 | 1932.33 | 9 | 1690.98 |
| rc107 | 10 | 2157.11 | 9 | 1928.62 | 9 | 1596.79 |
| rc108 | 10 | 1961.10 | 9 | 1734.48 | 9 | 1429.49 |
| **Avg. RC1** | 11.25 | 2211.27 | 10.13 | 1917.59 | 9.50 | 1611.56 |
| rc201 | 4 | 2259.05 | 3 | 2164.68 | 3 | 1754.83 |
| rc202 | 3 | 2201.77 | 3 | 1804.64 | 3 | 1365.07 |
| rc203 | 3 | 1916.23 | 2 | 1814.77 | 2 | 1576.25 |
| rc204 | 2 | 1845.46 | 2 | 1582.20 | 2 | 1233.26 |
| rc205 | 3 | 2506.93 | 3 | 2114.84 | 3 | 1789.13 |
| rc206 | 3 | 2024.26 | 3 | 1709.44 | 2 | 1633.32 |
| rc207 | 3 | 1910.54 | 3 | 1551.02 | 2 | 1579.12 |
| rc208 | 2 | 1888.23 | 2 | 1593.99 | 2 | 1399.95 |
| **Avg. RC2** | 2.88 | 2069.06 | 2.63 | 1791.95 | 2.38 | 1541.37 |
| **CNV/CWT** | 396 | 228387.28 | 367 | 186132.64 | 350 | 143007.91 |

**Table B.6:** Results of TS-DST for the secondary objective of minimizing working time on the VRPTWDST benchmark sets generated with the *Cluster* mechanism. The best solution found in 10 runs is reported in columns #Rts. and WT.

| | $\Gamma = 0.9$ | | $\Gamma = 0.7$ | | $\Gamma = 0.5$ | |
|---|---|---|---|---|---|---|
| | **#Rts.** | **WD** | **#Rts.** | **WD** | **#Rts.** | **WD** |
| c101 | 10 | 9614.15 | 10 | 9374.99 | 10 | 8978.95 |
| c102 | 10 | 9614.15 | 10 | 9336.51 | 10 | 8954.71 |
| c103 | 10 | 9611.7 | 10 | 9217.54 | 10 | 8773.68 |
| c104 | 10 | 9616.51 | 10 | 9227.40 | 10 | 8797.07 |
| c105 | 10 | 9614.15 | 10 | 9221.15 | 10 | 8889.87 |
| c106 | 10 | 9615.54 | 10 | 9254.54 | 10 | 8941.25 |
| c107 | 10 | 9614.15 | 10 | 9185.80 | 10 | 8778.76 |
| c108 | 10 | 9614.15 | 10 | 9156.79 | 10 | 8762.97 |
| c109 | 10 | 9614.15 | 10 | 9185.80 | 10 | 8637.81 |
| **Avg. C1** | 10.00 | 9614.29 | 10.00 | 9240.06 | 10.00 | 8835.01 |
| c201 | 3 | 9338.4 | 3 | 9275.77 | 3 | 9225.83 |
| c202 | 3 | 9322.12 | 3 | 9068.98 | 3 | 8890.88 |
| c203 | 3 | 9303.17 | 3 | 8996.78 | 3 | 8429.89 |
| c204 | 3 | 9281.43 | 3 | 8806.86 | 3 | 8601.70 |
| c205 | 3 | 9274.47 | 3 | 9076.04 | 3 | 8733.90 |
| c206 | 3 | 9274.39 | 3 | 8969.61 | 3 | 8548.08 |
| c207 | 3 | 9282.03 | 3 | 8818.74 | 3 | 8560.58 |
| c208 | 3 | 9286.28 | 3 | 8707.02 | 3 | 8292.71 |
| **Avg. C2** | 3.00 | 9295.29 | 3.00 | 8964.98 | 3.00 | 8660.45 |
| r101 | 18 | 3251.62 | 18 | 3175.15 | 17 | 3044.99 |
| r102 | 17 | 2969.87 | 16 | 2776.43 | 15 | 2624.51 |
| r103 | 13 | 2439.99 | 13 | 2372.05 | 12 | 2169.72 |
| r104 | 9 | 2012.18 | 9 | 1931.37 | 9 | 1811.98 |
| r105 | 14 | 2562.78 | 13 | 2455.10 | 13 | 2347.70 |
| r106 | 12 | 2289.86 | 11 | 2224.32 | 11 | 2128.80 |
| r107 | 10 | 2120.35 | 10 | 2029.46 | 9 | 1900.83 |
| r108 | 9 | 1950.96 | 9 | 1850.05 | 8 | 1728.47 |
| r109 | 11 | 2230.38 | 11 | 2162.44 | 10 | 2093.26 |
| r110 | 10 | 2146.86 | 10 | 2077.43 | 9 | 2049.90 |
| r111 | 10 | 2122.77 | 10 | 2059.46 | 10 | 1957.97 |
| r112 | 9 | 1969.64 | 9 | 1877.13 | 9 | 1821.01 |
| **Avg. R1** | 11.83 | 2338.94 | 11.58 | 2249.20 | 11.00 | 2139.93 |
| r201 | 4 | 2931.9 | 3 | 2562.27 | 3 | 2503.81 |
| r202 | 3 | 2367.19 | 3 | 2290.68 | 3 | 2360.87 |
| r203 | 3 | 2213.56 | 2 | 1916.98 | 2 | 1801.99 |
| r204 | 2 | 1788.27 | 2 | 1716.57 | 2 | 1428.68 |
| r205 | 3 | 2180.56 | 3 | 2141.78 | 3 | 2141.90 |
| r206 | 3 | 2018.88 | 2 | 1875.89 | 2 | 1704.44 |
| r207 | 2 | 1840.04 | 2 | 1696.58 | 2 | 1586.07 |
| r208 | 2 | 1692.13 | 2 | 1575.73 | 2 | 1462.10 |
| r209 | 3 | 2142.93 | 3 | 2082.39 | 2 | 1720.08 |
| r210 | 3 | 2183.69 | 2 | 1952.20 | 2 | 1799.19 |
| r211 | 2 | 1832.79 | 2 | 1684.43 | 2 | 1429.79 |
| **Avg. R2** | 2.73 | 2108.36 | 2.36 | 1954.14 | 2.27 | 1812.63 |
| rc101 | 14 | 2829.41 | 14 | 2724.68 | 13 | 2622.91 |
| rc102 | 12 | 2588.11 | 12 | 2491.46 | 11 | 2314.92 |
| rc103 | 10 | 2298.97 | 10 | 2190.76 | 10 | 2087.58 |
| rc104 | 10 | 2166.25 | 9 | 2036.69 | 9 | 1951.27 |
| rc105 | 13 | 2761.81 | 13 | 2579.38 | 12 | 2480.55 |
| rc106 | 11 | 2463.01 | 11 | 2326.02 | 11 | 2260.69 |
| rc107 | 10 | 2321.2 | 10 | 2188.72 | 10 | 2048.04 |
| rc108 | 10 | 2186.46 | 10 | 2056.96 | 9 | 1961.14 |
| **Avg. RC1** | 11.25 | 2451.90 | 11.13 | 2324.33 | 10.63 | 2215.89 |
| rc201 | 4 | 3002.08 | 4 | 2986.84 | 3 | 2524.47 |
| rc202 | 3 | 2525.12 | 3 | 2478.75 | 3 | 2446.29 |
| rc203 | 3 | 2190.09 | 3 | 2158.33 | 2 | 1840.86 |
| rc204 | 3 | 1960.57 | 2 | 1780.69 | 2 | 1681.47 |
| rc205 | 4 | 2868.13 | 3 | 2549.83 | 3 | 2429.20 |
| rc206 | 3 | 2379.7 | 3 | 2284.33 | 3 | 2241.47 |
| rc207 | 3 | 2185.02 | 3 | 2126.88 | 3 | 2016.39 |
| rc208 | 3 | 1988.18 | 2 | 1813.93 | 2 | 1653.68 |
| **Avg. RC2** | 3.25 | 2387.36 | 2.88 | 2272.45 | 2.63 | 2104.23 |
| **CNV/CWD** | 402 | 250864.25 | 391 | 240140.46 | 377 | 228977.63 |

**Table B.7:** Results of TS-DST for the secondary objective of minimizing working duration on the VRPTWDST benchmark sets generated with the *Random* mechanism. The best solution found in 10 runs is reported in columns #Rts. and WD.

| | $\Gamma = 0.9$ | | $\Gamma = 0.7$ | | $\Gamma = 0.5$ | |
|---|---|---|---|---|---|---|
| | **#Rts.** | **WD** | **#Rts.** | **WD** | **#Rts.** | **WD** |
| c101 | 10 | 9383.61 | 10 | 9090.23 | 10 | 8805.46 |
| c102 | 10 | 9307.61 | 10 | 8320.03 | 10 | 7516.19 |
| c103 | 10 | 9256.93 | 10 | 7910.39 | 10 | 7136.65 |
| c104 | 10 | 9083.67 | 10 | 7446.69 | 10 | 6072.88 |
| c105 | 10 | 9111.40 | 10 | 8597.01 | 10 | 8054.07 |
| c106 | 10 | 9117.84 | 10 | 8431.47 | 10 | 7857.17 |
| c107 | 10 | 8936.28 | 10 | 8281.23 | 10 | 7584.93 |
| c108 | 10 | 8935.04 | 10 | 8009.95 | 10 | 7365.69 |
| c109 | 10 | 8917.09 | 10 | 7647.07 | 10 | 6967.86 |
| **Avg. C1** | 10.00 | 9116.61 | 10.00 | 8192.67 | 10.00 | 7484.54 |
| c201 | 3 | 9307.61 | 3 | 9174.63 | 3 | 8982.65 |
| c202 | 3 | 9120.73 | 3 | 8924.81 | 3 | 8785.26 |
| c203 | 3 | 9034.21 | 3 | 7659.56 | 3 | 7140.65 |
| c204 | 3 | 8756.34 | 3 | 7543.72 | 3 | 6607.21 |
| c205 | 3 | 9058.30 | 3 | 8935.06 | 3 | 8667.69 |
| c206 | 3 | 8905.27 | 3 | 8450.89 | 3 | 8153.35 |
| c207 | 3 | 8682.28 | 3 | 8215.25 | 3 | 7978.90 |
| c208 | 3 | 8642.98 | 3 | 8387.71 | 3 | 8234.10 |
| **Avg. C2** | 3.00 | 8938.47 | 3.00 | 8411.45 | 3.00 | 8068.73 |
| r101 | 18 | 3245.84 | 17 | 3059.23 | 15 | 2662.06 |
| r102 | 17 | 2959.66 | 15 | 2609.74 | 14 | 2484.91 |
| r103 | 13 | 2396.29 | 13 | 2261.80 | 11 | 2012.63 |
| r104 | 9 | 1917.87 | 8 | 1653.72 | 8 | 1507.46 |
| r105 | 13 | 2479.85 | 12 | 2209.93 | 11 | 2009.74 |
| r106 | 11 | 2296.11 | 10 | 1913.32 | 9 | 1785.38 |
| r107 | 10 | 2016.21 | 9 | 1738.48 | 8 | 1545.63 |
| r108 | 9 | 1808.44 | 8 | 1571.81 | 8 | 1313.39 |
| r109 | 11 | 2127.14 | 9 | 1960.86 | 9 | 1637.07 |
| r110 | 10 | 2061.07 | 9 | 1774.02 | 8 | 1564.77 |
| r111 | 10 | 2011.88 | 9 | 1749.85 | 8 | 1569.98 |
| r112 | 9 | 1863.12 | 8 | 1586.90 | 8 | 1261.48 |
| **Avg. R1** | 11.67 | 2265.29 | 10.58 | 2007.47 | 9.75 | 1779.54 |
| r201 | 4 | 2913.27 | 3 | 2513.66 | 3 | 2379.94 |
| r202 | 3 | 2340.39 | 3 | 2289.59 | 3 | 2194.64 |
| r203 | 3 | 2184.29 | 2 | 1787.18 | 2 | 1775.96 |
| r204 | 2 | 1740.01 | 2 | 1508.59 | 2 | 1390.29 |
| r205 | 3 | 2130.86 | 2 | 1780.65 | 2 | 1625.54 |
| r206 | 2 | 1950.31 | 2 | 1714.42 | 2 | 1612.96 |
| r207 | 2 | 1757.99 | 2 | 1531.44 | 2 | 1526.41 |
| r208 | 2 | 1613.31 | 2 | 1353.69 | 2 | 1167.44 |
| r209 | 3 | 2115.39 | 2 | 1738.28 | 2 | 1583.08 |
| r210 | 3 | 2109.73 | 2 | 1775.35 | 2 | 1756.09 |
| r211 | 2 | 1741.35 | 2 | 1403.69 | 2 | 1177.58 |
| **Avg. R2** | 2.64 | 2054.26 | 2.18 | 1763.32 | 2.18 | 1653.63 |
| rc101 | 14 | 2770.49 | 13 | 2525.79 | 11 | 2202.74 |
| rc102 | 12 | 2503.39 | 11 | 2236.56 | 10 | 1930.73 |
| rc103 | 10 | 2219.61 | 9 | 2021.24 | 9 | 1819.21 |
| rc104 | 10 | 2077.14 | 9 | 1764.47 | 9 | 1490.23 |
| rc105 | 13 | 2643.50 | 11 | 2348.47 | 10 | 2115.62 |
| rc106 | 11 | 2347.05 | 10 | 2064.39 | 9 | 1825.01 |
| rc107 | 10 | 2223.00 | 9 | 2024.54 | 9 | 1761.23 |
| rc108 | 10 | 2029.24 | 9 | 1774.82 | 9 | 1491.64 |
| **Avg. RC1** | 11.25 | 2351.68 | 10.13 | 2095.04 | 9.50 | 1829.55 |
| rc201 | 4 | 2979.55 | 3 | 2526.42 | 3 | 2422.15 |
| rc202 | 3 | 2491.51 | 3 | 2245.40 | 3 | 2131.40 |
| rc203 | 3 | 2180.34 | 2 | 1824.37 | 2 | 1728.76 |
| rc204 | 2 | 1854.06 | 2 | 1703.58 | 2 | 1535.52 |
| rc205 | 3 | 2612.62 | 3 | 2464.71 | 3 | 2427.34 |
| rc206 | 3 | 2323.68 | 3 | 2236.96 | 2 | 1688.14 |
| rc207 | 3 | 2147.50 | 3 | 2025.24 | 2 | 1640.43 |
| rc208 | 2 | 1903.33 | 2 | 1637.01 | 2 | 1439.47 |
| **Avg. RC2** | 2.88 | 2311.57 | 2.63 | 2082.96 | 2.38 | 1876.65 |
| **CNV/CWD** | 396 | 240643.58 | 367 | 217935.87 | 350 | 201104.76 |

**Table B.8:** Results of TS-DST for the secondary objective of minimizing working duration on the VRPTWDST benchmark sets generated with the *Cluster* mechanism. The best solution found in 10 runs is reported in columns #Rts. and WD.

# Appendix C

# Improved Problem Formulation for the Green VRP

The mathematical model provided in Erdogan and Miller-Hooks (2012) contains several errors:

1. In their Constraints (3), the depot is included to have less than or equal one outgoing arc, although it is clearly allowed to have several.
2. In Constraints (7), they subtract service time instead of adding it.
3. Constraints (12) state that enough fuel is available at each customer to reach a refueling station and then the depot. It should be enough fuel to reach a recharging station or the depot.
4. They do not consider that the refueling visit identified by Constraint (12) is not required to be part of the solution by Constraints (10) and (12). Thus, with their model a customer is only required to have sufficient fuel to reach a service station but is not forced to travel to this station.

We compile a corrected version of their Green VRP (G-VRP) model that we use in Section 5.3.3 to solve the set of small G-VRP benchmark instances with CPLEX. We use the original notation from Erdogan and Miller-Hooks (2012) to formulate the corrected model:

| | |
|---|---|
| $v_0$ | depot |
| $I$ | set of customers |
| $I_0$ | set of customers and the depot, $I_0 = I \cup \{v_0\}$ |
| $F$ | set of alternative fuel station (AFS) vertices |
| $F'$ | set of visits to AFS, which are dummy vertices created from $F$ |
| $F_0$ | set of visits and the depot, $F_0 = F' \cup \{v_0\}$ |

| | |
|---|---|
| $V$ | set of real vertices, $V = \{v_0\} \cup I \cup F$ |
| $V'$ | set of vertices, including dummy vertices, $V' = \{v_0\} \cup I \cup F'$ |
| $Q$ | vehicle fuel capacity |
| $r$ | vehicle fuel consumption rate |
| $y_j$ | remaining fuel level when reaching vertex $j$ |
| $T_{max}$ | maximal duration of each route |
| $t_{ij}$ | travel time between vertices $i$ and $j$ |
| $p_i$ | service time at vertex $i$ |
| $\tau_j$ | time of arrival at vertex $j$ |
| $x_{ij}$ | binary decision variable to indicate if vehicle travels from vertex $i$ to $j$ |

Using this notation, the G-VRP can be defined as follows:

$$\min \sum_{i,j \in V', i \neq j} d_{ij} x_{ij} \tag{C.1}$$

$$\sum_{j \in V', i \neq j} x_{ij} = 1 \quad \forall i \in I \tag{C.2}$$

$$\sum_{j \in V', i \neq j} x_{ij} \leq 1 \quad \forall i \in F' \tag{C.3}$$

$$\sum_{i \in V', i \neq j} x_{ji} - \sum_{i \in V', i \neq j} x_{ij} = 0 \quad \forall j \in V' \tag{C.4}$$

$$\sum_{j \in V' \setminus \{v_0\}} x_{0j} \leq m \tag{C.5}$$

$$\sum_{j \in V' \setminus \{v_0\}} x_{j0} \leq m \tag{C.6}$$

$$0 \leq \tau_i + (t_{ij} + p_i) x_{ij} - T_{max}(1 - x_{ij}) \leq \tau_j \quad \forall i \in V', \forall j \in V' \setminus \{v_0\} \text{ and } i \neq j \tag{C.7}$$

$$\tau_j + t_{j0} + p_j \leq T_{max} \quad \forall j \in V' \setminus \{v_0\} \tag{C.8}$$

$$0 \leq \tau_0 \leq T_{max} \tag{C.9}$$

$$0 \leq y_j \leq y_i - (r \cdot d_{ij}) x_{ij} + Q(1 - x_{ij}) \quad \forall j \in I, \forall i \in V', i \neq j \tag{C.10}$$

$$0 \leq y_i - (r \cdot d_{ij}) x_{ij} \quad \forall j \in F_0, \forall i \in V', i \neq j \tag{C.11}$$

$$y_j = Q \quad \forall j \in F_0 \tag{C.12}$$

$$x_{ij} \in \{0, 1\} \quad \forall i, j \in V' \tag{C.13}$$

Also note that the G-VRP benchmark instance files proposed in Erdogan and Miller-Hooks (2012) contain coordinates for each vertex $v$ given in longitude and latitude. They have to be converted to distances between vertices $v$ and $w$ by applying the haversine formula. The problem instances whose solutions are presented in their paper are obtained assuming an erroneous earth radius of 4182.449 miles, instead of the actual earth radius of 3963.191 miles.

# Curriculum Vitae

## Michael Schneider

---

### Kontaktdaten


| | |
|---|---|
| Name | Michael David Schneider |
| Dienstanschrift | Lehrstuhl für Wirtschaftsinformatik und Operations Research, Erwin-Schrödinger-Straße 42, 67653 Kaiserslautern |
| Email | schneider@wiwi.uni-kl.de |


---

### Wissenschaftlicher Werdegang

| | |
|---|---|
| seit 01/2009 | Wissenschaftlicher Mitarbeiter am Lehrstuhl für Wirtschaftsinformatik und Operations Research, Technische Universität Kaiserslautern. |
| 09/2007 | Diplom in Betriebwirtschaftslehre, Universität Mannheim. |
| 06/2007 | Diplom in Informatik, Universität Mannheim. |
| 06/1997 | Abitur, Rosensteingymnasium Heubach. |