The Electric Fleet Size and Mix Vehicle Routing Problem with Time Windows and Recharging Stations

Hiermann, Gerhard; Puchinger, Jakob; Røpke, Stefan; Hartl, Richard F.

Published in:
European Journal of Operational Research

Link to article, DOI:
10.1016/j.ejor.2016.01.038

Publication date:
2016

Document Version
Peer reviewed version

The Electric Fleet Size and Mix Vehicle Routing Problem with Time Windows and Recharging Stations

Gerhard Hiermann[a,*], Jakob Puchinger[a,d,e], Stefan Ropke[b], Richard F. Hartl[c]

[a] *Mobility Department, Dynamic Transportation Systems, AIT Austrian Institute of Technology GmbH*
[b] *DTU Management Engineering, Technical University of Denmark*
[c] *Department of Business Administration, University of Vienna, Austria*
[d] *Laboratoire Genie Industriel, CentraleSupélec, Université Paris-Saclay, France*
[e] *Institut de Recherche Technologique SystemX, Palaiseau, France*

Abstract

Due to new regulations and further technological progress in the field of electric vehicles, the research community faces the new challenge of incorporating the electric energy based restrictions into vehicle routing problems. One of these restrictions is the limited battery capacity which makes detours to recharging stations necessary, thus requiring efficient tour planning mechanisms in order to sustain the competitiveness of electric vehicles compared to conventional vehicles. We introduce the Electric Fleet Size and Mix Vehicle Routing Problem with Time Windows and recharging stations (E-FSMFTW) to model decisions to be made with regards to fleet composition and the actual vehicle routes including the choice of recharging times and locations. The available vehicle types differ in their transport capacity, battery size and acquisition cost. Furthermore, we consider time windows at customer locations, which is a common and important constraint in real-world routing and planning problems. We solve this problem by means of branch-and-price as well as proposing a hybrid heuristic, which combines an Adaptive Large Neighbourhood Search with an embedded local search and labelling procedure for intensification. By solving a newly created set of benchmark instances for the E-FSMFTW and the existing single vehicle type benchmark using an exact method as well, we show the effectiveness of the proposed approach.

Keywords: Heterogenous Fleet, Electric Vehicle Routing, Efficient constraint handling

1. Introduction

Current research in sustainable and energy efficient mobility is strongly motivated by increasing concerns about climate change and rising green house gas emissions. The introduction of electrically powered vehicles is one of the major directions taken in order to address these concerns. Pure battery electric vehicles, as studied in this work, are only powered by an electric engine, using the energy stored in a rechargeable battery. One of the main operational challenges in transport applications is their limited range and long recharging times. Besides acquisition cost, the acceptance of electric vehicles in the transportation business will strongly depend on methods alleviating the range and recharging limitations. Selecting the right vehicles for specific transport requirements while minimizing overall cost is therefore of crucial importance.

Companies have a variety of available electric vehicles with certain variability concerning range, payload, and price to consider (see e.g., AustriaTech (2010)). Especially with electric vehicles, acquisition cost play an important role in economic considerations of fleet managers. This means that larger vehicles might be able to serve the transportation needs without recharging operations. But the difference in price, using smaller vehicles in the fleet mix could reduce the overall cost. However, smaller vehicles have a smaller capacity and

*Corresponding author
Email addresses: gerhard.hiermann.fl@ait.ac.at (Gerhard Hiermann), jakob.puchinger@irt-systemx.fr (Jakob Puchinger), ropke@dtu.dk (Stefan Ropke), richard.hartl@univie.ac.at (Richard F. Hartl)

battery size, thus need to be recharged in order to serve longer tours, which in turn takes time. It is to be expected that smaller and cheaper vehicles will be used alongside larger vehicles depending on the typical customer distribution over the urban area. In this work we will show that a fleet composed of different vehicle types can indeed be beneficial.

We propose to address this task by introducing a new optimization problem, the so-called *Electric Fleet Size and Mix Vehicle Routing Problem with Time Windows and Recharging Stations* (E-FSMFTW). It combines and subsumes the well known Fleet Size Mix Vehicle Routing Problem with Time Windows (FSMFTW) and the recently defined Electric Vehicle Routing Problem with Time Windows and Recharging Stations (E-VRPTW).

1.1. Related Work

Solving and optimizing problems involving tour assignments of vehicles is a well known and well studied field of research, known as *Vehicle Routing Problem* (VRP) (Toth and Vigo, 2014). The basic problem variant is the *Capacitated VRP* (CVRP), where each customer has a given demand that has to be satisfied, with respect to a maximum vehicle capacity. The *VRP with Time Windows* (VRPTW) extends the CVRP by adding time windows to the depot and the customers. A survey on this problem class is provided in Bräysy and Gendreau (2005a,b); Toth and Vigo (2014).

The classical VRP has been extended in various ways to account for additional real world aspects. These more complicated problems are often called "rich VRPs" or "multi-attribute VRPs"; see Vidal et al. (2013a). Our research combines two such streams of research, (1) the use of electrical/zero-emission vehicles in "green" VRPs and (2) the analysis of VRPs with heterogeneous fleet. The first stream is part of research in the field of green logistics. A general survey on the subject is provided in Sbihi and Eglese (2010) and Dekker et al. (2011). A recent discussion on electric vehicles for distribution goods is provided by Pelletier et al. (2015). For a summary of the work on non-electric "green" VRPs, minimizing for example emissions by speed optimization, we refer to Toth and Vigo (2014) as well.

Erdoğan and Miller-Hooks (2012) started by extending the CVRP to the *Green VRP* where tours for Alternative Fuel Vehicles are optimized. The uneven distribution of *Alternative Fuelling Stations* (AFS) leads to the problem of deciding when a vehicle has to visit AFS during its tour (possibly multiple times) in order to minimize the distance travelled but avert to run out of fuel. Two construction heuristics are presented: A *Clarke and Wright savings heuristic* (Clarke and Wright, 1964) that is extended to include AFS nodes during the merge process, and a Density Based Clustering exploiting spatial properties of the problem. Both approaches terminate after creating a solution containing a feasible set of routes, which is then improved by means of local search. The methods were tested on a randomly generated test set as well as a real world case study considering up to 500 (randomly located) customers and 21 existing AFS. For smaller random instances the presented methods obtain solutions that are, on average, less than 10% worse than the best known solutions obtained with CPLEX.

Schneider et al. (2014) adapted the Green VRP to electric vehicles (EV) and added time window constraints, introducing the *Electric VRPTW with Recharging Stations* (E-VRPTW). The aim is to find tours satisfying charge constraints (the state of charge may never fall below zero) and time window constraints. The recharging process complicates the time calculations, since the required recharging time depends on the state of the charge. The problem is solved by a *Variable Neighbourhood Search* (VNS) approach using *Tabu Search* (TS) as local optimization technique. The proposed approach was tested on a new benchmark set based on the traditional Solomon instances for the VRPTW that have been extended with recharging stations as well as the instances of Erdoğan and Miller-Hooks (2012) for the GreenVRP. In addition, the presented approach has been adapted to solve instances of the related *Multi Depot VRP with Inter-depot routes* (MDVRPI) (Crevier et al., 2007; Tarantilis et al., 2008), where vehicles can visit depots between customers to restock in order satisfy the demand of the customers. The presented methods were able to improve upon previous results for the GreenVRP and new best solutions have been obtained for the MDVRPI. Based on the proposed benchmark set, smaller instances allowing a direct comparison with CPLEX have been created. The comparison shows, that the VNS/TS approach is able to find optimal solutions (where known).

A different electric vehicle routing problem has been presented in Conrad and Figliozzi (2011), the so-called *Recharging VRP* (RVRP). Instead of using dedicated recharging stations the authors assume that a set

of customers provides the option to recharge at their location. The EV can perform a recharging operation to a certain percentage of the maximum capacity, a so-called 'quick charge'. It is assumed that this takes a fixed amount of time, independent of the current level of charge. Recharging operations and service at the customer can be performed simultaneously. Different problem instances are proposed and solved with various parameter settings using a modified iterative construction and improvement algorithm. The paper focuses on the analysis of the instance parameters and their contribution to the solutions obtained to generate meaningful solution bounds for the average tour distance.

Recently, Goeke and Schneider (2015) studied a rich fleet mixing problem where not only conventional and electric vehicles are considered, but also load-dependent energy consumption based on a real-world network. They developed an adaptive large neighbourhood search approach with an embedded local search procedure using an surrogate function to evaluate changes efficiently. In parallel to our work, Desaulniers et al. (2014) tackled the original E-VRPTW as well as variations concerning variable recharging or allowing a single stop. They proposed a branch-price-and-cut algorithm with efficient labelling and cutting procedures applicable for all studied variants. In their computational results, they showed that their approach is able to solve instances with up to 100 customers, while some instances with 50 customers cannot be solved to optimality.

The second stream of research related to our work is that of VRPs with heterogeneous fleet. The *Mixed Fleet* or *Heterogenous VRP* considers problems where different types of vehicles are available. It was first introduced in Golden et al. (1984). Baldacci et al. (2008) identifies five major subclasses differing in the number of vehicles available (limited, unlimited), whether a fixed cost per vehicle is considered or not and if the routing cost depend on the vehicle type. The original formulation by Golden et al. (1984) considers an unlimited number of vehicles with fixed acquisition costs and vehicle type independent routing costs, which is classified as a *Fleet Size and Mix VRP with Fixed costs* (FSMF) (Baldacci et al., 2008; Toth and Vigo, 2014).

Liu and Shen (1999) reformulate the FSMF to consider time windows, creating the *Fleet Size and Mix Vehicle Routing Problem with Time Windows* (FSMFTW). The so-called *En Route* time, i.e., the time between departing from and returning to the depot minus the cumulative service time at the customers in the respective route is considered as routing cost. The proposed approach was applied to a new benchmark set based on the well known Solomon instances for the VRPTW. This benchmark extends the 56 VRPTW instances by providing three classes of vehicle type settings (A,B,C) varying from 3 to 6 vehicle types with different cost and capacity margins, resulting in 168 problem instances in total.

Bräysy et al. (2008a) propose a three phase *multi-start deterministic annealing* metaheuristic (MSDA) to solve the FSMFTW. A threshold acceptance criterion is used where the maximum threshold of accepting a worse solution is reduced after a number of iteration until no worsening is allowed. The solution itself is created using a systematic and deterministic multi-phase approach, starting with a modified *Clarke and Wright savings heuristic*, followed by a route elimination procedure and a systematic local search where three operators are applied every single, second or third iteration. The proposed algorithm shows a very good performance when run for a similar amount of time compared to previous approaches. Furthermore, with longer run-times new best solutions for almost every instance are obtained.

An *Adaptive Memory Program* (AMP) was proposed by Repoussis and Tarantilis (2010). A memory of good solution features (route assignments and orderings) is maintained within an *Iterated Local Search* (ILS) using TS as local improvement procedure. A specialized construction heuristic uses the information of the memory to create new solutions for the following run of the ILS which is repeated until the predefined maximum runtime has been reached. The approach was able to find new best solutions for over a half of the instances and found comparable results for the others.

Most solution approaches in the literature use heuristic methods. However, a few exact approaches were also proposed, most notably by Baldacci et al. (2010). They presented a general exact branch-and-cut-and-price framework to solve the FSMFTW among others. Their approach was able to solve some larger benchmark instances to optimality for the first time, however, there are still a number of instances left.

Recently, Vidal et al. (2013b) presented a hybrid evolutionary algorithm using population management mechanisms and a generalized solution representation with modular evaluation which is able to solve a large class of VRPs, including the FSMFTW. The approach was able to find new best solutions for many instances

in a reasonable amount of time.

Further noteworthy research for the FSMFTW can be found in Dullaert et al. (2002), Dell'Amico et al. (2007), Paraskevopoulos et al. (2008) and Bräysy et al. (2008b).

Recently, Çağri Koç et al. (2014) presented a related heterogeneous fleet problem, the *Fleet Size and Mix Pollution-Routing Problem* (FSMPRP), focussing on conventional vehicles only. The FSMPRP contains additional operational details, such as adaptation of travel speed to minimize fuel consumption and CO2 emissions.

1.2. Contributions of this article

We combine the streams of research on the EVRPTW and the FSMFTW and introduce a new vehicle routing problem: the Electric Fleet Size and Mix Vehicle Routing Problem with Time Windows and recharging stations (E-FSMFTW). For this new problem, we introduce a new set of benchmark instances for our computational experiments as well as for future research in the field.

To define the problem precisely, we provide a mathematical formulation as a MIP model and use a state-of-the-art branch-and-price algorithm designed for the VRPTW to solve a set of smaller instances to have benchmarks for heuristic approaches. In addition we applied it to larger instances as well to obtain good lower bounds for comparison. In order to tackle problem instances of realistic size, we propose a metaheuristic approach based on Adaptive Large Neighbourhood Search (ALNS) with embedded local search and labelling procedures. We describe a sophisticated move evaluation process, which enables us to calculate the change in the objective value using common moves in constant time. We propose to compute optimal positions of recharging stations in a post-local-search step using a variant of the labelling procedure applied in the branch-and-price.

The presented computational experiments show that our approach is able to find optimal solutions for small instances and that high quality feasible solutions can be found for the larger instances. Furthermore, a sensitivity analysis conducted on additional experiments is included, presenting and discussing the impact of the fleet mix and the recharging constraint in our problem.

Overview of the article

In Section 2, a mixed-integer linear programming formulation is given alongside detailed formal notation. Section 3 provides a general insight into the branch-and-price method, and describes the bi-directional labelling algorithm in detail. Section 4 describes the modules of our ALNS approach in detail. Results of our experiments for the new benchmark are presented in Section 5. Section 6 provides a short summary and a conclusion of our work.

2. Problem Description and Model

The *Electric Fleet Size and Mix Vehicle Routing Problem with Time Windows and recharging stations* (E-FSMFTW) consists of finding admissible tours for vehicles of different types such that every customer is covered by exactly one tour while minimizing acquisition costs and the total distance travelled. Our formulation is based on Bräysy et al. (2008a) for the FSMVRPTW and Schneider et al. (2014) for the E-VRPTW. As for the E-VRPTW, a vehicle is also assumed to always recharge to full capacity every time it visits a recharging station.

2.1. Mixed Integer Programming Model

An instance of the E-FSMFTW consists of a set of customers C, a set of recharging stations F, a depot node and k different vehicle types. N is defined as a set of nodes $N = C \cup F'$ consisting of a set of customers C and a set of dummy nodes F' representing the recharging stations of F multiple times. These copies of stations are needed in the model to account for multiple visits at the original one (even for the same vehicle) . u_0 and u_{n+1} are used to represent the start and end depot nodes respectively. N_0 (N_{n+1}) denotes the set of nodes with the start depot node (end depot node) whereas $N_{0,n+1}$ addresses $N \cup \{u_0, u_{n+1}\}$. The same

notation is used for C and F'. A binary variable x_{ij}^k is used for indicating whether a vehicle of type k visits node j after visiting node i.

The maximum capacity of vehicle type k is defined in Q^k, and p_i is the capacity demand of node i. Variables q_i^k hold the current load of a vehicle of type k in node i. Similar to the load capacity, Y^k is the maximum energy capacity of vehicle type k and y_i^k is the variable holding the current energy level of a vehicle of type k in node i. r^k represents the assumed energy consumption per distance unit travelled of a vehicle of type k and g^k corresponds to the recharging time per energy unit.

For each node i, a range $[e_i, l_i]$ is defined representing the time windows. The service time of each node is represented by s_i and the actual start of service time is stored in variable τ_i. The distance and the travel time between two nodes i and j are denoted as d_{ij} and t_{ij} respectively.

Furthermore, f^k is used for the acquisition cost of a vehicle of type k, whereas d_{ij} indicate the distance between node i and j.

The E-FSMFTW can be modelled as follows:

$$min \quad \sum_{k\in V}\sum_{j\in N} f^k x_{0j}^k + \sum_{k\in V}\sum_{i\in N_0, j\in N_{n+1}, i\neq j} c_{ij}^k x_{ij}^k \tag{2.1}$$

$$s.t. \quad \sum_{k\in V}\sum_{j\in N_{n+1}, i\neq j} x_{ij}^k = 1 \qquad \forall i \in C \tag{2.2}$$

$$\sum_{k\in V}\sum_{j\in N_{n+1}, i\neq j} x_{ij}^k \leq 1 \qquad \forall i \in F' \tag{2.3}$$

$$\sum_{i\in N_{n+1}, i\neq j} x_{ji}^k - \sum_{i\in N_0, i\neq j} x_{ij}^k = 0 \qquad \forall j \in N, \forall k \in V \tag{2.4}$$

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

$$\tau_i + (t_{ij} + s_i)x_{ij}^k - l_0(1 - x_{ij}^k) \leq \tau_j \qquad \forall k \in V, \forall i \in C_0, \forall j \in N_{n+1}, i \neq j \tag{2.6}$$

$$\tau_i + t_{ij}x_{ij}^k + g^k(Y^k - y_i^k) - (l_0 + g^k Y^k)(1 - x_{ij}^k) \leq \tau_j \qquad \forall k \in V, \forall i \in F', \forall j \in N_{n+1}, i \neq j \tag{2.7}$$

$$q_j^k \leq q_i^k - p_i x_{ij}^k + Q^k(1 - x_{ij}^k) \qquad \forall k \in V, \forall i \in N_0, \forall j \in N_{n+1}, i \neq j \tag{2.8}$$

$$0 \leq q_j^k \leq Q^k \qquad \forall k \in V, \forall j \in N_{0,n+1} \tag{2.9}$$

$$0 \leq y_j^k \leq y_i^k - (r^k d_{ij})x_{ij}^k + Y^k(1 - x_{ij}^k) \qquad \forall k \in V, \forall i \in C, \forall j \in N_{n+1}, i \neq j \tag{2.10}$$

$$0 \leq y_j^k \leq Y^k - (r^k d_{ij})x_{ij}^k \qquad \forall k \in V, \forall i \in F_0', \forall j \in N_{n+1}, i \neq j \tag{2.11}$$

$$y_0^k = Y^k \qquad \forall k \in V \tag{2.12}$$

$$x_{ij}^k \in \{0, 1\} \qquad i \in N_0, j \in N_{n+1}, i \neq j, \forall k \in V \tag{2.13}$$

The problem is a minimization problem with an objective function (2.1) consisting of two parts. The first part is the sum of the costs of all vehicles used, i.e., if a vehicle is driving from the depot to any other node than the depot (indicated by a value greater than zero) the corresponding acquisition cost f^k is added. The second part counts the total travelling cost or each car – in the case of the E-FSMFTW, the distance travelled, i.e., $c_{ij}^k = d_{ij}$. Equation (2.2) ensures that every customer is visited by any vehicle exactly once while (2.3) covers the fact that a recharge station does not need to be used in a solution at all. Furthermore, each dummy node representing a recharge station is restricted to be visited at most once by any vehicle to ensure correct assignments to node specific variables like the start of service time τ_i. Equation (2.13) forces the value of x_{ij}^k to be either zero or one.

The timing constraints are covered by (2.5) - (2.7). The constraint described in (2.5) ensures that the start of service time τ_i has to be inside the time window $[e_i, l_i]$ of node i. Constraint (2.6) ensures that the starting time of the next node τ_j has to consider the start time τ_i plus the service time s_i of the previous node i in addition to the travel time t_{ij} in case that node i is a customer node or the start depot. If the previous node is a recharging station, i.e., $i \in F'$, constraint (2.7) considers the recharge time instead of a

service time. The recharging time depends on the remaining energy when arriving at the recharging station y_i^k, the maximum energy capacity of the vehicle Y^k and the recharging rate g^k.

To ensure that the demand of the customers can be fulfilled by the assigned vehicle k, constraint (2.8) ensures that the load of the vehicle in the next node q_j^k depend on the load of the previous node q_i^k plus the demand p_i. Equation (2.9) restricts the load q_i^k never to exceed the maximum capacity Q^k of the vehicle k as well as ensures a positive value.

Constraint (2.10) restricts the current available energy y_j^k to be smaller than the previous y_i^k and to consider the energy consumption per km/mile r^k when travelling from a customer node $i \in C$ to any other node $j \in N_{n+1}$. As it is assumed, that a vehicle is recharged to the maximum capacity when visiting a recharge station, constraint (2.11) ensures that the current load is assumed to be the maximum minus the consumed amount. Both constraints also ensure that the available energy is always positive in any node $i \in N_{0,n+1}$.

2.2. Set Partitioning Formulation

To formulate the problem as a generalized set partitioning problem, let Ω_k denote the set of all feasible routes of vehicle type k that satisfy constraints (2.2)-(2.13). Let c_r^k be the cost of the route $r \in \Omega_k$ and a_{ir}^k be the number of times node $i \in C$. Furthermore, let λ_r^k be a binary variable equal to 1 if and only if route $r \in \Omega_k$ is used in the solution. The E-FSMFTW can then be formulated as follows:

$$min \quad \sum_{k \in V} \sum_{r \in \Omega_k} c_r^k \lambda_r^k \tag{2.14}$$

$$\sum_{k \in V} \sum_{r \in \Omega_k} a_{ir}^k \lambda_r^k = 1, \forall i \in C \tag{2.15}$$

$$\sum_{r \in \Omega_k} a_{ir}^k \leq U^k, \forall k \in V \tag{2.16}$$

$$\lambda_r^k \in \{0,1\}, \forall r \in \Omega_k, \forall k \in V \tag{2.17}$$

The objective function (2.14) minimizes the cost of the selected routes, whereas constraint (2.15) ensures that every request is served once.

As the number of routes is usually very large, enumerating all columns may not be possible. In order to solve the LP relaxation of such a problem, a so-called *restricted master problem* (Desrosiers and Lübbecke, 2010) is used whose columns are extended as needed using column generation. Such a restricted master problem can be optained by replacing Ω_k with a set of restricted routes Ω_k' as well as replacing (2.17) with $\lambda_r^k \geq 0, r \in \Omega_k'$.

Additional columns with negative reduced cost to the restricted master problem are created by solving the corresponding *pricing problems* (one per vehicle type k):

$$min \quad f^k + \sum_{i \in N_0, j \in N_{n+1}, i \neq j} c_{ij}^k x_{ij}^k - \sum_{i \in C, j \in N_{n+1}} x_{ij}^k \pi_i \tag{2.18}$$

subject to constraints (2.2)-(2.13), where π_i is the negative reduced cost associated with node i. This type of pricing problem is also called a *Shortest Path Problem with Resource Constraints* (SPPRC). The next section will describe how this general problem is adapted for the E-FSMFTW and how it is solved.

3. Branch and Price Algorithm

The set partitioning formulation described in the previous section is solved using a branch-and-price algorithm. The algorithm is similar to existing branch-and-price algorithms for the VRPTW (see e.g. Desaulniers et al. (2008)), but differs in the algorithm for solving the pricing problem since it has to incorporate

the constraints related to charging. The procedure for solving the pricing problem is described in details in Section 3.1.

In order to obtain integer solutions, the algorithm branches on the number of vehicles as long as this number is fractional. Once the number of vehicles used is integer, the algorithm switches to branching on the original variables. For a pair of nodes $i, j \in N_0, i < j$ the algorithm computes $\sum_{k \in V}(x_{ij}^k + x_{ji}^k)$ as well as $\sum_{k \in V} x_{ij}^k$ and $\sum_{k \in V} x_{ji}^k$,and ensures that this number is integer. It also ensures that $x_{i,n+1}^k$ is integer for all $i \in N$ and all $k \in V$. When both conditions are satisfied an integer solution is obtained

This section will now focus on describing how the new pricing problem - called the *Elementary Shortest Path Problem with Time Windows and Recharging Stations* (ESPPTWRS) - is solved using a bi-directional labelling algorithm. The SPPRC in context of vehicle routing problems is primarily solved using dynamic programming approaches, so-called labelling algorithms. An overview of SPPRC and solution techniques are provided in Irnich and Desaulniers (2005).

3.1. Labelling for the ESPPTWRS

In this work, the pricing problem is solved using a bi-directional labelling algorithm. This variant of a labelling algorithm utilizes a forward (labels are extended by adding an additional visit at the end of the current path) and a backward (extension by adding a visit before to the current path) labelling procedure. Labels of both directions are extended in turn until a termination criterion is reached followed by a merging process of the remaining non-dominated labels.

The procedures follows the general steps (see Irnich and Desaulniers (2005)), but differ in their label definition, extension and dominance criterion. The description of the labelling algorithm uses a function style notation to address the value of a field of a specific label, i.e., $node(L)$ refers to the node of label L, the resources are referred by $R_1(L), \ldots, R_\gamma(L)$, and $prev(L)$, $\mathcal{V}(L)$ for the pointer to the previous label, and the bit set of unreachable nodes respectively. In the following, the forward and backward cases are covered first before describing the merging procedure resulting in the final bi-directional algorithm.

3.1.1. Forward Labelling

The forward variant extends the labels in such a way, that the next node is added at the end of the current path. To ensure feasibility and perform dominance checks efficiently, four resources are used: the current cost of the path (R_c), the load after visiting the current node (R_q), the earliest begin of service time at the current node (R_t) and the energy usage since the last visit of a recharging station (R_y). The forward label denoted as $L^F = \{node, (R_c, R_q, R_t, R_y), prev, \mathcal{V}\}$

Label extension. An extension of a label L^F to a node j, where $(node(L^F), j) \in A$, is feasible if following constraints are fulfilled:

$$R_q(L^F) + p_j \leq Q^k, \quad R_t(L^F) + t_{node(L^F)j} \leq l_j, \quad R_y(L^F) + d_{node(L^F)j} \cdot r \leq Y^k \tag{3.1}$$

The constraints in (3.1) only ensure the feasibility of the next extension, but to reduce the number of labels created – and thus reducing the overall computational time – more restrictive constraints are introduced. These harness the requirement to return to the depot and the possibility to recharge along the way.

$$l_{n+1} \geq max(R_t(L^F) + t_{node(L^F)j}, e_j) + t_{jn+1} + \begin{cases} s_j & j \in C \\ g(R_y(L^F) + d_{node(L^F)j} \cdot r) & j \in F \end{cases} \tag{3.2}$$

$$R_y(L^F) + d_{node(L^F)j} \geq min(d_{jn+1}, d_{jf}) \cdot r \quad f = min\{f | d_{fj}, f \in F\}, j \in C \tag{3.3}$$

With equation (3.2) it is ensured that the depot can be reached in time to satisfy the latest arrival time, if the label L^F is extended to node j. Apart from the travel time and the possible waiting time at node j, the time spent in the node has to be accounted for too. There are two possibilities, depending on whether node j being a customer, where a service time is present, or a recharging station. Due to the fixed recharging

policy of the problem – always recharging to full capacity – the time spent recharging is calculated using the information of label L^F. By satisfying equation (3.3) it is ensured that a recharging station (or the depot) can be reached after visiting customer node j.

If (3.1)-(3.3) are valid for an extension of L^F to node j, then a new label L^F_{new} is created as follows:

$$node(L^F_{new}) = j \tag{3.4}$$

$$R_c(L^F_{new}) = R_c(L^F) + \hat{c}_{node(L^F)j} \tag{3.5}$$

$$R_q(L^F_{new}) = \begin{cases} R_q + p_j & \text{if } j \in C \\ R_q & \text{if } j \in F \end{cases} \tag{3.6}$$

$$R_t(L^F_{new}) = \max(R_t(L^F) + t_{node(L^F)j}, e_j) \begin{cases} + s_j & \text{if } j \in C \\ + g(R_y(L^F) + d_{node(L^F)j} \cdot r) & \text{if } j \in F \end{cases} \tag{3.7}$$

$$R_y(L^F_{new}) = \begin{cases} R_y(L^F) + d_{node(L^F)j} \cdot r & \text{if } j \in C \\ 0 & \text{if } j \in F \end{cases} \tag{3.8}$$

$$\mathcal{V}(L^F_{new}) = \mathcal{V}(L^F) \cup \{j\} \cup unreachables(L^F_{new}) \tag{3.9}$$

The equations in (3.4)-(3.8) set the last visited node, cost, load, time and consumed energy of label L^F_{new}. The cost resource uses the modified cost $\hat{c}_{ij}$, which incorporates the dual value of the current pricing problem. The set of unreachable nodes is updated in equation (3.9) by adding the new node j and – substituted by the function $unreachables(L^F)$ – all nodes $u, (node(L^F), u) \in A$, violating the capacity or time constraint in (3.1) and (3.2). The energy constraint is not part of the check for unreachable nodes, as $R_y(L^F)$ can be replenished, i.e., re-set to 0, see equation(3.8). Therefore a node u might only be temporarily unreachable, and might lead to an undesired and invalid domination.

Dominance criterion. To reduce the number of labels extended, a label elimination method based on *dominance checks* (as described in Feillet et al. (2004)) is used. By identifying and removing dominated labels, the number of labels can be reduced significantly which directly impacts the overall runtime of the algorithm.

For the ESPPTWRS, using forward labelling, the criterion is as follows: a forward label L^F_1 dominates L^F_2 if

$$\begin{gathered} node(L^F_1) = node(L^F_2), R_q(L^F_1) \leq R_q(L^F_2), R_t(L^F_1) \leq R_t(L^F_2), R_y(L^F_1) \geq R_y(L^F_2), \\ \mathcal{V}(L^F_1) \subseteq \mathcal{V}(L^F_2) \end{gathered} \tag{3.10}$$

The proof of this criterion is an extension of Feillet et al. (2004). Let L^F_1 and L^F_2 be two forward labels with $i = node(L^F_1) = node(L^F_2)$ where L^F_1 dominates L^F_2 according to criterion (3.10). Let j be a node L^F_1 can be extended to. If L^F_2 cannot be extended to j then it is clearly dominated. Otherwise, based on cost, time, and energy, L^F_1 dominates L^F_2 after an extension as well, if they are extended to node j directly. Let assume a recharge at station f is needed in order to reach j. After recharging at station f, both labels have the same amount of energy consumed when arriving at node j, i.e., $d_{fj} \cdot r$, thus the criterion is still fulfilled with respect to the energy resource. The time resource calculation contains the starting time, travel times from i to f and further to j and the recharging time at f. As the travel time is equal for both labels, it is omitted in the further consideration. Due to the direct correlation between the energy consumed and the recharging time needed, the following can be deduced: $R_t(L^F_1) + g(R_y(L^F_1) + d_{if} \cdot r) \leq R_t(L^F_2) + g(R_y(L^F_2) + d_{if} \cdot r)$. This is true as $R_t(L^F_1) \leq R_t(L^F_2)$ and $R_y(L^F_1) \leq R_y(L^F_2)$ holds as assumed previously. As our method to update the set of unreachable nodes do not consider the energy constraint, the remaining proof follows directly from Feillet et al. (2004).

3.1.2. Backward Labelling

For the backward case, the next node in the extension is added at the beginning of the current path. In addition to the resources used in the forward label, three additional resources are maintained: the minimum

(R_{rs}) and the maximum ($R_{\overline{rs}}$) time available to recharge at the first recharging station in the current path; and the earliest begin of service $R_{\overline{t}}$, with the recharging time for the current energy consumption already included. In contrast to the forward label, the time resource R_t denote the the latest begin of the route. The backward label denoted as $L^B = \{node, (R_c, R_q, R_t, R_y, R_{\overline{t}}, R_{rs}, R_{\overline{rs}}), prev, \mathcal{V}\}$

Label extension. An extension of a label L^B to a node i, where $(i, node(L^B)) \in A$, is valid if following constraints are fulfilled:

$$p_i + R_q(L^B) \leq Q^k, \quad R_t(L^B) - s_i - t_{inode(L^B)} \geq e_i, \quad d_{inode(L^B)} \cdot r + R_y(L^B) \leq Y^k \tag{3.11}$$

$$\min(R_t(L^B) - s_i - t_{inode(L^B)} - e_i + R_{rs}(L^B), R_{\overline{rs}}(L^B)) \geq g(R_y(L^B) + d_{inode(L^B)} \cdot r) \tag{3.12}$$

Constraints (3.11) ensures the feasibility of the next extension on a basic level by checking the capacity, time window and energy constraint similar to the forward label constraints. The time needed to recharge at the next possible recharging station is not yet fully known when visiting a customer. Therefore constraint (3.12) ensures that after adding node i, a recharge of the energy consumed (rhs) is between the minimal and maximal recharging time available. Further restrictive constraints are used to reduce the number of labels created, i.e., equation (3.3) and

$$e_0 + t_{0i} \leq \min(l_i, R_t(L^B) - t_{inode(L^B)} \begin{cases} -s_i) & j \in C \\ -\Delta rs - g(d_{0i} \cdot r)) & j \in F \end{cases}, \tag{3.13}$$

where $\Delta rs = \max(0, g(R_y(L^B) + d_{inode(L^B)} \cdot r) - R_{rs}(L^B))$ represents the additional time needed for the recharging operation. Notice that the maximum time will never exceeded due to constraint (3.12). With (3.13), extensions to a node i are prohibited, if the depot cannot be reached after that visit.

If (3.11)-(3.13) and (3.3) are valid for an extension of L^B to node i, then a new label L^B_{new} is created as follows:

$$node(L^B_{new}) = i \tag{3.14}$$

$$R_c(L^B_{new}) = R_c(L^B) + \hat{c}_{inode(L^B)} \tag{3.15}$$

$$R_q(L^B_{new}) = \begin{cases} R_q + p_j & \text{if } i \in C \\ R_q & \text{if } i \in F \end{cases} \tag{3.16}$$

$$R_t(L^B_{new}) = \min(l_i, R_t(L^B) - t_{inode(L^B)} \begin{cases} -s_i) & i \in C \\ -\Delta rs) & i \in F \end{cases} \tag{3.17}$$

$$R_y(L^B_{new}) = \begin{cases} R_y(L^B) + d_{inode(L^B)} \cdot r & \text{if } i \in C \\ 0 & \text{if } i \in F \end{cases} \tag{3.18}$$

$$R_{\overline{t}}(L^B_{new}) = \min(l_i, R_t(L^B) - t_{inode(L^B)} - \Delta rs) \tag{3.19}$$

$$R_{rs}(L^B_{new}) = \begin{cases} \min(R_{\overline{t}}(L^B_{new}), Y^k \cdot g, R_{rs}(L^B) + \Delta_{WT}) & \text{if } i \in C \\ 0 & \text{if } i \in F \end{cases} \tag{3.20}$$

$$R_{\overline{rs}}(L^B_{new}) = \begin{cases} R_{\overline{t}}(L^B_{new}) & \text{if } i \in C \\ 0 & \text{if } i \in F \end{cases} \tag{3.21}$$

$$\mathcal{V}(L^B_{new}) = \mathcal{V}(L^B) \cup \{i\} \cup unreachables(L^F_{new}) \tag{3.22}$$

where $\Delta_{WT} = \max(0, l_i - R_t(L^B) - t_{inode(L^B)} - s_i)$ denotes the amount of unavoidable waiting time at customer i and $\Delta rs = \max(0, g(R_y(L^B) + d_{inode(L^B)} \cdot r) - R_{rs}(L^B))$ the additional recharging time as before.

Again, equations (3.14)-(3.18) set the last visited node, cost, load, time and remaining energy of label L^B_{new}. Equation (3.19) stores the earliest begin of service with the recharging time of the current energy consumption included for later use in the dominance checks. With (3.20) and (3.21) the time spent at least at the next recharging station and the maximum time available are updated. Equation (3.22) updates the set of unreachable nodes.

Dominance criterion. A backward label L_1^B dominates L_2^B if

$$\begin{gathered} node(L_1^B) = node(L_2^B), R_q(L_1^B) \leq R_q(L_2^B), R_{\bar{t}}(L_1^B) \geq R_{\bar{t}}(L_2^B), R_y(L_1^B) \geq R_y(L_2^B), \\ \mathcal{V}(L_1^B) \subseteq \mathcal{V}(L_2^B) \end{gathered} \tag{3.23}$$

Similar to the forward label domination criterion, the capacity, time and energy resources are compared. However, instead of using the current time at the label, the time with recharging is used. The proof of this criterion follows the same rules as for the forward label case.

3.1.3. Bi-directional Labelling

By combining forward and backward labelling, the overall number of labels created is expected to be smaller than for the single directional cases. A bounded version of this method is known to improve the runtime of ESPPRC for VRP's as shown by Righini and Salani (2006).

Forward and backward labels are retrieved and extended in turn, using the same extension procedures and dominance checks as described before. A resource bound was used as suggested by Righini and Salani (2006). For the ESPPTWRS the time resource is utilized. The labels are extended using priority queues, which are sorted by the time resource – ascending for the forward labels and descending for the backward labels. As soon as the time resource of the next element in the queues have met, the extension can be stopped and a *splicing* (*joining*) procedure is called. This is used to combine all remaining non-dominated labels to obtain a set of feasible ESPPRC paths.

3.2. Heuristic Pricing Algorithm

To accelerate the search, the pricing problem is solved heuristically as long as columns with negative costs can be generated. This is achieved by iteratively searching on a reduced graph G_l with increasing complexity, as used by Irnich and Villeneuve (2006). In step l, the reduced graph G_l is build by connecting each customer node to its $(2l-1)$ nearest neighbours and the pricing problem is solved in G_l. The algorithm let l run from 1 to 6 but aborts the search as soon a path with negative reduced cost has been found (i.e. skipping the more complex graphs if possible).

4. Heuristic Solver

In order to be able to solve benchmark instances of realistic sizes, an heuristic solver has been developed based on the *Adaptive Large Neighbourhood Search* (ALNS) described in Ropke and Pisinger (2006) . The ALNS is combined with a local search procedure for intensification, as well as a labelling procedure to optimize the position of recharging stations within the routes at certain points during search. First the data structures for representing, evaluating and improving solutions are described. Then the local search and labelling procedures are presented. Finally the ALNS including the used destroy and repair operators will be discussed.

4.1. Representation and Evaluation

The evaluation of solutions in this approach is based on the work of Vidal et al. (2013b), where a sequence-based evaluation approach is introduced. It exploits the fact that solutions resulting from a bounded number of edge exchanges and node relocations performed on a solution can also be achieved using a recombination of a bounded number of sequences of visits of the same solution (see Kindervater and Savelsbergh, 1997; Vidal et al., 2013b).

For two sequences of nodes $\sigma_1 = \{u, \ldots, v\}$ and $\sigma_2 = \{u', \ldots, v'\}$ a concatenation results in a new sequence $\sigma' = \sigma_1 \oplus \sigma_2$, where the data of σ_1 and σ_2 has been combined to reflect the data needed to evaluate $\sigma' = \{u, \ldots, v, u', \ldots, v'\}$. The effectiveness of this approach depends substantially on efficient concatenation operator. However, as shown in Vidal et al. (2013b), there exists a large number of objective values and constraints that can be evaluated in $O(1)$.

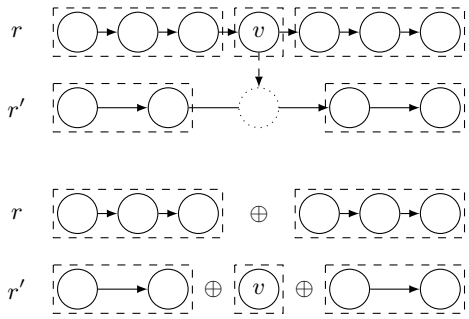


Figure 4.1: Example of a shift (relocate) move from r to r' (dashed line) using data of preprocessed sequences (dashed boxes) and the concatenation operator $\oplus$.

Figure 4.1 shows an example of a move evaluation by concatenation of preprocessed sequences. Starting with two routes, as shown at the top, the changed state (at the bottom) can be evaluated using three concatenation operations.

In the following, we will present a new constance-ctime concatenation operator for evaluating routes in of the E-VRPTW. For the E-FSMFTW we restrict these operations to preprocessed sequences using the same vehicle type. When sequences of different vehicle types are combined, one part has to be recalculated, which can be done in time linear to the number of nodes.

4.1.1. Concatenation operators from the literature.

According to the categorization introduced by Vidal et al. (2013a), the E-FSMFTW consists of several *EVAL* attributes, i.e., capacity, distance, and state of charge, as well as a single *ASSIGN* attribute: the vehicle type. For the concatenation of the capacity and distance attributes – and the time attribute to some extent, – the definitions also presented in Vidal et al. (2013b) are used. The general battery charge concatenation is based on the work of Schneider et al. (2014) for the E-VRPTW, but was adapted here for the sequence-based evaluation approach. To simplify the equations, the vehicle type k is omitted in the following description of the concatenation operators.

Capacity and distance. Both, capacity $Q(\sigma)$ and distance $Dist(\sigma)$ can be concatenated by constant operations:

$$Q(\sigma_1 \oplus \sigma_2) = Q(\sigma_1) + Q(\sigma_2) \tag{4.1}$$

$$Dist(\sigma_1 \oplus \sigma_2) = Dist(\sigma_1) + d_{ij} + Dist(\sigma_2) \tag{4.2}$$

where i is the last node of σ_1 and j first node of σ_2 The load capacity of a concatenation is simply the sum of both capacities. The distance of a concatenation is the sum of the individual distances plus the distance from the last node of the first sequence to first node of the second sequence.

State of charge. In order to compute the information concerning the state of charge, four values for each sequence are required. First, a boolean value $f(\sigma)$ indicating whether a sequence contains at least one recharging station. Second, the energy required to reach the first recharging station in the sequence is stored in $\overleftarrow{Y}(\sigma)$. Third, $\overrightarrow{Y}(\sigma)$ stores the energy needed to get from the last recharging station to the last node in the sequence. In sequences with no recharging station both values are set to the total energy needed to travel through the sequence. The final value $EY(\sigma)$ stores the overall violation of charge in a sequence, i.e.,

how much extra energy would be additionally needed to satisfy the constraint.

$$f(\sigma_1 \oplus \sigma_2) = f(\sigma_1) \vee f(\sigma_2) \tag{4.3}$$

$$\overleftarrow{Y}(\sigma_1 \oplus \sigma_2) = \begin{cases} \overleftarrow{Y}(\sigma_1) & \text{if } f(\sigma_1) \\ \overleftarrow{Y}(\sigma_1) + r \cdot d_{\sigma_1\sigma_2} + \overleftarrow{Y}(\sigma_2) - \Delta_Y & \text{otherwise} \end{cases} \tag{4.4}$$

$$\overrightarrow{Y}(\sigma_1 \oplus \sigma_2) = \begin{cases} \overrightarrow{Y}(\sigma_2) & \text{if } f(\sigma_2) \\ \overrightarrow{Y}(\sigma_1) + r \cdot d_{\sigma_1\sigma_2} + \overrightarrow{Y}(\sigma_2) - \Delta_Y & \text{otherwise} \end{cases} \tag{4.5}$$

$$EY(\sigma_1 \oplus \sigma_2) = EY(\sigma_1) + EY(\sigma_2) + \Delta_Y \tag{4.6}$$

where $\Delta_Y = max\{\overrightarrow{Y}(\sigma_1) + r \cdot d_{\sigma_1\sigma_2} + \overleftarrow{Y}(\sigma_2) - Y, 0\}$. For simplicity, the index of the vehicle type for the energy consumption r is omitted in these equations.

For the concatenation process, first the additional violation with regards to energy consumption – which is obtained when combining two sequences – is calculated. This value, denoted as Δ_Y, is obtained using the information of the amount of charge needed from the last charging station of the first sequence to the first charging station of the second sequence. If such an additional violation exists ($\Delta_Y > 0$), the amount is added to the total violation of the new sequence; see (4.6). To avoid counting violations multiple times throughout concatenations, the value Δ_Y is subtracted in Equation (4.4) and (4.5) in the cases where the first respectively second sequence does not contain a recharging station. This also implies that neither $\overleftarrow{Y}(\sigma)$ nor $\overrightarrow{Y}(\sigma)$ will ever exceed the maximum battery charge.

Time windows. An effective way to relax the time window constraint and efficiently calculate the time window violation was presented by Nagata et al. (2010) and was enhanced by Schneider et al. (2013) in the context of VRPTW. A time window penalty (TW) counting the cumulative time needed to travel back in time in order to satisfy the time window constraints is introduced. By 'repairing' the time using this time window penalty, violations in one location do not propagate to later locations. The concept of slack variables presented by Kindervater and Savelsbergh (1997) to evaluate moves efficiently in constant time is used.

Proper concatenation operators for this approach have been presented in Vidal et al. (2013b). In addition to the duration $Dur(\sigma)$ and the amount of time window violation $TW(\sigma)$, the earliest $E(\sigma)$ and latest departure $L(\sigma)$ from the first node are stored.

$$Dur(\sigma_1 \oplus \sigma_2) = Dur(\sigma_1) + Dur(\sigma_2) + t_{\sigma_1\sigma_2} + \Delta_{WT} \tag{4.7}$$

$$TW(\sigma_1 \oplus \sigma_2) = TW(\sigma_1) + TW(\sigma_2) + \Delta_{TW} \tag{4.8}$$

$$E(\sigma_1 \oplus \sigma_2) = max\{E(\sigma_2) - \Delta, E(\sigma_1)\} - \Delta_{WT} \tag{4.9}$$

$$L(\sigma_1 \oplus \sigma_2) = min\{L(\sigma_2) - \Delta, L(\sigma_1)\} + \Delta_{TW} \tag{4.10}$$

where $\Delta = Dur(\sigma_1) - TW(\sigma_1) + t_{\sigma_1\sigma_2}$, $\Delta_{WT} = max\{E(\sigma_2) - \Delta - L(\sigma_1), 0\}$ and $\Delta_{TW} = max\{E(\sigma_1) + \Delta - L(\sigma_2), 0\}$

For each concatenation of two sequences σ_1, σ_2 the additional waiting time Δ_{WT} is calculated as well as the additional time window violation Δ_{TW} introduced by the concatenation. These are used to compute the concatenation values of (4.7)-(4.10) properly. We use Δ as an auxiliary variable for the time needed to reach the first node of the second segment σ_2.

With this operator definition the time window violations can be calculated, however, the charging times and its effect on the time calculation are still unaccounted for. In Schneider et al. (2014) a slack-based approach was presented. Here the authors utilized the concept of time travelling and showed that for combining two partial routes, $\{u, \ldots, v\}$ and $\{w, \ldots, f, \ldots, h\}$, only a part of the second route has to be recalculated, i.e., until the first recharging station f is encountered. After recalculating the additional charge needed at the recharging station as well as the new arrival time, a violation in the time constraint can be checked using this new time values. In case there is no recharging station in the second part, no recalculation is needed.

The complexity of this approach is $O(B)$, where B is the number of nodes prior to the first recharging station in the second partial route. In most situations, B might only be a small number. However, if additional nodes are placed prior to the new partial route, the subsequent re-evaluations are going to have an impact on the performance. In the following, a new concatenation operator is proposed to address this issue.

4.1.2. An efficient concatenation operator for the E-VRPTW

In this work, the definition for the capacity and distance data concatenation introduced at the beginning of this chapter is used. However, the energy constraint has a direct impact on the time window constraint. Therefore the modified time window and energy calculation of Schneider et al. (2014) are adapted and extended for the sequence-based evaluation approach of Vidal et al. (2013b). As this calculation also depends on the vehicle type (see state of charge operator), this concatenation operator is also limited to sequence data concatenations of the same vehicle type. However, as described in the following, this restriction lead to an efficient constant time operator formulation.

The problem in the calculation of time window violations with recharging is the possibility of increasing recharging time at the first recharging station due to increasing energy demand by adding additional nodes before this visit. However, after the first recharging station has been visited, the precise recharging time required in each following station is known. This is due to the assumption to recharge to full capacity at each recharging station visit and the nature of the sequence-based evaluation, where a sequence can only be added before or after another sequence.

To solve this problem, the calculation of the final amount recharged and its effect on the remaining sequence is delayed by splitting the sequence into three sub-sequences right before and after the first recharging station encountered. That is, for $\sigma = \{u, \dots, v, f, v', \dots, w\}$, let $\overleftarrow{\sigma} = \{u, \dots, v\}$, $\sigma^{RC} = \{f\}$, $\overrightarrow{\sigma} = \{v', \dots, w\}$, where $f \in F'$ is the first recharging station in σ. If $\sigma = \{u, \dots, w\}$ does not have any recharging stations, then let $\overleftarrow{\sigma} = \{u, \dots, w\}$ and $\sigma^{RC} = \overrightarrow{\sigma} = \{\}$. To get the evaluation data for the full sequence, constant concatenation operations are performed on the three sub-sequences, i.e., $\sigma = \overleftarrow{\sigma} \oplus \sigma^{RC} \oplus \overrightarrow{\sigma}$.

The concatenation operator for two sequences σ_1 and σ_2 is defined as a set of concatenation operations on the sub-sequences $\{\overleftarrow{\sigma_1}, \sigma_1^{RC}, \overrightarrow{\sigma_1}\}$ and $\{\overleftarrow{\sigma_2}, \sigma_2^{RC}, \overrightarrow{\sigma_2}\}$. Depending on whether recharging stations exists on the sequences, there exists four cases:

- **no recharging station in both sequences.** Both sequences can be linked in a single concatenation operation $\sigma' = \{\overleftarrow{\sigma_1} \oplus \overleftarrow{\sigma_2}, \{\}, \{\}\}$
- **no recharging station in σ_1, but one in σ_2.** In this case, the additional distance and time travelled in σ_1 is considered and the changes when combining this information with the first part of σ_2 is calculated, i.e., $\sigma' = \{\overleftarrow{\sigma_1} \oplus \overleftarrow{\sigma_2}, \sigma_2^{RC}, \overrightarrow{\sigma_2}\}$
- **a recharging station in σ_1 but not in σ_2.** Although the recharging time of the first recharging station in σ_1 will not change, the information of the second part $\overrightarrow{\sigma}$ has to be updated to account for the reduced energy capacity available after travelling the additional distance of σ_2. The concatenation of both sequences is therefore $\sigma' = \{\overleftarrow{\sigma_1}, \sigma_1^{RC}, \overrightarrow{\sigma_1} \oplus \overleftarrow{\sigma_2}\}$
- **a recharging station in both sequences.** If both sequences contain recharging stations, the changes in σ_2 need to be evaluated using two concatenations: First the changes in distance and time for σ_2 is calculated by concatenating $\overrightarrow{\sigma_1}$ with $\overleftarrow{\sigma_2}$. With the final distance and time bounds stored, the actual recharging time is calculated by concatenating $\overrightarrow{\sigma_1} \oplus \overleftarrow{\sigma_2}$ with $\overrightarrow{\sigma_2}$. This newly acquired information is also the second part of the new sequence, i.e., $\sigma' = \{\overleftarrow{\sigma_1}, \sigma_1^{RC}, \overrightarrow{\sigma_1} \oplus \overleftarrow{\sigma_2} \oplus \sigma_2^{RC} \oplus \overrightarrow{\sigma_2}\}$.

In summary, the concatenation operations for time windows with recharging are as follows:

$$\sigma_1 \oplus \sigma_2 = \begin{cases} \{\overleftarrow{\sigma_1} \oplus \overleftarrow{\sigma_2}, \{\}, \{\}\} & \text{if } \sigma_1^{RC} = \{\}, \sigma_2^{RC} = \{\} \\ \{\overleftarrow{\sigma_1} \oplus \overleftarrow{\sigma_2}, \overrightarrow{\sigma_2}\} & \text{if } \sigma_1^{RC} = \{\}, \sigma_2^{RC} \neq \{\} \\ \{\overleftarrow{\sigma_1}, \overrightarrow{\sigma_1} \oplus \overleftarrow{\sigma_2}\} & \text{if } \sigma_1^{RC} \neq \{\}, \sigma_2^{RC} = \{\} \\ \{\overleftarrow{\sigma_1}, \overrightarrow{\sigma_1} \oplus \overleftarrow{\sigma_2} \oplus \overrightarrow{\sigma_2}\} & \text{if } \sigma_1^{RC} \neq \{\}, \sigma_2^{RC} \neq \{\} \end{cases} \tag{4.11}$$

Equation (4.11) shows that the constraint data of a combination of two sequences can be build using at most three concatenation operations. The actual values is calculated by combining the data from $\overleftarrow{\sigma}$, σ^{RC} and $\overrightarrow{\sigma}$, i.e., perform two additional concatenation operation $\overleftarrow{\sigma} \oplus \sigma^{RC} \oplus \overrightarrow{\sigma}$. This leads to a total of three constant time operations per concatenation. By storing additional information, inter-route moves as described in Schneider et al. (2014) can be evaluated in constant time – regardless of whether recharging stations are present or not – without the need to recalculate a subsequence. Furthermore, this approach is applicable to any move resulting from a bounded number of edge changes.

For the E-FSMFTW using differentvehicle types, only preprocessed sequences of the one route can be used. The new part has to be recalculated using this vehicle type before it can be concatenated. This recalculation depends on the length of the sequence added, and is thus no longer constant. We therefore restriced our neighbourhood creation in our local search to consider only moves between routes of the same vehicle type, and use additional, smaller neighbourhoods for combining different vehicle types.

4.1.3. Feasibility

The feasibility of a route can be checked using the data calculated for the corresponding sequences:

$$isFeasible(\sigma^k) = Q(\sigma^k) \leq Q^k \wedge EY(\sigma^k) \leq 0 \wedge TW(\sigma^k) \leq 0 \tag{4.12}$$

where $k \in V$ is the vehicle type used to calculate the penalties. We note that the calculation of $Q(\sigma)$ is independent of the vehicle type, but the charge and time window penalty are not. Although identical travel times for each vehicle are assumed, the time window penalty calculation is influenced by the battery charge constraint, which itself depends on the vehicle type.

Penalizing infeasibility. The heuristic solver proposed in this paper works with infeasible solutions and uses penalty values for violations of the constraints (thus relaxing the problem) in the objective function:

$$obj(\sigma^k) = Dist(\sigma^k) + \rho_Q(Q(\sigma^k) - Q^k) + \rho_{EY} EY(\sigma^k) + \rho_{TW} TW(\sigma^k) \tag{4.13}$$

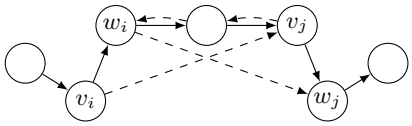
Each constraint has a corresponding penalty weight ρ which is multiplied with the amount of violation for each of the three relaxed constraints and added to the total distance travelled. We use ρ_Q to weight load capacity violations, ρ_{EY} for energy capacity violations and ρ_{TW} for time window violations.

4.2. Construction and Insertion

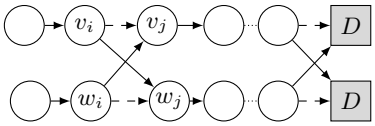
For the construction of an initial solution and later also in the repair step of each iteration of the ALNS, partial solutions have to be extended, i.e., solutions with at least one customer not being assigned to any route, in order to create a complete solution. This is done using different insertion based approaches, i.e., inserting nodes into existing or newly created routes until all nodes have been assigned. To diversify the search, the concept of a *Restricted Candidate List* (RCL) introduced by Hart and Shogan (1987) is employed to select the next node and insertion position in a probabilistic way. Each position in the RCL is selected based on its contribution to the sum of all insertion costs in the RCL.

As recharging stations do not have to be part of a route and can be inserted multiple times, a special procedure is implemented to insert recharging stations during the insertion and construction phase. Besides trying to insert a node v at position i of a route r, three additional attempts are performed, where a recharging station f is inserted right before v, a station g right after v, or both. This means a total of four sequences are tested to be inserted for each node v: $\{v\}$, $\{f, v\}$, $\{v, g\}$ and $\{f, v, g\}$, where f and g can be the same station. To determine $f, g \in F'$, the recharging station which is reachable with the available energy capacity and which has the smallest additional detour is selected. This approach is myopic in the sense that just a subset of recharging stations is considered, and that only the direct effect on the value of the route (and not the effect on future inserts). Furthermore, all previously inserted recharging stations are kept unchanged.

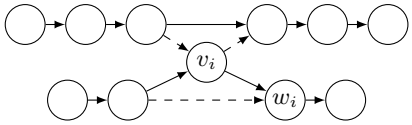
Similar to the work of Paraskevopoulos et al. (2008) and Repoussis and Tarantilis (2010), an iterative route construction heuristic is implemented. In each iteration, a single route for each vehicle type is created using only unassigned nodes until the capacity constraint is violated. These routes are constructed independently,



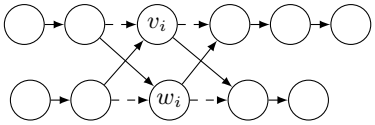
(a) **2-Opt.** Inverting a sequence of at least two node. This example shows the inversion of the sequence between v_j and w_i



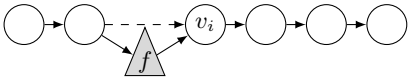
(b) **2-Opt*.** Reconnecting a substring of one route at node v_i with a substring of another route at w_j and vice versa.



(c) **Relocate.** Moves a single node v_i from the assigned route prior to w_i of another route.



(d) **Exchange.** Swaps (exchanges) two nodes v_i and w_j of different routes.



(e) **InsertRemoveIF.** Inserts or removes a recharging station f prior to v_i.

Figure 4.2: List of all moves used. Solid lines shows the path of the routes after the move is performed and dashed lines the removed edges.

i.e., nodes can be used in multiple routes, and the route with the best lowest cost is added to the current partial solution. We use the *Average Cost per Unit Transfered* (ACUT_k) value defined in Paraskevopoulos et al. (2008) to measure the cost of a tour. This value represents the relative costs of serving the demand of a route σ of vehicle type k (see Equation 4.14).

$$ACUT_k(\sigma) = (f^k + Dist(\sigma))/(Q(\sigma)) \tag{4.14}$$

The remaining routes are dismissed and the next iteration is performed until no further unassigned nodes are available.

4.3. Neighbourhoods and Local Search

The approach described in this paper combines the general search methodology of ALNS with a local search method to intensify the search in each iteration. A mix of well-researched neighbourhoods and specifically tailored problem specific neighbourhoods is used. First the moves defining these neighbourhoods are presented followed by a description of the local search method applied.

2-Opt. Figure 4.2a. This intra-route move (i.e., operation on a single route) optimizes a route by inverting a subsequence. As shown in the figure, this move involves the deletion and reinsertion of two edges as well as the inversion of the directions between the nodes of the selected sequence. To further reduce the neighbourhood size and thus the runtime, only moves including subsequences of size two at maximum are considered.

2-Opt.* Figure 4.2b. The 2-Opt* move removes two edges of two distinct routes of the same vehicle type and reconnects the remaining sequences as shown in the figure. An additional cross is made at the end of the route to ensure that the depots are assigned to the same node after the move is performed. As described in Section 4.1, this restriction is needed to compute the move changes using concatenation efficiently.

Relocate. Figure 4.2c. In this move, a single node is shifted (relocated) from one route in between nodes of a different route.

Exchange. Figure 4.2d. This moves switches (exchanges) the position of two nodes assigned to different routes.

InsertRemoveIF. Figure 4.2e. In this move, a recharging station is inserted prior to a node v_i in order to improve the value of a route due to repairing violations of the battery capacity constraint, or an existing recharging station is removed from the route. This problem specific move is the only one capable to insert and/or remove an optional recharging station in the local search procedure.

Resize. All moves described so far only change node assignments or positions but never the vehicle type assignment. The Resize move changes the vehicle assigned without changing the route itself.

RelocateAndResize. This move works like the *Relocate* move described before, but with an additional procedure performed after relocating a node. If a node v is moved from a route r_i to r_j, the vehicle type of r_i and r_j are tried to be changed at the same time in order to benefit from either lower costs from selecting a cheaper vehicle type or reducing the penalty of constraint violations due to the change to a vehicle type with more capacities.

In the implementation of this move, the effect of changing the vehicle type on the evaluation process has to be considered, which requires an additional preprocessing of the subsequences with the new type. In this approach, the preprocessing is delayed until the neighbourhood is used and store the data from this point onwards until the route changes again (which triggers a recalculation on the next use of this neighbourhood).

4.3.1. Local Search

The embedded local search procedure utilizes a list of different neighbourhoods, which are searched in a cyclic manner (see Di Gaspero and Schaerf, 2002; Hiermann et al., 2015). Each neighbourhood is searched until no further improvement can be found, after which the next neighbourhood in the list is selected and searched. When the end of the list is reached, the search starts again from the first entry in the list. The procedure terminates, if a local optimum is reached in every single neighbourhood. By cycling through the list, it is ensured that a local optimum has been reached in each neighbourhood. Preliminary tests showed that random ordering of the list of neighbourhoods at the beginning of the local search call improves the overall performance.

As improvement strategy, a *best of 50*-policy is used, where the best of the first 50 moves encountered during a single search iteration is applied. If none of them is an improving move, the search continues until an improving move is found or the neighbourhood has been searched completely. This way the search terminates faster than with a *best improvement*-policy, but it still provides higher quality solutions than *first improvement*.

4.3.2. MakeFeasible

As described in section 4.1.3, a varying penalty cost is used to guide the search through the infeasible search space. To reach a feasible solution using the existing tools, a similar approach as described in Vidal et al. (2013b) is employed. A feasible solution is tried to be obtained twice by multiplying each penalty costs by 100 and call the local search procedure. If the solution is still not feasible, the penalty costs are further multiplied by ten each and the local search is called again. No further attempts are made to make the solution feasible, i.e. solutions might still be infeasible after applying this repair operator. However, using this approach, feasible regions close to the infeasible solution may be found at low cost.

4.3.3. Reducing neighbourhood size

The heuristic solver described in this paper depends heavily on the embedded local search procedure, which in turn uses a number of neighbourhoods to improve the search. As the number of neighbours is large and computationally expensive, a pruning approach as described in Vidal et al. (2013b) is used to reduce the number of neighbours considered. In this approach, a set of so-called *promising* arcs is calculated for each customer node. This set is then used to prohibit introducing arcs into the solution that are not in this set. With Equation (4.15), the so-called *customer correlations* measure is calculated, which is used for the pruning.

$$\begin{aligned}\gamma(u,v) = d_{uv} &+ \gamma^{WT} \cdot max(e_v - s_u - t_{uv} - l_u, 0) \\ &+ \gamma^{TW} \cdot max(e_u + s_u + t_{uv} - l_v, 0)\end{aligned} \tag{4.15}$$

For a customer v_i not only the direct distance, but fractions of the waiting (γ^{WT}) and time window violation (γ^{TW}) are considered as well. The final set of *promising* arcs $\Gamma(u)$ consists of the $|\Gamma|$ closest customers v with respect to the correlation measure $\gamma(u,v)$. The same settings as in Vidal et al. (2013b) are used, setting $|\Gamma| = 40$, $\gamma^{WT} = 0.2$, and $\gamma^{TW} = 1.0$.

4.4. Labelling Algorithm

Recharging stations are handled explicitly in this work, i.e., stations are added directly as entries in the routes. As the construction and insertion heuristics described earlier have a rather narrow view on the positioning of recharging stations in order to keep the runtime low, a post-processing procedure is used to improve the selection and positioning of recharging stations in a route of fixed visiting orders.

The method employed is a labelling algorithm similar to the procedure described in Section 3.1 for the ESPPTWRS. However, several simplifications can be derived and exploiting from the fixed visiting order:

Let r be the route for which the optimal assignment of recharging stations $r*$ has to be found. Furthermore, let $r' = r \setminus f \in F$ and u_i be the ith node of r'. The set of possible nodes to extend to is restricted by r'. Only recharging stations and a next customer (or depot) in r', i.e., node u_{i+1}, can be considered. Additionally, not every recharging station has to be considered, just a subset relevant for a detour between u_1 and u_{i+1}. The visits are restricted to r', thus the set of already visited (unreachable) nodes is no longer necessary.

To exploit this features, a modified forward labelling procedure with a label $L' = \{i, (R_c, R_t, R_y), prev, f\}$ is used, where i is the index in r'. The load resource R_q is omitted in the modified label, as the total demand will never increase by adding recharging stations. f stores the recharging station used before arriving in node node u_i. Algorithm 1 outlines this modified procedure.

Algorithm 1: Pseudo code of a label setting algorithm for the ESPPTWRS (similar to Ropke and Cordeau (2009)).

```
Data: route r'
U_1 = {(L' = {1, (0, 0, g · Y), null, {})};
for i ∈ 1, ..., |r'| do
    for L ∈ U_i do
        if no label in L_i dominates L then
            L_i = L_i ∪ {L};
            extend L along arc (u_i, u_{i+1});
            extend L via all relevant recharging station using arcs (u_i, f), (f, u_{i+1});
            add all feasible extensions to U_{i+1};
return path corresponding to the best label in L_{|r'|}
```

For this procedure, only a single recharging station is assumed to be needed between two consecutive nodes. We are aware that this assumption might prohibit the labelling procedure to reach an optimum. However, preliminary experiments with implementations permitting two or three recharging stations between nodes in r' showed that the additional computation time leads to no improvement in the overall performance.

This simple but effective procedure is performed for each route after construction, reinsertion and local search.

4.5. Adaptive Large Neighbourhood Search

The E-FSMFTW is solved using an *Adaptive Large Neighbourhood Search* (ALNS) as proposed by Ropke and Pisinger (2006), but extended by an intensification mechanism in form of an embedded *Local Search*

Algorithm 2: ALNS main loop

Input: initial solution s
Output: best feasible solution found s_f^*

$s^* \leftarrow s$;
if *isFeasible(s)* **then** $s_f^* \leftarrow s$;
$i, i^{lastImp} \leftarrow 0$;
while $i < \eta_{max}$ ***and*** $(i - i^{lastImp}) < \eta_{maxNoImp}$ **do**
  $s' \leftarrow DestroyAndRepair(s)$;
  $s' \leftarrow LocalSearch.improve(s')$;
  $s' \leftarrow Labelling.optimize(s')$;
  if $isBetter(s', s)$ **then**
    $s \leftarrow s'$;
    if $isBetter(s', s^*)$ **then** $s^* \leftarrow s'$;
    $s'_f \leftarrow MakeFeasible(s')$;
    if $isFeasible(s'_f)$ ***and*** $isBetter(s'_f, s_f^*)$ **then** $s_f^* \leftarrow s'_f$;
    $i^{lastImp} \leftarrow i$;
  updateScore(s');
  updatePenalty(s');
  if $0 \equiv (i - i^{lastImp} + 1) \mod \eta_R$ **then**
    $s \leftarrow s^*$;
  if $0 \equiv (i + 1) \mod \eta_L$ **then**
    $adaptSelectionScore()$;
  $i \leftarrow i + 1$;
return s_f^*;

procedure. A general introduction to LNS and its variants and extensions can be found in Pisinger and Ropke (2010). An outline of the approach proposed in this paper is shown in Algorithm 2.

In Shaw (1998), a search procedure for a larger neighbourhood defined by a *destroy* and *repair* operators is presented. A neighbour is a solution which is reachable by removing some nodes using the destroy operator, followed by the repair operator.

Ropke and Pisinger (2006) extend this by using learning mechanisms to bias a selection of a variety of destroy and repair operators effectively. For each phase (destroy or repair) the corresponding operator i is selected using a roulette wheel based on so-called weights ν_i. These weights ν_i are changed to adapt the algorithm based on the performance of the selected operators for the last couple of iterations. This adaption is done using exponential smoothing of the observed scores $\bar{\nu}_i$. Similar to Ropke and Pisinger (2006), the score of an operator is increased in the following cases, if the newly generated solution

- is an overall best solution
- has not been accepted before and is better than the current solution
- has not been accepted before and is worse, but was accepted in this iteration

After a learning period (of η_L iterations) this score $\bar{\nu}_i$ is used to update the selection probabilities of operator i using the following equation (cf. Ropke and Pisinger, 2006)

$$\nu_{i,j+1} = \phi(\frac{\bar{\nu}_{i,j}}{a_i}) + (1 - \phi)\nu_{i,j} \tag{4.16}$$

The weight $\nu_{i,j+1}$ of heuristic i in the next period $j+1$ is calculated using iteration (4.16). The counted score $\bar{\nu}_{i,j}$ for the last period is divided by the number of times the heuristic has been used a_i, which influences the

new score for period $j+1$. The step size $\phi = [0;1]$ measures the influence of the new observation compared to the past. If $\phi = 1$ the score reflects the performance of the previous iteration only.

As described in Section 4.1.3, some constraints are relaxed during search in order to also investigate infeasible regions, while penalizing infeasible solutions. The penalties are managed using an adaptive mechanism as presented in Cordeau et al. (2001). However, instead of adapting the penalty weights in each iteration of the embedded local search, these values are controlled within the ALNS iteration. The embedded intensification procedure is therefore used to improve the newly generated solution towards a local optimum for the given penalty settings. As the procedure moves rather slowly towards feasible solutions, the repair approach described in Section 4.3.2 is employed to find these earlier in the search.

During the search two types of best solutions are stored. On the one hand the best feasible solution is stored and on the other hand the best solution with the current penalty setting is remembered as well. Every time the penalty values change, the value of the best (possibly infeasible) solution is adapted as well. This might lead to an objective value for the best solution which is worse than the value for the best feasible solution. In this case the best is replaced by the best feasible solution. The penalty weight is multiplied by w_i if the constraint i was violated in the last iteration or divide it by the same value if it was satisfied.

To avoid searching too deep in infeasible regions and to escape possible local optima, a restarting mechanism is used. After a certain number of iterations (η_R) with no overall improvement the current solution is reset to the best overall solution found so far. In contrast to other restarting mechanisms, only the current solution is changed and no other values (e.g., no change in the penalty values).

Similar to the original ALNS, an SA approach to handle the acceptance of new solutions generated in an iteration was tested. However, preliminary experiments showed that using an 'accept only improvements' policy yields better results for the given problem. This might be due to the use of adaptive penalties and the restart mechanism, which already contributes to the diversification process sufficiently.

An important aspect of ALNS is the number and functionality of the operators used. In the remainder of this section the applied operators will be presented.

4.5.1. Destroy operators.

A solution is destroyed by removing at least q nodes from the current solution, where q is a random number between $[\zeta_{max}, \zeta_{min}]$.

The ***RandomRemoval*** operator simply removes q nodes from the current solution randomly with equal probability.

RandomAndRelatedRemoval operator is based on the definition of Shaw (1998) where nodes are iteratively select and removed at random (with equal probability). Following that, a node is selected which is similar using an RCL of five nodes and roulette wheel selection and remove it as well. The similarity of nodes is calculated using the relatedness measure (4.15). The procedure continues until at least q nodes have been removed in total.

A similar operator is the ***WorstAndRelatedRemoval***. Here, however, a node is first selected and removed based on the detour needed to travel in the solution induced by it. This is done using an RCL of the five worst nodes where a roulette wheel is used based on the detour cost. Then up to $q-1$ additional nodes are removed based on their relatedness to the selected node. The relatedness is again calculated using (4.15).

With the ability to remove whole routes the ***InefficientRouteAndNeighbourRemoval*** starts by selecting and removing a route at random using roulette wheel based on the ACUT (4.14). Then neighbouring routes are selected and removed iteratively – starting from the route with the smallest maximum distance between any pair of nodes – until at least q nodes have been removed in total.

The ***TargetRemoval*** operator is similar to the target operator used by Dell'Amico et al. (2007) but simplified. Instead of trying different combinations of a target route, a single node is considered, the so-called *target node*. The node with the highest contribution to the total distance of its assigned tour is selected. Afterwards, routes based on the minimum distance from any node of a route to the target node are selected and removed. Starting from the route with the highest minimum distance, they are destroyed until at least $q-1$ nodes have been removed.

4.5.2. Insertion Based Repair.

The repair operators are all based on insertion heuristics with the ability to insert recharging stations as described in Section 4.2.

The first basic **SequentialNodeInsertion** heuristic implemented is a myopic but fast approach. Nodes are processed in a sequential manner based on the removal order. One after another, the insertion cost of each route and position is calculated first, followed by the selection using a RCL containing the five best options. To handle different vehicle types, the vehicle type assignment is changed if necessary (when a constraint is violated) and add the resulting costs to the insertion costs. This is also done in the other insertion heuristics. When used as a repair operator in the ALNS, the sequential order of the insertion depends on removal order of the nodes .

The second heuristic is a **ParallelRegretInsertion** approach. Basic parallel greedy insertion heuristics tend to place 'difficult' nodes late in the process, leaving only a few possibilities. To avoid this behaviour, the regret heuristic uses a so-called *regret* value which represents the expected costs of inserting a node not in this iteration but in a future iteration. Such heuristics have been used by Potvin and Rousseau (1993) for the VRPTW and by Trick (1992) for the Generalized Assignment Problem. The regret value for a node v is obtained as follows. For each route σ_i in the current partial solution, the position and cost c_i of the best insertion (based on the objective value) of v into σ_i is calculated . Let $l \in L$ be the index of the routes where L is sorted based on c_l, i.e., $l \leq l'$ if $c_l \leq c_{l'}$. The regret value is calculated considering the k best insertion costs using following equation:

$$regret(v) = \sum_{i=2}^{k} (c_{l_i} - c_{l_1}) \tag{4.17}$$

The **SemiParallelConstruction** heuristic used to create the initial solution can be easily modified to work as a repair operator too. Instead of creating a full solution from scratch this approach starts with the partial (destroyed) solution and creates new routes without considering existing ones.

A variant, the **SemiParallelInsertion** procedure, extends the previous approach as also existing routes are considered. Instead of creating only new routes, existing routes are extended by inserting unassigned nodes until the capacity constraint is violated.

5. Computational Results

In this section we present computational results of the proposed ALNS approach for solving the E-FSMFTW. We use our branch-and-price solver – further refered as BnP – to obtain optimal solutions for smaller instances, and lower bounds for larger instances of our new benchmark.

Our experiments using the BnP procedure were run on a cluster system on single 3.3 GHz cores (Intel Xeon 2643) and a maximum of 7 GB RAM, running on with CentOS (Red Hat Enterprise Linux). The solver uses Cplex 12.6 through the Open Solver Interface (OSI; v0.105.2).

The experiments with the ALNS were run on a single core of a cluster system with an Intel Core2 Quad CPU Q6600 with 2.40 GHz where a memory of 4 GB RAM is shared between 4 cores, operating on the Linux distribution openSuse 12.1 (Asparagus) 64 Bit. The ALNS with embedded local search and labelling is a single-thread implementation in Java 7 and was run using the Java Runtime Environment 1.7, Update 25 (JRE 7u25).

In calibrating the parameters of our ALNS approach, we mostly relied on parameters chosen in the literature. Indeed, in our experience ALNS is a rather robust approach which is not very sensitive w.r.t parameters. After some preliminary experiments on a reduced number of instances we chose the parameter setting as shown in Table 5.1, if not stated otherwise. Ten test runs were performed and average and best results are reported.

We also tested our approach for the related E-VRPTW with both proposed solver. The BnP was able to prove all results on the small instances reported by Schneider et al. (2014) to be optimal. For the larger instances, six instances were solved to optimality. The results are presented in the appendix. Our heuristic solver was also able to obtain two new best solutions for the larger instances, with an average gap to the best known below one percent (see Goeke and Schneider (2015)).

5.1. E-FSMFTW Benchmark Instances

Our benchmark instances are based on the modified Solomon instances of Schneider et al. (2014) and the description of the vehicle type classes of Liu and Shen (1999). The instances of Schneider et al. (2014) consists of six data sets varying in the distribution of the customers, i.e., whether they are clustered (C), randomly distributed (R) or a combination of both (RC). Furthermore, they are divided in instances with a shorter (1) or a longer (2) scheduling horizon. In Liu and Shen (1999), the vehicle types are defined for each of these instance sets, differing in their acquisition cost. These vehicle types are extended for our problem by energy consumption per km/mile, battery size and recharging rate. The consumption and recharging rate is directly taken from the instances of Schneider et al. (2014) and set as the same value for each vehicle type. For the battery size we use the value defined in the E-VRPTW instances as base value and scale it upon the rank of the vehicle types as follows: Given an instance with $|V|$ vehicle types defined by Liu and Shen (1999) and the battery size Y for the E-VRPTW, we set the battery size of vehicle type k for the E-FSMFTW to

$$Y^k = (1.0 + (s \cdot (k/|V|) - (s/2)) \cdot Y, \tag{5.1}$$

where $s = 0.1 \cdot |V|$. Using this approach we encourage the use of larger vehicles due to their larger battery size which leads to a different fleet composition compared to the non-electrical vehicles used by Liu and Shen (1999). Although a different choice of the recharging and energy consumption factors may result in more realistic scenarios, we note that this could be easily included in our model and algorithms. However, changes in the battery size and other energy related factors have to be weighted against each other to avoid vehicle type dominations.

5.2. Results on small E-FSMFTW instances

For the exact approach we solve the smaller sets created by Schneider et al. (2014) using our BnP solver (which uses CoinOR and CPLEX12.6) with the model and modifications described in Section 2. The sizes of these instances are either 5, 10 or 15 customers with 2 to 8 recharging stations. In preliminary experiements we tried to solve these smaller instances with the MIP model also described in Section 2 using branch-and-bound only. However, only 5 customer instances could be solved, but almost none of the instances with 10 and 15 customer within a 12 hours runtime.

Table 5.2 presents the comparison of the proposed ALNS with the BnP for the 15 customer instances. The results for 5 and 10 customer instances can be found in Appendix A. The BnP was able to obtain an optimal solution within the time limit of two hours for all instances. We show the objective value (obj) and the included sum of the acquisition costs (F) for the optimal fleet composition specified in the column mix as well as the runtime in seconds ($t[s]$). The fleet composition is presented as in Repoussis and Tarantilis (2010), where the vehicle type is referenced using capital letters (starting with the cheapest vehicle type using the letter 'A') and the number of vehicles used shown superscript after the corresponding letter. The objective of the initial solution (i.e., the improved solution after construction using LS and the labelling procedure) is shown in the column *init*. For the ALNS solutions we present these values for the best solution (obj) as well as the average objective function ($\overline{obj}$) of all ten runs.

The BnP is able to find the optimal solutions for all of the instances in a matter of seconds in most cases, only for one instance it took several minutes. Our ALNS approach is able to obtain optimal solutions in all cases as well, requiring less than a minute of runtime.

general		ALNS		penalty	
η_{max}	2000	η_R	200	$\rho_Q, \rho_{EY}, \rho_{TW}$	10
$\|\Gamma\|$	40	η_L	50	$\rho_Q^{min}, \rho_{EY}^{min}, \rho_{TW}^{min}$	0.1
$(\gamma^{WT}, \gamma^{TW})$	(0.2,1.0)	ϕ	0.8	$\rho_Q^{max} \rho_{EY}^{max}, \rho_{TW}^{max}$	5000
$\|RCL\|$	5	$[\zeta_{min}, \zeta_{max}]$	[0.05,0.15]	w_i	1.1

Table 5.1: Parameters of the ALNS with embedded LS/labelling approach

		BnP				init.	ALNS					
type	name	obj	F	mix	t[s]	obj	$\overline{obj}$	obj	F	mix	t[s]
A	c103c15	**1291,03**	900	A^3	1,54	1596,11	1291,46	**1291,03**	900	A^3	30,16
	c106c15	**1253,59**	900	A^3	0,18	1532,38	**1253,59**	**1253,59**	900	A^3	29,25
	c202c15	**2403,35**	2000	A^2	0,79	3419,19	**2403,35**	**2403,35**	2000	A^2	38,51
	c208c15	**2325,89**	2000	A^2	2,22	2327,88	**2325,89**	**2325,89**	2000	A^2	32,00
	r102c15	**850,58**	390	A^3B^3	0,13	943,28	854,99	**850,58**	390	A^3B^3	30,11
	r105c15	**760,13**	380	B^3C^1	1,04	889,40	762,48	**760,13**	380	B^3C^1	24,80
	r202c15	**1311,24**	900	A^2	4,26	1397,12	1316,84	**1311,24**	900	A^2	54,90
	r209c15	**1033,50**	700	B^1	1,25	2029,22	**1033,50**	**1033,50**	700	B^1	38,33
	rc103c15	**840,97**	420	A^2B^2	0,10	912,16	840,98	**840,97**	420	A^2B^2	31,08
	rc108c15	**1013,70**	570	$A^2B^1C^1$	4,52	1204,29	1033,14	**1013,70**	570	$A^2B^1C^1$	28,06
	rc202c15	**1101,61**	700	A^1C^1	0,43	1434,09	**1101,61**	**1101,61**	700	A^1C^1	33,55
	rc204c15	**810,90**	500	A^1B^1	953,47	925,38	**810,90**	**810,90**	500	A^1B^1	32,54
B	c103c15	**571,03**	180	A^3	1,28	649,36	571,78	**571,03**	180	A^3	32,91
	c106c15	**533,59**	180	A^3	0,08	576,30	538,28	**533,59**	180	A^3	32,90
	c202c15	**803,35**	400	A^2	0,29	1020,57	**803,35**	**803,35**	400	A^2	36,84
	c208c15	**725,89**	400	A^2	1,29	927,88	**725,89**	**725,89**	400	A^2	28,45
	r102c15	**511,55**	90	$A^3B^2C^1$	0,21	580,80	514,84	**511,55**	90	$A^3B^2C^1$	33,74
	r105c15	**436,89**	98	B^3D^1	0,29	566,99	**436,89**	**436,89**	98	B^3D^1	26,02
	r202c15	**591,24**	180	A^2	3,54	673,31	**591,24**	**591,24**	180	A^2	44,64
	r209c15	**473,50**	140	B^1	1,71	709,59	**473,50**	**473,50**	140	B^1	38,03
	rc103c15	**499,67**	102	A^1B^3	0,31	632,91	**499,67**	**499,67**	102	A^1B^3	26,04
	rc108c15	**514,78**	132	A^1C^2	0,21	755,78	**514,78**	**514,78**	132	A^1C^2	26,60
	rc202c15	**541,61**	140	A^1C^1	0,42	586,78	**541,61**	**541,61**	140	A^1C^1	30,86
	rc204c15	**410,90**	100	A^1B^1	26,55	563,39	**410,90**	**410,90**	100	A^1B^1	35,27
C	c103c15	**481,03**	90	A^3	0,84	496,48	481,64	**481,03**	90	A^3	25,78
	c106c15	**415,13**	140	A^2B^1	0,12	452,38	440,52	**415,13**	140	A^2B^1	28,98
	c202c15	**603,35**	200	A^2	0,73	693,34	607,82	**603,35**	200	A^2	36,52
	c208c15	**525,89**	200	A^2	2,05	627,21	526,29	**525,89**	200	A^2	23,07
	r102c15	**465,94**	46	$A^2B^1C^2$	0,00	551,82	467,00	**465,94**	46	$A^2B^1C^2$	30,32
	r105c15	**387,89**	49	B^3D^1	0,64	552,71	387,94	**387,89**	49	B^3D^1	22,55
	r202c15	**495,64**	115	A^1B^1	1,08	568,08	**495,64**	**495,64**	115	A^1B^1	33,19
	r209c15	**403,50**	70	B^1	1,30	486,61	**403,50**	**403,50**	70	B^1	32,96
	rc103c15	**448,67**	51	A^1B^3	0,13	587,04	**448,67**	**448,67**	51	A^1B^3	25,52
	rc108c15	**445,25**	75	B^1C^2	0,21	714,44	445,95	**445,25**	75	B^1C^2	24,81
	rc202c15	**471,61**	70	A^1C^1	0,17	567,12	**471,61**	**471,61**	70	A^1C^1	26,54
	rc204c15	**360,90**	50	A^1B^1	22,25	585,51	**360,90**	**360,90**	50	A^1B^1	31,00
dev. %							0,01	0,00			

Table 5.2: Results for the E-FSMFTW instances with 15 customers compared to BnP

5.3. Results on larger E-FSMFTW instances

In order to further evaluate the performance of our algorithm, we conduct experiments on the larger benchmark instances (as described earlier in this section). We test four settings of our approach denoted as

- $ALNS_{2000}$ as our default settings, with a limit of 2000 iterations,
- $ALNS^{800}_{1500}$ using a limit of 1500 iterations or 800 iterations without an improvement,
- $ALNS_{800}$ with a limit of 800 iterations and
- $\overline{ALNS}^{800}_{1500}$ where we use the same termination criteria as $ALNS^{800}_{1500}$, but without calling the labelling algorithm described in Section 4.4

Table 5.3 shows the average deviations from the best solutions (BKS) we have been able to find by the BnP algorithm or in ten runs ($\overline{obj}$) as well as the average runtime in minutes ($t[m]$) grouped by instance type (C,R,RC).

As expected the best average performance is achieved using a setting with a higher iteration limit ($ALNS_{2000}$) which itself results in higher runtime (∼25 minutes on average). However, terminating earlier ($ALNS^{800}_{1500}$) reduces the runtime by around one third while keeping the average solution quality on a high level. Although almost 2% worse than the best known solutions the fast variant ($ALNS_{800}$) is around 60% faster than the default variant, on average.

		$ALNS_{2000}$		$ALNS^{800}_{1500}$		$ALNS_{800}$		$\overline{ALNS}^{800}_{1500}$	
type	name	$\overline{obj}$	t[m]	$\overline{obj}$	t[m]	$\overline{obj}$	t[m]	$\overline{obj}$	t[m]
A	C1	0,24	17,68	0,28	12,80	0,44	7,35	0,45	13.64
	C2	0,39	42,54	0,61	24,36	0,61	19,21	0,78	26,58
	R1	0,89	15,51	1,24	11,64	1,69	6,49	1,34	12,60
	R2	0,87	45,90	1,12	30,76	1,31	18,03	1,12	33,70
	RC1	1,23	14,92	1,39	11,18	2,21	6,35	1,74	12,08
	RC2	0,61	16,04	0,68	11,98	1,06	6,55	0,86	12,62
	avg.	0,66	25,68	0,84	17,38	1,15	10,70	1,00	18,84
B	C1	0,48	14,68	0,62	10,38	0,78	6,09	0,84	11,23
	C2	1,08	37,05	1,24	18,92	1,29	14,81	1,62	20,73
	R1	1,47	15,19	1,70	11,28	2,49	6,26	1,82	12,39
	R2	1,30	29,48	1,49	20,82	2,20	11,85	1,70	23,19
	RC1	1,40	14,57	1,79	10,73	2,37	6,08	1,92	11,65
	RC2	1,43	18,87	1,54	13,34	1,80	7,55	1,83	14,89
	avg.	1,17	21,47	1,38	14,32	1,82	8,71	1,59	15,77
C	C1	0,43	14,77	0,39	10,45	0,72	6,02	0,56	11,33
	C2	0,89	31,95	0,96	18,50	1,07	13,08	1,23	21,11
	R1	1,59	15,51	1,76	11,44	2,57	6,29	1,90	12,40
	R2	1,70	28,72	1,97	20,22	2,33	11,37	1,99	22,36
	RC1	1,40	14,80	1,76	10,90	2,43	6,08	1,65	11,60
	RC2	1,29	19,71	1,53	13,70	2,01	7,87	1,59	15,26
	avg.	1,23	20,83	1,40	14,26	1,90	8,41	1,49	15,72

Table 5.3: Average deviation from the best known solution and average runtime for all E-FSMFTW instances.

When comparing the fast approach $ALNS^{800}_{1500}$ including the labelling procedure with the version without labelling, one can observe that, on average, the labelling procedure increases solution quality while slightly reducing the overall runtime. This is due to the termination criterion of 800 iterations without improvement. Furthermore, a detailed analysis of our test runs shows, that the contribution of the labelling procedure to the overall runtime is only around 1%.

For future reference, we included the results for each of the newly proposed instances in the appendix.

5.4. Sensitivity analysis of the E-FSMFTW instances

In this work we proposed instances for using different types of electric vehicles. To analyse the gain of considering a fleet mix, we conducted experiments where the set of vehicle types is reduced to a single one. Table 5.4 shows the average objective value over 10 runs grouped by instance type (C,R,RC) for each vehicle type set (A,B,C). The best known solution (BKS), i.e., the best solution found in any run (heuristic or BnP), is shown in the third column. The fourth shows the average objective value for runs using all vehicle types. The following columns uses only a single vehicle type (A,B,C,D,E, or F) and tries to solve the instance with this restriction. In case of the R1 instances, the algorithm couldn't find a single feasible solution using only the first vehicle type (A) due to customers with demands larger than the vehicle's transport capacity. In the other cases, where no values are provided, the instances do not have more than three (C1), four (C2, R2, RC1), or five (R1) vehicle types defined.

As we can see in the results, considering the whole mix of vehicles does indeed improve the quality of the solution. In some cases, where good solutions uses only a single type, the algorithm tends to produce slightly inferior results on average. This can be seen in Table 5.5, where the average fleet mix is presented as defined before. For most instances, the mix does not only contain two, but three and four different vehicles used in the final solution.

To evaluate the impact on the number of recharging stations available in the instances, Table 5.6 shows three values for the best known solution (BKS) and again for the mix (All) and single vehicle case (A,B,C,D,E, or F): First the number of recharging stations used in total ($\#rs$), followed by the number of distinct recharging stations visited ($\{rs\}$) and the average number of visits per route ($\overline{rs}$). In terms of number of visits to recharging stations per vehicle, we see that at most two are visited on average. Even when increasing the transport capacity (by using larger vehicles), the number does not increase over two as well. These results are in line with the ones in Desaulniers et al. (2014), where the impact of allowing only a single or multiple visits to recharging stations per route was presented. Allowing multiple visits does lead

type	name	BKS	All	Only A	Only B	Only C	Only D	Only E	Only F
A	C1	7141,24	7156,84	7157,37	9735,86	15310,54	–	–	–
	C2	5708,36	5731,47	5724,10	6287,15	8645,76	11439,26	–	–
	R1	4104,28	4138,51	n.a.	4411,49	4162,40	4800,86	7564,91	–
	R2	3145,79	3171,16	3168,48	3728,16	4542,67	7592,27	–	–
	RC1	5000,13	5046,58	6112,60	5293,53	5544,67	7203,49	–	–
	RC2	4211,63	4228,28	4335,74	4298,72	4380,71	5090,72	5542,89	9145,25
dev	min		0,22%	0,23%	2,07%	1,42%	16,97%	31,61%	117,14%
	avg		0,60%	5,28%	13,40%	37,76%	64,73%	57,97%	117,14%
B	C1	2450,12	2459,95	2582,36	2797,31	3867,05	–	–	–
	C2	1703,79	1723,50	1715,41	1801,14	2242,43	2794,94	–	–
	R1	1885,69	1908,91	n.a.	2482,07	2010,20	1975,35	2507,54	–
	R2	1335,32	1357,48	1352,04	1455,14	1646,23	2267,73	–	–
	RC1	2182,13	2208,12	3995,52	2638,15	2274,61	2551,96	–	–
	RC2	1683,90	1710,72	2172,11	1765,76	1718,01	1855,14	1984,55	2775,84
dev	min		0,40%	0,68%	4,86%	2,03%	4,75%	17,85%	64,85%
	avg		1,21%	23,89%	14,37%	20,93%	33,15%	25,42%	64,85%
C	C1	1762,33	1768,21	2007,27	1916,76	2455,76	–	–	–
	C2	1197,63	1211,77	1206,19	1240,43	1440,83	1713,66	–	–
	R1	1566,55	1588,42	n.a.	2239,99	1740,83	1622,22	1860,29	–
	R2	1109,18	1127,32	1129,40	1171,62	1281,25	1605,60	–	–
	RC1	1794,58	1818,69	3727,44	2303,84	1861,39	1964,59	–	–
	RC2	1356,02	1378,07	1901,78	1452,22	1385,61	1441,87	1528,52	1969,90
dev	min		0,33%	0,71%	3,57%	2,18%	3,55%	12,72%	45,27%
	avg		1,25%	32,88%	16,07%	15,37%	21,44%	15,74%	45,27%

Table 5.4: Average objective value from the best known solutions (BKS) and average results of 10 runs using all or only a single vehicle type (A,B,C,D,E,F) for all E-FSMFTW instances.

to a lower objective value for these instances, as was expected. On average, although, for our instances we see that the best known solutions tend to have less than one stop per route scheduled. This differs from the E-VRPTW results, where more than one station was scheduled per route on average. We deduce that this is a feature of using a mix of vehicles, where larger vehicles with more battery capacity are used, which in turn need fewer (or no) recharging operations during their route.

The instances we propose consists – like the original instances for the E-VRPTW – of 21 recharging stations per instance. As we can deduce from the results, not even half of the recharging stations are needed in the solutions obtained (on average, less than a third are needed). Most distinct recharging stations are used in the instances with random spatial distribution and short time windows (R1). This was to be expected, as the tight time windows forces vehicles to travel larger distances between visits, thus covering a larger area. The fewest are visited on the set of instances with larger time windows and a combination of clustered and spatial distribution (RC2).

We also conducted the same set of experiments on the same instances, but without the energy constraint. The problem is therefore relaxed to the FSMF problem with a minimal distance as objective. Table 5.7 shows the comparison of the objective value when allowing a mix or restricting the fleet to a single type as in Table 5.4. The results are very similar, also showing a gain of using a mix, but also a lower objective value in general due to the missing detours. However, experimental results do not show any significant differences between the two problem variants in terms of gap between the mix and restricted types. Table 5.8 shows a compact view on these results. From these results we can see, that the total number of vehicles required is only slightly higher, with up to a single vehicle more on average. We then compared the fleet composition with the E-FSMFTW variant, to see whether the mix might be different, but the results showed a similar distribution of vehicle types in both variants. We therefore can conclude, that the transport capacity is a dominant factor in finding a good fleet composition. Including the energy constraint does not influence the distribution of vehicles in the fleet. However, the constraint requires different routing decisions with additional detours to recharging stations. This can be deduced from the comparison of the objective value, where the E-FSMFTW is, as expected, always higher. The table showing the average fleet composition can be found in the appendix as well.

type	name	BKS	All	Only A	Only B	Only C	Only D	Only E	Only F
A	C1	$A^{19,0}$	$A^{19,0}$	$A^{19,0}$	$B^{10,9}$	$C^{10,6}$	–	–	–
	C2	$A^{5,0}$	$A^{5,0}$	$A^{5,0}$	$B^{4,0}$	$C^{4,0}$	$D^{4,0}$	–	–
	R1	$A^{0,3}B^{1,6}C^{16,5}D^{0,7}$	$A^{0,3}B^{2,4}C^{15,4}D^{1,0}$	n.a.	$B^{30,1}$	$C^{19,1}$	$D^{14,0}$	$E^{12,7}$	–
	R2	$A^{5,0}$	$A^{5,0}$	$A^{5,0}$	$B^{4,0}$	$C^{3,0}$	$D^{2,7}$	–	–
	RC1	$A^{3,3}B^{6,8}C^{7,3}$	$A^{4,1}B^{7,4}C^{6,6}D^{0,1}$	$A^{44,0}$	$B^{22,1}$	$C^{13,6}$	$D^{12,9}$	–	–
	RC2	$A^{5,6}B^{4,5}C^{1,1}$	$A^{6,0}B^{4,1}C^{1,3}$	$A^{18,0}$	$B^{9,0}$	$C^{6,0}$	$D^{5,0}$	$E^{4,0}$	$F^{3,2}$
B	C1	$A^{5,7}B^{6,7}$	$A^{5,8}B^{6,6}$	$A^{19,0}$	$B^{11,1}$	$C^{10,6}$	–	–	–
	C2	$A^{5,0}$	$A^{4,7}B^{0,2}C^{<0,1}$	$A^{5,0}$	$B^{4,0}$	$C^{4,0}$	$D^{4,0}$	–	–
	R1	$A^{0,3}B^{1,1}C^{5,0}D^{7,3}E^{0,9}$	$A^{0,3}B^{1,1}C^{5,3}D^{7,7}E^{0,6}$	n.a.	$B^{30,3}$	$C^{19,3}$	$D^{14,2}$	$E^{12,8}$	–
	R2	$A^{5,0}$	$A^{5,0}$	$A^{5,0}$	$B^{4,0}$	$C^{3,0}$	$D^{2,6}$	–	–
	RC1	$A^{0,8}B^{3,8}C^{9,0}D^{0,9}$	$A^{0,8}B^{4,2}C^{8,0}D^{1,4}$	$A^{44,6}$	$B^{22,2}$	$C^{13,8}$	$D^{13,1}$	–	–
	RC2	$A^{0,9}B^{1,3}C^{4,9}$	$A^{1,5}B^{1,7}C^{3,6}D^{0,6}E^{<0,1}$	$A^{18,0}$	$B^{9,0}$	$C^{6,0}$	$D^{5,0}$	$E^{4,0}$	$F^{3,2}$
C	C1	$A^{9,0}B^{5,0}$	$A^{9,1}B^{5,0}$	$A^{19,0}$	$B^{11,0}$	$C^{10,5}$	–	–	–
	C2	$A^{2,9}B^{1,1}C^{0,4}$	$A^{3,3}B^{0,8}C^{0,4}$	$A^{5,0}$	$B^{4,0}$	$C^{4,0}$	$D^{4,0}$	–	–
	R1	$A^{0,1}B^{0,9}C^{4,1}D^{7,5}E^{1,5}$	$A^{0,2}B^{1,0}C^{4,4}D^{7,4}E^{1,3}$	n.a.	$B^{30,4}$	$C^{19,3}$	$D^{14,3}$	$E^{12,9}$	–
	R2	$A^{5,0}$	$A^{4,9}B^{0,1}C^{<0,1}$	$A^{5,0}$	$B^{4,0}$	$C^{3,1}$	$D^{2,7}$	–	–
	RC1	$A^{0,6}B^{2,9}C^{8,4}D^{2,1}$	$A^{0,7}B^{3,1}C^{7,8}D^{2,5}$	$A^{45,0}$	$B^{22,3}$	$C^{14,0}$	$D^{13,1}$	–	–
	RC2	$A^{0,9}B^{0,6}C^{4,1}D^{0,9}$	$A^{1,3}B^{1,3}C^{3,3}D^{1,0}E^{0,1}$	$A^{18,0}$	$B^{9,0}$	$C^{6,1}$	$D^{5,0}$	$E^{4,1}$	$F^{3,3}$

Table 5.5: Average fleet mix from the best known solutions (BKS) and average results of 10 runs using all or only a single vehicle type (A,B,C,D,E,F) for all E-FSMFTW instances.

6. Conclusion

In this paper, we introduced a new fleet composition and routing problem using electric vehicles: the E-FSMFTW considers multiple vehicle types differing in their capacities and costs. Each vehicle has the possibility of recharging on tour at dedicated recharging stations. Each recharge operation consumes an amount of time depending on the distance travelled and the resulting battery charge. This adds additional complexity to the problem since time windows at customer locations have to be observed.

We proposed a hybrid solution method based on Adaptive Large Neighbourhood Search (ALNS) extended with an embedded intensification mechanism. This intensification was done using intelligent local search for route improvement and efficient labelling procedures for the optimal placement of charging stations. For defining the problem rigorously, we also provided a MIP formulation and presented a Set-Partitioning formulation which was solved using a branch-and-price procedure. We constructed a new benchmark set for the E-FSMFTW and performed extensive computational experiments with our branch-and-price solver and our extended ALNS. On small instances, the ALNS was able to find all optimal solutions. We also presented results of our algorithms on larger instances, comparable in size to those of benchmark instances published for related problems. For these our heuristic solver is able to find solutions within a gap of around one percent to the best known solution. Several of these best known solution have been proven optimal by our branch-and-price procedure.

A sensitivity analysis was conducted, showing the expected benefit of considering a fleet mix of different vehicle types. Furthermore, usage of recharging stations in the obtained results was investigated. Our computational study shows that not even half of the recharging stations are used in the current benchmark instances, and at most one is visited per vehicle in good solutions.

Since our work extends the research on E-VRPTW, it also shares some of the limiting assumptions of this model. We assume that the vehicle is always charged to full capacity every time a vehicle reaches a recharging station. While this can be partly defended by the fact that in reality drivers will prefer to do this rather than to leave with a partially charged vehicle, it would be interesting to relax this limiting assumption. Furthermore, like most of the vehicle routing literature we assume that the resource (energy) consumption rate is constant and does not depend on the carrying load of the vehicles. A more realistic modeling of this aspect e.g. in the sense of the pollution-routing problem (Bektas and Laporte, 2011) will be one of the subjects of future work focusing on extending and applying our model to real world cases.

type	name		BKS	All	Only A	Only B	Only C	Only D	Only E	Only F
A	C1	$\#rs$	17,22	16,91	16,61	9,43	8,64	–	–	–
		$\{rs\}$	8,00	8,57	8,28	6,67	6,30	–	–	–
		$\overline{rs}$	0,91	0,89	0,87	0,87	0,82	–	–	–
	C2	$\#rs$	6,38	6,83	6,70	5,44	4,21	3,43	–	–
		$\{rs\}$	5,63	6,30	6,21	5,05	3,99	3,24	–	–
		$\overline{rs}$	1,28	1,37	1,34	1,36	1,05	0,86	–	–
	R1	$\#rs$	17,83	19,23	n.a.	26,48	19,21	15,47	13,20	–
		$\{rs\}$	11,83	12,02	n.a.	13,38	12,09	11,06	10,21	–
		$\overline{rs}$	0,93	1,00	–	0,88	1,00	1,10	1,04	–
	R2	$\#rs$	2,82	3,01	2,94	2,99	3,43	3,42	–	–
		$\{rs\}$	2,73	2,91	2,82	2,86	3,25	3,26	–	–
		$\overline{rs}$	0,56	0,60	0,59	0,75	1,14	1,28	–	–
	RC1	$\#rs$	18,13	18,12	46,31	23,13	16,98	13,49	–	–
		$\{rs\}$	9,50	9,89	9,23	10,04	10,16	9,22	–	–
		$\overline{rs}$	1,05	1,00	1,05	1,05	1,25	1,04	–	–
	RC2	$\#rs$	0,25	0,48	0,64	0,70	2,03	2,39	3,66	4,33
		$\{rs\}$	0,25	0,48	0,59	0,66	1,84	2,28	3,43	4,04
		$\overline{rs}$	0,03	0,04	0,04	0,08	0,34	0,48	0,92	1,38
B	C1	$\#rs$	9,67	9,78	15,96	9,68	8,50	–	–	–
		$\{rs\}$	6,67	6,66	8,09	6,23	6,23	–	–	–
		$\overline{rs}$	0,69	0,70	0,84	0,88	0,81	–	–	–
	C2	$\#rs$	5,75	6,51	6,58	5,46	3,76	3,25	–	–
		$\{rs\}$	5,63	6,08	6,09	5,13	3,71	3,18	–	–
		$\overline{rs}$	1,15	1,32	1,32	1,37	0,94	0,81	–	–
	R1	$\#rs$	14,42	15,16	n.a.	26,23	18,69	14,55	12,52	–
		$\{rs\}$	11,25	11,27	n.a.	13,33	12,02	10,64	9,90	–
		$\overline{rs}$	0,98	1,01	–	0,87	0,97	1,02	0,98	–
	R2	$\#rs$	2,91	2,92	2,75	2,85	3,43	3,46	–	–
		$\{rs\}$	2,82	2,73	2,61	2,74	3,27	3,35	–	–
		$\overline{rs}$	0,58	0,58	0,55	0,71	1,14	1,31	–	–
	RC1	$\#rs$	16,75	16,23	46,11	23,07	16,93	12,89	–	–
		$\{rs\}$	9,50	9,67	9,38	9,76	9,91	9,05	–	–
		$\overline{rs}$	1,17	1,12	1,03	1,04	1,23	0,99	–	–
	RC2	$\#rs$	1,13	1,25	0,68	0,40	1,86	2,36	3,41	3,90
		$\{rs\}$	0,88	1,23	0,56	0,38	1,69	2,16	3,31	3,71
		$\overline{rs}$	0,19	0,19	0,04	0,04	0,31	0,47	0,85	1,22
C	C1	$\#rs$	9,44	9,70	15,41	9,64	8,28	–	–	–
		$\{rs\}$	6,33	6,56	7,68	6,11	6,18	–	–	–
		$\overline{rs}$	0,77	0,79	0,81	0,87	0,78	–	–	–
	C2	$\#rs$	5,13	5,99	6,13	5,43	3,89	3,30	–	–
		$\{rs\}$	5,13	5,65	5,65	5,08	3,79	3,20	–	–
		$\overline{rs}$	1,18	1,33	1,22	1,36	0,97	0,83	–	–
	R1	$\#rs$	15,00	15,04	n.a.	26,24	18,38	14,56	12,33	–
		$\{rs\}$	10,58	11,09	n.a.	13,33	12,02	10,85	9,62	–
		$\overline{rs}$	1,04	1,04	–	0,86	0,95	1,02	0,95	–
	R2	$\#rs$	2,64	2,66	2,86	2,74	3,34	3,42	–	–
		$\{rs\}$	2,45	2,55	2,75	2,63	3,25	3,31	–	–
		$\overline{rs}$	0,53	0,53	0,57	0,68	1,08	1,30	–	–
	RC1	$\#rs$	15,63	15,45	45,00	22,51	16,86	12,77	–	–
		$\{rs\}$	9,50	9,39	9,39	9,79	9,44	8,85	–	–
		$\overline{rs}$	1,10	1,09	1,00	1,01	1,21	0,97	–	–
	RC2	$\#rs$	1,38	1,58	0,55	0,51	1,81	2,38	3,19	3,66
		$\{rs\}$	1,25	1,53	0,53	0,49	1,71	2,15	3,09	3,47
		$\overline{rs}$	0,25	0,25	0,03	0,06	0,30	0,48	0,78	1,13
avg		$\#rs$	9,03	9,27	14,35	11,27	8,90	7,41	8,05	3,96
		$\{rs\}$	6,11	6,37	5,32	6,31	6,16	5,72	6,59	3,74
		$\overline{rs}$	0,80	0,83	0,75	0,82	0,91	0,93	0,92	1,24

Table 5.6: Average total number of recharging stations used ($\#rs$), number of distinct recharging stations used ($\{rs\}$), and average number of recharging stations visited per vehicle ($\overline{rs}$) from the best known solutions (BKS) and average results of 10 runs using all or only a single vehicle type (A,B,C,D,E,F) for all E-FSMFTW instances.

Acknowledgments

Thanks are due to our colleagues at the Austrian Institute of Technology and the University of Vienna as well as the participants of conference VeRoLog 2013 and the workshop AWGM 2013 (where preliminary results were presented) for useful discussions and suggestions. We would also like to thank the reviewers for their valuable feedback which helped to improve the overall quality of this paper.

type	name	BKS	All	Only A	Only B	Only C	Only D	Only E	Only F
A	C1	7116,04	7132,50	7129,85	9182,20	13988,17	–	–	–
	C2	5676,01	5686,60	5686,13	6269,85	8636,52	10779,75	–	–
	R1	4044,53	4077,80	n.a.	4368,87	4097,34	4568,51	6810,36	–
	R2	3144,30	3170,36	3168,11	3720,51	4540,06	7501,06	–	–
	RC1	4871,55	4928,88	6055,38	5240,69	5154,38	6504,40	–	–
	RC2	4206,42	4226,43	4335,28	4301,09	4372,71	5082,41	5527,46	9030,47
dev	min		0,19%	0,18%	2,25%	1,31%	12,96%	31,41%	114,68%
	avg		0,62%	5,70%	12,61%	34,03%	59,16%	49,89%	114,68%
B	C1	2413,50	2421,93	2562,76	2622,39	3572,68	–	–	–
	C2	1675,58	1687,79	1684,81	1783,23	2233,35	2684,60	–	–
	R1	1812,56	1830,90	n.a.	2445,79	1962,27	1878,99	2293,91	–
	R2	1333,01	1350,26	1352,64	1450,82	1638,60	2246,12	–	–
	RC1	2071,01	2094,06	3940,12	2596,31	2137,55	2367,54	–	–
	RC2	1683,40	1707,95	2173,45	1766,28	1716,46	1850,72	1974,44	2748,00
dev	min		0,35%	0,55%	4,92%	1,96%	3,66%	17,29%	63,24%
	avg		0,99%	25,51%	14,86%	19,61%	31,33%	21,92%	63,24%
C	C1	1712,62	1719,22	1987,72	1797,05	2269,90	–	–	–
	C2	1168,94	1177,18	1176,21	1222,31	1430,15	1667,08	–	–
	R1	1488,37	1504,60	n.a.	2205,23	1694,02	1539,12	1722,84	–
	R2	1107,74	1126,69	1125,50	1170,56	1274,69	1591,38	–	–
	RC1	1693,89	1713,18	3671,52	2264,07	1754,29	1842,05	–	–
	RC2	1354,96	1374,39	1902,49	1450,66	1385,43	1442,51	1522,21	1955,02
dev	min		0,39%	0,62%	4,57%	2,25%	3,41%	12,34%	44,29%
	avg		1,08%	35,09%	17,34%	14,93%	20,98%	14,05%	44,29%

Table 5.7: Average objective value from the best known solutions (BKS) and average results of 10 runs using all or only a single vehicle type (A,B,C,D,E,F) for all E-FSMFTW instances solved using no energy restrictions.

type	variant		BKS	All	Only A	Only B	Only C	Only D	Only E	Only F
A	FSM	*obj*	4735,68	4763,13	5173,46	5401,52	6612,70	6729,07	6297,20	9030,47
		m	12,4	12,6	17,3	13,9	9,4	7,5	8,4	3,1
	EFSM	*obj*	4776,56	4803,70	5196,57	5510,81	6897,89	7042,40	6756,10	9145,25
		m	12,9	13,1	17,3	14,0	9,8	7,9	9,2	3,2
B	FSM	*obj*	1813,84	1831,06	2280,25	2108,51	2186,16	2180,38	2166,12	2748,00
		m	9,5	9,7	17,4	13,9	9,5	7,5	8,5	3,2
	EFSM	*obj*	1855,83	1877,10	2299,50	2153,71	2266,33	2260,97	2298,34	2775,84
		m	10,1	10,3	17,4	14,1	9,8	8,0	9,3	3,2
C	FSM	*obj*	1414,31	1429,28	1915,27	1696,58	1631,03	1608,25	1642,59	1955,02
		m	8,8	9,0	17,5	13,9	9,5	7,6	8,5	3,2
	EFSM	*obj*	1457,97	1475,78	1935,73	1731,97	1689,07	1661,47	1727,58	1969,90
		m	9,5	9,7	17,5	14,1	9,9	8,0	9,4	3,3
avg	FSM	*obj*	2654.61	2674.49	3122.99	3068.87	3476.63	3505.90	3368.64	4577.83
		m	10.2	10.4	17.4	13.9	9.5	7.5	8.5	3.2
	EFSM	*obj*	2696.79	2718.86	3143.93	3132.16	3617.76	3654.95	3594.01	4630.33
		m	10.9	11.0	17.4	14.1	9.8	8.0	9.3	3.2

Table 5.8: Average objective value (obj) and total vehicles used (m) from the best known solutions (BKS) and average results of 10 runs using all or only a single vehicle type (A,B,C,D,E,F) for all E-FSMFTW instances solved without (FSM) or with energy restriction (EFSM).

This work is partially funded by the Austrian Climate and Energy Fund within the "Electric Mobility Flagship Projects" program under grant 834868 (project VECEPT).

References

References

AustriaTech. 2010. Annex: Electric fleets in urban logistics, project ENCLOSE. URL `http://www.austriatech.at/files/get/9e26eb124ad90ffa93067085721d4942/austriatech_electricfleets_annex.pdf`. Last accessed 2015-07-21.

Baldacci, Roberto, Enrico Bartolini, Aristide Mingozzi, Roberto Roberti. 2010. An exact solution framework for a broad class of vehicle routing problems. *Computational Management Science* **7**(3) 229–268.

Baldacci, Roberto, Maria Battarra, Daniele Vigo. 2008. Routing a heterogeneous fleet of vehicles. Bruce Golden, Subramanian Raghavan, Edward Wasil, eds., *The Vehicle Routing Problem: Latest Advances and New Challenges*, *Operations Research/Computer Science Interfaces*, vol. 43. Springer, 3–27. doi:10.1007/978-0-387-77778-8_1. URL `http://dx.doi.org/10.1007/978-0-387-77778-8\_1`.

Bektas, Tolga, Gilbert Laporte. 2011. The pollution-routing problem. *Transportation Research Part B: Methodological* **45**(8) 1232–1250.

Bräysy, Olli, Wout Dullaert, Geir Hasle, David I. Mester, Michel Gendreau. 2008a. An effective multirestart deterministic annealing metaheuristic for the fleet size and mix vehicle-routing problem with time windows. *Transportation Science* **42**(3) 371–386.

Bräysy, Olli, Michel Gendreau. 2005a. Vehicle routing problem with time windows, part i: Route construction and local search algorithms. *Transportation Science* **39**(1) 104–118.

Bräysy, Olli, Michel Gendreau. 2005b. Vehicle routing problem with time windows, part ii: Metaheuristics. *Transportation Science* **39**(1) 119–139.

Bräysy, Olli, P. Porkka, Wout Dullaert, Panagiotis P. Repoussis, Christos D. Tarantilis. 2008b. A well-scaleable metaheuristic for the fleet size and mix vehicle routing problem with time windows. *Expert Systems with Applications* **36**(4) 8460–8475.

Çağri Koç, Tolga Bektaş, Ola Jabali, Gilbert Laporte. 2014. The fleet size and mix pollution-routing problem. *Transportation Research Part B: Methodological* **70**(0) 239–254. doi:http://dx.doi.org/10.1016/j.trb.2014.09.008.

Clarke, G, JW Wright. 1964. Scheduling of vehicles from a central depot to a number of delivery points. *Operations research* **12**(4) 568–581.

Conrad, Ryan G., Miguel A. Figliozzi. 2011. The recharging vehicle routing problem. *Proceedings Industrial Engineering Research Conference. Reno, NV* .

Cordeau, Jean-François, Gilbert Laporte, Anne Mercier. 2001. A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operational research society* 928–936.

Crevier, Benoit, Jean-François Cordeau, Gilbert Laporte. 2007. The multi-depot vehicle routing problem with inter-depot routes. *European Journal of Operational Research* **176**(2) 756–773. doi:http://dx.doi.org/10.1016/j.ejor.2005.08.015. URL `http://www.sciencedirect.com/science/article/pii/S0377221705006983`.

Dekker, Rommert, Jacqueline Bloemhof, Ioannis Mallidis. 2011. Operations research for green logistics – an overview of aspects, issues, contributions and challenges. *European Journal of Operational Research* **219**(3) 671–679.

Dell'Amico, Mauro, Michele Monaci, Corrado Pagani, Daniele Vigo. 2007. Heuristic approaches for the fleet size and mix vehicle routing problem with time windows. *Transportation Science* **41**(4) 516–526. doi:10.1287/trsc.1070.0190. URL `http://dx.doi.org/10.1287/trsc.1070.0190`.

Desaulniers, Guy, Fausto Errico, Stefan Irnich, Michael Schneider. 2014. Exact algorithms for electric vehicle-routing problems with time windows. Tech. rep., Technical Report Les Cahiers du GERAD, G-2014-110, Montral, Canada. URL `https://www.gerad.ca/en/papers/G-2014-110/view`.

Desaulniers, Guy, François Lessard, Ahmed Hadjar. 2008. Tabu search, partial elementarity, and generalized k-path inequalities for the vehicle routing problem with time windows. *Transportation Science* **42**(3) 387–404.

Desrosiers, Jacques, Marco Lübbecke. 2010. Branch-price-and-cut algorithms. *Wiley Encyclopedia of Operations Research and Management Science (EORMS)*. John Wiley and Sons, Inc.

Di Gaspero, Luca, Andrea Schaerf. 2002. Multi-neighbourhood local search with application to course timetabling. Edmund Burke, Patrick De Causmaecker, eds., *Practice and Theory of Automated Timetabling IV* . Springer, DE, 262–275.

Dullaert, Wout, Gerrit K. Janssens, Kenneth Sörensen, Bert Vernimmen. 2002. New heuristics for the fleet size and mix vehicle routing problem with time windows. *Journal of the Operations Research Society* **53**(11) 1232–1238.

Erdoğan, Sevgi, Elise Miller-Hooks. 2012. A green vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review* **48**(1) 100–114. doi:http://dx.doi.org/10.1016/j.tre.2011.08.001. URL `http://www.sciencedirect.com/science/article/pii/S1366554511001062`. Select Papers from the 19th International Symposium on Transportation and Traffic Theory.

Feillet, Dominique, Pierre Dejax, Michel Gendreau, Cyrille Gueguen. 2004. An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks* **44**(3) 216–229.

Goeke, Dominik, Michael Schneider. 2015. Routing a mixed fleet of electric and conventional vehicles. *European Journal of Operational Research* **245**(1) 81–99.

Golden, Bruce, Arjang Assad, Larry Levy, Filip Gheysens. 1984. The fleet size and mix vehicle routing problem. *Computers & Operations Research* **11**(1) 49–66. doi:http://dx.doi.org/10.1016/0305-0548(84)90007-8. URL `http://www.sciencedirect.com/science/article/pii/0305054884900078`.

Hart, J.P., A.W. Shogan. 1987. Semi-greedy heuristics: An empirical study. *Operations Research Letters* **6**(3) 107–114.

Hiermann, Gerhard, Matthias Prandtstetter, Andrea Rendl, Jakob Puchinger, Günther R. Raidl. 2015. Metaheuristics for solving a multimodal home-healthcare scheduling problem. *Central European Journal of Operations Research* **23**(1) 89–113. doi:10.1007/s10100-013-0305-8.

Irnich, Stefan, Guy Desaulniers. 2005. Shortest path problems with resource constraints. Guy Desaulniers, Jacques Desrosiers, Marius M. Solomon, eds., *Column Generation*. Springer US, 33–65.

Kindervater, Gerard A. P., Martin W. P. Savelsbergh. 1997. Vehicle routing: Handling edge exchanges. Emile Aarts, Jan K. Lenstra, eds., *Local Search in Combinatorial Optimization*. John Wiley & Sons Ltd., 337–360.

Liu, Fuh-Hwa, Sheng-Yuan Shen. 1999. The fleet size and mix vehicle routing problem with time windows. *The Journal of the Operational Research Society* **50**(7) 721–732.

Nagata, Yuichi, Olli Bräysy, Wout Dullaert. 2010. A penalty-based edge assembly memetic algorithm for the vehicle routing problem with time windows. *Computers & Operations Research* **37**(4) 724–737. doi:10.1016/j.cor.2009.06.022. URL `http://dx.doi.org/10.1016/j.cor.2009.06.022`.

Paraskevopoulos, Dimitris C., Panagiotis P. Repoussis, Christos D. Tarantilis, George Ioannou, Gregory P. Prastacos. 2008. A reactive variable neighborhood tabu search for the heterogeneous fleet vehicle routing problem with time windows. *Journal of Heuristics* **14**(5) 425–455. doi:10.1007/s10732-007-9045-z. URL `http://dx.doi.org/10.1007/s10732-007-9045-z`.

Pelletier, Samuel, Ola Jabali, Gilbert Laporte. 2015. Goods distribution with electric vehicles: review and research perspectives. *Transportation Science* Accepted for publication.

Pisinger, David, Stefan Ropke. 2010. Large neighborhood search. Michel Gendreau, Jean-Yves Potvin, eds., *Handbook of Metaheuristics*, *International Series in Operations Research & Management Science*, vol. 146. Springer US, 399–419. doi:10.1007/978-1-4419-1665-5_13. URL `http://dx.doi.org/10.1007/978-1-4419-1665-5\_13`.

Potvin, Jean-Yves, Jean-Marc Rousseau. 1993. A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *European Journal of Operational Research* **66**(3) 331 – 340. doi:http://dx.doi.org/10.1016/0377-2217(93)90221-8. URL `http://www.sciencedirect.com/science/article/pii/0377221793902218`.

Repoussis, Panagiotis P., Christos D. Tarantilis. 2010. Solving the fleet size and mix vehicle routing problem with time windows via adaptive memory programming. *Transportation Research Part C: Emerging Technologies* **18**(5) 695–712.

Righini, Giovanni, Matteo Salani. 2006. Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. *Discrete Optimization* **3**(3) 255–273.

Ropke, Stefan, Jean-François Cordeau. 2009. Branch and cut and price for the pickup and delivery problem with time windows. *Transportation Science* **43**(3) 267–286.

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

Sbihi, Abdelkader, Richard W. Eglese. 2010. Combinatorial optimization and green logistics. *Annals of Operations Research* **175**(1) 159–175.

Schneider, Michael, Bastian Sand, Andreas Stenger. 2013. A note on the time travel approach for handling time windows in vehicle routing problems. *Computers & Operations Research* **40**(10) 2564 – 2568. doi:http://dx.doi.org/10.1016/j.cor.2013.02.002. URL `http://www.sciencedirect.com/science/article/pii/S0305054813000348`.

Schneider, Michael, Andreas Stenger, Dominik Goeke. 2014. The electric vehicle routing problem with time windows and recharging stations. *Transportation Science* **48**(4) 500–520.

Shaw, Paul. 1998. Using constraint programming and local search methods to solve vehicle routing problems. *Proceedings of the 4th International Conference on Principles and Practice of Constraint Programming*. CP '98, Springer-Verlag, London, UK, 417–431. URL `http://dl.acm.org/citation.cfm?id=647485.726320`.

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

Toth, Paolo, Daniele Vigo, eds. 2014. *Vehicle Routing: Problems, Methods, and Applications. SIAM, 2nd edition*. MOS-SIAM Series on Optimization., Philadelphia, PA, USA.

Trick, Michael A. 1992. A linear relaxation heuristic for the generalized assignment problem. *Naval Research Logistics (NRL)* **39**(2) 137–151. doi:10.1002/1520-6750(199203)39:2⟨137::AID-NAV3220390202⟩3.0.CO;2-D. URL `http://dx.doi.org/10.1002/1520-6750(199203)39:2<137::AID-NAV3220390202>3.0.CO;2-D`.

Vidal, Thibaut, Teodor G. Crainic, Michel Gendreau, Christian Prins. 2013a. Heuristics for multi-attribute vehicle routing problems: A survey and synthesis. *European Journal of Operational Research* **231**(1) 1–21.

Vidal, Thibaut, Teodor G. Crainic, Michel Gendreau, Christian Prins. 2013b. A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Computers & Operations Research* **40**(1) 475–489. doi:10.1016/j.cor.2012.07.018. URL `http://dx.doi.org/10.1016/j.cor.2012.07.018`.

Appendix A. Detailed Branch-and-Price Results

E-VRPTW. Table A.1 shows more detailed results for the E-VRPTW instances by Schneider et al. (2014). An upper bound of the best feasible integer solution (m,obj) found during the search is presented where available. If the upper bound is proven optimal, the field of column (opt.) is marked with an X.

The BnP approach calculated 24 lower bounds, 20 upper bounds, from which 6 are proven to be optimal.

name	LB	m	obj	opt.	t[s]	name	LB	m	obj	opt.	t[s]
c101	121053.83	12	1053.83	X	41.84	r107	109339.714	12	1170.87		28800.00
c102	111051.38	11	1051.38	X	6253.38	r109	115959.121	13	1311.19		28800.00
c105	111075.37	11	1075.37	X	5513.96	r110	105679.642	-	-		28800.00
c106	102267.114	11	1063.11		28800.00	r111	107582.835	12	1145.34		28800.00
c107	100035.954	11	1083.12		28800.00	rc101	146056.597	15	1823.23		28800.00
c108	98768.154	11	1150.51		28800.00	rc102	134062.462	14	1661.76		28800.00
c109	95413.933	-	-		28800.00	rc103	117271.134	13	1366.96		28800.00
r101	171859.51	17	1859.51	X	261.63	rc104	103517.437	-	-		28800.00
r102	151659.87	15	1659.87	X	1626.49	rc105	128516.809	14	1502.65		28800.00
r103	131267.35	13	1267.35	X	8827.98	rc106	121502.420	13	1425.70		28800.00
r105	134710.929	14	1471.98		28800.00	rc107	108830.543	13	1431.56		28800.00
r106	125294.609	13	1321.08		28800.00	rc108	103567.304	-	-		28800.00

Table A.1: Results for the E-VRPTW instances using branch-and-price. Only instances where a lower bound (LB) was calculated inside the time limit of eight hours are presented.

E-FSMFTW. Table A.1 shows more detailed results for the E-FSMFTW instances used in our studies. An upper bound of the best feasible integer solution (obj) found during the search is presented where available. If the upper bound is proven optimal, the field of column (opt.) is marked with an X.

Our BnP approach calculated 75 lower bounds, 51 upper bounds with 7 also proven to be optimal.

To results for each instance of the proposed new benchmark set is shown in Table A.3, A.4 and A.5. Here we show the best known solution obtained using BnP or any run in our experiments, the average of 10 runs using our algorithm as well as the fleet mix.

To complete our results, we also show the results obtained for the smaller benchmarks with only 5 and 10 customers similar to Table 5.2 in Table A.6 and A.7.

The additional results on the fleet composition using the E-FSMFTW instances but without the energy restriction is shown in Table A.8.

type	name	LB	obj	opt.	t[s]	type	name	LB	obj	opt.	t[s]	type	name	LB	obj	opt.	t[s]
A	c101	6837.241	7236.512		28800.00	B	c101	2421.263	2527.576		28800.00	C	c101	1809.935	1809.935	X	2123.43
	c102	6798.166	-		28800.00		c102	2365.057	2561.651		28800.00		c102	1724.010	1763.266		28800.00
	c103	6790.929	-		28800.00		c103	2344.125	-		28800.00		c103	1694.208	-		28800.00
	c105	6819.319	7220.868		28800.00		c105	2381.633	2479.885		28800.00		c105	1736.300	1783.977		28800.00
	c106	6808.444	7209.730		28800.00		c106	2364.513	2562.269		28800.00		c106	1720.111	1791.293		28800.00
	c107	6807.764	7185.731		28800.00		c107	2361.278	2565.659		28800.00		c107	1709.232	1783.601		28800.00
	c108	6798.539	7194.518		28800.00		c108	2349.746	2604.617		28800.00		c108	1701.279	1780.915		28800.00
	c109	6786.900	7181.010		28800.00		c109	2324.910	-		28800.00		c109	1666.582	1843.747		28800.00
	r101	4318.924	4379.129		28800.00		r101	2249.140	2249.140	X	2655.46		r101	1954.003	1954.003	X	503.62
	r102	4136.216	4195.824		28800.00		r102	2047.886	2047.886	X	16489.06		r102	1757.128	1757.128	X	2468.54
	r103	4008.244	-		28800.00		r103	1858.405	1900.629		28800.00		r103	1555.924	1597.732		28800.00
	r105	4101.137	4178.399		28800.00		r105	1997.751	1997.751	X	17231.29		r105	1699.339	1699.339	X	6299.56
	r106	4036.996	-		28800.00		r106	1894.424	1932.634		28800.00		r106	1588.263	1604.546		28800.00
	r107	3963.154	-		28800.00		r107	1765.407	-		28800.00		r107	1449.559	-		28800.00
	r109	3979.568	4234.333		28800.00		r109	1820.195	1948.187		28800.00		r109	1510.929	1550.398		28800.00
	r110	3926.277	-		28800.00		r110	1715.231	-		28800.00		r110	1392.624	-		28800.00
	r111	3929.345	-		28800.00		r111	1722.599	-		28800.00		r111	1401.570	1438.812		28800.00
	rc101	5143.233	5272.301		28800.00		rc101	2419.133	2514.501		28800.00		r112	1334.615	-		28800.00
	rc102	4979.408	5175.751		28800.00		rc102	2255.876	2374.218		28800.00		rc101	2051.311	2134.295		28800.00
	rc103	4807.277	-		28800.00		rc103	2037.336	-		28800.00		rc102	1890.929	1947.705		28800.00
	rc105	4946.539	5193.986		28800.00		rc105	2197.437	2266.204		28800.00		rc103	1666.695	1881.321		28800.00
	rc106	4881.093	6069.702		28800.00		rc106	2130.475	2226.335		28800.00		rc104	1525.902	-		28800.00
	rc107	4740.984	-		28800.00		rc107	1955.423	2311.475		28800.00		rc105	1823.419	1911.277		28800.00
	rc108	4704.022	-		28800.00		rc108	1892.580	-		28800.00		rc106	1759.369	1829.856		28800.00
													rc107	1581.566	-		28800.00
													rc108	1509.491	-		28800.00
													rc201	1580.153	1595.270		28800.00

Table A.2: Results for the E-FSMFTW instances using branch-and-price. Only instances where a lower bound (LB) was calculated inside the time limit of eight hours are presented

	BKS			ALNS_{2000}		
name	obj_{LB}	obj	mix	$\overline{obj}$	$\overline{mix}$	t[m]
c101	6837,24	7180,42	$A^{19,0}$	7190,21	$A^{19,0}$	17,53
c102	6798,17	7154,50	$A^{19,0}$	7162,24	$A^{19,0}$	17,95
c103	6790,93	7126,29	$A^{19,0}$	7149,31	$A^{19,0}$	18,30
c104	–	7100,22	$A^{19,0}$	7110,43	$A^{19,0}$	17,75
c105	6819,32	7155,23	$A^{19,0}$	7182,56	$A^{19,0}$	17,60
c106	6808,44	7146,88	$A^{19,0}$	7168,94	$A^{19,0}$	17,69
c107	6807,76	7156,18	$A^{19,0}$	7171,04	$A^{19,0}$	17,42
c108	6798,54	7141,49	$A^{19,0}$	7153,77	$A^{19,0}$	17,53
c109	6786,90	7120,33	$A^{19,0}$	7132,19	$A^{19,0}$	17,32
c201	–	5737,57	$A^{5,0}$	5757,53	$A^{5,0}$	39,28
c202	–	5744,65	$A^{5,0}$	5765,52	$A^{5,0}$	42,22
c203	–	5726,08	$A^{5,0}$	5751,99	$A^{5,0}$	47,69
c204	–	5705,82	$A^{5,0}$	5727,18	$A^{5,0}$	50,91
c205	–	5703,48	$A^{5,0}$	5725,41	$A^{5,0}$	46,16
c206	–	5708,77	$A^{5,0}$	5714,39	$A^{5,0}$	31,20
c207	–	5697,99	$A^{5,0}$	5713,44	$A^{5,0}$	32,58
c208	–	5685,40	$A^{5,0}$	5707,65	$A^{5,0}$	50,23
r101	4318,92	* 4379,13	$A^{1,0}B^{12,0}C^{11,0}$	4465,51	$A^{1,0}B^{10,9}C^{11,6}D^{0,1}$	14,47
r102	4136,22	* 4195,82	$B^{6,0}C^{15,0}$	4270,92	$A^{0,2}B^{5,8}C^{14,6}D^{0,3}$	14,97
r103	4008,24	4103,35	$B^{4,0}C^{16,0}$	4130,86	$A^{0,2}B^{1,8}C^{16,6}D^{0,5}$	16,43
r104	–	4007,28	$B^{1,0}C^{18,0}$	4025,60	$A^{0,2}B^{1,1}C^{16,4}D^{0,9}$	15,27
r105	4101,14	* 4178,40	$B^{4,0}C^{16,0}$	4215,34	$A^{0,3}B^{2,8}C^{16,7}D^{0,1}$	15,37
r106	4037,00	4120,23	$A^{1,0}B^{1,0}C^{16,0}D^{1,0}$	4155,24	$A^{0,3}B^{1,3}C^{16,8}D^{0,6}$	15,54
r107	3963,15	4057,06	$C^{17,0}D^{1,0}$	4093,59	$A^{0,1}B^{1,0}C^{15,6}D^{1,5}$	15,30
r108	–	3992,57	$A^{1,0}C^{18,0}$	4025,75	$A^{0,3}B^{0,8}C^{15,0}D^{1,9}$	15,77
r109	3979,57	4067,14	$C^{17,0}D^{1,0}$	4110,98	$A^{0,2}B^{1,1}C^{16,3}D^{1,0}$	15,58
r110	3926,28	4024,71	$A^{1,0}C^{18,0}$	4045,96	$A^{0,2}B^{0,1}C^{16,1}D^{1,5}$	15,73
r111	3929,35	4023,38	$C^{17,0}D^{1,0}$	4048,42	$A^{0,2}B^{0,6}C^{16,6}D^{1,0}$	15,93
r112	–	4001,87	$A^{1,0}C^{15,0}D^{2,0}$	4023,01	$A^{0,2}B^{0,1}C^{15,2}D^{2,1}$	15,80
r201	–	3413,93	$A^{5,0}$	3432,83	$A^{5,0}$	42,20
r202	–	3270,49	$A^{5,0}$	3295,26	$A^{5,0}$	44,95
r203	–	3136,47	$A^{5,0}$	3169,97	$A^{5,0}$	49,40
r204	–	3008,01	$A^{5,0}$	3026,09	$A^{5,0}$	46,32
r205	–	3234,26	$A^{5,0}$	3261,16	$A^{5,0}$	40,89
r206	–	3172,50	$A^{5,0}$	3194,12	$A^{5,0}$	47,73
r207	–	3079,39	$A^{5,0}$	3099,52	$A^{5,0}$	46,87
r208	–	3010,51	$A^{5,0}$	3026,57	$A^{5,0}$	51,26
r209	–	3142,72	$A^{5,0}$	3161,57	$A^{5,0}$	45,06
r210	–	3110,90	$A^{5,0}$	3143,79	$A^{5,0}$	45,94
r211	–	3041,93	$A^{5,0}$	3079,24	$A^{5,0}$	44,29
rc101	5143,23	* 5272,30	$A^{7,0}B^{13,0}C^{3,0}$	5346,49	$A^{6,8}B^{12,5}C^{3,4}$	14,13
rc102	4979,41	5121,53	$A^{7,0}B^{9,0}C^{5,0}$	5180,03	$A^{6,2}B^{9,3}C^{5,2}$	14,63
rc103	4807,28	4958,51	$A^{4,0}B^{5,0}C^{8,0}$	5007,37	$A^{3,8}B^{7,1}C^{6,7}D^{0,2}$	14,53
rc104	–	4804,00	$B^{3,0}C^{10,0}$	4862,65	$A^{2,3}B^{3,0}C^{9,5}$	16,03
rc105	4946,54	5074,43	$A^{4,0}B^{7,0}C^{7,0}$	5117,09	$A^{4,6}B^{9,5}C^{5,5}$	14,48
rc106	4881,09	5028,28	$A^{2,0}B^{8,0}C^{7,0}$	5102,46	$A^{3,2}B^{9,5}C^{5,9}$	14,69
rc107	4740,98	4864,78	$A^{2,0}B^{4,0}C^{9,0}$	4913,90	$A^{2,6}B^{4,5}C^{8,6}$	15,24
rc108	4704,02	4814,33	$A^{4,0}B^{3,0}C^{9,0}$	4862,41	$A^{2,8}B^{2,8}C^{9,2}D^{0,2}$	15,64
rc201	–	4346,25	$A^{8,0}B^{5,0}$	4361,17	$A^{8,3}B^{4,7}C^{0,1}$	14,27
rc202	–	4273,74	$A^{6,0}B^{6,0}$	4295,27	$A^{7,8}B^{4,8}C^{0,2}$	14,59
rc203	–	4152,94	$A^{7,0}B^{4,0}C^{1,0}$	4186,28	$A^{5,7}B^{4,8}C^{0,9}$	15,98
rc204	–	4113,49	$A^{3,0}B^{3,0}C^{3,0}$	4127,11	$A^{3,8}B^{4,1}C^{2,0}$	19,18
rc205	–	4246,52	$A^{6,0}B^{6,0}$	4273,59	$A^{7,9}B^{4,9}C^{0,1}$	14,86
rc206	–	4237,75	$A^{5,0}B^{5,0}C^{1,0}$	4270,25	$A^{5,7}B^{5,1}C^{0,7}$	15,09
rc207	–	4177,23	$A^{2,0}B^{8,0}$	4199,60	$A^{3,9}B^{4,8}C^{1,5}$	16,20
rc208	–	4097,04	$A^{2,0}B^{2,0}C^{4,0}$	4122,12	$A^{3,3}B^{3,9}C^{2,3}$	18,13
avg.		4776,24	$A^{6,1}B^{2,1}C^{4,7}D^{0,1}$	4803,80	$A^{6,2}B^{2,2}C^{4,5}$	25,68
dev.				0,58		

Table A.3: Results for the E-FSMFTW instances, vehicle type A. * mark solutions found by BnP only. The deviation from the best known solution (BKS) is in percent.

	BKS			ALNS$_{2000}$		
name	obj$_{LB}$	obj	mix	$\overline{obj}$	$\overline{mix}$	t[m]
c101	2421,26	2495,00	$A^{9,0}B^{5,0}$	2505,73	$A^{9,6}B^{4,7}$	14,43
c102	2365,06	2445,99	$A^{9,0}B^{5,0}$	2450,73	$A^{9,0}B^{5,0}$	14,41
c103	2344,13	2438,54	$A^{9,0}B^{5,0}$	2452,40	$A^{9,0}B^{5,0}$	15,03
c104	–	2404,97	$A^{9,0}B^{5,0}$	2428,95	$A^{9,0}B^{5,0}$	15,07
c105	2381,63	2472,93	$A^{9,0}B^{5,0}$	2475,95	$A^{9,2}B^{4,9}$	14,45
c106	2364,51	2462,54	$A^{9,0}B^{5,0}$	2468,13	$A^{9,2}B^{4,9}$	14,71
c107	2361,28	2458,37	$A^{9,0}B^{5,0}$	2461,32	$A^{9,0}B^{5,0}$	14,60
c108	2349,75	2450,17	$A^{9,0}B^{5,0}$	2463,02	$A^{9,4}B^{4,8}$	14,33
c109	2324,91	2436,41	$A^{9,0}B^{5,0}$	2452,57	$A^{9,8}B^{4,6}$	15,07
c201	–	1730,41	$A^{1,0}B^{3,0}$	1739,26	$A^{4,6}B^{0,3}$	35,76
c202	–	1737,57	$A^{5,0}$	1745,24	$A^{4,3}B^{0,4}C^{0,1}$	38,14
c203	–	1716,29	$A^{5,0}$	1742,76	$A^{4,3}B^{0,4}C^{0,1}$	39,38
c204	–	1699,07	$A^{1,0}B^{3,0}$	1709,43	$A^{3,8}B^{0,9}$	37,02
c205	–	1697,01	$A^{5,0}$	1715,09	$A^{5,0}$	37,83
c206	–	1693,15	$A^{5,0}$	1712,38	$A^{4,4}B^{0,2}C^{0,2}$	36,78
c207	–	1694,61	$A^{5,0}$	1710,70	$A^{4,0}B^{0,5}C^{0,2}$	35,05
c208	–	1681,47	$A^{5,0}$	1707,11	$A^{4,4}B^{0,2}C^{0,2}$	36,46
r101	2249,14	* 2249,14	$A^{2,0}B^{2,0}C^{12,0}D^{4,0}$	2281,28	$A^{1,0}B^{4,3}C^{12,6}D^{2,9}$	13,68
r102	2047,89	* 2047,89	$B^{2,0}C^{9,0}D^{6,0}$	2095,87	$A^{0,5}B^{2,4}C^{11,9}D^{3,9}$	14,45
r103	1858,41	1894,98	$B^{2,0}C^{4,0}D^{9,0}$	1927,52	$A^{0,3}B^{0,9}C^{6,4}D^{7,8}E^{0,1}$	15,11
r104	–	1747,65	$C^{1,0}D^{10,0}E^{1,0}$	1775,33	$A^{0,2}B^{0,2}C^{2,4}D^{8,8}E^{1,2}$	15,28
r105	1997,75	* 1997,75	$B^{2,0}C^{9,0}D^{6,0}$	2030,12	$A^{0,3}B^{1,1}C^{8,7}D^{6,4}E^{0,1}$	15,08
r106	1894,42	* 1932,63	$B^{2,0}C^{3,0}D^{10,0}$	1963,88	$A^{0,3}B^{1,1}C^{6,6}D^{7,3}E^{0,3}$	14,51
r107	1765,41	1824,88	$B^{1,0}C^{2,0}D^{9,0}E^{1,0}$	1844,20	$B^{0,7}C^{2,4}D^{8,9}E^{1,1}$	15,50
r108	–	1729,18	$C^{1,0}D^{10,0}E^{1,0}$	1753,09	$A^{0,1}B^{0,1}C^{1,5}D^{9,3}E^{1,3}$	16,27
r109	1820,20	1871,54	$B^{1,0}C^{2,0}D^{11,0}$	1904,12	$A^{0,2}B^{0,6}C^{4,5}D^{9,0}E^{0,3}$	15,37
r110	1715,23	1759,69	$D^{11,0}E^{1,0}$	1793,48	$A^{0,3}B^{0,1}C^{3,3}D^{9,0}E^{0,8}$	15,54
r111	1722,60	1786,97	$A^{1,0}C^{3,0}D^{7,0}E^{2,0}$	1808,36	$A^{0,2}B^{0,1}C^{3,4}D^{9,1}E^{0,7}$	15,66
r112	–	1721,79	$D^{11,0}E^{1,0}$	1746,02	$B^{0,1}C^{1,5}D^{9,7}E^{1,1}$	15,77
r201	–	1594,58	$A^{5,0}$	1618,25	$A^{5,0}$	31,21
r202	–	1468,05	$A^{5,0}$	1479,42	$A^{5,0}$	29,61
r203	–	1340,00	$A^{5,0}$	1354,24	$A^{5,0}$	30,70
r204	–	1203,89	$A^{5,0}$	1211,63	$A^{5,0}$	27,35
r205	–	1430,70	$A^{5,0}$	1455,08	$A^{5,0}$	30,16
r206	–	1361,69	$A^{5,0}$	1376,34	$A^{5,0}$	31,35
r207	–	1256,22	$A^{5,0}$	1268,66	$A^{5,0}$	28,18
r208	–	1198,39	$A^{5,0}$	1208,89	$A^{5,0}$	29,02
r209	–	1333,33	$A^{5,0}$	1345,50	$A^{5,0}$	30,30
r210	–	1314,16	$A^{5,0}$	1324,07	$A^{5,0}$	30,07
r211	–	1231,38	$A^{5,0}$	1244,73	$A^{5,0}$	26,30
rc101	2419,13	* 2514,50	$A^{2,0}B^{10,0}C^{7,0}$	2560,33	$A^{1,6}B^{8,9}C^{6,7}D^{0,9}$	13,71
rc102	2255,88	2330,50	$A^{2,0}B^{4,0}C^{10,0}$	2359,92	$A^{2,4}B^{4,8}C^{9,1}D^{0,2}$	14,35
rc103	2037,34	2105,84	$A^{1,0}B^{4,0}C^{7,0}D^{2,0}$	2136,78	$A^{1,4}B^{3,6}C^{6,8}D^{2,3}$	14,14
rc104	–	1986,35	$A^{2,0}C^{9,0}D^{2,0}$	2002,33	$A^{0,6}B^{1,4}C^{7,1}D^{3,1}$	15,56
rc105	2197,44	2259,97	$B^{7,0}C^{6,0}D^{2,0}$	2287,95	$A^{0,7}B^{6,0}C^{6,8}D^{1,7}$	13,87
rc106	2130,48	2209,73	$B^{4,0}C^{9,0}D^{1,0}$	2232,05	$A^{0,7}B^{4,0}C^{8,7}D^{1,1}$	14,50
rc107	1955,42	2037,25	$C^{11,0}D^{1,0}$	2050,20	$A^{0,3}B^{1,2}C^{9,2}D^{1,8}$	15,43
rc108	1892,58	1962,87	$B^{2,0}C^{7,0}D^{3,0}$	1995,41	$A^{0,1}B^{1,3}C^{7,6}D^{2,9}$	14,99
rc201	–	1899,99	$A^{3,0}B^{6,0}C^{1,0}$	1931,42	$A^{3,3}B^{4,0}C^{2,1}D^{0,1}$	15,69
rc202	–	1807,30	$A^{3,0}B^{1,0}C^{3,0}D^{1,0}$	1825,07	$A^{2,7}B^{2,1}C^{2,9}D^{0,6}$	16,19
rc203	–	1642,43	$A^{1,0}B^{1,0}C^{5,0}$	1660,93	$A^{1,7}B^{1,1}C^{3,6}D^{0,7}E^{0,1}$	19,00
rc204	–	1521,80	$C^{6,0}$	1543,04	$A^{0,6}B^{0,6}C^{3,8}D^{1,2}$	22,13
rc205	–	1753,79	$A^{1,0}B^{1,0}C^{5,0}$	1774,22	$A^{1,5}B^{1,7}C^{4,1}D^{0,2}$	19,03
rc206	–	1751,75	$A^{2,0}B^{2,0}C^{4,0}$	1767,75	$A^{1,4}B^{1,9}C^{4,0}D^{0,2}$	17,79
rc207	–	1616,96	$A^{1,0}B^{1,0}C^{5,0}$	1640,23	$A^{0,6}B^{1,0}C^{4,2}D^{0,7}$	19,07
rc208	–	1497,95	$C^{6,0}$	1520,76	$A^{0,4}B^{0,1}C^{3,2}D^{1,7}E^{0,2}$	22,03
avg.		1857,06	$A^{3,4}B^{1,9}C^{2,6}D^{2,1}E^{0,1}$	1875,70	$A^{3,5}B^{1,8}C^{2,8}$	21,47
dev.				1,00		

Table A.4: Results for the E-FSMFTW instances, vehicle type B. * mark solutions found by BnP only; underscored are proven to be optimal. The deviation from the best known solution (BKS) is in percent.

	BKS			ALNS_{2000}		
name	obj_{LB}	obj	mix	$\overline{obj}$	$\overline{mix}$	t[m]
c101	1809,93	* 1809,93	$A^{7,0}B^{6,0}$	1816,06	$A^{7,0}B^{6,0}$	14,08
c102	1724,01	1759,73	$A^{7,0}B^{6,0}$	1766,14	$A^{6,8}B^{6,1}$	14,36
c103	1694,21	1755,02	$A^{7,0}B^{6,0}$	1759,20	$A^{6,6}B^{6,2}$	15,09
c104	–	1719,67	$A^{5,0}B^{7,0}$	1735,86	$A^{5,2}B^{6,6}C^{0,2}$	15,60
c105	1736,30	1783,25	$A^{7,0}B^{6,0}$	1785,43	$A^{5,8}B^{6,6}$	14,51
c106	1720,11	1774,77	$A^{7,0}B^{6,0}$	1777,67	$A^{6,2}B^{6,4}$	14,64
c107	1709,23	1764,02	$A^{5,0}B^{7,0}$	1768,33	$A^{5,8}B^{6,6}$	14,57
c108	1701,28	1761,41	$A^{5,0}B^{7,0}$	1769,76	$A^{5,4}B^{6,8}$	14,80
c109	1666,58	1740,18	$A^{5,0}B^{7,0}$	1749,07	$A^{5,0}B^{7,0}$	15,30
c201	–	1210,41	$A^{1,0}B^{3,0}$	1213,63	$A^{1,3}B^{2,4}C^{0,3}$	31,30
c202	–	1209,73	$A^{1,0}B^{3,0}$	1220,97	$A^{2,0}B^{1,5}C^{0,6}$	34,17
c203	–	1212,34	$A^{2,0}B^{1,0}C^{1,0}$	1227,69	$A^{2,9}B^{0,7}C^{0,7}$	33,07
c204	–	1179,25	$A^{1,0}B^{3,0}$	1199,37	$A^{2,0}B^{2,0}C^{0,2}$	32,15
c205	–	1188,92	$A^{2,0}B^{1,0}C^{1,0}$	1195,24	$A^{2,3}B^{0,9}C^{0,9}$	31,64
c206	–	1183,42	$A^{2,0}B^{1,0}C^{1,0}$	1192,30	$A^{2,6}B^{0,8}C^{0,8}$	31,01
c207	–	1183,42	$A^{2,0}B^{1,0}C^{1,0}$	1190,37	$A^{2,6}B^{0,8}C^{0,8}$	31,44
c208	–	1181,47	$A^{5,0}$	1192,96	$A^{3,5}B^{0,5}C^{0,5}$	30,82
r101	1954,00	* 1954,00	$A^{3,0}B^{2,0}C^{9,0}D^{6,0}$	1977,89	$A^{0,6}B^{2,3}C^{12,2}D^{4,7}$	14,36
r102	1757,13	* 1757,13	$B^{1,0}C^{12,0}D^{5,0}$	1791,03	$A^{0,7}B^{2,3}C^{9,9}D^{5,4}E^{0,1}$	14,67
r103	1555,92	* 1597,73	$B^{1,0}C^{5,0}D^{9,0}$	1618,81	$A^{0,5}B^{0,8}C^{4,8}D^{8,4}E^{0,5}$	15,52
r104	–	1424,30	$C^{1,0}D^{9,0}E^{2,0}$	1448,31	$B^{0,4}C^{1,6}D^{8,3}E^{2,0}$	16,20
r105	1699,34	* 1699,34	$B^{2,0}C^{7,0}D^{6,0}E^{1,0}$	1728,12	$A^{0,3}B^{1,6}C^{7,6}D^{6,5}E^{0,6}$	14,68
r106	1588,26	* 1604,55	$A^{1,0}C^{3,0}D^{9,0}E^{1,0}$	1635,42	$A^{0,4}B^{1,2}C^{4,5}D^{7,2}E^{1,5}$	14,67
r107	1449,56	1490,04	$B^{1,0}C^{1,0}D^{7,0}E^{3,0}$	1514,01	$A^{0,1}B^{0,3}C^{1,9}D^{8,0}E^{2,1}$	15,99
r108	–	1399,27	$B^{1,0}D^{9,0}E^{2,0}$	1417,39	$A^{0,2}B^{0,4}C^{1,4}D^{7,3}E^{2,6}$	16,42
r109	1510,93	* 1550,40	$C^{5,0}D^{8,0}E^{1,0}$	1580,14	$A^{0,2}B^{0,3}C^{4,5}D^{8,5}E^{0,8}$	15,65
r110	1392,62	1446,48	$B^{2,0}C^{1,0}D^{8,0}E^{2,0}$	1471,66	$B^{0,4}C^{2,6}D^{7,9}E^{1,8}$	15,64
r111	1401,57	* 1438,81	$B^{1,0}C^{3,0}D^{7,0}E^{2,0}$	1479,75	$B^{0,2}C^{2,5}D^{9,2}E^{1,1}$	16,16
r112	1334,62	1389,87	$C^{2,0}D^{5,0}E^{4,0}$	1403,82	$C^{1,5}D^{7,7}E^{2,5}$	16,10
r201	–	1366,63	$A^{5,0}$	1378,77	$A^{5,2}$	29,69
r202	–	1236,97	$A^{5,0}$	1249,65	$A^{5,0}$	29,13
r203	–	1104,85	$A^{5,0}$	1124,07	$A^{5,0}$	30,23
r204	–	977,72	$A^{5,0}$	983,97	$A^{5,0}$	26,81
r205	–	1217,77	$A^{5,0}$	1232,63	$A^{4,9}B^{0,1}$	29,15
r206	–	1136,83	$A^{5,0}$	1155,47	$A^{4,9}B^{0,1}$	30,95
r207	–	1031,22	$A^{5,0}$	1057,22	$A^{5,0}$	26,31
r208	–	971,15	$A^{5,0}$	984,87	$A^{4,4}B^{0,3}C^{0,1}$	28,21
r209	–	1099,24	$A^{5,0}$	1117,68	$A^{5,0}$	29,62
r210	–	1087,21	$A^{5,0}$	1100,27	$A^{4,6}B^{0,3}$	29,88
r211	–	1006,38	$A^{5,0}$	1026,07	$A^{4,9}B^{0,1}$	25,93
rc101	2051,31	* 2134,30	$B^{11,0}C^{6,0}D^{1,0}$	2153,24	$A^{1,3}B^{7,4}C^{6,9}D^{2,1}$	13,80
rc102	1890,93	* 1947,71	$A^{2,0}B^{4,0}C^{9,0}D^{1,0}$	1972,85	$A^{1,5}B^{4,3}C^{8,6}D^{1,5}$	14,68
rc103	1666,70	1736,25	$A^{1,0}B^{2,0}C^{7,0}D^{3,0}$	1764,22	$A^{1,1}B^{3,4}C^{6,8}D^{2,6}$	14,54
rc104	1525,90	1595,44	$B^{1,0}C^{8,0}D^{3,0}$	1614,09	$A^{0,3}B^{1,3}C^{7,0}D^{3,6}$	15,80
rc105	1823,42	1885,63	$B^{4,0}C^{7,0}D^{3,0}$	1900,42	$A^{0,1}B^{5,3}C^{7,5}D^{2,0}$	14,14
rc106	1759,37	1823,89	$B^{3,0}C^{9,0}D^{2,0}$	1844,99	$A^{0,9}B^{2,8}C^{9,4}D^{1,3}$	15,18
rc107	1581,57	1639,84	$B^{1,0}C^{8,0}D^{3,0}$	1675,58	$A^{0,1}B^{1,5}C^{7,8}D^{3,0}$	15,31
rc108	1509,49	1578,51	$B^{2,0}C^{6,0}D^{4,0}$	1601,47	$A^{0,1}B^{0,9}C^{6,7}D^{4,0}$	14,98
rc201	1580,15	1589,99	$A^{3,0}B^{6,0}C^{1,0}$	1617,52	$A^{3,1}B^{3,9}C^{2,2}D^{0,2}$	15,64
rc202	–	1485,13	$A^{2,0}B^{1,0}C^{2,0}D^{2,0}$	1497,99	$A^{1,9}B^{2,5}C^{2,5}D^{0,9}$	16,72
rc203	–	1310,37	$A^{1,0}B^{1,0}C^{5,0}$	1333,25	$A^{1,1}B^{1,4}C^{3,5}D^{0,4}E^{0,4}$	19,35
rc204	–	1183,16	$C^{2,0}D^{3,0}$	1193,93	$A^{0,7}C^{2,7}D^{2,3}$	22,69
rc205	–	1424,75	$A^{1,0}C^{3,0}D^{2,0}$	1440,00	$A^{0,8}B^{1,1}C^{4,0}D^{0,5}E^{0,2}$	20,95
rc206	–	1431,21	$A^{1,0}B^{4,0}C^{3,0}$	1439,17	$A^{1,0}B^{2,6}C^{3,4}D^{0,4}$	18,11
rc207	–	1277,71	$C^{6,0}$	1299,14	$A^{0,4}B^{0,4}C^{3,7}D^{1,3}E^{0,1}$	21,30
rc208	–	1161,57	$C^{2,0}D^{3,0}$	1171,52	$A^{0,1}C^{2,8}D^{2,0}E^{0,3}$	22,91
avg.		1458,70	$A^{2,5}B^{2,2}C^{2,4}D^{2,1}E^{0,3}$	1474,22	$A^{2,6}B^{2,1}C^{2,6}$	20,83
dev.				1,06		

Table A.5: Results for the E-FSMFTW instances, vehicle type C. * mark solutions found by BnP only; underscored are proven to be optimal. The deviation from the best known solution (BKS) is in percent.

		BnP				init.	ALNS				
type	name	obj	F	mix	t[s]	obj	$\overline{obj}$	obj	F	mix	t[s]
A	c101c5	**857,75**	600	A^2	0,01	867,33	**857,75**	**857,75**	600	A^2	16,56
	c103c5	**476,05**	300	A^1	0,01	493,15	**476,05**	**476,05**	300	A^1	12,72
	c206c5	**1261,88**	1000	A^1	0,02	2262,46	1262,19	**1261,88**	1000	A^1	13,07
	c208c5	**1164,34**	1000	A^1	0,03	**1164,34**	**1164,34**	**1164,34**	1000	A^1	10,76
	r104c5	**327,25**	190	A^1C^1	0,08	393,52	**327,25**	**327,25**	190	A^1C^1	12,83
	r105c5	**346,08**	190	A^1C^1	0,02	422,92	**346,08**	**346,08**	190	A^1C^1	11,51
	r202c5	**609,18**	450	A^1	0,04	618,86	**609,18**	**609,18**	450	A^1	9,78
	r203c5	**645,63**	450	A^1	0,02	1095,63	**645,63**	**645,63**	450	A^1	11,78
	rc105c5	**511,96**	270	A^2B^1	0,02	524,99	**511,96**	**511,96**	270	A^2B^1	8,51
	rc108c5	**532,04**	360	A^1B^2	0,01	640,51	**532,04**	**532,04**	210	A^1B^1	10,87
	rc204c5	**496,74**	300	A^2	0,19	**496,74**	**496,74**	**496,74**	300	A^2	7,31
	rc208c5	**328,89**	150	A^1	0,08	502,91	**328,89**	**328,89**	150	A^1	9,04
B	c101c5	**377,75**	120	A^2	0,01	387,33	**377,75**	**377,75**	120	A^2	13,51
	c103c5	**236,05**	60	A^1	0,01	287,11	**236,05**	**236,05**	60	A^1	8,79
	c206c5	**461,88**	200	A^1	0,03	662,46	**461,88**	**461,88**	200	A^1	9,39
	c208c5	**364,34**	200	A^1	0,03	**364,34**	**364,34**	**364,34**	200	A^1	6,84
	r104c5	**175,25**	38	A^1C^1	0,01	233,52	**175,25**	**175,25**	38	A^1C^1	6,97
	r105c5	**194,08**	38	A^1C^1	0,01	**194,08**	**194,08**	**194,08**	38	A^1C^1	7,84
	r202c5	**249,18**	90	A^1	0,01	**249,18**	**249,18**	**249,18**	90	A^1	6,77
	r203c5	**285,63**	90	A^1	0,01	**285,63**	**285,63**	**285,63**	90	A^1	6,65
	rc105c5	**295,96**	54	A^2B^1	0,01	**295,96**	**295,96**	**295,96**	54	A^2B^1	9,28
	rc108c5	**313,93**	30	B^1	0,01	**313,93**	**313,93**	**313,93**	60	B^2	10,36
	rc204c5	**255,55**	70	B^1	0,03	260,35	255,79	**255,55**	70	B^1	10,09
	rc208c5	**208,89**	30	A^1	0,03	**208,89**	**208,89**	**208,89**	30	A^1	8,61
C	c101c5	**317,75**	60	A^2	0,01	340,04	**317,75**	**317,75**	60	A^2	10,26
	c103c5	**206,05**	30	A^1	0,01	233,84	**206,05**	**206,05**	30	A^1	8,89
	c206c5	**361,88**	100	A^1	0,01	445,98	**361,88**	**361,88**	100	A^1	9,69
	c208c5	**264,34**	100	A^1	0,01	265,55	**264,34**	**264,34**	100	A^1	8,70
	r104c5	**156,25**	19	A^1C^1	0,01	213,52	**156,25**	**156,25**	19	A^1C^1	8,64
	r105c5	**175,08**	19	A^1C^1	0,01	**175,08**	**175,08**	**175,08**	19	A^1C^1	7,73
	r202c5	**204,18**	45	A^1	0,04	222,43	**204,18**	**204,18**	45	A^1	6,43
	r203c5	**240,63**	45	A^1	0,02	263.46	**240,63**	**240,63**	45	A^1	7,62
	rc105c5	**268,96**	27	A^2B^1	0,02	308,99	**268,96**	**268,96**	27	A^2B^1	8,01
	rc108c5	**283,93**	30	B^2	0,01	408,35	**283,93**	**283,93**	30	B^2	10,07
	rc204c5	**220,55**	35	B^1	0,06	228,06	220,86	**220,55**	35	B^1	9,59
	rc208c5	**193,89**	15	A^1	0,07	212,31	**193,89**	**193,89**	15	A^1	7,55
dev. %							0,01	0,00			

Table A.6: Results for the E-FSMFTW instances with 5 customers compared to BnP

		BnP				init.	ALNS					
type	name	obj	F	mix	t[s]	obj	$\overline{obj}$	obj	F	mix	t[s]
A	c101c10	**1302,15**	900	A^3	0,17	**1302,15**	**1302,15**	**1302,15**	900	A^3	14,24
	c104c10	**902,71**	600	A^2	1,10	917,07	**902,71**	**902,71**	600	A^2	18,25
	c202c10	**1304,32**	1000	A^1	0,09	2268,00	1493,84	**1304,32**	1000	A^1	14,83
	c205c10	**2631,97**	2400	A^1B^1	0,15	2693,16	**2631,97**	**2631,97**	2400	A^1B^1	14,39
	r102c10	**595,34**	320	$A^2B^1C^1$	0,31	787,67	**595,34**	**595,34**	320	$A^2B^1C^1$	13,67
	r103c10	**478,07**	270	$A^1B^1C^1$	4,35	651,73	478,23	**478,07**	270	$A^1B^1C^1$	14,22
	r201c10	**1138,38**	900	A^2	0,31	1156,54	1140,45	**1138,38**	900	A^2	16,30
	r203c10	**952,16**	700	B^1	0,93	1424,84 0	956,50	**952,16**	700	B^1	13,77
	rc102c10	**1100,27**	660	$A^1B^2C^1$	0,03	1190,05	**1100,27**	**1100,27**	660	$A^1B^2C^1$	16,16
	rc108c10	**693,27**	240	A^4	0,06	816,19	**693,27**	**693,27**	240	A^4	20,83
	rc201c10	**684,63**	300	A^2	2,69	842,74	684,88	**684,63**	300	A^2	18,03
	rc205c10	**1064,59**	700	A^1C^1	0,63	1246,55	1067,44	**1064,59**	700	A^1C^1	17,27
B	c101c10	**582,15**	180	A^3	0,03	648,14	**582,15**	**582,15**	180	A^3	15,26
	c104c10	**422,71**	120	A^2	0,55	534,28	**422,71**	**422,71**	120	A^2	18,51
	c202c10	**504,32**	200	A^1	0,07	669,77	548,61	**504,32**	200	A^1	14,19
	c205c10	**711,97**	480	A^1B^1	0,06	972,74	**711,97**	**711,97**	480	A^1B^1	14,50
	r102c10	**325,19**	76	$A^1B^1D^1$	0,02	507,67	**325,19**	**325,19**	76	$A^1B^1D^1$	14,13
	r103c10	**262,07**	54	$A^1B^1C^1$	0,54	335,48	**262,07**	**262,07**	54	$A^1B^1C^1$	17,51
	r201c10	**418,38**	180	A^2	0,11	471,79	**418,38**	**418,38**	180	A^2	16,98
	r203c10	**392,16**	140	B^1	0,33	496,63	438,29	**392,16**	140	B^1	13,81
	rc102c10	**572,27**	132	$A^1B^2C^1$	0,02	626,05	**572,27**	**572,27**	132	$A^1B^2C^1$	13,57
	rc108c10	**422,12**	72	A^1B^2	0,01	538,15	**422,12**	**422,12**	72	A^1B^2	15,82
	rc201c10	**429,17**	90	A^3	0,46	476,30	**429,17**	**429,17**	90	A^3	26,72
	rc205c10	**504,59**	140	A^1C^1	0,17	575,21	**504,59**	**504,59**	140	A^1C^1	22,62
C	c101c10	**492,15**	90	A^3	0,15	536,90	**492,15**	**492,15**	90	A^3	16,39
	c104c10	**362,71**	60	A^2	0,19	419.47	**362,71**	**362,71**	60	A^2	16,60
	c202c10	**404,32**	100	A^1	0,17	449,44	413,84	**404,32**	100	A^1	14,55
	c205c10	**471,97**	240	A^1B^1	0,07	499,03	**471,97**	**471,97**	240	A^1B^1	10,45
	r102c10	**287,19**	38	$A^1B^1D^1$	0,04	472,67	**287,19**	**287,19**	38	$A^1B^1D^1$	12,09
	r103c10	**235,07**	27	$A^1B^1C^1$	0,25	299,73	**235,07**	**235,07**	27	$A^1B^1C^1$	20,27
	r201c10	**328,38**	90	A^2	0,31	342,99	**328,38**	**328,38**	90	A^2	17,55
	r203c10	**322,16**	70	B^1	0,95	378,32	350,29	**322,16**	70	B^1	13,12
	rc102c10	**498,51**	75	B^3C^1	0,03	548,05	**498,51**	**498,51**	75	B^3C^1	13,09
	rc108c10	**386,12**	36	A^1B^2	0,05	484,44	**386,12**	**386,12**	36	A^1B^2	14,76
	rc201c10	**384,17**	45	A^3	0,52	426,96	**384,17**	**384,17**	45	A^3	19,28
	rc205c10	**429,69**	95	A^1D^1	0,13	451,47	**429,69**	**429,69**	95	A^1D^1	16,06
dev. %							1,18	0,00			

Table A.7: Results for the E-FSMFTW instances with 10 customers compared to BnP

type	name	BKS	All	Only A	Only B	Only C	Only D	Only E	Only F
A	C1	$A^{19,0}$	$A^{19,0}$	$A^{19,0}$	$B^{10,2}$	$C^{9,6}$	–	–	–
	C2	$A^{5,0}$	$A^{5,0}$	$A^{5,0}$	$B^{4,0}$	$C^{4,0}$	$D^{3,8}$	–	–
	R1	$A^{0,3}B^{0,8}C^{16,8}D^{0,8}$	$A^{0,3}B^{1,1}C^{15,8}D^{1,3}$	n.a.	$B^{30,0}$	$C^{19,0}$	$D^{13,4}$	$E^{11,3}$	–
	R2	$A^{5,0}$	$A^{5,0}$	$A^{5,0}$	$B^{4,0}$	$C^{3,0}$	$D^{2,6}$	–	–
	RC1	$A^{2,5}B^{2,8}C^{9,5}$	$A^{3,4}B^{4,6}C^{8,2}D^{0,1}$	$A^{44,0}$	$B^{22,0}$	$C^{12,6}$	$D^{11,5}$	–	–
	RC2	$A^{5,4}B^{4,6}C^{1,1}$	$A^{5,6}B^{4,4}C^{1,2}$	$A^{18,0}$	$B^{9,0}$	$C^{6,0}$	$D^{5,0}$	$E^{4,0}$	$F^{3,1}$
B	C1	$A^{7,7}B^{5,7}$	$A^{7,9}B^{5,5}$	$A^{19,0}$	$B^{10,2}$	$C^{9,6}$	–	–	–
	C2	$A^{5,0}$	$A^{4,7}B^{0,2}C^{<0,1}$	$A^{5,0}$	$B^{4,0}$	$C^{4,0}$	$D^{3,8}$	–	–
	R1	$A^{0,1}B^{0,7}C^{3,9}D^{7,4}E^{1,3}$	$A^{0,1}B^{0,5}C^{3,6}D^{8,1}E^{1,1}$	n.a.	$B^{30,1}$	$C^{19,1}$	$D^{13,5}$	$E^{11,5}$	–
	R2	$A^{5,0}$	$A^{5,0}$	$A^{5,0}$	$B^{4,0}$	$C^{3,0}$	$D^{2,6}$	–	–
	RC1	$A^{0,8}B^{2,1}C^{8,1}D^{1,9}$	$A^{0,8}B^{2,5}C^{7,7}D^{2,1}$	$A^{44,4}$	$B^{22,0}$	$C^{12,8}$	$D^{11,6}$	–	–
	RC2	$A^{0,9}B^{1,0}C^{4,9}D^{0,1}$	$A^{1,5}B^{1,7}C^{3,5}D^{0,6}E^{<0,1}$	$A^{18,0}$	$B^{9,0}$	$C^{6,0}$	$D^{5,0}$	$E^{4,0}$	$F^{3,2}$
C	C1	$A^{4,0}B^{7,3}C^{0,1}$	$A^{4,3}B^{7,3}C^{<0,1}$	$A^{19,0}$	$B^{10,2}$	$C^{9,7}$	–	–	–
	C2	$A^{1,9}B^{1,3}C^{0,9}$	$A^{2,7}B^{1,0}C^{0,6}$	$A^{5,0}$	$B^{4,0}$	$C^{4,0}$	$D^{3,8}$	–	–
	R1	$A^{0,3}B^{0,3}C^{3,2}D^{6,4}E^{2,4}$	$A^{0,3}B^{0,5}C^{3,1}D^{6,6}E^{2,3}$	n.a.	$B^{30,3}$	$C^{19,1}$	$D^{13,6}$	$E^{11,5}$	–
	R2	$A^{5,0}$	$A^{4,9}B^{0,1}$	$A^{5,0}$	$B^{4,0}$	$C^{3,1}$	$D^{2,6}$	–	–
	RC1	$A^{0,8}B^{1,8}C^{6,1}D^{3,8}$	$A^{0,7}B^{2,2}C^{6,4}D^{3,4}$	$A^{45,0}$	$B^{22,1}$	$C^{12,9}$	$D^{11,9}$	–	–
	RC2	$A^{0,9}B^{0,9}C^{3,6}D^{1,1}$	$A^{1,3}B^{1,2}C^{3,2}D^{1,1}E^{0,1}$	$A^{18,0}$	$B^{9,0}$	$C^{6,1}$	$D^{5,0}$	$E^{4,0}$	$F^{3,2}$

Table A.8: Average fleet mix from the best known solutions (BKS) and average results of 10 runs using all or only a single vehicle type (A,B,C,D,E,F) for all E-FSMFTW instances using no energy restrictions.