European Journal of Operational Research 302 (2022) 700–716

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Innovative Applications of O.R.

The consistent electric-Vehicle routing problem with backhauls and charging management

Pamela C. Nolz[a,c,*], Nabil Absi[b], Dominique Feillet[b], Clóvis Seragiotto[c]

[a] St. Pölten University of Applied Sciences, Campus-Platz 1, 3100 St. Pölten, Austria
[b] Mines Saint-Etienne, Univ Clermont Auvergne, CNRS, UMR 6158 LIMOS, Centre CMP, F - 13541 Gardanne France
[c] AIT Austrian Institute of Technology, Center for Energy, Giefinggasse 4, 1210 Vienna, Austria

ARTICLE INFO

Article history:
Received 27 November 2020
Accepted 11 January 2022
Available online 15 January 2022

Keywords:
Distribution
Metaheuristics
Transportation
Combinatorial optimization

ABSTRACT

We consider a consistent vehicle routing problem for the delivery of parcels with electric vehicles. Stemming from a real-world problem, we assume that vehicles can only be charged with electricity between their delivery tours in the morning and their pickup tours in the afternoon. For this purpose, a charging station with a limited amount of charging slots is available at the depot. We aim at generating a set of vehicle routes that are driver- and time-consistent and efficiently use limited charging resources, while optimizing the sum of vehicle fixed cost, vehicle/driver operating time, arrival time consistency and driver consistency. We present a mathematical model to describe the problem in detail. For solving the real-world problem, a template-based Adaptive Large Neighborhood Search is developed, complemented with constraint programming for charging management and quadratic programming for delivery and pickup trip scheduling. Computational experiments for different settings and scenarios, based on data from an Austrian parcel delivery company, are presented and analysed.

1. Introduction

The worldwide pandemic in 2020 has led to an increasing demand for home deliveries of small packages, i.e. daily goods and groceries. Yet before, the parcel delivery sector has been growing steadily, creating a strong competition and reinforcing the need for high-quality customer service. At the same time, the increasing population in urban areas and the pollution induced by fuel vehicles, have put the environmental impact of freight transport in cities and residential areas on the corporate agenda. The European Strategy for low-emission mobility (EC2, 2020) states that almost a quarter of Europe's greenhouse gas (GHG) emissions are caused by transport activities. Since road transport is the biggest emitter of GHG emissions from transport, the global shift towards a low-carbon, circular economy is inevitable. The European Commission's low-emission mobility strategy aims to ensure that Europe stays competitive and able to respond to the increasing mobility needs of people and goods, by advancing the transition towards zero-emission vehicles (EC2, 2020).

Innovative low-emission logistics concepts are required, in order to contribute to efficient and sustainable transport activities. In last mile logistics, specifically in the parcel delivery sector, the steady relationship between shipping companies and customers is essential. Customers appreciate being served by the same person at roughly the same time of day and at regular intervals. Companies can improve their customer service by establishing personal relationships between drivers and customers. Companies profit from the familiarity of drivers with customers, being able to reduce travel and service times and increase customer loyalty. Drivers operate more effectively, knowing customer specifics as well as local peculiarities. By this routine, driver satisfaction and resilience can be improved.

In this paper, we consider the operative problem of a parcel delivery company in Austria. Driven by the challenges mentioned before, the company decided to convert their vehicle fleet, using electric vehicles instead of fuel-powered vehicles. The main drawback of electric vehicles is their limited autonomy and significant recharging time, requiring an efficient planning of vehicle routes together with the electric charging management. The focal company possesses a charging station at the depot with a limited number of charging slots. Drivers perform deliveries in the morning and pickups in the afternoon, spending their break in between at

* Corresponding author.
E-mail addresses: pamela.nolz@fhstp.ac.at (P.C. Nolz), absi@emse.fr (N. Absi), feillet@emse.fr (D. Feillet), clovis.seragiotto@ait.ac.at (C. Seragiotto).

https://doi.org/10.1016/j.ejor.2022.01.024

the depot. If the pickup tour cannot be completed with the current energy level of the vehicle, a partial recharging can be performed after the delivery tour. Since the number of slots simultaneously available at the charging station is limited, the problem of charging management has to be faced in addition to customer-oriented service policies. These policies refer to time and driver consistency in the current problem setting. Ensuring this consistency is important because most customers ask for a service every day or almost every day.

In our paper, we formally describe this problem with a mathematical model, encompassing conflicting preferences in the objective function. Then, we propose an innovative version of an Adaptive Large Neighborhood Search (ALNS) to address the real-world problem stated above. The ALNS algorithm is complemented with constraint programming as well as quadratic programming to efficiently solve the problem. Therefore, a two-phase solution procedure is developed and evaluated by means of various performance indicators. Results for realistic test instances give an indication of solution quality and benefits of different policies.

The remainder of the article is organized as follows. In Section 2, a literature review is presented. In Section 3, the problem is described and a mathematical model is provided. In Section 4, our solution approach is introduced and explained. Finally, instance description and experimental results are presented in Section 6 to show the effectiveness of our solution approach and leading to our conclusions in Section 7.

2. Literature review

Section 2 presents a collection of papers investigating consistent vehicle routing problems and electric vehicle routing problems, both linked to the problem examined in this paper.

2.1. Consistent vehicle routing problems

For service-related vehicle routing problems, the consideration of consistency requirements when planning vehicle routes is essential in order to provide a high service level and improve customer loyalty. In order to fulfill consistency requirements, a multi-period planning horizon needs to be considered instead of independent one-period VRPs. For a survey on VRPs in which consistency considerations are important, the interested reader is referred to (Kovacs, Golden, Hartl, & Parragh, 2014a). The Consistent VRP (ConVRP) was first considered by Gröer, Golden, & Wasil (2009) and has since then been studied by various authors in different aspects. Gröer et al. (2009) aim to find a set of minimum cost routes to service a set of customers with known demands over multiple periods, respecting time and driver consistency. The authors assign to each customer a fixed driver, delivering goods within a maximum arrival time difference each day a customer requires service. Tarantilis, Stavropoulou, & Repoussis (2012) impose a maximum difference between the latest and earliest arrival times at any customer location. Feillet, Garaix, Léhuéde, Péton, & Quadri (2014) propose to discretize the day into disjoint time classes and minimize the maximum number of time classes in which a customer is visited. Time classes are limited by bounds specified for the earliest and the latest visit to a customer. Kovacs, Golden, Hartl, & Parragh (2015a) introduce the Generalized ConVRP, where a customer may be served by a limited number of drivers instead of only one. The time consistency is not enforced by constraints but is penalized in the objective function. Haughton (2007) maximize the number of times a unique driver visits each customer and Braekers & Kovacs (2016) restrict the number of different drivers serving each customer.

Only few exact solution methods have been developed (c.f. Goeke, Roberti, & Schneider (2019)) due to the complexity of the ConVRP. Instead, several metaheuristics have been proposed, where a common approach to address the problem is the generation of template routes (c.f. Gröer et al. (2009), Tarantilis et al. (2012), Kovacs, Parragh, & Hartl (2014b)). For the generation of template routes, customers are divided into frequent customers requiring service more than once and non-frequent customers requiring service only once along the whole planning horizon. Template routes are generated for frequent customers, considering their mean demand over the planning horizon. The template routes are then replicated in order to generate daily routes, where frequent customers are removed if they do not require service on a specific day, while non-frequent customers are added, considering the daily demands. Tarantilis et al. (2012) propose a Tabu Search algorithm to minimize the total travel time, modifying both the template routes and the actual daily schedules sequentially. Kovacs et al. (2014b) develop an Adaptive Large Neighborhood Search (ALNS). Other approaches have been proposed, considering the multi-period problem at once (c.f. Kovacs et al. (2015a), Kovacs, Parragh, & Hartl (2015b), Braekers and Kovacs Braekers & Kovacs (2016), Stavropoulou, Repoussis, & Tarantilis (2019), Campelo, Neves-Moreira, Amorim, & Almada-Lobo (2019)). Kovacs et al. (2015a) apply a Large Neighborhood Search algorithm to the solve the Generalized ConVRP.

2.2. Electric vehicle routing problems

In their survey of operational research in green freight transportation, Bektaş, Ehmke, Psaraftis, & Puchinger (2019) examine the major differences between classical vehicle routing problems and vehicle routing problems for electric vehicles, namely their limited range, long recharging times and limited recharging station availability. Vidal, Laporte, & Matl (2020) present a literature review of vehicle routing problems of the past sixty years, providing an overview of existing and emerging problem variants and applications. These include fleet composition problems involving battery-powered and conventional vehicles (e.g. Felipe, Ortuño, Righini, & Tirado (2014), Hiermann, Puchinger, Ropke, & Hartl (2016)). Schneider, Stenger, & Goeke (2014) introduce the electric vehicle-routing problem (EVRP) with time windows and recharging stations. Due to the limited battery capacities of electric vehicles, visits to recharging stations during delivery tours are scheduled. The authors propose a hybrid heuristic combining a variable neighborhood search algorithm with Tabu search. Koç, Jabali, Mendoza, & Laporte (2019) introduce the EVRP with shared charging stations, considering nonlinear charging functions. The objective is to minimize the sum of the fixed opening cost of charging stations and the driver's cost. The authors develop a multi-start heuristic for ALNS coupled with the solution of mixed integer linear programs. In Bruglieri, Mancini, Pezzella, & Pisacane (2019) vehicles can be recharged at alternative fuel stations during their trips. The authors propose a two-phase solution approach, first generating paths limited by dominance rules, and then selecting and combining paths to routes with mixed integer linear programming.

Ferro, Paolucci, & Robba (2018) present different power levels consumption models for the recharging process of electric vehicles. They develop a MIP model that includes time windows and partial recharges for vehicle visiting a charging station, considering time-variant energy prices, based on a case study of Italy. Echeverri, Froger, Mendoza, & Néron (2019) study a multi-period EVRP, where electric vehicles are charged at the depot at any time, subject to limited charging infrastructure (e.g., number of available chargers, power grid constraints, duration of the charging operations). The authors present a two-phase matheuristic, first applying a set of randomized route-first cluster-second heuristics, and then using an algorithm to generate a solution to the multi-period EVRP out of the pool of routes. Cortés-Murcia, Prodhon, & Afsar (2019) al-

low customer visits by an alternative mode of transport while the electric vehicle is at a recharging station. The authors present a mathematical model and apply Iterated Local Search, reinforced with Variable Neighborhood Descent and set partitioning to solve the envisaged problem. Macrina, Laporte, Guerriero, Di, & Puglia (2019b) and Macrina, Di, Puglia, Guerriero, & Laporte (2019a) investigate the use of a mixed vehicle fleet composed of electrical and conventional vehicles. Partial battery recharging is done at any of the available stations. In Macrina et al. (2019b), the authors propose a matheuristic embedded within a large neighborhood search scheme, while in Macrina et al. (2019a) an iterative local search heuristic is used. Breunig, Baldacci, Hartl, & Vidal (2019) introduce the electric two-echelon VRP. They propose a large neighborhood search metaheuristic as well as an exact mathematical programming algorithm to evaluate the impact of various features for optimized battery-powered distribution networks.

2.3. Contribution beyond state-of-the-art

Our study contains several innovative and new aspects. (i) First of all, we propose an extended template-based adaptive large neighborhood search algorithm including recharging management in addition to consistency, backhauling and cost efficiency. (ii) Second, we integrate the scheduling of recharging operations within routing optimization. We consider a single recharging station, with a partial recharging policy and a limited number of slots for recharging. Furthermore, we accept any form of recharging time function. This leads to a recharging management problem which we solve with constraint programming (CP) which is not already addressed in the known literature for this category of problem. (iii) Third, we propose consistency and operating time optimization at daily route scale with convex mixed integer quadratic programming (MIQP convex).

We do not consider time and driver consistency as hard constraints, like in most of the above-mentioned papers. We penalize time and driver consistency in the objective function. Driver consistency is not restricted to just one driver visiting a customer, but we minimize the number of drivers per customer. In addition, if a customer is visited by more than one customer, we favour the equal division of the number of visits to a customer by each driver. We do not bound the maximum arrival time difference of each customer, but we minimize the difference between the latest and earliest arrival times at any customer location. We use a square in these differences to provide some flexibility by only slightly penalizing small changes in visit times. Recharging management of electric vehicles at resource-constrained charging infrastructure is a novelty.

3. The consistent electrical vehicle routing problem with backhauls and charging management

In Section 1, we highlighted the main characteristics of the problem addressed in this paper. In this section, we introduce this problem formally, under the name of Consistent Electrical Vehicle Routing Problem with Backhauls and Charging Management (CEVRP-BCM). Before entering the formal definition, we quickly recall the basic features of the problem.

The CEVRP-BCM consists in optimizing pickup and delivery operations for a set of customers over a time horizon of several days. Because customers do not require pickup or delivery every day, consistency criteria are introduced to favor consistent operation schedules and consistent drivers for customers. The service is carried out with a fleet of electric vehicles. A backhauling policy is followed, *i.e.* all delivery operations in a vehicle route are performed before starting pickups. Contrary to private customer parcel services, in this paper we consider deliveries to retailers and business customers, who have a high interest in receiving goods as early as possible to be able to efficiently perform their daily business. These retailers and business customers, on the other hand, want to send goods as late as possible to have more preparation time (so that the day activity is complete and everything is ready for departure). These customers are present at the shipping address throughout the day, so there is no need to specifically wait for deliveries or pickups. To deal with the limited range of the electric vehicles, a recharging station is available. Recharging is, however, only allowed once deliveries are finished and before starting pickups. Furthermore, a limited number of terminals are available at the recharging station and the scheduling of recharging operations in the station has to be considered.

3.1. Problem description

The CEVRP-BCM can be formally formulated as follows. Let $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ be a complete directed graph. $\mathcal{V} = \{v_0, \dots, v_{n+1}\}$ is composed of a depot v_0, a recharging station v_{n+1} and n customers v_1 to v_n. $\mathcal{A} = \{(v_i, v_j) : v_i \in \mathcal{V}, v_j \in \mathcal{V} \setminus \{v_i\}\}$. We denote $\mathcal{N} = \{v_1, \dots, v_n\}$ the customer set. The time horizon consists of a set $\mathcal{D}$ of days. The daily delivery demand of customer v_i on day d is denoted q^-_{id}, the daily pickup demand is denoted q^+_{id}. The service time at customer v_i on day d is denoted s^-_{id} for a delivery and s^+_{id} for a pickup. With every arc $(v_i, v_j) \in \mathcal{A}$ is associated a non-negative travel time t_{ij} and an energy consumption e_{ij}. For the sake of simplicity, we assume that energy consumption and travel times are proportional. A fleet $\mathcal{K}$ of identical electric vehicles is assumed to be available at the depot. Vehicles are subject to a load capacity Q and a battery capacity F.

At a given day, all vehicles start from the depot with a full-loaded battery at a nonnegative time (≥ 0) and have to return to the depot at the latest at time H. Three types of vehicle routes are allowed. Pure delivery routes only include delivery operations. Pickup and delivery routes include both types of operations, with a backhauling policy: all delivery operations are constrained to be performed before pickup operations. Pure pickup routes only include pickup operations. When an operation is carried out by a vehicle, it has to be entirely fulfilled (no split pickup, no split delivery).

Whatever the type of route (pickup and delivery route, pure delivery or pure pickup), a lunch break is required, starting within the time window $[a_{\text{lunch}}, b_{\text{lunch}}]$. Even if drivers do not have to complete any more routes on a day (pickup or delivery), they have other duties to fulfill, i.e. administrative tasks. Therefore, a lunch break seems natural. The minimal duration of the break is t_{lunch}. All deliveries have to be carried out before lunch, all pickups after lunch. The break is spent at the recharging station v_{n+1}, and gives the opportunity of recharging the vehicle. No other visits to the recharging station are allowed. The vehicle battery can be recharged, partially or totally. Recharging is possible within the time interval $[a_{\text{rech}}, \dots, b_{\text{rech}}]$, with $a_{\text{rech}} \leq a_{\text{lunch}}$ and $b_{\text{rech}} \geq b_{\text{lunch}}$. The recharging station is equipped with L terminals that can each receive at most one vehicle at a time. Recharging time is described by a function $\zeta(e, t)$ that gives the amount of energy in the battery after t units of recharging time, starting from an initial level e. Please note, that the focus of this paper is not on how to model battery charging. Indeed, in our model, battery charging can be modeled with a simple linear function, as it is done in the majority of the literature (Schneider et al. (2014), Desaulniers, Errico, Irnich, & Schneider (2016), Hiermann et al. (2016)), or it can be modeled using nonlinear functions, to capture the nonlinear behavior of the charging process (Montoya, Guéret, Mendoza, & Villegas (2017), Froger, Mendoza, Jabali, & Laporte (2019), Macrina et al. (2019a)). No preemption is allowed when recharging a vehicle. Note that while the number of vehicles being simultaneously

recharged is limited to L, no limit on the total number of vehicles waiting at the station is imposed. Note also that lunch breaks are not considered for vehicles that are not used on a given day.

The CEVRP-BCM consists in performing all pickup and delivery operations over the time horizon, in compliance with the previous constraints and while minimizing a complex objective function described below.

A first component in the objective function consists of a fixed cost, added every time a vehicle is active on a given day, weighted by a coefficient α_1.

A second component concerns operating costs, which are counted per unit of driver working time (waiting and lunch break included). In other words, if τ is the difference between the return time to the depot and the starting time of a vehicle on a given day, the vehicle route contributes to operating costs with a value $\alpha_2 \times \tau$.

A third component stands for time consistency. Given a customer i that requires delivery operations on a subset $\mathcal{D}_i^{del} \subseteq \mathcal{D}$ of days, and denoting t_{id}^{del} the delivery time of the customer on day d, the delivery time-consistency of customer i is:

$$\left(\max_{d\in\mathcal{D}_i^{del}} t_{id}^{del} - \min_{d\in\mathcal{D}_i^{del}} t_{id}^{del}\right)^2$$

Equivalently, the pickup time-consistency is:

$$\left(\max_{d\in\mathcal{D}_i^{pick}} t_{id}^{pick} - \min_{d\in\mathcal{D}_i^{pick}} t_{id}^{pick}\right)^2$$

The sum of these two time-consistencies, for all customers, is added to the objective function, weighted by a coefficient α_3. The rationale behind using a square in these formulas is to provide some flexibility by only slightly penalizing small changes in visit times. Observe that in order to comply with the time-consistency objective, vehicle waiting times are not constrained in the CEVRP-BCM (even if they are penalized thanks to operating costs).

A fourth and last component stands for driver consistency. Given a customer i, we define $\mathcal{K}_i \subseteq \mathcal{K}$ the subset of vehicles (drivers) visiting v_i at least once, either for pickup or delivery, during the time horizon $\mathcal{D}$. For every $k \in \mathcal{K}_i$, we also compute σ_{ki}, the number of occurrences of a visit by vehicle k. Then, the driver consistency for customer v_i is set to:

$$|\mathcal{K}_i| + \sum_{k\in\mathcal{K}_i} \frac{1}{\sigma_{ki}}$$

The rationale here is to give the most importance to the number $|\mathcal{K}_i|$ of different drivers visiting customer i. It follows observed trends in the literature, where this number is usually constrained or minimized (see, e.g., Kovacs et al. (2014a), Kovacs et al. (2015a) or Rodríguez-Martín, Salazar-González, & Yaman (2019)). In addition, we propose to slightly deviate from the traditional measure by favouring a balanced sharing of customers between drivers. Indeed, we believe that avoiding infrequent visits is beneficial both to drivers and customers. Given $|\mathcal{K}_i|$, minimizing $\sum_{k\in\mathcal{K}_i} \frac{1}{\sigma_{ki}}$ helps distributing evenly the visits between drivers, seeing that $\sum_{k\in\mathcal{K}_i} \sigma_{ki}$ is constant and that small σ_{ki} values are highly penalizing. Finally, the driver consistency is stated at the sum of the driver-consistencies for the different customers and is multiplied by a coefficient α_4 before being added to the objective function.

It is important to underline that the consistency criteria deeply complicate the problem, because they prevent from computing pickup and delivery routes independently on each day.

These four components clearly set the problem as a multiobjective optimization problem. However, for the sake of practicability, the CEVRP-BCM is defined as a monoobjective problem with a weighted sum of the four components. The way the weights are defined and normalized is discussed in Section 6. An analysis on different weight combinations follows in Section 6.4.

Table 1 reports the notation introduced in the definition of the CEVRP-BCM. For a better readability, we introduce the additional following notation. $\mathcal{N}_d^-$ is the subset of customers with a positive delivery demand q_{id}^- on day d; $\mathcal{N}^- = \cup_{d\in\mathcal{D}} N_d^-$. $\mathcal{N}_d^+$ and $\mathcal{N}^+$ are equivalent notation for pickup operations.

3.2. Mathematical formulation

We now propose a mathematical formulation for the CEVRP-BCM. Decision variables are reported in Table 2.

$$\begin{aligned}\text{minimize } & \alpha_1 \sum_{d\in\mathcal{D}}\sum_{k\in\mathcal{K}} z_{kd} + \alpha_2 \sum_{d\in\mathcal{D}}\sum_{k\in\mathcal{K}} (a_{0kd}^+ - a_{0kd}^-) \\ & + \alpha_3 \sum_{v_i\in\mathcal{N}^-} (a_i^{\max -} - a_i^{\min -})^2 \\ & + \alpha_3 \sum_{v_i\in\mathcal{N}^+} (a_i^{\max +} - a_i^{\min +})^2 + \alpha_4 \sum_{k\in\mathcal{K}}\sum_{v_i\in\mathcal{N}} \left(z_{ik} + \frac{1}{n_{ik}}\right) \qquad (1)\end{aligned}$$

subject to:

[delivery trips]

$$\sum_{k\in\mathcal{K}} y_{ikd}^- = 1 \qquad (d\in\mathcal{D}, v_i\in\mathcal{N}_d^-), \qquad (2)$$

$$z_{kd} = \sum_{v_i\in\mathcal{N}_d^-\cup\{v_{n+1}\}} x_{0ikd}^- \qquad (d\in\mathcal{D}, k\in\mathcal{K}), \qquad (3)$$

$$\sum_{v_i\in\{v_0\}\cup\mathcal{N}_d^-\setminus\{v_j\}} x_{ijkd}^- = \sum_{v_i\in\mathcal{N}_d^-\setminus\{v_j\}\cup\{v_{n+1}\}} x_{jikd}^- = y_{jkd}^- \quad (d\in\mathcal{D}, k\in\mathcal{K}, v_j\in\mathcal{N}_d^-), \qquad (4)$$

$$a_{0kd}^- + t_{0i} + M(1 - x_{0ikd}^-) \le a_{id}^- \qquad (d\in\mathcal{D}, k\in\mathcal{K}, v_i\in\mathcal{N}_d^-), \qquad (5)$$

$$a_{id}^- + s_{id}^- + t_{ij} - M(1 - \sum_{k\in\mathcal{K}} x_{ijkd}^-) \le a_{jd}^- \quad (d\in\mathcal{D}, v_i\in\mathcal{N}_d^-, v_j\in\mathcal{N}_d^-\setminus\{v_i\}), \qquad (6)$$

$$a_{id}^- + s_{id}^- + t_{i,n+1} - M(1 - x_{i,n+1,k,d}^-) \le a_{n+1,k,d}^- \quad (d\in\mathcal{D}, k\in\mathcal{K}, v_i\in\mathcal{N}_d^-), \qquad (7)$$

$$a_{0,k,d}^+ + t_{0,n+1} - M(1 - x_{0,n+1,k,d}^+) \le a_{n+1,k,d}^+ \qquad (d\in\mathcal{D}, k\in\mathcal{K}), \qquad (8)$$

[lunch break]

$$a_{n+1,k,d}^- \le b_{\text{lunch}} z_{kd} \qquad (d\in\mathcal{D}, k\in\mathcal{K}), \qquad (9)$$

$$a_{n+1,k,d}^+ \ge a_{\text{lunch}} + t_{\text{lunch}} z_{kd} \qquad (d\in\mathcal{D}, k\in\mathcal{K}), \qquad (10)$$

$$a_{n+1,k,d}^+ \ge a_{n+1,k,d}^- + t_{\text{lunch}} z_{kd} \qquad (d\in\mathcal{D}, k\in\mathcal{K}), \qquad (11)$$

[pickup trips]

$$\sum_{k\in\mathcal{K}} y_{ikd}^+ = 1 \qquad (d\in\mathcal{D}, v_i\in\mathcal{N}_d^+), \qquad (12)$$

$$\sum_{v_i\in\mathcal{N}_d^+\cup\{v_0\}} x_{n+1,i,k,d}^+ = z_{kd} \qquad (d\in\mathcal{D}, k\in\mathcal{K}), \qquad (13)$$

$$\sum_{v_i\in\{v_{n+1}\}\cup\mathcal{N}_d^+\setminus\{v_j\}} x_{ijkd}^+ = \sum_{v_i\in\mathcal{N}_d^+\setminus\{v_j\}\cup\{v_0\}} x_{jikd}^+ = y_{jkd}^+ \quad (d\in\mathcal{D}, k\in\mathcal{K}, v_j\in\mathcal{N}_d^+), \qquad (14)$$

Table 1
Notation .

Sets	
$\mathcal{V}$	nodes (v_0 depot, v_{n+1} recharging station, v_1 to v_n customers)
$\mathcal{N}$	customers
$\mathcal{A}$	arcs
$\mathcal{D}$	days
$\mathcal{K}$	vehicles
Parameters	
H	latest possible return time to the depot
n	number of customers
Q	vehicle capacity (maximal load)
F	battery capacity (maximal energy)
L	number of terminals at the recharging station
q_{id}^-, q_{id}^+	daily delivery/pickup demand of customer v_i on day d
s_{id}^-, s_{id}^+	delivery/pickup service time at customer v_i on day d
t_{ij}	travel time on arc (v_i, v_j)
e_{ij}	required energy to drive from v_i to v_j
$\zeta(e,t)$	recharging function
t_{lunch}	minimal gap between delivery and pickup tours (lunch break duration)
a_{lunch}	earliest start time of lunch break
b_{lunch}	latest start time of lunch break
a_{rech}	earliest starting time for recharging
b_{rech}	latest ending time for recharging
α_1	coefficient for vehicle fixed costs
α_2	coefficient for driver working hours
α_3	coefficient for visiting time consistency
α_4	coefficient for driver consistency

Table 2
Decision variables .

Variables	
z_{kd}	1 if vehicle k travels on day d, 0 otherwise
x_{ijkd}^-	1 if vehicle k drives from v_i to v_j for delivery on day d, 0 otherwise
x_{ijkd}^+	1 if vehicle k drives from v_i to v_j for pickup on day d, 0 otherwise
y_{ikd}^-	1 if v_i is visited by vehicle k for delivery on day d, 0 otherwise
y_{ikd}^+	1 if v_i is visited by vehicle k for pickup on day d, 0 otherwise
a_{id}^-	arrival time at customer v_i for delivery on day d
a_{id}^+	arrival time at customer v_i for pickup on day d
$a_{n+1,k,d}^-$	arrival time of vehicle k at v_{n+1} on day d
$a_{n+1,k,d}^+$	departure time of vehicle k at v_{n+1} on day d
$a_{0,k,d}^+$	arrival time of vehicle k at v_0 on day d
$a_{0,k,d}^-$	departure time of vehicle k at v_0 on day d
z_{ik}	1 if v_i is visited by vehicle k on any day, 0 otherwise
n_{ik}	the number of times v_i is visited by vehicle k on any day
$a_i^{\max -}$	latest arrival time at customer v_i for delivery
$a_i^{\min -}$	earliest arrival time at customer v_i for delivery
$a_i^{\max +}$	latest arrival time at customer v_i for pickup
$a_i^{\min +}$	earliest arrival time at customer v_i for pickup
h_{kdt}^{start}	1 if the driver of vehicle k starts lunch at time t on day d, 0 otherwise
h_{kdt}^{end}	1 if the driver of vehicle k ends lunch at time t on day d, 0 otherwise
v_{kd}	beginning of the charging time of vehicle k on day d
w_{kd}	end of the charging time of vehicle k on day d
F_{kd}^-	energy consumed in the delivery trip of vehicle k on day d
F_{kd}^+	energy consumed in the pickup trip of vehicle k on day d

$$a_{id}^+ + s_{id}^+ + t_{ij} - M(1 - \sum_{k\in\mathcal{K}} x_{ijkd}^+) \le a_{jd}^+ \quad (d \in \mathcal{D}, v_i \in \mathcal{N}_d^+, v_j \in \mathcal{N}_d^+ \setminus \{v_i\}), \tag{15}$$

$$a_{n+1,k,d}^+ + t_{n+1.j} - M(1 - x_{n+1,j,k,d}^+) \le a_{jd}^+ \quad (d \in \mathcal{D}, k \in \mathcal{K}, v_j \in \mathcal{N}_d^+), \tag{16}$$

$$a_{id}^+ + s_{id}^+ + t_{i,0} - M(1 - x_{i,0,k,d}^+) \le a_{0,k,d}^+ \quad (d \in \mathcal{D}, k \in \mathcal{K}, v_i \in \mathcal{N}_d^+), \tag{17}$$

$$a_{n+1,k,d}^+ + t_{n+1,0} - M(1 - x_{n+1,0,k,d}^+) \le a_{0,k,d}^+ \quad (d \in \mathcal{D}, k \in \mathcal{K}), \tag{18}$$

[capacity constraints]

$$\sum_{v_i\in\mathcal{N}_d^-} q_{id}^- y_{ikd}^- \le Q \qquad (d \in \mathcal{D}, k \in \mathcal{K}), \tag{19}$$

$$\sum_{v_i\in\mathcal{N}_d^+} q_{id}^+ y_{ikd}^+ \le Q \qquad (d \in \mathcal{D}, k \in \mathcal{K}), \tag{20}$$

[depot closing time]

$$a_{0,k,d}^+ \le H \qquad (d \in \mathcal{D}, k \in \mathcal{K}), \tag{21}$$

[driver consistency]

$$y_{ikd}^- \le z_{ik} \qquad (d \in \mathcal{D}, k \in \mathcal{K}, v_i \in \mathcal{N}_d^-), \tag{22}$$

$$y_{ikd}^+ \le z_{ik} \qquad (d \in \mathcal{D}, k \in \mathcal{K}, v_i \in \mathcal{N}_d^+), \tag{23}$$

$$\sum_{\{d\in\mathcal{D}: v_i\in\mathcal{N}_d^-\}} y_{ikd}^- + \sum_{\{d\in\mathcal{D}: v_i\in\mathcal{N}_d^+\}} y_{ikd}^+ = n_{ik} \qquad (k \in \mathcal{K}, v_i \in \mathcal{N}), \tag{24}$$

[time consistency]

$$a_i^{\min -} \le a_{id}^- \le a_i^{\max -} \qquad (d \in \mathcal{D}, v_i \in \mathcal{N}_d^-), \tag{25}$$

$$a_i^{\min +} \le a_{id}^+ \le a_i^{\max +} \qquad (d \in \mathcal{D}, v_i \in \mathcal{N}_d^+), \tag{26}$$

[recharging station planning]

$$a_{n+1,k,d}^- \le v_{kd} \le w_{kd} \le a_{n+1,k,d}^+ \qquad (d \in \mathcal{D}, k \in \mathcal{K}), \tag{27}$$

$$v_{kd} = \sum_{t \in \{a_{\text{rech}}, \ldots, b_{\text{rech}}\}} t \times h_{kdt}^{start} \qquad (d \in \mathcal{D}, k \in \mathcal{K}), \tag{28}$$

$$w_{kd} = \sum_{t \in \{a_{\text{rech}}, \ldots, b_{\text{rech}}\}} t \times h_{kdt}^{end} \qquad (d \in \mathcal{D}, k \in \mathcal{K}), \tag{29}$$

$$\sum_{t \in \{a_{\text{rech}}, \ldots, b_{\text{rech}}\}} h_{kdt}^{start} = \sum_{t \in \{a_{\text{rech}}, \ldots, b_{\text{rech}}\}} h_{kdt}^{end} = 1 \qquad (d \in \mathcal{D}, k \in \mathcal{K}), \tag{30}$$

$$\sum_{k \in \mathcal{K}} \sum_{t' \in \{a_{\text{rech}}, \ldots, t\}} (h_{kdt'}^{start} - h_{kdt'}^{end}) \le L \qquad (d \in \mathcal{D}, t \in \{a_{\text{rech}}, \ldots, b_{\text{rech}}\}), \tag{31}$$

[autonomy]

$$F_{kd}^- = \sum_{v_i \in \{v_0\} \cup \mathcal{N}_d^-} \sum_{v_j \in \mathcal{N}_d^- \cup \{v_{n+1}\} \setminus \{v_i\}} e_{ij} x_{ijkd}^- \qquad (d \in \mathcal{D}, k \in \mathcal{K}), \tag{32}$$

$$F_{kd}^+ = \sum_{v_i \in \{v_{n+1}\} \cup \mathcal{N}_d^+} \sum_{v_j \in \mathcal{N}_d^+ \cup \{v_0\} \setminus \{v_i\}} e_{ij} x_{ijkd}^+ \qquad (d \in \mathcal{D}, k \in \mathcal{K}), \tag{33}$$

$$F_{kd}^- \le F \qquad (d \in \mathcal{D}, k \in \mathcal{K}), \tag{34}$$

$$\zeta(F - F_{kd}^-, w_{kd} - v_{kd}) \le F \qquad (d \in \mathcal{D}, k \in \mathcal{K}), \tag{35}$$

$$F_{kd}^+ \le \zeta(F - F_{kd}^-, w_{kd} - v_{kd}) \qquad (d \in \mathcal{D}, k \in \mathcal{K}), \tag{36}$$

[decision variables]

$$z_{kd} \in \{0,1\} \qquad (d \in \mathcal{D}, k \in \mathcal{K}), \tag{37}$$

$$x_{ijkd}^- \in \{0,1\} \quad (d \in \mathcal{D}, k \in \mathcal{K}, v_i \in \{v_0\} \cup \mathcal{N}_d^-, v_j \in \mathcal{N}_d^- \cup \{v_{n+1}\} \setminus \{v_i\}), \tag{38}$$

$$x_{ijkd}^+ \in \{0,1\} \quad (d \in \mathcal{D}, k \in \mathcal{K}, v_i \in \{v_{n+1}\} \cup \mathcal{N}_d^+, v_j \in \mathcal{N}_d^+ \cup \{v_0\} \setminus \{v_i\}), \tag{39}$$

$$y_{ikd}^- \in \{0,1\} \qquad (d \in \mathcal{D}, k \in \mathcal{K}, v_i \in \mathcal{N}_d^-) \tag{40}$$

$$y_{ikd}^+ \in \{0,1\} \qquad (d \in \mathcal{D}, k \in \mathcal{K}, v_i \in \mathcal{N}_d^+) \tag{41}$$

$$a_{id}^- \ge 0 \qquad (d \in \mathcal{D}, v_i \in \mathcal{N}_d^-) \tag{42}$$

$$a_{id}^+ \ge 0 \qquad (d \in \mathcal{D}, v_i \in \mathcal{N}_d^+) \tag{43}$$

$$a_{n+1,k,d}^-, a_{n+1,k,d}^+, a_{0kd}^-, a_{0kd}^+ \ge 0 \qquad (d \in \mathcal{D}, k \in \mathcal{K}) \tag{44}$$

$$z_{ik} \in \{0,1\} \qquad (k \in \mathcal{K}, v_i \in \mathcal{N}), \tag{45}$$

$$n_{ik} \ge 0 \qquad (k \in \mathcal{K}, v_i \in \mathcal{N}), \tag{46}$$

$$a_i^{\min -}, a_i^{\max -} \ge 0 \qquad (v_i \in \mathcal{N}^-), \tag{47}$$

$$a_i^{\min +}, a_i^{\max +} \ge 0 \qquad (v_i \in \mathcal{N}^+), \tag{48}$$

$$v_{kd}, w_{kd} \ge 0 \qquad (d \in \mathcal{D}, k \in \mathcal{K}), \tag{49}$$

$$h_{kdt}^{start}, h_{kdt}^{end} \in \{0,1\} \qquad (d \in \mathcal{D}, k \in \mathcal{K}, t \in \{a_{\text{rech}}, \ldots, b_{\text{rech}}\}), \tag{50}$$

$$F_{kd}^-, F_{kd}^+ \qquad (d \in \mathcal{D}, k \in \mathcal{K}). \tag{51}$$

In this model, M is a large number. The objective function is given by (1). Here, the total vehicle fixed cost, operating time, arrival time and driver consistency are minimized. Constraints (2) make sure that all deliveries are satisfied. Constraints (3) state that a vehicle is used on a given day if and only if it leaves the depot. Constraints (4) guarantee that every visited customer has exactly one successor and predecessor. Constraints (5) set the earliest possible visit time for deliveries. Constraints (6) then establishes the time gap for successive deliveries. Constraints (7)-(8) determine arrival times to the recharging station. Constraints (9)-(11) set the time constraints for lunch breaks and compute the availability time for vehicles after lunch. Constraints (12) to (18) are equivalent to Constraints (2)-(8) for pickups. Constraints (19) and (20) guarantee that the vehicle capacity is not exceeded. Constraints (21) ensure respecting the maximal duration for vehicle routes. To evaluate driver consistency, Constraints (22) and (23) capture whether customer v_i is visited by vehicle k. Constraints (24) count the number of visits. Constraints (25) and (26) determine the earliest and latest arrival time at customer v_i, for deliveries and pickups, respectively. Constraints (27) guarantee a logical sequence in time at the recharging station. Constraints (28) and (29) link continuous and discrete variables for the starting and ending times of recharging operations, respectively. Constraints (30) impose that a recharging is started and ended exactly once for each vehicle on each day. This recharging is however fictive when the recharging starts and ends at the same time index. Constraints (31) make sure that at most $|L|$ recharging are executed simultaneously. Note that this model does not explicitly assign recharging tasks to recharging stations. It remains valid because scheduling on $|L|$ identical parallel machines is equivalent to scheduling on a single cumulative resource of capacity $|L|$, assuming a resource consumption equal to 1 for the tasks (Baptiste, Le Pape, & Nuijten (1999)). Constraints (32) and (33) compute energy consumption for deliveries and pickups, respectively. Constraints (34) make sure that the energy consumed for deliveries is less than the battery capacity. Constraints (35) limit the amount of energy after recharging at the station not to exceed the battery capacity. Constraints (36) check that the available energy after recharging is enough for pickups. Constraints (37) to (51) define the variables of the problem.

4. Solution approach: Template-based ALNS

4.1. General framework

In view of the complexity of the problem and the sizes of the practical instances that we have to deal with, we propose

to tackle the problem with a heuristic approach. We present this approach in this section. The solution scheme that we have retained is a Template-based Adaptative Large Neighborhood Search (TALNS). TALNS is a two-phase solution scheme that was initially introduced and successfully applied by Kovacs et al. (2014b) for the Consistent Vehicle Routing Problem. It fits particularly well our context in which customers express demands very frequently. It is composed of the following two phases.

The **first phase** considers a worst-case daily projection of the problem: vehicle routes are optimized on a single period in which customer delivery (pickup) demands are assumed to be at their maximal value. Denoting q_i^- the delivery demand (q_i^+ the pickup demand) for customer v_i, we set $q_i^- = \max_{d\in\mathcal{D}} q_{id}^-$ ($q_i^+ = \max_{d\in\mathcal{D}} q_{id}^+$). Solutions generated at the first phase are called templates. Indeed, as will be seen in the second phase, solutions of the CEVRP-BCM can be easily derived by replicating on every day of the horizon the routing provided by the template and by eventually skipping the customers with zero demands.

The generation of templates is addressed with an Adaptative Large Neighborhood Search (ALNS) framework (Pisinger & Ropke, 2010). ALNS is based on the destroy & repair principle and consists in the repetition of two successive steps: the destruction of a large part of an existing solution with a destroy operator, the repair of this destroyed solution with a repair operator. ALNS applies multiple destroy and repair operators that are randomly selected according to probabilities depending on the rate of acceptance of the solutions they lead to. When dealing with templates, the objective function introduced in the CEVRP-BCM is not entirely relevant, especially with regards to consistency. In insertion operators, best insertions are those that minimize the increase in driving time.

Given a template solution, the **second phase** of the algorithm derives daily routes. In each day of the horizon, the daily demand cannot exceed demands q_i^- and q_i^+ considered in the first phase. Furthermore, it is possible that some customers are visited while they do not express any delivery or pickup demand on that day. These customers are then skipped. The scheduling of the remaining operations is then optimized, separately for each vehicle but considering the complete time horizon $\mathcal{D}$. Details on the models and algorithms developed for this optimization are given in Section 5. In this phase, the true objective function of the CEVRP-BCM is considered, including time and driver consistency. This phase thus permits to associate an actual cost with a template. Based on this cost, the template is kept or not for next iterations.

Algorithm 1 presents the TALNS algorithm. The algorithm is initialized with a template solution built with a best insertion heuristic (Line 1), described in Section 4.3. A CEVRP-BCM is obtained from this solution using the phase two algorithm (daily route generation) and is kept as the temporary best known solution (Line 2). The main loop of the algorithm (Lines 3–15) is then started and repeated until a stopping criterion is met: here, the computation time limit or maximal number of iterations without success. For each of these iterations, a pair of destroy and repair operators are first selected with a roulette wheel depending on their historical efficiency (Line 4). The set of operators and the probability distributions used are detailed in Section 4.4. The selected destroy and repair operators are then successively applied (Lines 5–6). Considering the new template, daily route generation is performed to derive a CEVRP-BCM solution. An acceptance rule is applied to decide whether the new template solution will serve for the next iterations or not; furthermore, the new best known solution is updated if possible (Lines 8–13). Notation introduced in Algorithm 1 is summarized in Table 3.

An important point in this algorithm is the management of vehicle autonomy. Each time a customer is inserted into a vehicle route, either in the initial construction phase or within the repair operators, the feasibility of the template with regard to energy consumption is checked. Additionally, the recharging schedule is re-optimized for daily routes at phase 2 of the algorithm. These operations are performed with a Constraint Programming approach presented in Section 4.2. Other details on the algorithm follow in subsequent sections.

Algorithm 1 TALNS algorithm

1: *TSol* ← template solution obtained from the constructive heuristic
2: *BestSol* ← CEVRP-BCM solution obtained from *TSol* via phase two algorithm
3: **while** computation time limit or maximal number of iterations without improvement is not reached **do**
4: apply roulette wheel to select destroy and repair operators
5: destroy *TSol* with the selected destroy operator
6: *NewTSol* ← repair the destroyed solution with the selected repair operator
7: *NewSol* ← CEVRP-BCM solution obtained from *NewTSol* via phase two algorithm
8: **if** *NewSol* is accepted **then**
9: *TSol* ← *NewTSol*
10: **if** *NewSol* is better than *BestSol* **then**
11: *BestSol* ← *NewSol*
12: **end if**
13: **end if**
14: update selection probabilities for destroy and repair operators
15: **end while**
16: **return** *BestSol*

Table 3
Notation introduced in Algorithm 1.

TSol	template solution (before destroy and repair)
NewTSol	template solution (after destroy and repair)
NewSol	CEVRP-BCM solution (obtained from template *NewTSol*)
BestSol	CEVRP-BCM solution (best known)

4.2. Recharging management

In this section we detail how we manage the recharging operations for a given template or partial template solution (partial meaning that some customers are not present in the template). Details on how the recharging schedule is optimized for daily routes will be given in Section 5. We first show that the problem of finding a feasible recharging schedule is a standard scheduling problem on parallel machines with release dates and due dates ($Pm|r_i, d_i|-$, using Graham's notation). We then explain how we tackle this problem.

The (partial) template solution provides a set of at most $|\mathcal{K}|$ vehicle routes. Let us introduce the following notation. A route R is composed of a delivery trip R^- followed, after a visit to the recharging station and a lunch, by a pickup trip R^+. Both the delivery and pickup trips might be empty, in the sense that the vehicle might directly reach the recharging station from the depot or *vice versa*. The amounts of energy consumed in R^- and R^+ are known and equal to $E_{R^-} = \sum_{(v_i,v_j)\in R^-} e_{ij}$ and $E_{R^+} = \sum_{(v_i,v_j)\in R^+} e_{ij}$, respectively. If $E_{R^-} + E_{R^+} \leq F$, no recharging is needed at the recharging station: the vehicle only stops a time t_{lunch} for the lunch break at the station and restarts for the pickup trip. These vehicles are not considered when computing the recharging schedule. The remaining vehicles require a positive recharging time p such that $\zeta(F - E_{R^-}, p) = E_{R^+}$. Using the scheduling terminology, it defines a non-preemptive task of processing time p_R. This task is subject to a release date (earliest possible starting time) and

Table 4
Notation introduced in Section 4.2 .

R	a route in the current (partial) template
R^-	delivery trip of route R
R^+	pickup trip of route R
E_{R^-}	energy consumed in trip R^-
E_{R^+}	energy consumed in trip R^+
p_R	recharging time needed for route R
rd_R	release date for the recharging of route R
dd_R	due date for the recharging of route R
$t^-_{\max}$	maximal duration of a delivery trip in the current (partial) template
$t^+_{\max}$	maximal duration of a pickup trip in the current (partial) template
t_{rech}	duration of the recharging planning associated with the current (partial) template

a due date (latest possible ending time). The release date rd_R is given by the maximal value between the ending time of trip R^- (assuming it is starting at time 0) and the opening time for recharging: $rd_R = \max\left(\sum_{(v_i,v_j)\in R^-} t_{ij}, a_{\text{rech}}\right)$. The due date dd_R is given by the minimal value between the latest possible starting time of the pickup trip and the recharging closing time: $dd_R = \min\left(H - \sum_{(v_i,v_j)\in R^+} t_{ij}, b_{\text{rech}}\right)$. The recharging station is a resource composed of L terminals, which exactly correspond to L parallel identical machines. The problem at hand is then to find a feasible schedule for the tasks on these L parallel machines, taking into account the temporal constraints imposed by release dates and due dates.

Parallel machine scheduling with release dates and due dates is NP-complete, but remains tractable in our case seeing the small size of the instances that we have to solve (at most $|\mathcal{K}|$ tasks). We propose to solve it using constraint programming. Following Baptiste et al. (1999), we notice that our problem can then be seen as a special case of the Cumulative Scheduling Problem $PS1|r_i, d_i|-$ (also called CuSP). Indeed, scheduling on L identical parallel machines is equivalent to scheduling on a single cumulative resource of capacity L, assuming a resource consumption equal to 1 for the tasks. This problem can very efficiently be expressed and solved using constraint programming, taking as input the capacity of the resource and the set of tasks with their processing times, release dates and due dates and returning a feasible schedule if it exists.

Note that when dealing with the templates, evaluating the feasibility of recharging is enough: the detail of the schedule is not useful. Indeed, the exact schedule of the different pickup and delivery operations will be computed with Phase 2. However, we keep trace of three pieces of information that are useful for Phase 2: the maximal duration of a delivery trip, denoted $t^-_{\max}$; the maximal duration of a pickup trip: $t^+_{\max}$; the time needed for recharging (*i.e.*, completion time minus starting time): t_{rech}. Notation introduced in this section is summarized in Table 4.

4.3. Constructive heuristic

The constructive heuristic presented in this section initializes the TALNS algorithm with a first template solution. The heuristic is described in Algorithm 2.

The set of all pickup and delivery requests is identified (Line 1). Then the main loop of the algorithm (Lines 2–14) progressively constructs new vehicle routes, one after the other, with a best insertion policy. When all requests have been inserted, the algorithm finishes with a 2-opt descent applied separately to every pickup or delivery trip (Line 15).

The construction of a route is described with Lines 3 to 13. Line 3 starts the construction by introducing the new vehicle route. The route is composed of three parts: an empty delivery trip R^- starting at time 0 from the depot and going to the recharging station; a break for a time t_{lunch} at the recharging station, an empty pickup trip R^+ starting from the recharging station after the break and returning to the depot. Additionally, the set of eligible trips (initially composed of R^- and R^+ – Line 4) is created. At each iteration of loop 5–13, a new request is inserted as follows. First, an eligible trip is randomly selected in ET – with equal probabilities (Line 6). An insertion in this trip is then carried out if possible, as detailed with Algorithm 3 below (Lines 7–9). When the insertion fails, the trip becomes ineligible (Line 11). When both trips are ineligible, the construction of the route is stopped (Line 5). Notation introduced in Algorithm 2 is summarized in Table 5.

Algorithm 2 Constructive heuristic

```
1:  Ω ← set of all pickup and delivery requests
2:  while Ω ≠ ∅ do
3:      (R⁻, R⁺) ← empty route with lunch break
4:      ET ← {R⁻, R⁺}
5:      while ET ≠ ∅ do
6:          R◇ ⟵ random draw in ET
7:          perform a best insertion in R◇
8:          if the best insertion succeeded then
9:              remove the request inserted in R◇ from Ω
10:         else
11:             remove R◇ from ET
12:         end if
13:     end while
14: end while
15: improve the template with 2-opt
```

Algorithm 3 details how the insertion is carried out. If $R^\diamond$ is a pickup (delivery) trip, Line restricts the set of candidate requests to pickup (delivery) requests. The resulting subset is denoted $\Omega^\diamond$. All requests whose insertion is not compatible with the residual capacity of the vehicle are removed from $\Omega^\diamond$ (Lines 2–6). Then, the minimal detour that would be induced by a request insertion is computed (Lines 9–14). and the insertion of the request r^* that minimizes this detour is tried (Lines 15–23). Three feasibility checks are successively carried out. First it is checked that deliveries can finish before lunch closing time b_{lunch} if $R^\diamond$ is a delivery trip, or start after a_{lunch} and stop before H if $R^\diamond$ is a pickup trip. Then, it is checked, that the additional energy consumption, implied by the insertion of r^* in $R^\diamond$, is not greater than the residual energy in $R^\diamond$. Finally, it is checked, thanks to the approach presented in Section 4.2, that the current partial template solution (with the insertion of r^*) admits a feasible recharging schedule. Note that r^* both minimizes the increase in travel time and energy consumption among all the requests in $\Omega^\diamond$ because of the proportionality assumption made in Section 3 between t_{ij} and e_{ij}. Hence, a fail in the insertion ensures that no other request could be inserted. Notation introduced in Algorithm 3 is summarized in Table 6.

4.4. ALNS Operators

The management of destroy and repair operators is based on Pisinger & Ropke (2010), where a roulette wheel mechanism is used to control the selection of destroy and repair operators. Each

Table 5
Notation introduced in Algorithm 2.

Ω	set of pickup and delivery requests that have not (yet) been inserted into the template
R^-	the delivery trip in the template route under construction
R^+	the pickup trip in the template route under construction
ET	the subset of trips in $\{R^-, R^+\}$ that are eligible for a new insertion
$R^\diamond$	a trip randomly chosen in ET

Table 6
Notation introduced in Algorithm 3.

$\Omega^\diamond$	set of requests that could be inserted in trip $R^\diamond$
r	a request in $\Omega^\diamond$
$\delta_{rR^\diamond}$	the best insertion cost of request r in $R^\diamond$ (the insertion is not necessarily feasible)
δ^*	the best known insertion cost (the insertion is feasible)
r^*	the request associated with the best known insertion cost (the insertion is feasible)

Algorithm 3 Best insertion (constructive heuristic)

$\Omega^\diamond \leftarrow$ subset of all delivery requests (case $R^\diamond = R^-$) or all pickup requests (case $R^\diamond = R^+$) in Ω
for all requests $r \in \Omega^\diamond$ **do**
  if the demand associated with request r is greater than the residual capacity of trip $R^\diamond$ **then**
    remove r from $\Omega^\diamond$
  end if
end for
$\delta^* \leftarrow \infty$
$r^* \leftarrow null$
for all requests $r \in \Omega^\diamond$ **do**
  if the detour $\delta_{rR^\diamond}$ implied by the best insertion of r in $R^\diamond$ is less than δ^* **then**
    $\delta^* \leftarrow \delta_{rR^\diamond}$
    $r^* \leftarrow r$
  end if
end for
if $r^* \neq null$ **then**
  if the duration of $R^\diamond$ increased with detour δ^* complies with the lunch break window **then**
    if the energy consumption in $R^\diamond$ plus the additional energy consumption implied by the insertion of r^* in $R^\diamond$ is not greater than F **then**
      if the current partial template solution admits a feasible recharging schedule **then**
        insert r^* in $R^\diamond$ at its best insertion position
      end if
    end if
  end if
end if

operator is initially assigned an equal weight, which is adapted during the search according to the performance of the operator. The performance of each operator determines its weight based on the quality of the solutions obtained after applying the operator. The weight includes whether a new best solution is found (ω_1), whether an improving solution is found (ω_2), whether the new solution is accepted (ω_3), or whether the new solution is rejected (ω_4), and the impact of changes in the performance of the destroy and repair methods (λ). In order not to get trapped in a local optimum, we use an acceptance criterion from simulated annealing, occasionally accepting solutions that are worse than the current solution. This enables more diversification throughout the search, especially during early stages of the algorithm. Parameters of the ALNS are summarized in Table 7.

All repair operators follow an equivalent scheme. Every request that was previously removed from the template is successively reinserted. The repair operation is illustrated by Algorithm 4. Depending on the type of request, pickup or delivery, the subset of eligible trips is identified and a first step removes all trips whose residual capacity is insufficient. Then, the trips are sorted according to the evaluation measure included in the operator and insertion in these trips is successively tried. Feasibility checks are equivalent to

Algorithm 4 Insertion (repair operators)

$\Omega \leftarrow$ requests to be reinserted
for all $r \in \Omega$ **do**
  $\mathcal{R}^\diamond \leftarrow$ subset of delivery trips or pickups trips in the partial solution, depending on the type of r
  for all trip $R^\diamond \in \mathcal{R}^\diamond$ **do**
    if the demand associated with request r is greater than the residual capacity of trip $R^\diamond$ **then**
      remove $R^\diamond$ from $\mathcal{R}^\diamond$
    end if
  end for
  for all trip $R^\diamond \in \mathcal{R}^\diamond$ **do**
    compute the minimal cost $\delta_{rR^\diamond}$ of inserting r in $R^\diamond$
  end for
  sort $\mathcal{R}^\diamond$ in the increasing order of insertion costs $\delta_{rR^\diamond}$
  $k \leftarrow 1$
  repeat
    $R^\diamond \leftarrow$ first element in $\mathcal{R}^\diamond$
    if the duration of $R^\diamond$ increased with detour $\delta_{rR^\diamond}$ complies with the lunch break window **then**
      if the energy consumption in $R^\diamond$ plus the additional energy consumption implied by the insertion of r in $R^\diamond$ is not greater than F **then**
        if the current partial template solution admits a feasible recharging schedule **then**
          $R^\diamond_k \leftarrow R^\diamond$
          $k \leftarrow k + 1$
        end if
      end if
    end if
    remove $R^\diamond$ from $\mathcal{R}^\diamond$
  until $k > 1$ (best insertion operator) or $k > 2$ (regret insertion operator) or $\mathcal{R}^\diamond$ is empty
  BEST: $\delta_r \leftarrow \delta_{rR^\diamond_1}$ if $k > 1$, $\delta_r \leftarrow +\infty$ otherwise
  REGRET: $\delta_r \leftarrow \delta_{rR^\diamond_1} - \delta_{rR^\diamond_2}$ if $k > 2$, otherwise $\delta_r \leftarrow \delta_{rR^\diamond_1}$ if $k > 1$, otherwise $\delta_r \leftarrow +\infty$
end for
find r with a minimal evaluation δ_r
insert r in $R^\diamond_1$

Table 7
Parameters of the ALNS .

ω_1	contribution of the finding of a new best solution to the weight of an operator
ω_2	contribution of the finding of an improving solution to the weight of an operator
ω_3	contribution of the acceptance of a solution to the weight of an operator
ω_4	contribution of the rejection of a solution to the weight of an operator
λ	decay parameter controlling the sensitivity of weights to performance changes of the operators

Table 8
Notation introduced in Algorithm 4.

Ω	set of requests to be reinserted into the template
r	a request in Ω
$\mathcal{R}^\circ$	set of delivery trips or pickups trips (depending on the nature of r) in the (partial) template
R°	a trip in $\mathcal{R}^\circ$
δ_{rR°	the best insertion cost of request r in R° (the insertion is not necessarily feasible)
R°_1	trip R° in $\mathcal{R}^\circ$ with the best insertion cost for r (the insertion is feasible)
k	counter introduced to distinguish between greedy insertion and regret insertion
δ_r	measure (greedy or regret) that evaluates the impact of inserting r in R°_1

the ones carried out in the constructive heuristics (Algorithm 3). Notation introduced in Algorithm 4 is summarized in Table 8.

A destroy operator deletes a number of requests from the current template solution and keeps them in a pool, from which repair operators select requests to be inserted.

Random Removal This operator randomly removes whole pickup and delivery trips from the template.

Random Customer Removal This operator randomly removes customers from the template (delivery and pickup).

Worst Removal This operator removes customers from the template. It always chooses as the next customer to be removed the one that, with his/her pickups and deliveries, contributes the most to the duration of the trips in the template (worst detour).

A repair operator reinserts requests into the template solution from the pool of requests removed by the destroy operator.

Greedy Insertion This operator inserts back all pairs (customer, pickup) and (customer, delivery) that were removed by a destroy operator. The pair that contributes the least to the overall time of the template is inserted first; after this pair has been inserted, the insertion cost is computed again for the remaining pairs and the next pair that contributes the least is inserted. The procedure is repeated until all removed pairs have been reinserted.

Note: The insertion of a pair (customer, pickup/delivery) may actually fail because it makes the trip where it has been inserted infeasible (for example, because there is no possible charging schedule after the insertion). The algorithm sticks to that pair and tries to insert it in another trip, even if the cost of inserting the pair in another trip is greater. In the worst case, a new route must be created and the customer will be inserted in the empty delivery/pickup trip of the new route, which is surely feasible.

Regret Insertion The regret insertion is based on the idea that postponing the insertion of a customer which is not considered as the best to insert could lead to an even worse solution. The idea is to check the difference of insertion cost between the best route and position to insert a customer and its second best route and position to insert it. The customer who has the biggest gap is the one to be inserted because inserting it later on would probably fiercely degrade the solution.

This operator inserts back all pairs (customer, pickup) and (customer, delivery) that were previously removed by a destroy operator. The pair that has the greatest regret value is inserted first; after this pair has been inserted, the regret of all remaining pairs is computed again and the pair with the greatest regret value is inserted; this procedure is repeated until all pairs have been inserted.

5. Daily route generation (phase 2 algorithm)

In this section, we describe how we derive a CEVRP-BCM solution from a template. This procedure is a key component of the TALNS method. It aims at:

- Assigning an effective cost to a template: the cost of an actual CEVRP-BCM solution.
- Based on this cost, deciding if a new template should be kept or not for next iterations (acceptance criterion).
- Determine actual CEVRP-BCM solutions and keep trace of the best solution found during the solution process.

The method consists of three steps that respectively optimize: the delivery trip schedule, the recharging schedule, the pickup trip schedule. The second step is carried out with the procedure presented in Section 4.2. The first and third steps apply a similar methodology: the customers that have no request on a given day are skipped from the template on that day, and a quadratic program with linear constraints is solved to optimize the starting times of all requests over the horizon. Note that the issue here is to balance efficiently the operating costs that tend to limit waiting times, and the time consistency costs that tend to add waiting times to enable visiting customers approximately at the same time for every visit. Neither the assignment of requests to vehicles nor the sequencing of requests in vehicles are modified.

5.1. Delivery trip scheduling (step 1)

In order to present the quadratic program for Step 1, we need new notation. Let R^- be a delivery trip in the template and, for every $d \in \mathcal{D}$, let R^{d-} be the delivery trip obtained from R^- by skipping all customers $v_i \in V$ such that $q^-_{id} = 0$. Let us denote $i^d_1, \ldots, i^d_{n_d}$ the indices of the vertices visited in R^{d-}, in the order of their visit (thus starting at the depot and finishing at the recharging station).

We introduce t^{start}, the time at which we plan to start the recharging operations. By default, we set $t^{start} = a_{\text{lunch}}$. However, using notation introduced at the end of Section 4.2 (see Table 4), if $t^-_{\max} > a_{\text{lunch}}$, one cannot be sure that recharging can be started at time a_{lunch} and we set $t^{start} = t^-_{\max}$. Similarly, if $a_{\text{lunch}} > H - t^+_{\max} - t_{rech}$ or $a_{\text{lunch}} > b_{\text{rech}} - t_{rech}$, starting recharging at time a_{lunch} could be too late and we set $t^{start} = \min\left(H - t^+_{\max} - t_{rech}, b_{\text{rech}} - t_{rech}\right)$.

Table 9
Notation introduced for the delivery trip scheduling model .

R^-	the delivery trip under optimization
R^{d-}	the delivery trip obtained from R^- on day d with all zero-demand customers removed
$(i^d_1, \dots, i^d_{n_d})$	sequence of visits in trip R^d
t^{start}	the time at which we plan to start the recharging operations

Table 10
Decision variables of the delivery trip scheduling model .

τ_{id}	the starting time of service at customer v_i on day d when v_i has a delivery request on that day
τ_{0d}	the starting time of the trip on day d
τ^{end}_d	the ending time of the trip on day d
τ^-_i	the minimal value of variables τ_{id} over the horizon
τ^+_i	the maximal value of variables τ_{id} over the horizon

We introduce different series of decision variables. Variables τ_{id} ($i \in \{1, \dots, n\}, d \in \mathcal{D}$) indicate the starting time of service at customer v_i on day d when v_i has a delivery request on that day, and are meaningless otherwise. Variables τ_{0d} ($d \in \mathcal{D}$) indicate the starting time of the trip on day d. Variables τ^{end}_d ($d \in \mathcal{D}$) indicate the ending time of the trip on day d. Variables τ^-_i and τ^+_i ($i \in \{1, \dots, n\}$), respectively indicate the minimal and maximal values of variables τ_{id} over the horizon. Tables 9 and 10 report the introduced notation.

The quadratic program is then:

$$\text{minimize } \alpha_2 \sum_{d \in \mathcal{D}} (t^{start} - \tau_{0d}) + \alpha_3 \sum_{i=1}^{n} (\tau^+_i - \tau^-_i)^2 \tag{52}$$

subject to

$$\tau_{i^d_k d} \geq \tau_{i^d_{k-1} d} + s^-_{i^d_{k-1} d} + t_{i^d_{k-1} i^d_k} \quad (d \in \mathcal{D}, 2 \leq k \leq n_d - 1) \tag{53}$$

$$\tau^{end}_d = \tau_{i^d_{n_d-1} d} + s^-_{i^d_{n_d-1} d} + t_{i^d_{n_d-1} i^d_{n_d}} \quad (d \in \mathcal{D}) \tag{54}$$

$$\tau^{end}_d \leq t^{start} \quad (d \in \mathcal{D}) \tag{55}$$

$$\tau^-_i \leq \tau_{id} \leq \tau^+_i \quad (i \in \{1, \dots, n\} \tag{56}$$

$$\tau_{id} \geq 0 \quad (d \in \mathcal{D}, i \in \{0, \dots, n\}) \tag{57}$$

$$\tau^{end}_d \geq 0 \quad (d \in \mathcal{D}) \tag{58}$$

$$\tau^-_i, \tau^+_i \geq 0 \quad (i \in \{1, \dots, n\}) \tag{59}$$

The objective function (52) minimizes the sum of daily operating costs related to deliveries and the time consistency cost. Constraints (53) impose that when a customer is visited, the vehicle has the time to travel from its previous location. Constraints (54) compute the ending time of the trip (arrival time to vertex $i^d_{n_d}$) on the different days. Constraints (55) limit this arrival time. Constraints (56) compute the minimal and maximal time of service for the different customers. Constraints (57)–(59) define the decision variables.

5.2. Recharging scheduling (step 2)

The recharging schedule is computed independently on each day with the constraint programming model presented in Section 4.2. On a given day d and for a given vehicle, the release date, due date and processing times are computed as follows. Reusing the notation of Section 4.2 (see Table 4), we note R^{d-} and R^{d+} the delivery and pickup trips, respectively. R^{d-} is obtained from Step 1 above and arrives to the recharging station at a time τ (variable τ^{end}_d in Section 5.1). We construct R^{d+} by skipping from the template pickup trip all customers with $q^+_{id} = 0$. Then, if $\sum_{(v_i,v_j) \in R^{d-}} e_{ij} + \sum_{(v_i,v_j) \in R^{d+}} e_{ij} \leq F$, the vehicle is not considered in the recharging scheduling problem. Otherwise, it defines a task with a processing time p_{R^d} such that $\zeta(F - \sum_{(v_i,v_j) \in R^{d-}} e_{ij}, p_{R^d}) = \sum_{(v_i,v_j) \in R^{d+}} e_{ij}$, a release date set to $\max(\tau, a_{\text{rech}})$ and a due date set to $\min\left(H - \sum_{(v_i,v_j) \in R^{d+}} t_{ij}, b_{\text{rech}}\right)$.

5.3. Pickup trip scheduling (step 3)

Step 2 above provides a starting time for the pickup trips (computed from the recharging schedule and taking into account lunch breaks as explained at the end of Section 4.2).

As in Section 5.1, we denote R^{d+} the pickup trip obtained from a template pickup trip R^+ by skipping all customers $v_i \in V$ such that $q^+_{id} = 0$. Note that these trips have been computed at Step 2. We denote $i^d_1, \dots, i^d_{n_d}$ the indices of the vertices visited in R^{d+}, in the order of their visit (thus starting with the recharging station and finishing at the depot). Let t^{end}_d indicate the ending time of the recharging on day d.

We introduce different series of decision variables. Variables τ_{id} ($i \in \{1, \dots, n\}, d \in \mathcal{D}$) indicate the starting time of service at customer v_i on day d when v_i has a pickup request on that day, and are meaningless otherwise. Variables τ_{n+1d} ($d \in \mathcal{D}$) indicate the starting time of the trip on day d and are introduced for the sake of a better readability. Variables τ^{end}_d ($d \in \mathcal{D}$) indicate the ending time of the trip on day d. Variables τ^-_i and τ^+_i ($i \in \{1, \dots, n\}$), respectively indicate the minimal and maximal values of variables τ_{id} over the horizon. Tables 11 and 12 report the introduced notation.

The quadratic program is then:

$$\text{minimize } \alpha_2 \sum_{d \in \mathcal{D}} \tau^{end}_d + \alpha_3 \sum_{i=1}^{n} (\tau^+_i - \tau^-_i)^2 \tag{60}$$

subject to

$$\tau_{i^d_k d} \geq \tau_{i^d_{k-1} d} + s^+_{i^d_{k-1} d} + t_{i^d_{k-1} i^d_k} \quad (d \in \mathcal{D}, 2 \leq k \leq n_d - 1) \tag{61}$$

$$\tau_{n+1,d} = t^{end}_d \quad (d \in \mathcal{D}) \tag{62}$$

$$\tau^{end}_d = \tau_{i^d_{n_d-1} d} + s^+_{i^d_{n_d-1} d} + t_{i^d_{n_d-1} i^d_{n_d}} \quad (d \in \mathcal{D}) \tag{63}$$

$$\tau^{end}_d \leq H \quad (d \in \mathcal{D}) \tag{64}$$

$$\tau^-_i \leq \tau_{id} \leq \tau^+_i \quad (i \in \{1, \dots, n\} \tag{65}$$

Table 11
Notation introduced for the pickup trip scheduling model .

R^+	the pickup trip under optimization
R^{d+}	the pickup trip obtained from R^+ on day d with all zero-demand customers removed
$(i_1^d, \ldots, i_{n_d}^d)$	sequence of visits in trip R^{d+}
t_d^{end}	the ending time of the recharging on day d

Table 12
Decision variables of the pickup trip scheduling model .

τ_{id}	the starting time of service at customer v_i on day d when v_i has a pickup request on that day
τ_{n+1d}	the starting time of the trip on day d
τ_d^{end}	the ending time of the trip on day d
τ_i^-	the minimal value of variables τ_{id} over the horizon
τ_i^+	the maximal value of variables τ_{id} over the horizon

$$\tau_{id} \geq 0 \quad (d \in \mathcal{D}, i \in \{1, \ldots, n+1\}) \tag{66}$$

$$\tau_d^{end} \geq 0 \quad (d \in \mathcal{D}) \tag{67}$$

$$\tau_i^-, \tau_i^+ \geq 0 \quad (i \in \{1, \ldots, n\}) \tag{68}$$

The objective function (60) minimizes the sum of daily operating costs and the time consistency cost. Constraints (61) impose that when a customer is visited, the vehicle has the time to travel from its previous location. Constraints (62) force to start from the recharging station at time t_d^{end}. Constraints (63) compute the ending time of the trip (arrival time to vertex $i_{n_d}^d$) on the different days. Constraints (64) limit this arrival time. Constraints (65) compute the minimal and maximal time of service for the different customers. Constraints (66)–(68) define the decision variables.

6. Computational experiments

All numerical experiments were conducted using one core of an Intel Xeon 2643 machine with 3.3 GHz and 16 GB RAM each running Linux CentOS 6.5. The quadratic program (QP) was solved using IBM ILOG CPLEX, version 12.6.2, and the constraint program (CP) was solved using Choco Solver, version 4.0.6. Each run of the solution approach was performed 5 times and the average values are presented in the subsequent sections. The total time limit allowed to the proposed algorithm is set to 300 seconds and the time limit allowed for solving the QP and the CP is fixed to 5 seconds.

6.1. Test instances and parameter setting

We tested the proposed solution approaches on 30 test instances derived from historic data of an Austrian parcel delivery company. The time horizon is set to 5 days. The fleet is composed of an unlimited number of identical electric vehicles with a battery capacity of 2 hours range and a cargo capacity of 100 items. A full battery recharging operation takes 2 hours, with linear recharging time. Note, that more advanced functions can be used in order to model the behavior of the charging process (interested readers are referred to Montoya et al. (2017) for more details on nonlinear charging functions). The travel time is given in minutes and based on the real road network of the focal region. The customer demand is specified by delivery and pickup percentages, which define the percentage of customer demands that require delivery or pickup respectively, namely 70% and 30%. The visit percentage of 80% determines the customers that are visited every day. The number of customers varies between 20, 50 and 100 customers (10 instances for each).

Table 13
Implementation details .

Normalization factors for objective values	
α_1	$\frac{1}{\lvert\mathcal{D}\rvert}$
α_2	$\frac{1}{\lvert\mathcal{D}\rvert \times H}$
α_3	$\frac{1}{15^2}$
α_4	$\frac{1}{n}$
ALNS Parameters	
ω_1	10
ω_2	5
ω_3	1
ω_4	0.1
λ	0.5
destroy	15%

Table 13 presents the details of ALNS parameters and normalization factors for objective values used in the experimental analyses. Hereafter, we motivate the values given to normalization factors:

- Using decision variables introduced in Section 3, fixed costs are given by $\sum_{d\in\mathcal{D}}\sum_{k\in\mathcal{K}} z_{kd}$. Their maximal value is $|\mathcal{D}| \times |\mathcal{K}|$. We propose to normalize with a factor $\alpha_1 = \frac{1}{|\mathcal{D}|}$. The result is a daily utilization rate of the fleet and is bounded by $|\mathcal{K}|$
- Operating costs are equal to $\sum_{d\in\mathcal{D}}\sum_{k\in\mathcal{K}}(a_{0kd}^+ - a_{0kd}^-)$. They are bounded by $|\mathcal{D}| \times |\mathcal{K}| \times H$. We set $\alpha_2 = \frac{1}{|\mathcal{D}|\times H}$. The normalization gives again a daily fleet utilization rate bounded by $|\mathcal{K}|$ (and by the normalized fixed cost).
- Time consistency for delivery operations is $\sum_{v_i\in\mathcal{N}^-}(a_i^{\max -} - a_i^{\min -})^2$. We propose $\alpha_3 = \frac{1}{15^2}$. With this value a discrepancy of 15 minutes for all customers gives a time consistency equal to $|\mathcal{N}^-|$. The reasoning is equivalent for pickup operations.
- Finally, driver consistency is given by $\sum_{v_i\in\mathcal{N}}\sum_{k\in\mathcal{K}}(z_{ik} + \frac{1}{n_{ik}})$. In the worst case, a customer contributes with a value $2 \times |\mathcal{K}|$. We define $\alpha_4 = \frac{1}{n}$, which bounds the criterion by $2 \times |\mathcal{K}|$ and gives the average consistency per customer.

All ALNS parameters but one were set manually, based on the literature and preliminary tests. The percentage of requests deleted by the destroy operators (row *destroy* in the table) was set to 15% after some tuning described in the next subsection.

6.2. Tuning of the destroy operator

In order to derive the appropriate destroy percentage for the ALNS algorithm, we solved the 30 instances three times, with the destroy percentage ranging from 5% to 50%. We also ran some experiments that combine two destroy percentages (a high percentage and a low percentage) that are uniformly drawn during the execution of the algorithm. In these experiments the number of available terminals in the recharging station is two.

Table 14
Tuning of the destroy percentage parameter .

Method	Destroy%	All instances		20 customers		50 customers		100 customers	
		#best	avg.gap	#best	avg.gap	#best	avg.gap	#best	avg.gap
ANLS	5%	2	12.29%	0	20.67%	1	3.11%	1	13.09%
ANLS	10%	0	11.35%	0	19.24%	0	4.90%	0	9.91%
ANLS	15%	3	8.58%	0	13.78%	3	3.08%	0	8.86%
ANLS	20%	1	7.72%	0	10.34%	1	6.71%	0	6.12%
ANLS	30%	3	9.58%	3	9.97%	0	12.26%	0	6.52%
ANLS	40%	1	9.22%	0	9.87%	0	11.14%	1	6.65%
ANLS	50%	1	9.90%	1	9.92%	0	12.75%	0	7.03%
ANLS	5%,30%	4	7.67%	1	9.58%	2	6.52%	1	6.92%
ANLS	10%,30%	2	7.67%	0	11.72%	0	6.54%	2	4.75%
ANLS	10%,50%	0	9.61%	0	11.17%	0	9.11%	0	8.56%
LNS	10%	2	9.97%	0	19.18%	2	2.67%	0	8.04%
LNS	20%	0	8.56%	0	12.36%	0	8.19%	0	5.12%
LNS	40%	3	9.02%	0	10.07%	0	12.85%	3	4.13%
LNS	5%,30%	2	7.30%	2	9.51%	0	5.79%	0	6.61%
LNS	10%,50%	3	8.17%	0	11.43%	1	8.19%	2	4.88%
MS	-	4	22.08%	4	3.14%	0	34.91%	0	28.19%

In order to show the effectiveness of our algorithm and the importance of the ALNS approach, we compare the ALNS algorithm to a multi-start algorithm (MS) and our ALNS algorithm without adaptiveness (LNS).

The MS algorithm works as follows. While the stopping criterion is not reached (CPU Time), construct a template solution (Tsol) using Algorithm 2 (see Section 4.3) and turn Tsol into a CEVRP-BCM solution using phase two of our ALNS algorithm (see Section 5). If the obtained solution is better than the best known solution, update the best known solution. In order to diversify the search, the first request inserted in an empty trip in Algorithm 2 is randomly selected.

Table 14 shows the average gap to best known solutions ("avg.gap") as well as the number of obtained best known solutions ("#best"). These two values are reported for each method (column "Method") and each destroy percentage (column "Destroy%"). The destroy operator does not apply to the MS method. Best known solutions are given by the best value found among all methods. The results are given for all instances (column "All instances") and for each set of instances sharing the same number of customers (columns "20 customers", "50 customers" and "100 customers").

From Table 14, average values on the set of all instances show that the MS method obtains very poor results compared to ALNS and LNS. This means that the LNS scheme is more efficient than a simple multi-start scheme. Results also show that the adaptiveness has a relatively small impact on the quality of obtained solutions. It confirms the recent study by Turkeš, Sörensen, & Hvattum (2021) that show the limited impact of the adaptive layer on most ALNS implementations.

Regarding percentage values, we can observe that large and small destroy percentages are, on average, outperformed by the other destroy percentages or the combined destroy percentages. Among the latter (intermediate percentages and combined), no percentage clearly dominates. Since the focus is on realistic test instances of large size, the following experiments were conducted with a destroy percentage set to 15%. When analyzing the results more deeply, we also noticed that the standard deviation is better for instances with 100 customers when a destroy percentage of 15% is used.

6.3. Computational efficiency

In the following we further investigate the performance of our method by analyzing the CPU time spent at each phase of the algorithm and the number of iterations that could be performed within the computing time limit of 300 seconds. We run our experiments on our set of instances by changing the number of terminals at the recharging station. The number of terminals ranges between 1 and 9.

Table 15 summarizes the obtained results. Each row reports the average values for 10 instances sharing the same parameters (number of customers "#Customers", number of terminals at the recharging station "#Terminals"). In Table 15 we present the average number of iterations "#Iterations" and the average CPU time of the different phases of our solution algorithm. Column "Phase 1" gives the average total time in seconds dedicated to the first phase in the algorithm. Columns "CP1" and "no_sol1" detail the average time spent within the CP solver and the percentage of CP runs where no solution could be generated before achieving the time limit of 5 seconds, in this phase. Columns "Phase 2", "CP2" and "no_sol2" give equivalent information for the second phase.

From Table 15 we can notice that the number of iterations decreases with the increase in the problem size. This decrease is particularly huge when we move from 20 customers to 50. Clearly, this is due to a more difficult Phase 2, which consumes most of the computing time for instances with 50 and 100 customers.

Some variability can however be observed because of the CP component. In most cases the total time spent by the CP solver is negligible. But when the number of terminals is small, finding feasible recharging schedules is apparently more difficult in Phase 1 and CP can become a bottleneck. In these cases, the number of iterations is significantly reduced.

It is also interesting to notice that the number of runs of the CP without finding a solution is very low. In preliminary experiments, no computing time limit was given to the CP solver and the algorithm could get stuck in very long CP solving steps for the larger instances (sometimes resulting in not even finishing the first iteration of the algorithm in 300 seconds). Limiting execution time to 5 seconds for each call to CP resolves that with very limited risks of missing good feasible solutions.

6.4. Solution value analysis

In this section, we now analyze the values of the obtained solutions. First we evaluate more precisely solution values by decomposing them into their different cost components. Then, we evaluate the impact of the number of recharging terminals on solutions. Finally, we observe how solutions are affected when some criteria are ignored in the objective function.

Table 15
Computation times .

#Customers	#Terminals	Iterations	**Phase 1**(s)	CP1(s)	no_sol1(%)	**Phase 2**(s)	CP2(s)	no_sol2(%)
20	1	608606.98	186.81	92.66	0	113.19	1.21	0.00
20	2	344656.15	92.18	42.98	0	207.82	1.48	0.00
20	3	1259843.14	216.52	16.96	0	83.48	0.13	0.00
20	4	1407234.68	211.51	0	0	88.49	0	0.00
20	5	1456323.72	217.47	0	0	82.53	0	0.00
20	6	1434371.96	222.76	0	0	77.24	0	0.00
20	7	1463946.28	208.69	0	0	91.31	0	0.00
20	8	1291037.28	197.37	0	0	102.63	0	0.00
20	9	1338001.06	208.5	0	0	91.5	0	0.00
50	1	6291.2	19.36	6.38	0	280.64	1.29	0.00
50	2	5730.43	31.78	21.33	0	268.22	1.45	0.00
50	3	5523.05	15.28	5.87	0	284.72	1.1	0.00
50	4	5868.23	13.74	3.67	0	286.26	0.66	0.00
50	5	5179.63	9.64	0.3	0	290.36	0.05	0.00
50	6	5087.61	9.36	0.01	0	290.64	0	0.00
50	7	5342.27	9.8	0	0	290.2	0	0.00
50	8	5406.43	9.12	0	0	290.88	0	0.00
50	9	5419.39	9.1	0	0	290.9	0	0.00
100	1	2415.92	14.8	4.11	0	285.2	0.75	0.00
100	2	1571.74	142.36	134.33	0.07	157.64	0.52	0.00
100	3	502.94	237.45	248.12	0.65	62.55	0.26	0.00
100	4	1436.68	149.13	143.64	0.41	160.36	0.66	0.00
100	5	2492.9	25.75	15.66	0.01	274.21	1.13	0.00
100	6	2386.74	14.01	5.16	0	285.91	0.76	0.00
100	7	2503.18	13.69	4.29	0	286.2	0.37	0.00
100	8	2115.86	9.57	1.73	0	290.38	0.06	0.00
100	9	2388.66	10.02	0.48	0	289.88	0.01	0.00

Table 16
Detailed cost analysis: Normalized values .

		20 customers				**50 customers**				**100 customers**			
		average	std.dev.	min	max	**average**	std.dev.	min	max	**average**	std.dev.	min	max
FC	initial	**3.68**	0.42	3.00	4.00	**7.79**	0.69	6.40	9.00	**14.94**	0.76	13.20	16.80
	final	**3.85**	0.33	3.00	5.00	**7.11**	0.38	6.20	8.00	**14.04**	0.65	12.40	15.80
OT	initial	**2.56**	0.24	2.28	2.94	**4.80**	0.33	3.94	5.45	**8.14**	0.43	7.34	9.38
	final	**2.48**	0.17	2.02	2.85	**4.11**	0.23	3.65	4.91	**7.59**	0.40	6.63	9.14
TC	initial	**3.63**	2.17	0.97	8.84	**22.25**	14.53	2.07	88.81	**32.13**	13.05	13.43	64.03
	final	**0.06**	0.14	0.00	0.78	**0.21**	0.39	0.00	1.92	**0.90**	0.98	0.00	5.45
DC	initial	**1.40**	0.19	1.21	1.76	**1.62**	0.15	1.39	1.94	**1.99**	0.19	1.71	2.45
	final	**1.56**	0.20	1.23	2.20	**2.10**	0.20	1.55	2.52	**2.37**	0.13	2.07	2.67
$\sum$	initial	**11.27**	2.11	8.13	15.41	**36.46**	14.31	17.40	101.54	**57.20**	13.25	37.27	88.11
	final	**7.95**	0.50	6.76	9.81	**13.54**	0.58	12.55	15.23	**24.90**	1.24	22.30	29.39

Table 17
Detailed cost analysis: Absolute values .

		20 customers				**50 customers**				**100 customers**			
		average	std.dev.	min	max	**average**	std.dev.	min	max	**average**	std.dev.	min	max
FC	initial	**18.40**	2.11	15	20	**38.96**	3.43	32	45	**74.72**	3.79	66	84
	final	**19.27**	1.64	15	25	**35.56**	1.88	31	40	**70.19**	3.25	62	79
OT	initial	**6140**	568	5464	7058	**11509**	794	9467	13,079	**19533**	1036	17,610	22518
	final	**5946**	405	4848	6849	**9875**	558	8748	11,773	**18223**	948	15,912	21941
TC	initial	**816**	489	219	1988	**5005**	3269	466	19,982	**7229**	2935	3022	14407
	final	**14**	31.08	0	175	**46**	88	0	433	**203**	220	0	1227
DC	initial	**26.59**	3.65	23.03	33.53	**79.55**	7.2	68.15	94.85	**195.04**	18.84	168.03	239.97
	final	**29.65**	3.88	23.28	41.78	**103.11**	9.88	75.87	123.33	**231.87**	12.49	202.58	261.63
$\sum$	initial	**7002**	697	5742	7950	**16632**	3177	11,059	30,408	**27031**	3208	21,757	33866
	final	**6009**	411	4901	6920	**10060**	538	9082	11,927	**18729**	1004	16,409	22588

Detailed analysis of solution values

In the following we provide a detailed analysis on solution values by disaggregating costs. These experiments were conducted with two terminals at the recharging station. Tables 16 and 17 show the values of the different components in the initial solution and after optimization (FC: Fixed cost, OT: Operating time, TC: Time consistency, DC: Driver consistency, $\sum$: Total cost). Values are normalized in Table 16, with the factors displayed in Table 13, and absolute in Table 17. In the latter, they are rounded for OT, TC and $\sum$.

A first observation from Table 16 is that the normalization is successful. The four components contribute with comparable weights to the total normalized cost. In initial solutions, time consistency is sometimes dominant, but it is completely leveled with the optimization.

Fixed costs (FC) and operating times (OT) seem only slightly impacted by the optimization. They are however on average decreased by more than 5%. These limited improvements can easily be interpreted. The normalized values indicate the average number of vehicles used each day and this number multiplied by the uti-

Table 18
Evaluation of the impact of the number of available terminals, 20 customers .

#Terminals	#Vehicles	Fixed cost		Operating time		Time cons.		Driver cons.		Total cost	
		relative	absolute	relative	absolute	relative	absolute	relative	absolute	relative	absolute
1	4.64	4.10	20.52	2.40	5769.68	0.11	24.12	1.75	33.29	8.37	5847.61
2	3.96	3.85	19.27	2.48	5946.36	0.06	14.12	1.56	29.65	7.95	6009.40
3	4.26	4.07	20.34	2.31	5551.66	0.16	35.76	1.61	30.60	8.15	5638.37
4	4.34	4.15	20.74	2.31	5532.96	0.16	36.74	1.56	29.60	8.17	5620.04
5	4.30	4.10	20.48	2.31	5532.06	0.19	42.32	1.60	30.42	8.19	5625.28
6	4.28	4.08	20.40	2.30	5509.60	0.22	49.34	1.58	30.06	8.18	5609.40
7	4.24	4.05	20.26	2.30	5526.16	0.20	45.00	1.57	29.86	8.13	5621.28
8	4.26	4.03	20.16	2.28	5467.04	0.33	73.16	1.57	29.87	8.21	5590.23
9	4.34	4.15	20.74	2.31	5555.30	0.18	39.40	1.57	29.81	8.21	5645.25

Table 19
Evaluation of the impact of the number of available terminals, 50 customers .

#Terminals	#Vehicles	Fixed cost		Operating time		Time cons.		Driver cons.		Total cost	
		relative	absolute	relative	absolute	relative	absolute	relative	absolute	relative	absolute
1	7.98	7.59	37.96	3.97	9523.98	0.52	118.12	2.20	107.70	14.28	9787.76
2	7.16	7.11	35.56	4.11	9875.56	0.21	46.20	2.10	103.11	13.54	10060.43
3	7.30	7.22	36.12	3.88	9320.84	0.24	53.08	2.09	102.26	13.43	9512.30
4	7.36	7.30	36.50	3.82	9162.78	0.15	34.68	2.01	98.66	13.29	9332.62
5	7.58	7.46	37.28	3.79	9085.64	0.14	30.76	2.04	99.75	13.41	9253.43
6	7.58	7.51	37.56	3.78	9060.76	0.13	30.34	2.09	102.43	13.51	9231.09
7	7.60	7.52	37.58	3.79	9085.30	0.11	24.34	2.07	101.19	13.47	9248.41
8	7.48	7.42	37.08	3.76	9034.88	0.12	27.70	2.08	102.07	13.39	9201.73
9	7.70	7.60	38.00	3.77	9057.66	0.11	24.92	2.07	101.60	13.56	9222.18

Table 20
Evaluation of the impact of the number of available terminals, 100 customers .

#Terminals	#Vehicles	Fixed cost		Operating time		Time cons.		Driver cons.		Total cost	
		relative	absolute	relative	absolute	relative	absolute	relative	absolute	relative	absolute
1	16.66	15.55	77.74	7.07	16977.44	1.26	283.42	2.37	232.49	26.25	17571.09
2	14.91	14.04	70.19	7.59	18223.35	0.90	203.42	2.37	231.87	24.90	18728.83
3	14.36	13.95	69.74	8.33	19985.74	1.72	386.06	2.37	232.35	26.36	20673.89
4	13.52	13.34	66.72	7.90	18959.06	0.68	152.22	2.32	227.65	24.24	19405.65
5	13.74	13.51	67.56	7.42	17803.10	0.41	92.32	2.33	228.09	23.67	18191.07
6	13.74	13.54	67.72	7.19	17260.96	0.14	32.32	2.28	223.42	23.16	17584.42
7	13.88	13.72	68.62	7.05	16926.22	0.14	31.36	2.29	224.47	23.21	17250.67
8	14.12	13.90	69.48	6.99	16781.60	0.14	32.30	2.28	223.03	23.31	17106.41
9	14.12	13.88	69.42	6.95	16682.32	0.17	38.28	2.27	222.34	23.27	17012.36

lization rate during the day, respectively. There is no hope to make these costs converging to zero while ensuring the service.

Driver consistency costs are also naturally lower bounded. Every customer has to be served by at most one vehicle. Normalized values show that this lower bound is reached for most customers. This is a natural outcome of the template mechanism that favor keeping the same driver. Contrary to FC and OT, the optimization has a negative impact on these costs. Probably, larger benefits can be obtained while breaking this consistency. However, the final values remain extremely good and still largely meet practical needs.

Contrary to previous criteria, time consistency costs TC could theoretically reach zero. Tables 16 shows that this indicator is improved substantially by our algorithm and actually almost reaches this target. Let us recall that the normalized factor would show a value $|\mathcal{N}^-| + |\mathcal{N}^+|$ if all pickup and delivery requests were satisfied with a 15 minutes consistency. It shows how time-consistent are the final schedules.

Impacts of the number of available terminals on the solution quality

The previous experiments were conducted with two terminals available at the recharging station. Now, we investigate the effect of the number of terminals available for electric recharging. Tables 18, 19 and 20 present the absolute and relative cost values and the required number of vehicles, depending on the number of terminals at the recharging station for respectively 20, 50 and 100 customers.

It can be seen that the cost values and the required number of vehicles are first decreased by increasing the number of terminals. However, after a certain threshold, no improvement can be achieved by further augmenting the number of terminals. From Table 18 we can notice that for 20 customers when there is only one recharging terminal, the number of vehicles is higher in order to face the recharging capacity. We can also notice that the operating time is higher when the number of terminals is 1 or 2, which represents in this case the critical number of terminals. When the number of terminals is higher than 2 the total cost is stable. The same behavior can be drawn from Tables 19 and 20 where the critical number of terminals is respectively between 1 and 3, and between 2 and 4. From all these results we can also notice that when the number of terminals is low (equal to 1), the number of used vehicles slightly increases in order to face the tight charging capacity. We also notice that in this case, the time consistency is impacted.

Another valuable information provided by this table is the importance of considering the recharging scheduling in the model. A number of terminals larger than the number of vehicles represents the situation when the recharging capacity is infinite. Results show that better solutions are obtained in this case, which also shows

Table 21
Evaluation of objective weights .

	Fixed cost		Operating time		Time cons.		Driver cons.		Total cost	
	relative	absolute	relative	absolute	relative	absolute	relative	absolute	relative	absolute
Initial	7.79	38.96	4.8	11509	22.25	5005	1.62	79.55	36.46	16633
1,1,1,1	7.11	35.56	4.11	9876	0.21	46.2	2.1	103.11	**13.54**	**10060**
1,0,0,0	**5.28**	**26.4**	4.2	10069	81.82	18409	2.39	117.16	93.69	28622
0,1,0,0	7	35	**3.32**	**7973**	67.58	15206	2.29	112.02	80.19	23326
0,0,1,0	8.72	43.58	9	20918	**0**	**0**	2	110.59	19.69	21073
0,0,0,1	7.29	36.44	5	11810	50.15	11284	**1**	**60.83**	63.6	23192
1,1,0,0	**6.1**	**30.5**	**4**	**8527**	63.87	14370	2	107.16	75.71	23036
0,0,1,1	8.6	43	9	20640	**0**	**0.92**	**2**	**81.04**	18.86	20765

that a model relaxing recharging capacity constraints would produce infeasible solutions.

In addition, results also invalidate another possible model. One might consider recharging capacity constraints by limiting the number of recharged vehicles to the number of terminals. Results clearly show that this approach would be overly pessimistic.

Impact of the optimization criteria

Finally, in Table 21 we evaluate different weights for the objective function components. For these tests, we only consider the 10 instances with 50 customers and two terminals at the recharging station (always with 5 runs each). The first column gives the weights applied to the four normalized components of the objective function (Fixed cost, Operating time, Time consistency, and Driver consistency). Row *initial* corresponds to the initial solution, which is not impacted by these weights. Row (1,1,1,1) corresponds to the standard setting of our model. In other rows, some criteria are deactivated.

Table 21 shows that the best value for each individual component is achieved when the optimization is driven by this component. It was expected but shows anyway that the algorithm is robust and is able to address each of the four components that we defined. Most importantly, the gap between the value obtained for a component when only this component is optimized and this value in the standard setting always remains relatively limited. It shows how the algorithm achieves a good optimization of the four components simultaneously. Conversely, when a component is ignored in the objective function, it always results in larger values for this component. Actually, with regards to total cost, none of the methods even approaches the quality of the solutions provided with the standard setting.

A second interesting observation concerns time-consistency. Table 21 very clearly shows how neglecting time-consistency affects solution quality. When the focus is on time-consistency, perfectly consistent solutions are obtained. When it is not considered, very large values, up to 81 (in relative), are found. In the standard setting, with all components active, time-consistency remains almost perfect. Practitioners are then left with a clear choice between the solution obtained with our standard setting and time-inconsistent solutions where other costs are only slightly improved.

7. Conclusions and future outlook

In this paper, we presented a driver- and time-consistent vehicle routing problem for the delivery of parcels with electric vehicles. We investigated the recharging management of these vehicles at a recharging station with a limited number of terminals. An important novelty of our model was to consider in detail the scheduling problem implied by the recharging. We proposed an original solution method able to combine efficiently all the aspects of our problem. The method derives solutions from a template-based Adaptive Large Neighborhood Search, complemented with constraint programming for charging management and quadratic programming for delivery and pickup trip scheduling. Computational experiments for different settings and scenarios showed the importance of an efficient recharging management and the impacts of different variants of the model on vehicle fixed cost, vehicle operating time, arrival time consistency and driver consistency. The presented solution approach can be embedded in a decision support system for companies aiming at reducing emissions in the transport sector and implementing innovative and efficient last-mile logistics services.

Future research could concentrate on the investigation of different vehicle fleets, e.g., mixed fleets of electric and conventional vehicles, and the efficient use of those with respect to their limited autonomy. A new compromise would then appear between the use of conventional vehicles, generating more pollution, and the use of electric vehicles, potentially saturating the recharging facilities. The studied problem could also be enriched by considering several recharging stations and multi-echelon distribution systems, which would better fit the context of large-scale cities. Finally, the development of an exact method, e.g., based on branch-and-price, would be useful to better evaluate the effectiveness of the proposed template-based ALNS.

Acknowledgments

The project EUFAL is funded by the ERA-NET Cofund Electric Mobility Europe (EMEurope). EMEurope is co-funded by the European Commission within the research and innovation framework programme Horizon 2020 and national and regional funding organizations (Project No. 723977). We gratefully acknowledge this financial support. We are also grateful to the reviewers for their valuable comments that helped improving the quality of this paper.

References

Baptiste, P., Le Pape, C., & Nuijten, W. (1999). Satisfiability tests and time-bound adjustmentsfor cumulative scheduling problems. *Annals of Operations research, 92*, 305–333.

Bektaş, T., Ehmke, J. F., Psaraftis, H. N., & Puchinger, J. (2019). The role of operational research in green freight transportation. *European Journal of Operational Research, 274*(3), 807–823.

Braekers, K., & Kovacs, A. A. (2016). A multi-period dial-a-ride problem with driver consistency. *Transportation Research Part B: Methodological, 94*, 355–377.

Breunig, U., Baldacci, R., Hartl, R. F., & Vidal, T. (2019). The electric two-echelon vehicle routing problem. *Computers & Operations Research, 103*, 198–210.

Bruglieri, M., Mancini, S., Pezzella, F., & Pisacane, O. (2019). A path-based solution approach for the green vehicle routing problem. *Computers & Operations Research, 103*, 109–122.

Campelo, P., Neves-Moreira, F., Amorim, P., & Almada-Lobo, B. (2019). Consistent vehicle routing problem with service level agreements: A case study in the pharmaceutical distribution sector. *European Journal of Operational Research, 273*(1), 131–145.

Cortés-Murcia, D. L., Prodhon, C., & Afsar, H. M. (2019). The electric vehicle routing problem with time windows, partial recharges and satellite customers. *Transportation Research Part E: Logistics and Transportation Review, 130*, 184–206.

Desaulniers, G., Errico, F., Irnich, S., & Schneider, M. (2016). Exact algorithms for electric vehicle-routing problems with time windows. *Oper. Res., 64*(6), 1388–1405.

European Commission. (2020). A european strategy for low-emission mobility. https://ec.europa.eu/clima/policies/transport_en, accessed 09.07.2020

Echeverri, L. C., Froger, A., Mendoza, J. E., & Néron, E. (2019). A matheuristic for the multi-period electric vehicle routing problem, 13th metaheuristics international conference, jul 2019. Cartagena de Indias, Colombia.

Feillet, D., Garaix, T., Léhuéde, F., Péton, O., & Quadri, D. (2014). A new consistent vehicle routing problem for the transportation of people with disabilities. *Networks, 63*(3), 211–224.

Felipe, A., Ortuño, M. T., Righini, G., & Tirado, G. (2014). A heuristic approach for the green vehicle routing problem with multiple technologies and partial recharges. *Transportation Research Part E: Logistics and Transportation Review, 71*, 111–128.

Ferro, G., Paolucci, M., & Robba, M. (2018). An optimization model for electrical vehicles routing with time of use energy pricing and partial recharging. *IFAC-PapersOnLine, 51*(9), 212–217.

Froger, A., Mendoza, J. E., Jabali, O., & Laporte, G. (2019). Improved formulations and algorithmic components for the electric vehicle routing problem with nonlinear charging functions. *Computers & Operations Research, 104*, 256–294.

Goeke, D., Roberti, R., & Schneider, M. (2019). Exact and heuristic solution of the consistent vehicle-routing problem. *Transportation Science, 53*(4), 1023–1042.

Gröer, C., Golden, B., & Wasil, E. (2009). The consistent vehicle routing problem. *Manufacturing & service operations management, 11*(4), 630–643.

Haughton, M. A. (2007). Assigning delivery routes to drivers under variable customer demands. *Transportation Research Part E: Logistics and Transportation Review, 43*(1), 157–172.

Hiermann, G., Puchinger, J., Ropke, S., & Hartl, R. F. (2016). The electric fleet size and mix vehicle routing problem with time windows and recharging stations. *European Journal of Operational Research, 252*(3), 995–1018.

Koç, c., Jabali, O., Mendoza, J. E., & Laporte, G. (2019). The electric vehicle routing problem with shared charging stations. International Transactions in Operational Research, 26, 4, 1211–1243

Kovacs, A. A., Golden, B. L., Hartl, R. F., & Parragh, S. N. (2014a). Vehicle routing problems in which consistency considerations are important: A survey. *Networks, 64*(3), 192–213.

Kovacs, A. A., Golden, B. L., Hartl, R. F., & Parragh, S. N. (2015a). The generalized consistent vehicle routing problem. *Transportation Science, 49*(4), 796–816.

Kovacs, A. A., Parragh, S. N., & Hartl, R. F. (2014b). A template-based adaptive large neighborhood search for the consistent vehicle routing problem. *Networks, 63*(1), 60–81.

Kovacs, A. A., Parragh, S. N., & Hartl, R. F. (2015b). The multi-objective generalized consistent vehicle routing problem. *European Journal of Operational Research, 247*(2), 441–458.

Macrina, G., Di, P., Puglia, L., Guerriero, F., & Laporte, G. (2019a). The green mixed fleet vehicle routing problem with partial battery recharging and time windows. *Computers & Operations Research, 101*, 183–199.

Macrina, G., Laporte, G., Guerriero, F., Di, P., & Puglia, L. (2019b). An energy-efficient green-vehicle routing problem with mixed vehicle fleet, partial battery recharging and time windows. *European Journal of Operational Research, 276*(3), 971–982.

Montoya, A., Guéret, C., Mendoza, J. E., & Villegas, J. G. (2017). The electric vehicle routing problem with nonlinear charging function. *Transportation Research Part B: Methodological, 103*, 87–110.

Pisinger, D., & Ropke, S. (2010). Large neighborhood search. In M. Gendreau, & J. Y. Potvin (Eds.), *In handbook of metaheuristics* (pp. 399–419). New York: Springer.

Rodríguez-Martín, I., Salazar-González, J.-J., & Yaman, H. (2019). The periodic vehicle routing problem with driver consistency. *European Journal of Operational Research, 273*(2), 575–584.

Schneider, M., Stenger, A., & Goeke, D. (2014). The electric vehicle-routing problem with time windows and recharging stations. *Transportation Science, 48*(4), 500–520.

Stavropoulou, F., Repoussis, P. P., & Tarantilis, C. D. (2019). The vehicle routing problem with profits and consistency constraints. *European Journal of Operational Research, 274*(1), 340–356.

Tarantilis, C. D., Stavropoulou, F., & Repoussis, P. P. (2012). A template-based tabu search algorithm for the consistent vehicle routing problem. *Expert Systems with Applications, 39*(4), 4233–4239.

Turkeš, R., Sörensen, K., & Hvattum, L. M. (2021). Meta-analysis of metaheuristics: Quantifying the effect of adaptiveness in adaptive large neighborhood search. *European Journal of Operational Research, 292*(2), 423–442.

Vidal, T., Laporte, G., & Matl, P. (2020). A concise guide to existing and emerging vehicle routing problem variants. *European Journal of Operational Research, 286*(2), 401–416.