The electric vehicle routing problem with capacitated charging stations

Aurélien Froger[1] Ola Jabali[2] Jorge E. Mendoza[3] Gilbert Laporte[3,4]

[1] Université de Bordeaux, UMR CNRS 5251 – Inria Bordeaux Sud-Ouest
[2] Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milan, Italy
[3] HEC Montréal, Montréal, Canada
[4] School of Management, University of Bath, Bath, United Kingdom

Submitted for publication in December 2020

Abstract

Electric vehicle routing problems (E-VRPs) deal with routing a fleet of electric vehicles (EVs) to serve a set of customers, while minimizing an operational criterion, e.g., cost or time. The feasibility of the routes is constrained by the autonomy of the EVs, which may be recharged along the route. Much of the E-VRP research neglects the capacity of charging stations (CSs), and thus implicitly assumes that an unlimited number of EVs can be simultaneously charged at a CS. In this paper, we model and solve E-VRPs considering these capacity restrictions. In particular, we study an E-VRP with non-linear charging functions, multiple charging technologies, en route charging, and variable charging quantities, while explicitly accounting for the number of chargers available at privately managed CSs. We refer to this problem as the E-VRP with non-linear charging functions and capacitated stations (E-VRP-NL-C). We introduce a continuous-time model formulation for the problem. We then introduce an algorithmic framework that iterates between two main components: 1) The route generator, which uses an iterated local search algorithm to build a pool of high-quality routes and 2) The solution assembler, which applies a branch-and-cut algorithm to combine a subset of routes from the pool into a solution satisfying the capacity constraints. We compare four assembly strategies on a set of instances. We show that our algorithm effectively deals with the E-VRP-NL-C. Furthermore, considering the uncapacitated version of the E-VRP-NL-C, our solution method identifies new best-known solutions for 80 out of 120 instances.

Keywords— Electric vehicle routing; non-linear charging function; synchronization constraints; mixed integer linear programming; matheuristic; iterated local search; branch-and-cut

1 Introduction

In recent years, competitive prices and technological advances have made electric vehicles (EVs) an attractive alternative to internal combustion engine-powered vehicles for logistics operations (Juan et al., 2016; Pelletier et al., 2016). To allow companies and private vehicle users to fully integrate EVs in their operations and their trips, the operations research community has turned its attention to EV route planing. The resulting literature can be divided into two streams: optimal path and vehicle routing problems. As the name suggests, the first stream focuses on problems in which the objective is to find an *optimal* path from an origin to a destination on a road network, while taking into account the EV's short driving range and accommodating stops at charging stations (CSs) to recharge the vehicle's battery. These problems are out of the scope of this paper, but we refer the reader to the works of Baum et al. (2019) and Baum et al. (2020) for a thorough literature review and a state-of-the-art results for these problems. We focus on the second stream, namely, electric vehicle routing problems (E-VRPs).

E-VRPs consist of designing routes to serve a set of customers using a fleet of EVs. Due to their relatively short driving range, EVs may need to detour to CSs to replenish their battery. This is especially critical in the context of

mid-haul or long-haul routing (Schiffer et al., 2018b; Villegas et al., 2018). Therefore, key decisions in E-VRPs concern not only the assignment of the customers to the vehicles and their sequencing, but also where and how much to charge the vehicles.

One of the main modeling elements in E-VRPs is the charging process of batteries. Some studies assume that EVs are fully recharged whenever they detour to a CS. That is the case, for instance, in the green vehicle routing problem (G-VRP). The G-VRP was introduced by Erdoğan and Miller-Hooks (2012) and tackled by Koç and Karaoglan (2016), Montoya et al. (2016), Andelmin and Bartolini (2017), and Bruglieri et al. (2019a), among others. One of the main assumptions in the G-VRP is that the charging time is constant, meaning that the vehicle takes a fixed amount of time to charge the battery to its full capacity, independently of the initial state of charge (SoC). Authors such as Schneider et al. (2014) and Hiermann et al. (2016) relaxed this assumption and incorporated charging times that linearly depend on the SoC upon arrival at the CS. As Montoya et al. (2017) pointed out, the full charge policy may lead to unnecessary out-of-the-depot charging, which translates into expensive driver idling time and overpriced energy purchases. To overcome this drawback, several researchers have studied problem variants in which the charging time is a decision variable; notable examples include the work of Felipe et al. (2014) Desaulniers et al. (2016), Montoya et al. (2017), Keskin and Çatay (2018), and Froger et al. (2019). In the first two examples, the authors assume that the charge retrieved at a CS is a linear function of the charging time. In reality, however, the battery charging process follows a non-linear function with respect to time (Pelletier et al. (2017)). To account for this, Montoya et al. (2017), Koç et al. (2018), and Froger et al. (2019) modeled the charging process using concave piecewise linear functions, while Lee (2020) considered a general concave and non-decreasing charging function. In the resulting problem, known as the electric vehicle routing problem with non-linear charging function (E-VRP-NL), the charging times no longer rely solely on the quantity of energy charged but also on the initial SoC of the EV.

In all the previously discussed research, the authors assume that CSs are always available to charge a vehicle, thus, implicitly assuming that CSs are *uncapacitated* and can simultaneously charge an unlimited number of EVs. In practice, however, each CS has a fixed and often small number of chargers. The intuition behind neglecting the CS capacity constraints is that accounting for the detour and charging times (or costs) while planning the routes is enough to capture the impact of the charging decisions on the cost and feasibility of solutions. Nonetheless, the long charging times (which may range from tens of minutes to several hours) and the small number of chargers typically available at CSs may generate congestion.

The nature of the congestion problem at CSs largely depends on the access options available to the fleet operator. For instance, a few networks of public (as in accessible to anyone) CSs allow for charging time reservations. In this case, congestion may be neglected and the fleet operator only needs to make sure that routes reach the CSs within the reserved time windows. To accurately model the availability of each charger at a CS, the fleet operator may also manage their allocation to the EVs within the reserved time windows (see for instance Bruglieri et al. (2019b) for an example). In practice, most public charging stations do not allow for reservations. In this case, the exact time at which EVs can access CSs is uncertain. Indeed, upon arrival, the chargers may be busy and the vehicle may have to wait in a queue or detour to a close by station. Keskin et al. (2019) dealt with this issue by explicitly considering expected (i.e., deterministic) time-dependent queuing times at CSs. Kullman et al. (2020) went a step further and proposed dynamic optimization policies that constantly adapt charging and routing decisions depending on the state of CS queues. While these authors demonstrated that their approaches can effectively handle CS access uncertainty, as Villegas et al. (2018) pointed out, most companies are unwilling to bear this risk, and thus decide to install their own private out-of-depot charging infrastructure (e.g., at satellite depots, company office branches, or exclusive-access parking spots in city centers).

Relying on a privately operated charging infrastructure only changes the nature of the congestion problem, but does not solve it. To illustrate this point, we ran a feasibility test on the 120 best-known solutions (BKSs) for the E-VRP-NL reported in Montoya et al. (2017) limiting the number of chargers per CS to one, two, three, and four. According to our results, 55 of the BKSs become infeasible if there is only one charger per CS. This figure drops to 23 and three for the cases with two and three chargers. The only scenario where all BKSs are feasible is when CSs have four chargers (see Appendix A for details). Intuitively, one may think that by merely shifting the starting time of the charging operations, the feasibility problem will be solved. However, our experiments show that this is not only unnecessarily expensive (albeit with a very limited time increase), but more importantly it may be infeasible. Another option to

mitigate the effects of congestion is to increase the number of chargers installed at each CS, but this may not be a viable solution in practice. Indeed, Gnann et al. (2018) predict that by the end of 2020, the purchase and installation costs of a fast charging point (i.e., power rates above 22 kW) will be around 40,000€ and its annual operation cost will reach 4,000€. Thus, if a company decides to invest in out-of-the-depot charging infrastructure, there is little chance that it will decide to install more than a couple of chargers at each CS. In conclusion, we argue that in most practical situations, the congestion at the CSs should be taken into account when planning the routes by explicitly modeling the capacity constraints.

We are aware of only one work taking into account CS capacity constraints when optimizing routing decisions in the context of a private charging infrastructure. Bruglieri et al. (2019b) extended the G-VRP to account for the number of pumps in the alternative refueling stations. They introduced a path-based formulation and solved it using a cutting plane algorithm. They reported optimal solutions on instances up to 20 customers. Out of the field of vehicle routing, some researchers have studied related problems. For instance, Sassi and Oulamara (2014) and Pelletier et al. (2018) considered the problem of scheduling charging operations at the depot, assuming that the routes are given as an input. Both papers considered not only constraints on the number of available chargers, but also electricity grid constraints limiting the amount of power that can be drawn from the electric grid at any given time. The latter is typically a binding constraint for central depots with a large number of available chargers, but is usually not a concern for CSs since they are designed to operate at full capacity. Brandstätter et al. (2020) studied the problem of finding optimal locations and sizes for charging stations based on the number of expected trips. They explicitly took charging station capacity into account using a time-expanded location graph.

In order to help the readers find their way through the different assumptions, Table 1 summarizes the main E-VRPs variants addressed in the literature (it excludes the literature on location-routing problems). The table is not exhaustive, but it provides a good overview of the current state of the research on E-VRPs.

In this paper we introduce the E-VRP-NL with capacitated CSs (E-VRP-NL-C). The problem extends classical E-VRPs to account for the fixed number of chargers available at each CS. We assume that the CSs are privately operated. Our research extends the work of Bruglieri et al. (2019b) by considering a more general charging function and charging policy, which introduces additional decisions. Specifically, rather than assuming that every EV is fully charged in constant time when it stops at a CS, the quantity of energy charged during every stop at a CS is a decision variable. Moreover, since charging functions are piecewise linear, the charging time of an EV depends on its initial SoC and the quantity of energy charged. Last but not least, we also propose an algorithm to tackle medium and large-size instances.

The E-VRP-NL-C is a complex combined routing-scheduling problem belonging to the family of VRPs with synchronization constraints. More specifically, according to the taxonomy introduced by Drexl (2012), the E-VRP-NL-C belongs to the class of VRPs with *resource synchronization*, where vehicles compete to access scarce resources (i.e., the chargers at every CS). The E-VRP-NL-C shares similarities with problems where vehicles need to wait while the loading equipment is busy. Examples of problems in this class include the log truck scheduling problem (El Hachemi et al., 2013; Rix et al., 2015) and routing and scheduling problems arising in public works (Grimault et al., 2017). The E-VRP-NL-C also relates to VRPs with inter-tour resource constraints. In these problems, the scarce resources are located only at the terminal vertex of the routes (i.e., the depot). An example of a problem in this category is the VRP introduced by Hempsch and Irnich (2008), where there is a limited number of ramps at the depot, and therefore only a fixed number of vehicles can be served simultaneously. The problem that is most closely related to the E-VRP-NL-C is the VRP with location congestion introduced by Lam and Van Hentenryck (2016). In this problem, each customer has a given number of requests and a limited number of resources (e.g., conveyors). When a vehicle arrives at a customer location, it must wait until at least one resource becomes available to handle the shipping. However, there exist two fundamental differences between their problem and our E-VRP-NL-C. The first is that in our problem, visiting locations with limited resources (the CSs) is needed for feasibility reasons. As a consequence, not only the number, but also the location and duration of the charging operations depend on the configuration of the routes. In other words, the *tasks* that must be synchronized are not known a priori. The second difference is that in contrast to their problem, in our E-VRP-NL-C, the task synchronization has a direct impact on the solution cost.

The contribution of this paper is twofold. First, we propose a mixed integer linear programming (MILP) formulation for the E-VRP-NL-C, showing how CS capacity constraints can be integrated into a standard E-VRP model. Second, we introduce a framework to solve E-VRPs with CS capacity constraints in general, and the E-VRP-NL-C in particular.

Table 1: Summary of the literature on E-VRPs

Reference	Charging infrastructure		Charging process		Fleet	Model(s)	Main algorithm
	Cong. | Access.	Multiple speed	Partial policy	Function			
Erdoğan and Miller-Hooks (2012)	NEG			C	H	CS-Repl	Heuristics
Felipe et al. (2014)	NEG	✓	✓	L	H	CS-Repl	Local-search heuristics
Schneider et al. (2014)	NEG			L	H	CS-Repl	Hybrid VNS/TS
Goeke and Schneider (2015)	NEG			L	M	CS-Repl	ALNS
Schneider et al. (2015)	NEG			C/L	H	CS-Repl	AVNS
Desaulniers et al. (2016)	NEG		✓	L	H	SP	Branch-and-price[1]
Hiermann et al. (2016)	NEG			L	M	CS-Repl, SP	Branch-and-price + ALNS
Keskin and Çatay (2016)	NEG		✓	L	H	CS-Repl	ALNS
Koç and Karaoglan (2016)	NEG			C	H	CS-Repl, CS-Path-1	Simulated annealing + B&C
Montoya et al. (2016)	NEG			C	H	-	Multi-space sampling heuristic
Montoya et al. (2017)	NEG	✓	✓	PL	H	CS-Repl	Matheuristic
Andelmin and Bartolini (2017)	NEG			C	H	SP (CS-Path)	Two-phase exact algorithm
Leggieri and Haouari (2017)	NEG			C	H	CS-Repl, CS-Path-1	B&C
Keskin and Çatay (2018)	NEG	✓	✓	L	H	CS-Repl	ALNS
Andelmin and Bartolini (2019)	NEG			C	H	CS-Path	Multi-start local search heuristic
Bruglieri et al. (2019a)	NEG			C	H	Customer-Path	B&C (exact and heuristic)
Bruglieri et al. (2019b)	SCH | PUB-R/PO			C	H	CS-Repl, Customer-Path	B&C
Froger et al. (2019)	NEG	✓	✓	PL	H	CS-Repl, CS-Path	MILP Models
Hiermann et al. (2019)	NEG		✓	L	M	-	Matheuristic (genetic + SP)
Macrina et al. (2019)	NEG	✓	✓	L	M	CS-Repl	Iterated local search
Lee (2020)	NEG		✓	Cv	SV	Customer-Path	Branch-and-price
Keskin et al. (2019)	DWT | PUB		✓	PL	H	CS-Repl	ALNS + MILP for route enhancement
Kullman et al. (2020)	DYN | PUB/PO	✓	✓	PL	SV	CS-Repl	Static and dynamic policies
Our research	SCH | PO	✓	✓	PL	H	CS-Path	Matheuristic (ILS + B&C)

[1] Improved results can be found in Desaulniers et al. (2020)

- **Charging infrastructure**:
  - Strategy to deal with congestion at CSs and accessibility of CSs (Cong. | Access.): neglected (NEG), scheduling of charging operations according to the number of available chargers (SCH), deterministic time-dependent waiting times (DWT), dynamic decision making (DYN) | public (PUB) –accessible to anyone, public with reservation (PUB-R), privately operated (PO)
  - Multiple speed: each station may charge at a different speed (e.g., fast, moderate, slow)
- **Charging process**:
  - Partial policy: the quantity of energy charged is a decision
  - Function: constant charging time (C), charging times linearly depending on the quantity of energy charged (L), charging times depending on the quantity of energy charged and on the initial SoC: concave piecewise linear charging function (PL), concave and non-decreasing charging function (Cv)
- **Fleet**: homogeneous (H), mixed (M), single vehicle (SV)
- **Model(s)**: CS replication-based formulation (CS-Repl), path-based formulation in which each path corresponds to a sequence of stops at CSs between customers and/or the depot (CS-Path), CS-Path in which every path contains at most one stop at a CS (CS-Path-1), path-based formulation in which each path corresponds to a sequence of visits to customers between CSs and/or the depot (Customer-Path), classical set partitioning with a variable per route (SP)

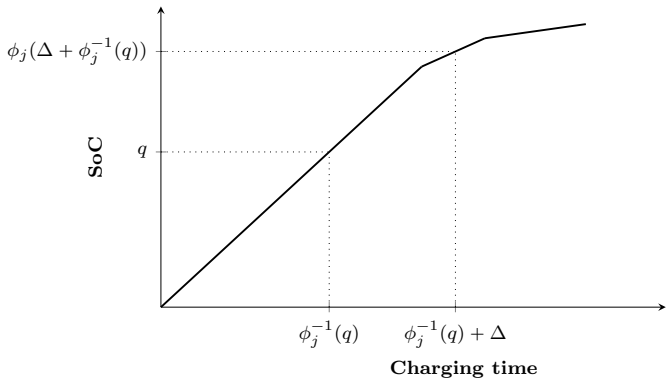

Figure 1: Piecewise linear charging function of a CS $j \in F$.

It is made up of two interacting components: a *route generator* and a *solution assembler*. The first component builds a set of high-quality solutions, while relaxing the capacity constraints. The routes making up these solutions are stored in a pool which is sent to the assembler every few iterations. The latter combines routes from the pool in an attempt to build a solution meeting the capacity constraints. If such a solution does not exist or cannot be found within a given computing time, the assembler sends a signal to the generator to modify the search space in order to favor feasibility. For the particular case of the E-VRP-NL-C, we have developed a route generator based on a new and efficient iterated local search (ILS) for the E-VRP-NL, and a solution assembler based on branch-and-cut. We present four assembly strategies allowing for different tradeoffs between efficiency and effectiveness. We have carried out extensive computational experiments on a large set of instances with different characteristics adapted from available benchmarks. The results demonstrate that our approach can handle the CS capacity constraints effectively. In addition, we report new BKS for 80 out of the 120 instances of a well-established benchmark set for the closely related E-VRP-NL.

The remainder of this paper is organized as follows. Section 2 formally introduces the E-VRP-NL-C. Section 3 describes a MILP formulation of the problem. Section 4 presents the proposed solution method. Section 5 shows the computational results. Finally, Section 6 concludes and outlines research perspectives.

2 Problem description

We define the E-VRP-NL-C as follows. Let I be the set of customers that need to be served and let F be the set of CSs at which recharging can take place. The CSs are privately operated, meaning that there is no uncertainty on their availability. Each customer $i \in I$ has a service time g_i. The customers are served using a homogeneous fleet of EVs. Each EV has a battery of capacity Q (expressed in Wh). At the beginning of the planning horizon, the EVs are located in a single depot, from which they leave fully charged. The depot is continuously open for T_{max} hours. Traveling from one location i (the depot, a customer, or a CS) to another location j incurs a driving time $t_{ij} \geqslant 0$ that corresponds to the shortest path in time from i to j and an energy consumption $e_{ij} \geqslant 0$ associated to this path.

Due to their limited battery capacity, EVs may need to stop en route at CSs. Charging operations can occur at any CS, they are non-preemptive, and EVs can be partially recharged. The CS $j \in F$ has a concave piecewise linear charging function ϕ_j that maps for an empty battery the time spent charging at j and the SoC of the vehicle upon departing from j. If q is the SoC of the EV upon arrival at j and Δ is the charging time, then the SoC of the EV upon departure from j is given by $\phi_j(\Delta + \phi_j^{-1}(q))$ (Figure 1). We denote by $B_j = \{(a_{j0}, q_{j0}), \ldots, (a_{jb_j}, q_{jb_j})\}$ the totally ordered set of breakpoints of the charging function at CS j, where every breakpoint is a (charging time, SoC) pair, sorted in non-decreasing order of time.

Each CS $j \in F$ also has a *capacity*, given by the number of available chargers C_j. Due to the limited capacity of CSs, vehicles may incur waiting times while they queue for a charger. We therefore note that optimal solutions are not necessarily *left-shifted schedules*, as is the case for the E-VRP-NL and other E-VRP variants assuming uncapacitated CSs.

Feasible solutions to the E-VRP-NL-C must satisfy the following conditions: 1) each customer is visited exactly

once by a single vehicle, 2) each route starts at the depot not before time 0 and ends at the depot not later than time T_{max}, 3) each route is energy-feasible, i.e., the SoC of an EV when it enters and leaves from any location lies between 0 and Q, and 4) no more than C_j EVs simultaneously charge at each CS $j \in F$. The objective of the E-VRP-NL-C is to minimize the total time needed to serve all customers, including driving, service, charging, and waiting times. Note that to avoid congestion at CSs, the starting times of the routes can be delayed at no cost. Moreover, an E-VRP-NL-C instance may have no feasible solution.

3 A mixed-integer linear programming formulation

We can classify explicit the mixed integer linear programming formulations for E-VRPs into two categories: CS replication-based formulations and path-based formulations. CS replication-based formulations are the compact MILP formulations typically used in the E-VRP literature to mathematically define the problem and solve toy instances. As their name suggests, in these formulations the nodes representing CSs are replicated and the number of stops at each CS replication is constrained to one. Note that in this case the number of stops at a CS is limited to its number of replications. This modeling artifice allows for tracking of the routes' travel time, distance, and SoC whenever there are several stops at the CS. To ensure that no optimal solution is cut off, the number of replications of each CS must be very large. For instance, Froger et al. (2019) provided an example where this value is equal to $4|I|$, when the congestion at each CS is neglected. CS replication-based formulations yield intractable MILPs and introduce symmetry issues.

Path-based formulations are based on a more intricate modeling strategy by which the problem is defined on a multigraph. In a nutshell, in these formulations, the nodes represent locations (the depot and the CSs or the depot and the customers) and the arcs represent possible paths between every pair of locations. A path may be defined as a sequence (possibly empty) of visits to customers between two CSs or the depot (Bruglieri et al., 2019a,b) or as a sequences of stops at CSs between two customers or the depot (Roberti and Wen, 2016; Andelmin and Bartolini, 2017; Froger et al., 2019). We refer to the former case a as *customer path* and to the latter as a *CS path*. In both cases, dominance rules are defined to limit the number of paths taken into consideration. These formulations no longer require CS vertices replication.

There also exist set partitioning-based formulations (see for instance Desaulniers et al. (2016)), but they are out of the scope of this paper. Table 1 provides details on the formulations used in the E-VRP research.

We introduce a path-based formulation for the E-VRP-NL-C. We decided to use CS paths rather than customer paths for two reasons. First, the number of customer paths grows much faster than the number of CS paths. Second, customer path-based formulations require introducing a sufficient number of clones of paths that do not visit any customer between two CSs, whereas a CS path can at most be traveled by a single EV. We point out that the modeling of the CS capacity constraints we introduce below can easily be adapted to customer path-based formulations.

The concept of CS paths leads to a redefinition of the problem on a directed multigraph $G = (V, P)$, where $V = \{0\} \cup I$, and P is the set of arcs connecting the vertices of V. An arc in P represents a CS path p, starting in vertex $\texttt{org}(p)$ visiting a sequence of CSs or none and ending in vertex $\texttt{dest}(p)$. We denote $\texttt{n}(p)$ the number of CSs in p. If $\texttt{n}(p)$ is equal to 0, then p does not stop at any CS between $\texttt{org}(p)$ and $\texttt{dest}(p)$. Otherwise, we denote by $\texttt{cs}(p, l)$ the l-th CS visited in p (with $l \in \{1, ..., \texttt{n}(p)\}$). We define inputs $\texttt{e}(p)$ and $\texttt{t}(p)$ as the energy consumption (energy used to travel the arcs of the path) and the driving time (time to travel the arcs of the path) initially associated with CS path p. Given two vertices $i, j \in V$, we define P_{ij} as the set of CS paths connecting i to j.

Our path-based formulation of the E-VRP-NL-C involves the following decisions variables. The binary variable x_p is 1 if and only if an EV travels CS path $p \in P$. The continuous variables τ_p and y_p track the time and SoC of an EV when it departs from vertex $\texttt{org}(p)$ to $\texttt{dest}(p)$ using CS path p. To model the charging process at $\texttt{cs}(p, l)$, we introduce a group of variables. The continuous variables $\underline{q}_{pl}$ and $\overline{q}_{pl}$ specify the SoC of an EV when it enters and leaves $\texttt{cs}(p, l)$. For $k \in B_i \backslash \{0\}$, the binary variables $\underline{w}_{plk}$ and $\overline{w}_{plk}$ are equal to 1 if and only if the SoC is between $q_{\texttt{cs}(p,l),k-1}$ and $q_{\texttt{cs}(p,l),k}$ when the EV enters and leaves $\texttt{cs}(p, l)$. The variables $\underline{a}_{pl}$ and $\overline{a}_{pl}$ are the scaled arrival and departure times, according to the charging function of $\texttt{cs}(p, l)$. The continuous variables $\underline{\lambda}_{plk}$ and $\overline{\lambda}_{plk}$ represent the coefficients associated with the breakpoints $(a_{\texttt{cs}(p,l),k}, q_{\texttt{cs}(p,l),k})$ of the piecewise linear charging function, when the EV enters and leaves $\texttt{cs}(p, l)$. The continuous variable Δ_{pl} and ∇_{pl} represents the duration of the charging operation performed at $\texttt{cs}(p, l)$ and the waiting time incurred before charging at this CS. Lastly, the continuous variables $\underline{s}_{pl}$ and $\overline{s}_{pl}$ represent the starting and

completion time of the charging operation performed at $\texttt{cs}(p,l)$.

To model the CS capacity constraints, we propose a flow-based formulation inspired from the one introduced by Artigues et al. (2003) for the Resource Constrained Scheduling Problem. For every CS $j \in F$, we consider C_j resources (represented by the chargers). Each resource can execute at most one operation at any given time. For convenience, we introduce the set O_j of potential charging operations at CS j. Every operation $o \in O_j$ represents a stop at j at a specific position $l(o)$ in a CS path going from $\texttt{org}(o)$ to $\texttt{dest}(o)$. Although there may exist several CS paths between two specific vertices that stop at j at the same position, at most one these CS paths can be selected in a solution. We also introduce two dummy operations $\texttt{o}_j^+$ and $\texttt{o}_j^-$, which act as the source and the sink of the flow for each CS j. Specifically, the resources flow from $\texttt{o}_j^+$ to $\texttt{o}_j^-$ through the charging operations. Let $(o,o') \in \left(O_j \cup \{\texttt{o}_j^+\}\right) \times \left(O_j \cup \{\texttt{o}_j^-\}\right)$ be a couple of charging operations; then the continuous flow variable $f_{oo'}$ denotes the number of chargers that are transferred from operation o to operation o'. When both these operations are not dummy, $f_{oo'}$ is equal to one if and only if these operations are scheduled on the same charger, o' is scheduled after o, and no other operation is scheduled on the charger between the completion of o and the beginning of o'. Binary variable $u_{oo'}$ is equal to 1 if and only if operation o' starts after the completion of operation $o \neq o'$. For notational convenience, we define $\texttt{C}(o) := 1$ for all $o \in O_j$ and $\texttt{C}(\texttt{o}_j^+) := \texttt{C}(\texttt{o}_j^-) := C_j$.

A path-based formulation for the E-VRP-NL-C, denoted as $[P^{path}]$, is as follows:

$$[F^{path}] \quad \text{minimize} \sum_{p \in P} \left(\texttt{t}(p) x_p + \sum_{l=1}^{\texttt{n}(p)} (\Delta_{pl} + \nabla_{pl}) \right) \tag{1}$$

$$\text{subject to} \sum_{j \in V, i \neq j} \sum_{p \in P_{ij}} x_p = 1 \quad i \in I \tag{2}$$

$$\sum_{j \in V, i \neq j} \sum_{p \in P_{ji}} x_p - \sum_{j \in V, i \neq j} \sum_{p \in P_{ij}} x_p = 0 \quad i \in V \tag{3}$$

$$\sum_{j \in V, j \neq i} \sum_{p \in P_{ji}} \left(y_p - \texttt{e}(p) x_p + \sum_{l=1}^{\texttt{n}(p)} (\overline{q}_{pl} - \underline{q}_{pl}) \right) = \sum_{j \in V, j \neq i} \sum_{p \in P_{ij}} y_p \quad i \in I \tag{4}$$

$$y_p - e_{\texttt{org}(p),\texttt{cs}(p,1)} x_p = \underline{q}_{p1} \quad p \in P \tag{5}$$

$$\overline{q}_{p,l-1} - e_{\texttt{cs}(p,l-1),\texttt{cs}(p,l)} x_p = \underline{q}_{pl} \quad p \in P, l \in \{2, ..., \texttt{n}(p)\} \tag{6}$$

$$y_p - \texttt{e}(p) x_p + \sum_{l=1}^{\texttt{n}(p)} (\overline{q}_{pl} - \underline{q}_{pl}) \geqslant 0 \quad i \in I, p \in P_{i0} \tag{7}$$

$$y_p = Q x_p \qquad i \in V \backslash \{0\}, p \in P_{0i} \tag{8}$$

$$y_p \leqslant Q x_p \quad p \in P \tag{9}$$

$$\underline{q}_{pl} = \sum_{k \in B_{\texttt{cs}(p,l)}} \underline{\lambda}_{plk} q_{\texttt{cs}(p,l),k} \qquad p \in P, l \in \{1, ..., \texttt{n}(p)\} \tag{10}$$

$$\underline{a}_{pl} = \sum_{k \in B_{\texttt{cs}(p,l)}} \underline{\lambda}_{plk} a_{\texttt{cs}(p,l),k} \qquad p \in P, l \in \{1, ..., \texttt{n}(p)\} \tag{11}$$

$$\sum_{k \in B_{\texttt{cs}(p,l)}} \underline{\lambda}_{plk} = \sum_{k \in B_{\texttt{cs}(p,l)} \backslash \{0\}} \underline{w}_{plk} \qquad p \in P, l \in \{1, ..., \texttt{n}(p)\} \tag{12}$$

$$\sum_{k \in B_{\texttt{cs}(p,l)} \backslash \{0\}} \underline{w}_{plk} = x_p \qquad p \in P, l \in \{1, ..., \texttt{n}(p)\} \tag{13}$$

$$\underline{\lambda}_{pl0} \leqslant \underline{w}_{pl1} \qquad p \in P, l \in \{1, ..., \texttt{n}(p)\} \tag{14}$$

$$\underline{\lambda}_{plk} \leqslant \underline{w}_{plk} + \underline{w}_{pl,k+1} \qquad p \in P, l \in \{1, ..., \texttt{n}(p)\}, k \in B_{\texttt{cs}(p,l)} \backslash \{0, b_{\texttt{cs}(p,l)}\} \tag{15}$$

$$\underline{\lambda}_{plb_{\texttt{cs}(p,l)}} \leqslant \underline{w}_{plb_{\texttt{cs}(p,l)}} \qquad p \in P, l \in \{1, ..., \texttt{n}(p)\} \tag{16}$$

$$\overline{q}_{pl} = \sum_{k \in B_{\texttt{cs}(p,l)}} \overline{\lambda}_{plk} q_{\texttt{cs}(p,l)k} \qquad p \in P, l \in \{1, ..., \texttt{n}(p)\} \tag{17}$$

$$\overline{a}_{pl} = \sum_{k \in B_{\texttt{cs}(p,l)}} \overline{\lambda}_{plk} a_{\texttt{cs}(p,l)k} \qquad p \in P, l \in \{1, ..., \texttt{n}(p)\} \tag{18}$$

$$\sum_{k \in B_{\texttt{cs}(p,l)}} \overline{\lambda}_{plk} = \sum_{k \in B_{\texttt{cs}(p,l)} \backslash \{0\}} \overline{w}_{plk} \qquad p \in P, l \in \{1, ..., \texttt{n}(p)\} \tag{19}$$

$$\sum_{k\in B_{\mathtt{cs}(p,l)}\setminus\{0\}} \overline{w}_{plk} = x_p \qquad p\in P, l\in\{1,...,\mathtt{n}(p)\} \tag{20}$$

$$\overline{\lambda}_{i0} \leqslant \overline{w}_{pl1} \qquad p\in P, l\in\{1,...,\mathtt{n}(p)\} \tag{21}$$

$$\overline{\lambda}_{plk} \leqslant \overline{w}_{plk} + \overline{w}_{pl,k+1} \qquad p\in P, l\in\{1,...,\mathtt{n}(p)\}, k\in B_{\mathtt{cs}(p,l)}\setminus\{0, b_{\mathtt{cs}(p,l)}\} \tag{22}$$

$$\overline{\lambda}_{plb_{\mathtt{cs}(p,l)}} \leqslant \overline{w}_{plb_{\mathtt{cs}(p,l)}} \qquad p\in P, l\in\{1,...,\mathtt{n}(p)\} \tag{23}$$

$$\Delta_{pl} = \overline{a}_{pl} - \underline{a}_{pl} \qquad p\in P, l\in\{1,...,\mathtt{n}(p)\} \tag{24}$$

$$\sum_{j\in V, j\neq i}\sum_{p\in P_{ji}}\left(\tau_p + \mathtt{t}(p)x_p + \sum_{l=1}^{\mathtt{n}(p)}(\Delta_{pl}+\nabla_{pl})\right) + g_i = \sum_{j\in V, j\neq i}\sum_{p\in P_{ij}} \tau_p \quad i\in I \tag{25}$$

$$\tau_p + \sum_{l=1}^{\mathtt{n}(p)}(\Delta_{pl}+\nabla_{pl}) \leqslant \left(T_{max} - \mathtt{t}(p) - g_{\mathtt{dest}(p)} - t_{\mathtt{dest}(p),0}\right)x_p \quad p\in P \tag{26}$$

$$\tau_p + t_{\mathtt{org}(p),\mathtt{cs}(p,1)}x_p + \nabla_{p1} = \underline{s}_{p1} \quad p\in P \tag{27}$$

$$\overline{s}_{p,l-1} + t_{\mathtt{cs}(p,l-1),\mathtt{cs}(p,l)}x_p + \nabla_{pl} = \underline{s}_{pl} \quad p\in P, l\in\{2,...,\mathtt{n}(p)\} \tag{28}$$

$$\overline{s}_{pl} = \underline{s}_{pl} + \Delta_{pl} \quad p\in P, l\in\{1,...,\mathtt{n}(p)\} \tag{29}$$

$$\sum_{o'\in O_j\cup\{\mathtt{o}_j^+\}} f_{o'o} = \sum_{\substack{p\in P_{\mathtt{org}(o),\mathtt{dest}(o)}\\ :\mathtt{cs}(p,l(o))=\mathtt{j}(o)}} x_p \quad j\in F, o\in O_j \tag{30}$$

$$\sum_{o'\in O_j\cup\{\mathtt{o}_j^+\}} f_{o'o} - \sum_{o'\in O_j\cup\{\mathtt{o}_j^-\}} f_{oo'} = 0 \quad j\in F, o\in O_j \tag{31}$$

$$\sum_{o\in O_j\cup\{\mathtt{o}_j^-\}} f_{\mathtt{o}_j^+,o} = C_j \quad j\in F \tag{32}$$

$$\sum_{o\in O_j\cup\{\mathtt{o}_j^+\}} f_{o,\mathtt{o}_j^-} = C_j \quad j\in F \tag{33}$$

$$\sum_{\substack{p\in P_{\mathtt{org}(o),\mathtt{dest}(o)}\\ :\mathtt{cs}(p,l(o))=\mathtt{j}(o)}} \underline{s}_{p,l(o)} - \sum_{\substack{p\in P_{\mathtt{org}(o'),\mathtt{dest}(o')}\\ :\mathtt{cs}(p,l(o'))=\mathtt{j}(o')}} \overline{s}_{p,l(o')} \geqslant T_{max}\left(u_{o'o}-1\right) \quad j\in F, o, o'\in O_j \tag{34}$$

$$f_{oo'} \leqslant \min(\mathtt{C}(o),\mathtt{C}(o'))u_{oo'} \quad j\in F, (o,o')\in\left(O_j\cup\{\mathtt{o}_j^+\}\right)\times\left(O_j\cup\{\mathtt{o}_j^-\}\right) \tag{35}$$

$$x_p\in\{0,1\} \quad p\in P \tag{36}$$

$$\tau_p\geqslant 0, y_p\geqslant 0 \quad p\in P \tag{37}$$

$$\underline{q}_{pl}, \overline{q}_{pl}, \underline{a}_{pl}, \overline{a}_{pl}, \underline{s}_{pl}, \overline{s}_{pl}, \Delta_{pl}, \nabla_{pl}\geqslant 0 \quad p\in P, l\in\{1,...,\mathtt{n}(p)\} \tag{38}$$

$$\underline{\lambda}_{plk}\geqslant 0, \overline{\lambda}_{plk}\geqslant 0 \quad p\in P, l\in\{1,...,\mathtt{n}(p)\}, k\in B_{\mathtt{cs}(p,l)}\setminus\{0\} \tag{39}$$

$$\underline{w}_{plk}\in\{0,1\}, \overline{w}_{plk}\in\{0,1\} \quad p\in P, l\in\{1,...,\mathtt{n}(p)\}, k\in B_{\mathtt{cs}(p,l)}\setminus\{0\} \tag{40}$$

$$u_{oo'}\in\{0,1\} \quad j\in F, o, o'\in O_j \tag{41}$$

$$f_{oo'}\geqslant 0 \quad j\in F, (o,o')\in\left(O_j\cup\{\mathtt{o}_j^+\}\right)\times\left(O_j\cup\{\mathtt{o}_j^-\}\right). \tag{42}$$

The objective function (1) minimizes the total driving, charging, and waiting time. Constraints (2) ensure that each customer is visited exactly once. Constraints (3) impose flow conservation: at each vertex the number of incoming EVs is equal to the number of outgoing EVs. Constraints (4) track the SoC of EVs at each customer location. Constraints (5) track the SoC at the arrival at the first CS of each CS path. Constraints (6) couple the SoC of an EV that leaves a CS with its SoC upon arrival to the next CS of the same path. Constraints (7) ensure that if the EV travels from a customer to the depot, it has sufficient energy to reach its destination. Constraints (8) state that every EV leaves the depot with a fully charged battery. Constraints (9) couple the SoC tracking variables with the path travel variables. Constraints (10)–(24) model the relationship between the SoCs of an EV upon entering and leaving a CS and the charging time. Constraints (25) track the departure time at each customer. Constraints (26) couple the time tracking variable with the path travel variables, and impose the route duration limit. Constraints (27) and (28) define the starting and completion times of every charging operation, as well as the potential waiting time before the start of the operation. Constraints (29) define the completion time of the charging operations based on their starting time and duration. Constraints (30) state that a charger has to be allocated to every charging operation of each selected CS path (at most one CS path in the sum can be selected in a solution). Constraints (31) ensure flow conservation or the chargers of each CS.

Constraints (32) and (33) compute the flow value at the beginning and at the end of the time horizon, which is equal to the number of available chargers at each CS. Constraints (34) couple the sequencing variables with the starting time of the corresponding charging operations. Constraints (35) couple the flow variables with the operation sequence variables. Specifically, a charger can be sent from a charging operation o' to another charging operation o if o starts after the completion of o'. Finally, constraints (36)–(42) set the domains of the decision variables.

Without preprocessing, the number of CS paths explodes with the number of CSs and the number of customers. However, a large number of these arcs cannot be part of an optimal solution. Froger et al. (2019) presented a filtering procedure to reduce the number of CS paths based on the definition of a dominance rule between two CS paths having the same origin and destination. Due to the potential waiting times that can occur at CSs, we cannot apply the dominance rule described in the work of Froger et al. (2019) between CS paths with the same origin and destination if they both contain CSs. This is due to the fact that waiting times depend on all the CS paths selected to build a complete solution. In our computational experiments, we apply the dominance rule between the unique CS path with no CS between i and j (if it exists) and every other CS path of P_{ij}, with i and j in V.

4 Solution method

In this section, we introduce a matheuristic to solve the E-VRP-NL-C. The method relies on an algorithmic framework based on two interacting components: a route generator and a solution assembler. The first component builds a pool of high-quality and diverse routes exploring the solution space of the problem without CS capacity constraints, while the second component recombines routes from the pool trying to build a solution satisfying the CS capacity constraints.

Algorithm 1 outlines the general structure of the method. The algorithms starts by generating an initial solution without taking the CS capacity constraints into account (line 1). In our implementation, this step is carried out using a modified version of the classical Clarke and Wright heuristic. Next, the algorithm enters the main loop (lines 3–27). During n_{max} iterations the algorithm alternates between the *route generation* and *solution assembly* phases. In particular, the algorithm uses procedure `generateRoutes(·)` to retrieve a triplet $(\Phi_O, \Phi_R, \Omega^{ST})$, where Φ_O, Φ_R are sets of high-quality solutions for the *original* and *relaxed* problems (i.e., with and without CS capacity constraints), and Ω^{ST} is a set of independent, feasible, and high-quality routes. The latter will be referred to as the *short-term* pool, since it is reset before each call to the route generator. Next (lines 4–12) the algorithm adds the routes making up the solutions in Φ_O and Φ_R to a *long-term* pool Ω^{LT} (never reset) while keeping track of the best solutions for both the original (s_O^*) and the relaxed problem (s_R^*). The intuition behind adding routes coming from potentially infeasible solutions to the original problem (i.e., solutions to the relaxed problem) to the long-term pool is to foster diversity. Indeed, these routes tend to be of high quality (in terms of the objective function) and might be recombined efficiently with others later.

The algorithm then enters the assembly phase (line 13). In the first step of this phase, the algorithm merges the short- and long-term pools Ω^{ST} and Ω^{LT}. Thus, seeking to combine the past and recent history of the search into a single set of routes that has a size that is manageable for the solution assembler. The assumption is that a new part of the search space has been explored since the last call to the assembler. The algorithm then calls procedure `assemble(·)` to retrieve a tuple (s', S'), where s' is the best solution and S' is the set of all improving solutions (with respect to s_O^*) for the original problem found during the call. If the assembler retrieves a solution, the algorithm adds its routes in S' to the long-term pool Ω^{LT} and updates the incumbent s_O^* (lines 14–17).

If after the call to the assembler, the algorithm still has not found a feasible solution (i.e., s_O^* is still equal to `NULL`), it implements two actions. First, it slightly modifies the search space to favor feasibility. In our implementation, we modify the search space by artificially reducing the opening hours of the depot (line 20). The underlying idea is that the reduction of the depot's operating hours would lead to shorter routes with more slack to accommodate a late departure from the depot or waiting times at charging stations during the assembly phase. Second, it tries to *repair* the best-known solution for the relaxed problem (i.e., s_R^*) by making it comply with the new solution space (line 21). In our implementation, if a route visiting n customers has a duration strictly greater than the new value T of the depot hours, the algorithm creates two new routes by adding a return to the depot after the $\lfloor n/2 \rfloor^{th}$ customer and reoptimizes the charging decisions within these routes. To avoid feasibility issues, the value of T is not considered when a route visits only one customer. The same procedure is performed on the newly created routes as long as the routes contain more than one customer and their duration exceeds T. On the other hand, if after the call to the assembler a feasible

solution is available (i.e., s_O^* is different than NULL), the algorithm then reestablishes (if needed) the original solution space (i.e., resets the value of the depot opening hours to T_{max}) and moves to a new iteration.

In our implementation of the framework, we develop an iterative local search algorithm as the route generator and a branch-and-cut algorithm as the solution assembler. The remainder of this section describes these two components.

Algorithm 1: Solution method - general structure

```
/* f(s) denotes the value of the objective function for a solution s and we assume that f(NULL) = +∞*/
s_R^0 ←generateInitialSolution()
n ← 0, T ← T_max, Ω^LT ← ∅, s_R^* ← s_R^0, s_R ← s_R^0, s_FO^* ← NULL, s_O ← NULL
while n < n_max do
    (Φ_O, Φ_R, Ω^ST) ←generateRoutes(s_R,s_O,T)      (see Algorithm 2)
    for each s'_R ∈ Φ_R do
        Add the routes of s'_R to Ω^LT
        if f(s'_R) < f(s_R^*) then s_R^* ← s'_R
    end
    for each s'_O ∈ Φ_O do
        Add the routes of s'_O to Ω^LT
        if f(s'_O) < f(s_O^*) then s_O^* ← s'_O
    end
    (s', S') ← assemble(Ω^ST ∪ Ω^LT, s_O^*)   (see Algorithm 3) /* s' = NULL if no improving solution is found */
    if s' ≠ NULL then
        s_O^* ← s'
        if f(s') < f(s_R^*) then s_R^* ← s'
        for each s ∈ S' do Add the routes of s to Ω^LT
    end
    if s_O^* = NULL then
        T ← max{T_min, α · T} /* T_min : minimum possible value for T, α < 1 a tuning parameter */
        s_R ← repair(s_R^*), s_O ← NULL
    else
        if T < T_max then T ← T_max
        s_O ← s_O^*, s_R ← s_R^*
    end
    n ← n + 1
end
return s_O^*
```

4.1 Route generator: an iterated local search algorithm for the E-VRP-NL

Our route generator is based on an ILS algorithm that solves the E-VRP-NL, while populating the pool of routes that is used to assemble solutions to the E-VRP-NL-C. Introduced by Lourenço et al. (2003), ILS is a metaheuristic that iteratively applies a local search phase to produce local optima, and perturbation mechanisms to escape from them. It is initialized with a solution generally provided by a constructive heuristic. In our implementation, we combine ILS with a variable neighborhood descent (VND) search strategy for the local search phase. Algorithm 2 outlines the general structure of our method. First, the current best solution s_R^* is set equal to the initial solution s_R^0 provided as input. The algorithm then enters an iterative process. Except during the first iteration, it perturbs the current best solution s_R^* to escape from the current local optimum and potentially explore a new region of the search space (see §4.1.2). This produces a new start point s'_R for the VND which computes a new local optimum s_R to the E-VRP-NL (see §4.1.1) and also returns the best solution s_O to the E-VRP-NL-C it has encountered. Note that we only check the CS capacity constraints after accepting a move (i.e., when building a new solution to the E-VRP-NL). If appropriate, the algorithm updates the best-known solutions s_R^* and s_O^*, as well as the sets Φ_R and Φ_O of improving solutions. It also populates a pool of routes Ω. Note that we only add a route to Ω if it does not already contains a route visiting the same sequence of vertices. This procedure is reiterated until the targeted number of iterations δ^{max} has been reached. The optimization returns the sets of improving solutions to the E-VRP-NL and E-VRP-NL-C and the generated pool of routes Ω. To

speed up the ILS algorithm, its implementation is based on the static move descriptor concept introduced by Zachariadis and Kiranoudis (2010) which prevents unnecessary reevaluations of moves and provides an efficient way of exploring neighborhoods (see Appendix B).

Algorithm 2: The route generating procedure

Input : a solution s_R^0 to the E-VRP-NL, a solution s_O^0 to the E-VRP-NL-C (possibly equal to `NULL`), a maximum route duration limit T

Output: a set Φ_R and a set Φ_O of improving solutions to the E-VRP-NL and to the E-VRP-NL-C, a pool of routes Ω

Procedure `generateRoutes`(s_R^0,s_O^0,T)**:**

`/* We denote` $f(s)$ `the value of the objective function for a solution` s `and we assume` f(`NULL`) $= +\infty$ `*/`

$\delta \leftarrow 0$, $\Phi_R \leftarrow \emptyset$, $\Phi_O \leftarrow \emptyset$, $\Omega \leftarrow \emptyset$

$s_R^* \leftarrow s_R^0$, $s_O^* \leftarrow s_O^0$

while $\delta < \delta^{max}$ **do**

  if $\delta = 0$ **then** $s'_R \leftarrow s_R^0$

  else $s'_R \leftarrow$`perturb(`s_R^*`,`T`)` (see §4.1.2)

  $(s_R, s_O) \leftarrow$ `VND(`s'_R`,`T`)` (see Algorithm 5)

  Add the routes of s_R to Ω

  if $f(s_R) < f(s_R^*)$ **then** $\Phi_R \leftarrow \Phi_R \cup \{s_R\}$, $s_R^* \leftarrow s_R$

  if $f(s_O) < f(s_O^*)$ **then** $\Phi_O \leftarrow \Phi_O \cup \{s_O\}$, $s_O^* \leftarrow s_O$

  $\delta \leftarrow \delta + 1$

end

return (Φ_R, Φ_O, Ω)

4.1.1 The VND search phase

The VND relies on an ordered list of local search operators. A single operator is considered at a time. If an improving move is found, the search restarts with the first operator of the list. Otherwise, it moves to the next operator. The search reaches a local optimum when the last operator fails to improve the current solution.

Our VND employs several classical VRP operators focusing on sequencing decisions. These operators are defined for solutions represented as sequences of customer visits without CSs. We use five vertex exchanges operators that work by relocating or exchanging customer visits : 1-0, 2-0, 1-1, 2-1, and 2-2 vertex exchanges. We also use the inter-route and intra-route versions of 2-opt. We also define a specific operator for the E-VRP-NL, referred to as *separate*. This operator creates two routes from a single route by inserting a return to the depot after a customer visit. It may improve the cost of a solution if at least one CS is part of the split route. Indeed, creating two routes rather than one may decrease the total time when an expensive detour to a CS is avoided. We stop applying an operator as soon as it has found an improving move. We refer to Algorithm 5 in Appendix C for a description of the general scheme of the VND search phase.

We only consider CSs when evaluating local search moves. In order to make charging decisions in such a way that every route involved in a move has the lowest possible duration, we solve one (inter-route moves) or more (intra-route moves) fixed route vehicle charging problems (FRVCPs). In a nutshell, this problem consists in inserting CSs into a fixed route (i.e., sequence of customers) trying to gain (energy) feasibility while minimizing the total duration of the route. To this end, we use the labeling algorithm described by Froger et al. (2019) that applies shortest path techniques similar to the one presented in Baum et al. (2019). After each call, the duration of the route is stored in a cache memory to avoid recomputing it. It is worth noting that this procedure yields a "true" evaluation of the move. Indeed, all local search-based metaheuristics for E-VRPs reported in the literature use only approximate evaluations of the moves. The reason is that, in a quest for computational efficiency, these approaches embed local search operators that either do not revise the charging decisions (where, when, and how much to charge) or solve the resulting FRVCP heuristically.

Since only improving solutions are accepted in the VND framework, we only solve FRVCPs for potentially improving moves. Therefore, we rely on a procedure that filters out unpromising or infeasible moves as done for example in Schiffer et al. (2018a) and Hiermann et al. (2019). Such moves are determined by establishing a lower bound on the duration of the routes resulting from a move. The value of this lower bound is computed as the sum of two terms: the minimum increase in time to detour to a CS between two successive vertices in the route, and a lower bound on the charging time. The latter is computed by dividing the sum of the energy consumption of the route and the minimum increase in

energy consumption to detour to a CS between two successive vertices in the route, by the steepest slope for a segment of the piecewise linear charging functions. Similar ideas are used in shortest path algorithms for electric vehicles within the computation of a lower bound on the trip time from any vertex to the target vertex (see Baum et al. (2019)). We note that this lower bound corresponds to the true duration of a route in the case where its energy consumption does not exceed Q. If the lower bound on the route duration exceeds the limit, then the route is infeasible. Otherwise, the procedure checks whether the route duration is stored in the cache memory and modifies the lower bound value accordingly. We use the lower bound on the duration of the routes to determine whether the move is strictly non-improving. We refer to Algorithm 6 in Appendix C for a detailed explanation on this procedure. If a move is not discarded by the lower bound on the basis that it is infeasible or non-improving, we solve FRVCPs for all the routes that have not been evaluated exactly, as long as the move remains feasible and potentially improving.

4.1.2 The perturbation phase

Whenever we reach a local optimum, we perturb the current solution by removing geographically close customers and by reinserting them at different positions. First, we randomly select a customer $i \in I$. We then remove the κ closest customers to i from their respective routes, with κ randomly selected in the interval $[\min\{|I|, 5\}, \max\{\min\{|I|, 5\}, \lceil\sqrt{|I|}\rceil\}]$. We set the distance between two customers i_1 and i_2 as equal to $0.5\left(t_{i_1,i_2} + t_{i_2,i_1} + (e_{i_1,i_2} + e_{i_2,i_1})/\rho^*\right)$. The value of ρ^* corresponds to the steepest slope for a segment of the piecewise linear charging functions. Finally, customers are reinserted in the solution one at a time and in a random order by applying the following rules. A removed customer cannot be reinserted in the same route from which it was removed. We evaluate the increase in time of every feasible insertion of the customer. This is done by reoptimizing the charging decisions. The probability of selecting a given feasible insertion is set inversely proportional to the time increase due to the insertion. If there exists no feasible insertion, we simply create a new route with the customer.

4.2 Solution assembler: a branch-and-cut method

The objective of the second component of the matheuristic is to assemble the best possible solution to the E-VRP-NL-C from the pool of routes Ω, obtained from the route generator component. We recall that the charging decisions made for a route are such that its total duration is minimized. The main challenge of this second component is to combine the routes in a solution that satisfies the CS capacity constraints.

Determining the best solution that can be built from selected routes in a pool is a strategy that has been successfully applied to several hardcore VRPs. Usually, this reduces to solving a set partitioning model (Alvarenga et al., 2007; Subramanian et al., 2013; Villegas et al., 2013; Montoya et al., 2017; Andelmin and Bartolini, 2019). The strategy is used either as a post-optimization phase or as an intensification phase within a methaheuritic. An example of the latter is the matheuristic proposed by Subramanian et al. (2013) to solve a class of VRPs. This approach has been mostly applied to problems without route coupling constraints (i.e., the feasibility of one route is independent of the feasibility of other routes). Due to the CS capacity constraints, the route coupling constraints need to be accounted for in the E-VRP-NL-C. To the best of our knowledge, only two studies have dealt with route coupling constraints in the assembly phase of a solution from a pool of routes: Morais et al. (2014) and Grangier et al. (2017), both in the context of cross-dock VRPs.

In this work, we propose assembling the solutions using a decomposition of the problem into a route selection master problem and a CS capacity management subproblem. The master problem consists in selecting a set of routes such that every customer is covered by exactly one route. The charging decisions within the selected routes (as imported from the first component) may lead to a violation of the CS capacity constraints. In such cases, the subproblem checks whether the CS capacity constraints can be met by revising some of the decisions in these routes.

We propose four versions of the subproblem, which primarily depend on the degree of allowed modifications on the routes selected by the master problem. In the first version (denoted by N – **N**o possible revision), we do not revise the charging decisions in the selected routes and we only check whether the CS capacity constraints are satisfied. In the second version (denoted by D – **D**elay), we allow delaying the starting time of each route (i.e., we postpone the departure of the EV from the depot) to satisfy the CS capacity constraints. We note that in versions N and D no increase in the total time of the solution is incurred. In the third version (denoted by DW – **D**elay+**W**aiting times), we

allow delaying the starting time of each route, and we also allow vehicles to wait for a charger if a CS is overcrowded by EVs. In DW, waiting times can only occur when a route stops several times for charging, since, when there is only one stop at a CS, delaying the starting time of the route is preferable, as this does not penalize the objective function. In the fourth version (denoted by DWR – **D**elay+**W**aiting times+**R**evision of charging amounts), we also revise the amount of energy charged at the CSs. If a route contains at least two stops at CSs, we may decide to charge more during the first stop in order to avoid waiting at the second stop before charging the EV. We note that this strategy may sometimes avoid detouring to one of the CSs visited in the original route.

We designed a branch-and-cut algorithm to efficiently solve the assembling problem while exploiting the decomposition discussed above. While solving the route selection problem, we dynamically solve the CS capacity management subproblem. More specifically, at each node of the branch-and-bound tree, we solve the subproblem for the current selection of routes. Depending on the version of the subproblem, we generate different cuts to discard infeasible route selections (see §4.2.3). For versions DW and DWR, we also add cuts to account for the underestimation of the increase in the total time to visit the customers.

We outline the general structure of the branch-and-cut method in Algorithm 3 (we focus on its most common used implementation based on interacting with the branch-and-bound procedure embedded into the MILP solver via callback routines). When provided as an input, we give to the solver the objective function value of a solution to the E-VRP-NL-C as a cutoff value (i.e., an upper bound on the value of the objective function of the master problem). Therefore, the solver only considers the solutions with an objective function value strictly less than this cutoff value. The solver returns the best solution assembled from a given pool of routes by the branch-and-cut algorithm. The component also returns a set Φ of improving solutions (compared to the initial solution s) computed throughout the algorithm. Whenever a selection of routes $\bar{x}$ at an integer node of the branch-and-bound tree does not lead to the introduction of a cut after solving the capacity management subproblem $CMP_.(\bar{x})$, we save it along with the updated charging decisions (delays, waiting times, revision of charging amounts) to build a complete solution.

Algorithm 3: The assembling procedure (focus on the implementation)

Input : a pool Ω of routes / a solution s (possibly equal to `NULL`) to the E-VRP-NL-C

Output: the best solution s^* computed by the branch-and-cut algorithm / a set Φ of improving solutions to the E-VRP-NL-C (in comparison with s)

Procedure `assemble(`Ω,s`)`**:**

- Build the MILP formulation $[HC1]$ or $[HC2]$ of the route selection problem from pool Ω (see §4.2.1 and §4.2.2)
- Define the callback function (function called at every node of the branch-and-bound tree) as procedure `SolveSubProblem`$(\bar{x})$ (see Algorithm 4) and associate to it a CPU time limit of τ^{SP} per call
- Give the model $[HC.]$, the callback function, the cutoff value $f(s)$ (objective function value of s) to the MILP solver and launch it with a CPU time limit equal to τ
- Retrieve the best solution s^* computed within the CPU time limit and the set Φ of improving solutions (compared to s)

return (s^*, Φ)

In the following subsections, we provide a detailed description of our four versions of the CS capacity management subproblem. We use the following notation. The binary parameter $\mathtt{a}(r, i)$ is equal to one if and only if route $r \in \Omega$ serves customer $i \in I$. We define parameter $\mathtt{t}(r)$ as the duration of a route $r \in \Omega$, obtained from the route generation component. We denote by $O(\Omega)$ the set containing all the charging operations occurring in the routes belonging to Ω, and by $O_j(\Omega) \subseteq O(\Omega)$ all the charging operations occurring at CS $j \in F$ in these routes. We denote by $O(\{r\})$ the list of charging operations occurring in route r. Let $\mathtt{r}(o)$ and $\mathtt{j}(o)$ be the route and the CS associated with a charging operation o. Let $\mathtt{S}(o)$ and $\mathtt{d}(o)$ be the starting time and duration of charging operation o in the route $\mathtt{r}(o) \in \Omega$. For each route $r \in \Omega$, we assume that the operations in $O(\{r\})$ are sorted in non-decreasing order of their starting times. We denote by $\mathtt{suc}(o)$ the charging operation following o in route $\mathtt{r}(o)$. If there does not exist any charging operation after o, we set $\mathtt{suc}(o) = -1$. For every route $r \in \Omega$ and for each charging operation $o \in O(\{r\})$ we denote by $\mathtt{t}^+(o)$ the total time (driving and customer serving) spent in r by the EV between its departure from $\mathtt{j}(o)$ and its arrival at $\mathtt{j}(\mathtt{suc}(o))$ or at the depot if $\mathtt{suc}(o) = -1$. For each operation o, we also introduce two parameters $\mathtt{ES}(o)$ and $\mathtt{LS}(o)$ representing its earliest and latest possible starting times. The values of these parameters depend on the different versions of the subproblem and they are specified in the following subsections.

4.2.1 N and D subproblem versions

To model the route selection problem, we introduce a binary variable x_r equal to 1 if and only if route $r \in \Omega$ is selected. A MILP formulation of this problem is then the following classical set partitioning model:

$$[HC1] \quad \text{minimize} \sum_{r\in\Omega} \mathtt{t}(r)x_r \tag{43}$$

$$\text{subject to} \sum_{r\in\Omega} \mathtt{a}(r,i)x_r = 1 \quad i \in I \tag{44}$$

$$x_r \in \{0,1\} \quad r \in \Omega. \tag{45}$$

The objective (43) is to select a subset of routes from Ω that minimizes the total duration. Constraints (44) ensure that each customer is visited exactly once. Constraints (45) set the domains of the decision variables.

Let $\Omega_{1^+}(\bar{x})$ denote the set of routes with at least one charging operation resulting from a feasible solution $\bar{x}$ to the above problem, i.e., $\Omega_{1^+}(\bar{x}) = \{r \in \Omega : \bar{x}_r = 1 \wedge O(\{r\}) \neq \varnothing\}$. Note that the routes without mid-route charging do not need to be considered when verifying the CS capacity constraints. We also define $\Omega_{1^+}(\bar{x}, j) \subseteq \Omega_{1^+}(\bar{x})$ as the set of routes with a charging operation at $j \in F$.

Version N In this version of the subproblem, the CS capacity management subproblem does not revise the charging operations scheduled in the selected routes. It only checks the feasibility of the charging operations with respect to the capacity constraints. We therefore set $\mathtt{ES}(o) = \mathtt{LS}(o) = \mathtt{S}(o)$. Let $\text{CMP}_1(\bar{x})$ be this subproblem defined for the routes that belong to $\Omega_{1^+}(\bar{x})$. It can be decomposed into $|F|$ independent problems, one for every CS. To solve $\text{CMP}_1(\bar{x})$, for every CS $j \in F$ we apply a polynomial algorithm to check the existence of subsets of operations overloading the CS. Specifically, we call procedure $\mathtt{CheckCapacityCut}(O_j(\Omega_{1^+}(\bar{x})), C_j)$ to check whether the capacity constraints are satisfied according to the scheduled charging operations in $O_j(\Omega_{1^+}(\bar{x}))$. We refer the reader to Algorithm 7 in Appendix D for the details of this procedure.

Version D The main drawback of version N is that it may discard promising routes. Indeed, in some cases simply delaying the starting time of some of the routes may allow their combination into a feasible solution to the problem. Note that shifting the start time of a route does not increase the objective function. In version D, we seek to resolve capacity violations by shifting starting times of the routes. Let $\text{CMP}_2(\bar{x})$ be this subproblem. In contrast to $\text{CMP}_1(\bar{x})$, $\text{CMP}_2(\bar{x})$ does not decompose into an independent problem for each CS. Let o be a charging operation. Its earliest starting time $\mathtt{ES}(o)$ is equal to $\mathtt{S}(o)$ since by construction the operations are left shifted in each route of the pool. The parameter $\mathtt{LS}(o)$ is computed by subtracting from T_{max} the time needed to complete the route (considering the duration of the next charging operations, the driving times, and no waiting times). Specifically, $\mathtt{LS}(o) = T_{max} - \sum_{o'\in O(\{\mathtt{r}(o)\}):\mathtt{S}(o')\geqslant\mathtt{S}(o)}(\mathtt{t}^+(o') + \mathtt{d}(o'))$.

We now define a continuous-time MILP formulation of $\text{CMP}_2(\bar{x})$. Let the variable S_o define the starting time of operation $o \in O(\Omega_{1^+}(\bar{x}))$. We model the capacity constraints using the flow-based formulation used in Section 3. For each CS $j \in F$, we consider two dummy operations $\mathtt{o}_j^+$ and $\mathtt{o}_j^-$, and we define $\mathtt{C}(o) := 1$ for all $o \in O_j(\Omega_{1^+}(\bar{x}))$ and $\mathtt{C}(\mathtt{o}_j^+) := \mathtt{C}(\mathtt{o}_j^-) := C_j$. We then introduce the continuous variable $f_{oo'}$ representing the quantity of resource (i.e., chargers) that is transferred from charging operation o to charging operation o'. We also define the sequential binary variable $u_{oo'}$ taking the value of 1 if operation o is processed before operation o'. The MILP formulation of $\text{CMP}_2(\bar{x})$ is as follows:

$$[\text{CMP}_2(\bar{x})] \quad \text{minimize } 0 \tag{46}$$

$$\text{subject to} \quad u_{oo'} + u_{o'o} \leqslant 1 \quad j \in F, o, o' \in O_j(\Omega_{1^+}(\bar{x})) : o < o' \tag{47}$$

$$u_{oo''} \geqslant u_{oo'} + u_{o'o''} - 1 \quad j \in F, o, o', o'' \in O_j(\Omega_{1^+}(\bar{x})) \tag{48}$$

$$S_{o'} \geqslant S_o + \mathtt{d}(o)u_{oo'} + (\mathtt{LS}(o) - \mathtt{ES}(o'))(u_{oo'} - 1) \quad j \in F, o, o' \in O_j(\Omega_{1^+}(\bar{x})) \tag{49}$$

$$S_{\mathtt{suc}(o)} = S_o + \mathtt{d}(o) + \mathtt{t}^+(o) \quad o \in O(\Omega_{1^+}(\bar{x})) : \mathtt{suc}(o) \neq -1 \tag{50}$$

$$\sum_{o'\in O_j(\Omega_{1^+}(\bar{x}))\cup\{\mathtt{o}_j^+\}} f_{o'o} = 1 \quad j \in F, o \in O_j(\Omega_{1^+}(\bar{x})) \tag{51}$$

$$\sum_{o'\in O_j(\Omega_{1^+}(\bar{x}))\cup\{\mathtt{o}_j^+\}} f_{o'o} - \sum_{o'\in O_j(\Omega_{1^+}(\bar{x}))\cup\{\mathtt{o}_j^-\}} f_{oo'} = 0 \quad j \in F, o \in O_j(\Omega_{1^+}(\bar{x})) \tag{52}$$

$$\sum_{o \in O_j(\Omega_{1^+}(\bar{x})) \cup \{\mathtt{o}_j^-\}} f_{\mathtt{o}_j^+, o} = C_j \quad j \in F \tag{53}$$

$$\sum_{o \in O_j(\Omega_{1^+}(\bar{x})) \cup \{\mathtt{o}_j^+\}} f_{o, \mathtt{o}_j^-} = C_j \quad j \in F \tag{54}$$

$$f_{oo'} \leqslant \max\left(\mathtt{C}(o), \mathtt{C}(o')\right) u_{oo'} \quad j \in F, o, o' \in O_j(\Omega_{1^+}(\bar{x})) \tag{55}$$

$$\mathtt{ES}(o) \leqslant S_o \leqslant \mathtt{LS}(o) \quad o \in O(\Omega_{1^+}(\bar{x})) \tag{56}$$

$$f_{oo'} \geqslant 0 \quad j \in F,$$

$$(o, o') \in \left(O_j(\Omega_{1^+}(\bar{x})) \cup \{\mathtt{o}_j^+\}\right) \cup \left(O_j(\Omega_{1^+}(\bar{x})) \cup \{\mathtt{o}_j^-\}\right) \tag{57}$$

$$u_{oo'} \in \{0, 1\} \quad j \in F, o, o' \in O_j(\Omega_{1^+}(\bar{x})). \tag{58}$$

$\mathrm{CMP}_2(\bar{x})$ is a feasibility problem. Constraints (47) state that for two distinct operations o and o', either o precedes o', o' precedes o, or o and o' are processed in parallel (if there is more than one charger at the CS). Constraints (48) express the transitivity of the precedence relations. Constraints (49) are the disjunctive constraints on the operations related to the same CS. Each such constraint is active when $u_{oo'} = 1$ and, in which case, it enforces the precedence relation between charging operations o and o'. Note that no waiting times can occur before a charging operation. Constraints (50) enforce the precedence relation and the time lag between the charging operations occurring in the same route. Constraints (51) state that a charger has to be allocated to each charging operation. Constraints (52) ensure flow conservation. Constraints (53) and (54) define the value of the flow leaving the source and the flow entering the sink. Constraints (55) couple the flow variables with the sequence variables. Constraints (56) and (58) define the domains of the decision variables.

4.2.2 DW and DWR subproblem versions

In these versions of the subproblem, we allow a possible increase in the total duration of the routes. Indeed, introducing waiting times at CSs or revising the amount of charged energy may help resolve capacity violations, but such modifications may extend routes due to the non-linearity of the charging functions and the consideration of multiple charging technologies. Let θ be a non-negative variable estimating the added delay when solving the CS capacity management subproblem. A MILP formulation of the route selection problem (derived directly from $[HC1]$) follows:

$$[HC2] \quad \text{minimize} \sum_{r \in \Omega} \mathtt{t}(r) x_r + \theta \tag{59}$$

$$\text{subject to} \quad (44), (45)$$

$$\theta \geqslant 0. \tag{60}$$

Thereafter, we assume that we have a fixed selection $\Omega_{1^+}(\bar{x})$ of routes given by fixing the variables $\{x_r\}_{r \in \Omega}$ to values respecting the current constraints of the route selection problem.

Version DW In this version of the subproblem, we assume that EVs can wait at CSs if delaying the starting times of the routes is not sufficient to avoid capacity violations. Let $\mathrm{CMP}_3(\bar{x})$ be the scheduling subproblem of the routes $\Omega_{1^+}(\bar{x})$, which has the objective of minimising the addition of waiting times. The MILP formulation of $\mathrm{CMP}_3(\bar{x})$ uses the decision variables S_o, $f_{oo'}$, $u_{oo'}$ defined in $\mathrm{CMP}_2(\bar{x})$. We also introduce variable ∇_o that represents the waiting time incurred before the start of charging operation $o \in O(\Omega_{1^+}(\bar{x}))$. For every charging operation o, its earliest starting time $\mathtt{ES}(o)$ is equal to $\mathtt{S}(o)$ and its latest starting time $\mathtt{LS}(o)$ is equal to $T_{max} - \sum_{o' \in O(\{\mathtt{r}(o)\}) : \mathtt{S}(o') \geqslant \mathtt{S}(o)} (\mathtt{t}^+(o') + \mathtt{d}(o'))$. The MILP formulation of $\mathrm{CMP}_3(\bar{x})$ is as follows:

$$[\mathrm{CMP}_3(\bar{x})] \quad \text{minimize} \sum_{o \in O(\Omega_{1^+}(\bar{x}))} \nabla_o \tag{61}$$

$$\text{subject to} \quad (47) - (49), (51) - (58)$$

$$S_{\mathtt{suc}(o)} = S_o + \mathtt{d}(o) + \mathtt{t}^+(o) + \nabla_{\mathtt{suc}(o)} \quad o \in O(\Omega_{1^+}(\bar{x})) : \mathtt{suc}(o) \neq -1 \tag{62}$$

$$\nabla_o \geqslant 0, \quad o \in O(\Omega_{1^+}(\bar{x})). \tag{63}$$

The objective (61) is to minimize the waiting time inserted in each route. Constraints (62) define the minimum time lag between the charging operations occurring in the same route. Constraints (63) define the domains of the waiting decision variables.

Version DWR In this version of the subproblem, in addition to the introduction of waiting times, resolving capacity violations at CSs can also be achieved by revising the amounts of energy charged at each CS in every route. We denote by $\text{CMP}_4(\bar{x})$ the subproblem in which we want to minimize the increase in the duration of the selected routes. Indeed, revising the charging operations leads to an increase in time when the CSs in a route have different charging technologies or when the charging functions are non-linear. Note that if we substantially increase the charging amounts at a CS we may not need to stop at the subsequent CS in the route. We therefore need to account for the potential removal of stops at CSs.

We denote by $\mathtt{e}^+(o)$ the energy consumption of the EV between its departure from $\mathtt{j}(o)$ and its arrival at $\mathtt{j}(\mathtt{suc}(o))$ if $\mathtt{suc}(o) \neq -1$. This takes into account the energy consumed to visit all the customers scheduled in the route between charging operations o and $\mathtt{suc}(o)$ or the depot. Since a charging operation can be skipped by charging more energy during the previous or next charging operations of the same route, we define $\widetilde{\mathtt{t}}(o)$ and $\widetilde{\mathtt{e}}(o)$ as the driving time and energy saved if the EV does not detour to perform the charging operation o. We define $\Omega_{2^+}(\bar{x})$ as the subset of $\Omega_{1^+}(\bar{x})$ that contains only the routes including at least two charging operations (i.e., $\Omega_{2^+}(\bar{x}) = \{r \in \Omega_{1^+}(\bar{x}) : |O(\{r\})| \geqslant 2\}$).

Our formulation of $\text{CMP}_4(\bar{x})$ draws upon formulation $[\text{CMP}_3(\bar{x})]$. Aside from using decision variables defined in the latter, $[\text{CMP}_4(\bar{x})]$ also uses the following decision variables for the operations in $O(\Omega_{2^+}(\bar{x}))$. The variables $\underline{t}_o$ and $\overline{t}_o$ are the scaled arrival and departure times, according to the charging function of CS $\mathtt{j}(o)$. The binary variables $\underline{w}_{ok}$ and $\overline{w}_{ok}$ are equal to 1 if and only if the SoC lies between $q_{\mathtt{j}(o),k-1}$ and $q_{\mathtt{j}(o),k}$, with $k \in B_{\mathtt{j}(o)} \backslash \{0\}$, upon starting and finishing operation o, respectively. The continuous variables $\underline{\lambda}_{ok}$ and $\overline{\lambda}_{ok}$ are the coefficients associated with the breakpoints $(a_{\mathtt{j}(o),k}, q_{\mathtt{j}(o),k})$ of $\phi_{\mathtt{j}(o)}$ upon starting and finishing operation o, respectively. The continuous variables $\underline{y}_o$ and $\overline{y}_o$ represent the SoC of the EV upon starting and finishing charging operation o. The continuous variable Δ_o represents the duration of charging operation o.

For each route, we check whether it might be possible for a charging operation o to be skipped by considering that the EV leaves the previous CS (or depot) with a fully replenished battery. If this allows the EV to reach the next CS or to return to the depot without performing o, then we allow the EV not to detour to the corresponding CS. To this end, we introduce the binary variable z_o equal to 1 if and only if the charging operation o is executed.

We also compute for every charging operation the time windows during which it must be scheduled. The earliest starting time $\mathtt{ES}(o)$ of a charging operation o is computed assuming that the EV skips (if the previous computation has shown it is possible) the previous charging operations (if any), and charges the maximum between the energy needed to recover the detour to the CS and the energy required to reach the next CS. To compute the latter, we consider that the SoC of the EV upon arriving at the CS is maximal (i.e., a full charge occurs at the previous CS). Then, we estimate the charging times assuming that the EV arrives with an empty battery. The latest starting time $\mathtt{LS}(o)$ of operation o is computed assuming that the EV returns to the depot at time T_{max} and assuming that the EV skips the next charging operations (if possible).

The MILP formulation of $\text{CMP}_4(\bar{x})$ is as follows:

$$[\text{CMP}_4(\bar{x})] \quad \text{minimize} \sum_{o \in O(\Omega_{1^+}(\bar{x}))} \nabla_o + \sum_{o \in O(\Omega_{2^+}(\bar{x}))} \big(\Delta_o - \mathtt{d}(o) - (1 - z_o)\widetilde{\mathtt{t}}(o)\big) \tag{64}$$

$$\text{subject to} \quad (47) - (48), (52) - (58), (63)$$

$$\underline{y}_o = \sum_{k \in B_{\mathtt{j}(o)}} \underline{\lambda}_{ok} q_{\mathtt{j}(o)k} \quad o \in O(\Omega_{2^+}(\bar{x})) \tag{65}$$

$$\underline{t}_o = \sum_{k \in B_{\mathtt{j}(o)}} \underline{\lambda}_{ok} a_{\mathtt{j}(o)k} \quad o \in O(\Omega_{2^+}(\bar{x})) \tag{66}$$

$$\sum_{k \in B_{\mathtt{j}(o)}} \underline{\lambda}_{ok} = \sum_{k \in B_{\mathtt{j}(o)} \backslash \{0\}} \underline{w}_{ok} \quad o \in O(\Omega_{2^+}(\bar{x})) \tag{67}$$

$$\sum_{k \in B_{\mathtt{j}(o)} \backslash \{0\}} \underline{w}_{ok} = z_o \quad o \in O(\Omega_{2^+}(\bar{x})) \tag{68}$$

$$\underline{\lambda}_{o0} \leqslant \underline{w}_{o1} \quad o \in O(\Omega_{2^+}(\bar{x})) \tag{69}$$

$$\underline{\lambda}_{ok} \leqslant \underline{w}_{ok} + \underline{w}_{o,k+1} \quad o \in O(\Omega_{2^+}(\bar{x})), k \in B_{\mathtt{j}(o)} \backslash \{0, b_{\mathtt{j}(o)}\} \tag{70}$$

$$\underline{\lambda}_{ob_{\mathtt{j}(o)}} \leqslant \underline{w}_{ob_{\mathtt{j}(o)}} \quad o \in O(\Omega_{2^+}(\bar{x})) \tag{71}$$

$$\overline{y}_o = \sum_{k \in B_{\mathtt{j}(o)}} \overline{\lambda}_o q_{\mathtt{j}(o)k} \quad o \in O(\Omega_{2^+}(\bar{x})) \tag{72}$$

$$\overline{t}_o = \sum_{k \in B_{\mathtt{j}(o)}} \overline{\lambda}_o a_{\mathtt{j}(o)k} \quad o \in O(\Omega_{2^+}(\bar{x})) \tag{73}$$

$$\sum_{k \in B_{\mathtt{j}(o)}} \lambda_{ok} = \sum_{k \in B_{\mathtt{j}(o)} \backslash \{0\}} \overline{w}_{ok} \quad o \in O(\Omega_{2^+}(\bar{x})) \tag{74}$$

$$\sum_{k \in B_{\mathtt{j}(o)} \backslash \{0\}} \overline{w}_{ok} = 1 \quad o \in O(\Omega_{2^+}(\bar{x})) \tag{75}$$

$$\overline{\lambda}_{o0} \leqslant \overline{w}_{o1} \quad o \in O(\Omega_{2^+}(\bar{x})) \tag{76}$$

$$\overline{\lambda}_{ok} \leqslant \overline{w}_{ok} + \overline{w}_{o,k+1} \quad o \in O(\Omega_{2^+}(\bar{x})), k \in B_{\mathtt{j}(o)} \backslash \{0, b_{\mathtt{j}(o)}\} \tag{77}$$

$$\overline{\lambda}_{ob_{\mathtt{j}(o)}} \leqslant \overline{w}_{ob_{\mathtt{j}(o)}} \quad o \in O(\Omega_{2^+}(\bar{x})) \tag{78}$$

$$\Delta_o = \overline{t}_o - \underline{t}_o \quad o \in O(\Omega_{2^+}(\bar{x})) \tag{79}$$

$$\Delta_o \leqslant a_{\mathtt{j}(o),b_{\mathtt{j}(o)}} z_o \quad o \in O(\Omega_{2^+}(\bar{x})) \tag{80}$$

$$\underline{y}_{\mathtt{o_{first}}(r)} = \mathtt{q_{first}}(r) \quad r \in \Omega_{2^+}(\bar{x}) \tag{81}$$

$$\underline{y}_{\mathtt{suc}(o)} = \overline{y}_o - \mathtt{e}^+(o) + \tilde{\mathtt{e}}(o)(1 - z_o) \quad o \in O(\Omega_{2^+}(\bar{x})) : \mathtt{suc}(o) \neq -1 \tag{82}$$

$$\overline{y}_o - \mathtt{e}^+(o) + \tilde{\mathtt{e}}(o)(1 - z_o) \geqslant 0 \quad o \in O(\Omega_{2^+}(\bar{x})) : \mathtt{suc}(o) = -1 \tag{83}$$

$$S_{o'} \geqslant S_o + \mathtt{d}(o) u_{oo'} + (\mathtt{LS}(o) - \mathtt{ES}(o'))(u_{oo'} - 1) \quad j \in F, o, o' \in O_j(\Omega_{1^+}(\bar{x}) \backslash \Omega_{2^+}(\bar{x})) \tag{84}$$

$$S_{o'} \geqslant S_o + \Delta_o + (\mathtt{LS}(o) - \mathtt{ES}(o'))(u_{oo'} - 1) \quad j \in F, o, o' \in O_j(\Omega_{2^+}(\bar{x})) \tag{85}$$

$$S_{\mathtt{suc}(o)} = S_o + \mathtt{d}(o) + \mathtt{t}^+(o) + \nabla_{\mathtt{suc}(o)} \quad o \in O(\Omega_{1^+}(\bar{x}) \backslash \Omega_{2^+}(\bar{x})) : \mathtt{suc}(o) \neq -1 \tag{86}$$

$$S_{\mathtt{suc}(o)} = S_o + \Delta_o + \mathtt{t}^+(o) - \tilde{\mathtt{t}}(o)(1 - z_o) + \nabla_{\mathtt{suc}(o)} \quad o \in O(\Omega_{2^+}(\bar{x})) : \mathtt{suc}(o) \neq -1 \tag{87}$$

$$S_o + \Delta_o + \mathtt{t}^+(o) - \tilde{\mathtt{t}}(o)(1 - z_o) \leqslant T_{max} \quad o \in O(\Omega_{2^+}(\bar{x})) : \mathtt{suc}(o) = -1 \tag{88}$$

$$\sum_{o' \in O_{\mathtt{j}(o)}(\Omega_{1^+}(\bar{x})) \cup \{\mathtt{o}^+_{\mathtt{j}(o)}\}} f_{o'o} = 1 \quad o \in O(\Omega_{1^+}(\bar{x}) \backslash \Omega_{2^+}(\bar{x})) \tag{89}$$

$$\sum_{o' \in O_{\mathtt{j}(o)}(\Omega_{1^+}(\bar{x})) \cup \{\mathtt{o}^+_{\mathtt{j}(o)}\}} f_{o'o} = z_o \quad o \in O(\Omega_{2^+}(\bar{x})) \tag{90}$$

$$z_o \in \{0,1\}, \Delta_o \geqslant 0, 0 \leqslant \underline{y}_o \leqslant Q, 0 \leqslant \overline{y}_o \leqslant Q \quad o \in O(\Omega_{2^+}(\bar{x})) \tag{91}$$

$$\underline{w}_{ok}, \overline{w}_{ok} \in \{0,1\}, \quad o \in O(\Omega_{2^+}(\bar{x})), k \in B_{\mathtt{j}(o)} \backslash \{0\} \tag{92}$$

$$\underline{\lambda}_{ok}, \overline{\lambda}_{ok} \in \{0,1\} \quad o \in O(\Omega_{2^+}(\bar{x})), k \in B_{\mathtt{j}(o)}. \tag{93}$$

The objective (64) is to minimize the total additional time inserted in each route. Constraints (65)–(79) model the piecewise linear charging functions. Constraints (80) impose a duration equal to 0 for each charging operation that is not executed anymore. For each route $r \in \Omega_{2^+}(\bar{x})$, constraints (81) define the SoC $\mathtt{q_{first}}(r)$ of the EV upon starting its first charging operation (denoted $\mathtt{o_{first}}(r)$). Constraints (82) couple the SoC of the EV after finishing a charging operation with its SoC when starting the next charging operation occurring in the route. Note that if z_o is equal to 0, then the SoC $\underline{y}_o = \overline{y}_o$ still takes into account the energy consumed to detour to CS $\mathtt{j}(o)$. The energy saved by not stopping at this CS is subtracted when computing the SoC at the beginning of the next operation of the route or at the arrival at the depot (see (83)). For each route, constraints (83) force the corresponding EV to have enough SoC at the end of the last charging operation to reach the depot. Constraints (84) and (85) are the disjunctive constraints on the operations related to the same CS. Constraints (86) and (87) define a minimum time lag between the charging operations occurring in the same route. Note that if z_o is equal to 0, then the starting time S_o still takes into account the detour to CS $\mathtt{j}(o)$. The time saved by not stopping at this CS is subtracted during the computation of the departure time for the next operation of the route. Constraints (88) limit the route duration. Constraints (89) and (90) assign a charger to each operation that is executed. Constraints (91)–(93) define the domains of the new decision variables.

4.2.3 Cut generation procedure

To couple the route selection problem with the CS capacity management subproblem, we generate constraints to cut off infeasible selections of routes in $\Omega_{1^+}(\bar{x})$ and to bound the variable θ (for versions DW and DWR). The efficiency of our branch-and-cut method is based on the strength of these cuts. Rather than only cutting the current solution, we investigate a strategy that generates cuts for a large portion of the solution space of the route selection problem. Moreover, we try to add several cuts at a time to speed up the convergence of the algorithm.

We first focus on the cuts that are applicable only to version N. Given a CS j, let $\mathcal{O}_j$ be a set containing subsets of operations with a cardinality strictly larger than C_j, which overlap in time. To discard the current solution $\bar{x}$ in the route selection problem, we add to $[HC1]$ the following cuts:

$$\sum_{r \in \Omega_{1^+}(\bar{x}): O(\{r\}) \cap U \neq \emptyset} x_r \leqslant C_j \quad j \in F, U \in \mathcal{O}_j. \tag{94}$$

For the other versions of the subproblem, Algorithm 4 summarizes the procedure used to solve the capacity management subproblem and to generate cuts to discard infeasible selection of routes. We first try to detect, before solving $\text{CMP}_\cdot(\bar{x})$, the potential infeasibility of the subproblem using two fast procedures focusing independently on each CS. If the subproblem is deemed infeasible, then we generate cuts specially crafted for our problem. If not, we solve the MILP formulation $[\text{CMP}_\cdot(\bar{x})]$ with a commercial solver. If possible, we decompose the subproblem into smaller independent problems beforehand. We present below the details of the cut generation procedure.

The two algorithms we apply to detect capacity violations that cannot be solved by the CS capacity management subproblem are frequently used in scheduling problems. They focus on a single CS and are based on a reasoning rooted in the time windows (derived from the opening hours of the depot) in which every charging operation can be scheduled. The first algorithm focuses on the fixed part of the charging operations (line 3). We call a fixed part of a charging operation o the time interval (if it exists) between the latest starting time $\texttt{LS}(o)$ and earliest completion time $\texttt{ES}(o)+\texttt{d}(o)$. Based on the fixed part of the charging operations, we detect subsets of operations that will necessarily lead to a violation of the capacity constraints and we add cuts to forbid the underlying subset of routes (line 6). Hereafter, we refer to this algorithm as the *fixed part based algorithm*. The details of this procedure are provided in Algorithm 8 (Appendix D). The second algorithm is based on an energy reasoning (line 8). The required energy consumption of a charging operation o during a time interval $[t_1, t_2)$ is equal to the minimum duration (possibly equal to 0) for which o is surely executed within the interval. For a CS j, the total required energy consumption by the charging operation o scheduled at j over time interval $[t_1, t_2)$ cannot exceed $C_j(t_2 - t_1)$. The difference between the last term and the total required energy consumption is called the slack over $[t_1, t_2)$. The slack over any time interval must always be non-negative. The number of intervals that needs consideration is bounded (see Baptiste et al. (2001) for more details). When there exists an interval for which a slack is strictly negative, the subset of operations having a non-zero energy consumption on this interval cannot be performed without leading to a violation of the capacity constraints. We add a no-good cut to discard it (line 10). Hereafter, we refer to this algorithm as the *energy reasoning based algorithm*. The details of this procedure are provided in Algorithm 9 (Appendix D). To limit the computation time, if the two previous algorithms prove the infeasibility of the subproblem, we add to the route selection problem the generated cuts and we do not solve the subproblem.

The decomposition of the subproblem $\text{CMP}_2(\bar{x})$, $\text{CMP}_3(\bar{x})$, or $\text{CMP}_4(\bar{x})$ into several independents smaller subproblems is based on reasoning about the stops at the CSs in the different routes. The interest is to potentially formulate cuts for each of these small subproblems. Let $G(\bar{x})$ be a graph in which each vertex represents a CS and there exists an edge between two CSs if there exists a route in $\Omega_{1^+}(\bar{x})$ with charging operations at these two CSs. We can decompose the subproblem into as many independent subproblems as the number of connected components of $G(\bar{x})$. Let $\mathcal{C}(G(\bar{x}))$ be the connected components of $G(\bar{x})$. For every connected component $C \in \mathcal{C}(G(\bar{x}))$, only the charging operations scheduled at the CSs in C need consideration. We denote by $\text{CMP}(\bar{x}, C)$ the subproblem restricted to these operations. When $\text{CMP}(\bar{x}, C)$ is infeasible, we generate an integer Benders cut, also called combinatorial Benders cut (Codato and Fischetti, 2006), to invalidate the current solution to the route selection problem (line 18). For versions DW and DWR of the subproblem, we also compare the current value $\bar{\theta}$ of θ with the increase in time computed over the set $\mathcal{C}^{\text{feas}}$ of connected components for which the corresponding subproblem is feasible. For each subset $\mathcal{C}$ of $\mathcal{C}^{\text{feas}}$, we generate an

integer Benders optimality cut if the route selection problem underestimates the increase in time needed to resolve the capacity violations at CSs (line 25).

Algorithm 4: Solving the CS capacity management subproblem and generating cuts for version D, DW, or DWR

Input: a solution $\bar{x}$ to the current master problem

1 **Procedure** `SolveSubProblem(`$\bar{x}$`):`
2 Compute `ES`(o) and `LS`(o) for each charging operation $o \in O(\Omega_{1+}(\bar{x}))$
3 **for** $j \in F$ **do**
4 $\mathcal{O} \leftarrow$`FixedPartsAlgorithm`$(O_j(\bar{x}), C_j)$ (see Algorithm 8)
5 **if** $\mathcal{O} \neq \varnothing$ **then**
6 **for** $U \in \mathcal{O}$ **do** Generate the following cut and add it to $[HC.]$: $\sum_{r \in \Omega_{1+}(\bar{x},j): O(\{r\}) \cap U \neq \varnothing} x_r \leqslant C_j$
7 **else**
8 $\mathcal{O} \leftarrow$`EnergeticAlgorithm`$(O_j(\bar{x}), C_j)$ (see Algorithm 9)
9 **if** $\mathcal{O} \neq \varnothing$ **then**
10 **for** $U \in \mathcal{O}$ **do** Generate the following cut and add it to $[HC.]$: $1 + \sum_{r \in \Omega_{1+}(\bar{x},j): O(\{r\}) \cap U \neq \varnothing} (x_r - 1) \leqslant 0$
11 **end**
12 **end**
13 **end**
14 **if** no cuts have been generated and $\bar{x}$ is integer **then**
15 $\mathcal{C}^{\text{feas}} \leftarrow \varnothing$
16 **for each** $C \in \mathcal{C}(G(\bar{x}))$ **do**
`/* Let CMP`$(\bar{x}, C)$ `be the subproblem restricted to the CSs in connected component` C `and` z`(CMP`$(\bar{x}, C)$`) denotes the value of the objective function (when a solution is found) */`
17 Solve the MILP formulation of $\text{CMP}(\bar{x}, C)$ that corresponds to the selected version of the subproblem
18 **if** no solution is found **then**
19 Generate the following cut and add it to $[HC.]$: $1 + \sum_{r \in \left(\bigcup_{j \in C} \Omega_{1+}(\bar{x},j) \right)} (x_r - 1) \leqslant 0$
20 **else**
21 $\mathcal{C}^{\text{feas}} \leftarrow \mathcal{C}^{\text{feas}} \cup \{C\}$ `/* Only for versions DW and DWR`
22 **end**
23 **end**
`/* Only for versions DW and DWR`
24 **for** every subset $\mathcal{C}$ of connected components of $\mathcal{C}^{\text{feas}}$ **do**
25 **if** $\sum_{C \in \mathcal{C}} z\left(\text{CMP}(\bar{x}, C)\right) > \bar{\theta}$ **then** Generate the following cut and add it to $[HC.]$: $\left(\sum_{C \in \mathcal{C}} z\left(\text{CMP}(\bar{x}, C)\right) \right) \left(1 + \sum_{C \in \mathcal{C}} \sum_{r \in \left(\bigcup_{j \in C} \Omega_{1+}(\bar{x},j) \right)} (x_r - 1) \right) \leqslant \theta$
26 **end**
27 **end**
28 **if** no cuts have been generated **then**
29 Save the solution result of calling the subproblem on the routes of $\Omega_{1+}(\bar{x})$
30 **end**

5 Computational results

We considered the 120-instance testbed of Montoya et al. (2017) (available from `http://www.vrp-rep.org/`). In this testbed, there are six sets of 20 instances, each with 10, 20, 40, 80, 160, or 320 customers. The EVs are Peugeot iOns which have a consumption rate of 0.125 kWh/km and a battery of 16 kWh. We note that in these instances the triangular inequalities hold for energy consumption. However, this assumption is not necessary for our model and solution method. When dealing with the E-VRP-NL-C, we adapted each instance by fixing a number of chargers for each CS. We decided to consider instances in which all the CSs have one or two chargers.

The first aim of our computational experiments is to assess the performance of a commercial MIP solver for solving small-size instances of the E-VRP-NL-C using the path-based formulation introduced in Section 3 (although the primary motivation for introducing the latter formulation was to give a precise definition of the problem). These results are presented in §5.1. The second aim of our computational experiments is to assess the quality of the ILS presented in §4.1 as a solution method for the E-VRP-NL. A comparison with results from the literature is presented in §5.2. The third aim of our experiments is to assess the effectiveness of our algorithmic framework to build high-quality solutions to the E-VRP-NL-C. We compare the results given by our matheuristic according to the version of the subproblem selected during the assembly phase. These results are presented in §5.3.

We implemented all the algorithms using Java 8. In all the tables, the CPU time is given in seconds and rounded to the nearest integer.

5.1 Results for the E-VRP-NL-C formulation

We considered the twenty 10-customer and the twenty 20-customer instances. We ran the path-based model with a three-hour CPU time limit. We used Gurobi 8.1.1 through its Java API. The results were obtained using a standard PC with an Intel(R) Core(TM) i7-9700K processor clocked at 3.6 GHz, equipped with 128 GB RAM, and running Debian GNU/Linux 10. Each instance was executed on a single thread.

Table 2 reports the number of instances with a solution proven to be optimal (#Opt), the average CPU time in seconds (Time) for these latter instances, and the average gap (Gap) for the remaining instances. We compute the gap as $(z - z^{LB})/z$, where z is the objective function value of the best integer solution returned by the solver, and z^{LB} is the best lower bound retrieved by the solver. The detailed results for all the tested instances are reported in Appendix E.1.

Our results show that the path-based model cannot solve all the 10-customer instances to optimality within the CPU time limit whereas the results in Froger et al. (2019) showed that they can be solved within an average computation time of roughly four minutes when CS capacity constraints are ignored. We also observe that the 20-customer instances are already very challenging for the model since zero instances were solved. As expected, since the linear relaxation of the model is weak, we conclude that solving the MILP formulations with a commercial solver does not constitute an efficient solution method for the E-VRP-NL-C. Still, 17 out of the 40 tested instances were optimally solved when considering one or two chargers per CS (the number of chargers appears to have no impact).

Table 2: Computational results for the CS path-based formulation on the 10-customer and 20-customer instances.

$\|I\|$	Capacity = 1			Capacity = 2		
	#Opt	Time	Gap	#Opt	Time	Gap
10	17	873	22%	17	776	19%
20	0	-	32%*	0	-	29%*

*: There are 16 out of 20 instances for which the solver reaches the CPU time limit without finding any solution. They are not considered when computing the gap.

5.2 Results for the E-VRP-NL

Since the route generator of our matheuristic essentially solves the E-VRP-NL, we wanted to assess its quality on this problem. We considered the 120 instances of the original Montoya et al. (2017) testbed. Each instance was executed on a single thread with 12 GB RAM and on a cluster of 27 computers, each having 12 cores and two Intel(R) Xeon® X5675 3.07 GHz processors. We performed 10 test runs for each instance using Algorithm 2 and compared the results with those obtained in the E-VRP-NL literature through two solutions methods: the metaheuristic of Montoya et al. (2017) which combines an ILS with a heuristic concentration (HC), and the large neighborhood search (LNS) of Koç et al. (2018). Table 3 shows the results of this comparison. Given a number of customers in the instances, it reports for each solution method the number of BKS to the E-VRP-NL (#BKS), the average gap to the BKS (Gap), and the average number of routes in the best computed solution. We compute the gap as $(z - z^*)/z$, where z is the objective function

value of the best solution returned by the solution method, and z^* is the objective function value of the BKS. Table 3 also reports the average gap to the best average objective function value (Gap BA) as $(z_{\text{avg}} - z^*_{\text{avg}})/z_{\text{avg}}$, where z_{avg} is the average objective function value of the solutions obtained in 10 runs by the solution method, and z^*_{avg} is the best average (BA) objective function value obtained in 10 runs by one of the existing solution method for the E-VRP-NL. The detailed results for all the tested instances are reported in Appendix E.2.

Table 3: Comparison of the results obtained by ILS with the results of Montoya et al. (2017) and Koç et al. (2018) for the E-VRP-NL.

	ILS + HC 2.33 GHz processor - 16GB of RAM					LNS 3.6 GHz processor - 32 GB of RAM					ILS 3.07 GHz processor - 12 GB of RAM				
$\|I\|$	#BKS	Gap BKS	#R	Gap BA	Time	#BKS	Gap BKS	#R	Gap BA	Time	#BKS	Gap BKS	#R	Gap BA	Time
10	20	0.0%	2.7	0.3%	6	20	0.0%	2.7	0.1%	8	20	0.0%	2.7	0.0%	5
20	11	0.3%	3.7	0.7%	11	12	0.2%	3.6	0.4%	14	20	0.0%	3.6	0.0%	8
40	3	1.0%	6.5	2.6%	35	6	0.9%	6.4	1.1%	45	20	0.0%	6.2	0.0%	20
80	0	3.8%	9.2	5.4%	80	0	3.8%	8.8	3.8%	99	20	0.0%	8.6	0.0%	64
160	0	7.3%	16.7	8.1%	568	0	7.7%	16.6	7.8%	632	20	0.0%	15.3	0.0%	295
320	0	11.2%	32.6	12.6%	4398	0	11.7%	32.6	12.4%	4555	20	0.0%	29.0	0.0%	1118
All	34	3.9%	11.9	4.9%	850	38	4.1%	11.8	4.3%	892	120	0.0%	10.9	0.0%	252

ILS + HC: (Montoya et al., 2017), LNS: (Koç et al., 2018).

The results presented in Table 3 clearly show that our ILS outperforms all existing methods for the E-VRP-NL. We have identified 80 new BKS and matched the existing BKS for the remaining instances. We have improved the previous solutions by about 4.0%. The improvement increases with the number of customers. The algorithm is also stable as the average results on 10 test runs for each instance are better than those reported for previous methods. While it is difficult to draw definitive conclusions on the computation times since these tests were run on different machines, it seems that the ILS is at least as fast as the other methods, if not the fastest one. The major difference between our method and the existing ones comes from the reoptimization of the charging decisions when evaluating a move and the use of larger neighborhoods.

5.3 Results for the E-VRP-NL-C

We have performed several tests to assess the effectiveness of our matheuristic in obtaining high-quality solutions for the E-VRP-NL-C. Each instance was executed on a single thread with 12 GB RAM and on a cluster of 27 computers, each having 12 cores and two Intel(R) Xeon® X5675 3.07 GHz processors. We used Gurobi 7.5.0 (through its Java API) to solve the MILP models.

After some preliminary experiments, we set the values of the parameters of the matheuristic as presented in Table 4. Setting the stopping criterion to 12 iterations and the number of ILS iterations to 200 proved to be a good compromise between solution quality and computation time. In some rare cases we were not able to optimally solve the assembly phase using version DWR of the subproblem. Indeed, when it is difficult to find a solution satisfying the CS capacity constraints or when such a solution does not exist, the MILP formulation becomes computationally expensive. Nonetheless, after testing higher CPU time limits for the second component, we observed that the impact on the results was negligible.

Table 4: Value of the matheuristic parameters.

n_{max}	α	T_{min}	δ^{max}	τ	τ^{SP}
12	0.9	$0.67 \cdot T_{max}$	200	180 s	5 s

We first compare the results obtained by our matheuristic for the four versions of the subproblem used by the solution assembler. For one or two chargers at each CS and each version of the subproblem, we performed 10 test runs for each instance. Table 5 reports the best results obtained during our tests. Specifically, given a number of chargers

per CS and a selected version of the subproblem used in the branch-and-cut algorithm, this table reports the number of instances with a solution in each of the 10 tests (#F), the number of instances with the best solution (BS) to the E-VRP-NL-C computed all over our tests (#BS), including those of the path-based formulation, and the average gap to the BS (Gap BS). Table 6 reports the average CPU time in seconds over 10 runs for the whole algorithm (T), and for the first (T_1) and second (T_2) components. It also reports the average gap with respect to the best average objective function value obtained using a given version of the subproblem (Gap BA). The gaps are computed as in §5.2. Detailed results for all the tested instances are reported in Appendix E.3.

Table 5: Comparison of the best results obtained by the matheuristic for the E-VRP-NL-C according to the version of the subproblem it uses and the number of chargers at each CS.

Cap.	$\|I\|$	Version N			Version D			Version DW			Version DWR		
		#F	#BS	Gap BS	#F	#BS	Gap BS	#F	#BS	Gap BS	#F	#BS	Gap BS
1	10	20	17	0.06%	20	20	0.00%	20	20	0.00%	20	20	0.00%
	20	20	15	0.09%	20	20	0.00%	20	20	0.00%	20	20	0.00%
	40	19	15	0.17%	19	18	0.01%	19	18	0.01%	19	19	0.00%
	80	20	15	0.02%	20	20	0.00%	20	20	0.00%	20	20	0.00%
	160	19	5	0.27%	20	13	0.06%	20	13	0.06%	20	15	0.09%
	320	10	5	0.32%	20	1	0.28%	20	2	0.27%	20	13	0.10%
	All	**105**	**72**	**0.14%**	**119**	**92**	**0.06%**	**119**	**93**	**0.06%**	**119**	**107**	**0.03%**
2	10	20	20	0.00%	20	20	0.00%	20	20	0.00%	20	20	0.00%
	20	20	20	0.00%	20	20	0.00%	20	20	0.00%	20	20	0.00%
	40	19	19	0.09%	20	20	0.00%	20	20	0.00%	20	20	0.00%
	80	20	18	0.01%	20	20	0.00%	20	20	0.00%	20	19	0.01%
	160	20	13	0.05%	20	10	0.18%	20	10	0.18%	20	14	0.09%
	320	19	4	0.33%	20	9	0.12%	20	9	0.12%	20	8	0.17%
	All	**118**	**94**	**0.08%**	**120**	**99**	**0.05%**	**120**	**99**	**0.05%**	**120**	**101**	**0.05%**

Table 6: Comparison of the average results obtained by the matheuristic for the E-VRP-NL-C according to the version of the subproblem it uses and the number of chargers at each CS.

Cap.	$\|I\|$	Version N				Version D				Version DW				Version DWR			
		T	T_1	T_2	Gap BA	T	T_1	T_2	Gap BA	T	T_1	T_2	Gap BA	T	T_1	T_2	Gap BA
1	10	6	5	1	0.06%	6	5	1	0.00%	6	5	1	0.00%	6	5	1	0.00%
	20	11	10	1	0.09%	11	10	1	0.00%	11	10	1	0.00%	11	10	1	0.00%
	40	24	23	1	0.31%	23	22	1	0.03%	23	22	1	0.03%	87	22	65	0.00%
	80	74	73	1	0.10%	74	73	1	0.00%	73	72	1	0.00%	74	73	1	0.04%
	160	340	335	5	0.45%	339	335	4	0.10%	340	336	4	0.10%	397	333	64	0.08%
	320	1345	1274	71	0.34%	1480	1253	227	0.17%	1487	1259	228	0.15%	1933	1244	689	0.05%
	All	**300**	**287**	**13**	**0.22%**	**322**	**283**	**39**	**0.05%**	**323**	**284**	**39**	**0.05%**	**418**	**281**	**137**	**0.03%**
2	10	5	5	0	0.00%	5	5	0	0.00%	5	5	0	0.00%	5	5	0	0.00%
	20	10	10	0	0.00%	10	10	0	0.00%	10	10	0	0.01%	10	10	0	0.00%
	40	23	23	0	0.05%	23	23	0	0.11%	23	23	1	0.09%	45	23	22	0.01%
	80	73	72	1	0.18%	73	73	1	0.19%	73	73	1	0.25%	73	73	1	0.14%
	160	336	334	2	0.24%	336	334	2	0.25%	337	335	2	0.24%	338	336	2	0.28%
	320	1340	1279	61	0.21%	1283	1276	7	0.12%	1276	1269	7	0.14%	1327	1269	58	0.08%
	All	**298**	**287**	**11**	**0.11%**	**288**	**287**	**2**	**0.11%**	**287**	**286**	**2**	**0.12%**	**300**	**286**	**14**	**0.09%**

First, it should be noted that our matheuristic returns an optimal solution for the 17 small-size instances for which the MILP path formulation yields an optimal solution employing on average only around 2% of the computation time the solver spent to reach the optimal solutions and less than 1% of the computation time the solver spent to prove optimality. Considering that a single charger exists in each CS, we observe that using version N of the subproblem prevents the algorithm from finding solutions to the E-VRP-NL-C for 15 instances. In this case, the algorithm finds the

best solution (computed all over our tests) for only 72 out of 120 instances. Delaying the starting time of the routes yields better solutions for 20 more instances. In contrast, allowing waiting times in version DW leads to very marginal improvements, compared with version D. Indeed, making the EVs wait for an available charger can eliminate capacity violations only if at least two CSs are visited in a route; otherwise it is sufficient to make the EVs leave the depot after time 0, as is done in version D. Although the same requirement (more than one CS) holds when allowing the revision of the amount of energy charged at the CSs in version DWR, the latter strategy yields almost all the best solutions to the E-VRP-NL-C.

When increasing the number of chargers to two per CS, the capacity constraints become less binding and all our assembly strategies yield very similar results. This was somewhat expected but it should be noted that delaying the starting time of the routes increases the probability of ending up with a feasible solution.

The first general conclusion is that allowing delays when solving the CS capacity management subproblem is a suitable and efficient way of to assembling high-quality solutions to the E-VRP-NL-C. Allowing the revision of the amounts of energy charged at the CSs on top of it significantly improves the results, but it is slightly more computationally expensive (Table 6). We also note that in general, most of the computation time is spent generating routes as assembling a solution is usually very fast, especially when a cutoff value is provided to the branch-and-cut algorithm. Not surprisingly the computation time increases with the number of customers but it remains reasonable since it takes around 30 minutes to tackle the largest instances.

To assess the relevance of our cuts, we show in Table 7 the distribution of the different type of cuts generated in the branch-and-cut tree. When the CS capacity constraints are very binding, the energetic reasoning based algorithm detects infeasible route selections much more frequently than the fixed part based algorithm. We observe that due to its high degree of flexibility, compared with the other versions, DWR requires solving the MILP formulation more often. Indeed, the two previously mentioned algorithms are weaker in that case since the mandatory part of each charging operation may have a very short duration due to the potential revision of the amount of energy charged at the CSs. When we increase the number of chargers at each CS, the reasoning algorithms lead to the generation of fewer cuts. This is not surprising since these are based on relaxations of the subproblem. However, they are useful since the generated cuts are stronger, which avoids computationally expensive calls to the MILP solver. We also see that integer optimality cuts are seldom generated. Since increases in the total time of the routes are penalized, the algorithm tends to avoid adding waiting times. This indicates that revising the charging operations does not always lead to an increase in time, and a trade-off between delay and a revision at no cost is often found. To support this conclusion, we have analyzed the solutions returned by the matheuristic. Given the number of chargers at each CS, Table 8 reports the percentage of solutions for which there exists at least one route with a delayed starting time, a waiting time before a charging operation, and a revision of the amount of energy charged in the EV. We see that for most of the solutions returned by the algorithm, satisfying the CS capacity constraints can be achieved without increasing the cost.

Table 7: Characteristics of the cuts generating in the solution assembly phase.

Cap.	Version D			Version DW				Version DWR			
	FP	NRG	BF	FP	NRG	BF	BO	FP	NRG	BF	BO
1	2.3%	94.5%	3.2%	2.3%	95.0%	2.5%	0.2%	0.6%	16.9%	80.5%	2.0%
2	29.0%	28.2%	42.8%	28.9%	27.1%	42.5%	1.5%	1.3%	0.7%	96.2%	1.8%

FP (feasibility cuts generated after calling the fixed part based algorithm), NRG (feasibility cuts generated after calling the energetic based reasoning algorithm), BF (feasibility cuts of type generated after solving the MILP formulation), BO (optimality cuts of type generated after solving the MILP formulation)

Since we have imposed a CPU time limit for the branch-and-cut execution and since the maximum route duration limit may decrease according to the state of the solution method, no version of the subproblem dominates the other versions. To perform a fair comparison between the different strategies, and to quantify the benefit of allowing the addition of waiting times and the revision of the amount of energy charged at the CSs, we took the 2,400 long-term pools of routes obtained at the end of the matheuristic for version DWR of the subproblem. For every pool of routes, we ran the branch-and-cut algorithm to generate the best E-VRP-NL-C solution. Table 9 reports for each version of the subproblem the number of pool of routes for which the algorithm obtains a feasible solution (#F), as well as the

Table 8: Analysis of the solutions retrieved by the matheuristic.

Cap.	Version D	Version DW		Version DWR				
	= D	= D	> W	= D	= NRG	> W	> NRG	> R
1	42.1%	42.2%	1.7%	42.0%	17.4%	1.3%	6.3%	2.5%
2	11.7%	11.7%	0.0%	11.4%	8.5%	0.0%	0.4%	0.0%

D (delay), W (waiting time), NRG (revision of the amount of energy charged), R (removal), = (no increase of the total time), > (increase of the total time)

best solution (#Best) among those obtained with the other versions on the same pool. It also shows the average gap between the solution returned and the best solution obtained during these tests under all the assembly strategies. The conclusions are very similar to those drawn from the results of the matheuristic. Compared to version N, version DWR makes nearly 400 extra routes feasible when assuming one charger. Not surprisingly, when increasing the number of chargers, the results tend to be more independent of the selected version of the subproblem.

Table 9: Comparison of the results of the branch-and-cut algorithm according to the different versions of the subproblem.

Cap.	Version N			Version D			Version DW			Version DWR		
	#F	#Best	Gap	#F	#Best	Gap	#F	#Best	Gap	#F	#Best	Gap
1	1916	1361	0.315%	2262	2095	0.096%	2270	2112	0.091%	2307	2307	0.000%
2	2321	2125	0.073%	2390	2371	0.013%	2390	2371	0.013%	2390	2390	0.000%

Lastly, we compare the BKSs for the E-VRP-NL and those for the E-VRP-NL-C obtained for each instance in our computational experiments. Table 10 reports aggregated results. The second column shows the number of BKSs to the E-VRP-NL that are not feasible for the EVRP-NL-C. As expected, many BKSs to the E-VRP-NL are also BKSs to the E-VRP-NL-C. Note that there is no obvious procedure to identify beforehand whether or not the CS capacity constraints are binding for a given instance. The other columns focus on the case when the BKSs to the E-VRP-NL-C and the E-VRP-NL are not equal. The third and fourth columns of the table show the number of BKSs that do not have the same number of routes and the average gap in terms of the objective function between the BKSs to the E-VRP-NL-C and the E-VRP-NL. We observe that i) the introduction of capacity constraints does not increase the number of routes in the vast majority of the cases and ii) the increase in the objective function is limited. The fifth and sixth columns of the table show the percentage of routes that make up the BKSs to both problems with and without capacity (differentiating the case when the starting time is only delayed). While the CS capacity is not problematic in many instances, when it is, the solutions differ widely.

Table 10: Comparison between the best known solutions to the E-VRP-NL and the E-VRP-NL-C.

Cap.	#BKSs ≠	#BKSs with ≠ number of routes	Objective function deviation	% of routes =	delayed	≠
1	53/119*	4/53	0.29%	36%	10%	54%
2	12/120	1/12	0.16%	38%	7%	55%

* No solution is known for a particular instance.

When a violation of the CS capacity constraints is observed for a BKS to the E-VRP-NL, we tested the use of version DWR of the capacity management subproblem to resolve it. We observed that 31 out of the 53 solutions and eight out of the 12 solutions can be repaired into a feasible solution to the E-VRP-NL-C when the capacity is equal to one and two, respectively. We conclude that a risk is incurred if CS capacity constraints are only considered a posteriori.

6 Conclusion and perspectives

We have modeled and solved an electric vehicle routing problem that embeds several features such as piecewise linear charging functions, multiple charging technologies, and multiple stops at CSs. Our methodology explicitly considers the fact that the number of EVs simultaneously charging at every CS is limited by the number of chargers. We have proposed a continuous-time path-based formulation of the E-VRP-NL-C. Our results show that optimally solving even small-size instances of the problem with a commercial MILP solver running this formulation is challenging. To handle larger instances, we have developed an algorithmic framework which iteratively calls a route generator and a solution assembler. The first component focuses on generating a pool of high-quality and diversified routes from solutions to the E-VRP-NL obtained by means of an ILS metaheuristic. Computational experiments have shown that this first component produces high quality E-VRP-NL solutions. Indeed, we have improved 80 out of 120 best-known E-VRP-NL solutions. The second component assembles a solution to the E-VRP-NL-C by selecting a subset of routes from the generated pool. We decomposed this assembly problem into a route selection problem and a CS capacity management subproblem. We have designed a branch-and-cut algorithm based on this decomposition scheme. We have developed and compared four versions of the CS capacity management subproblem, including a simple check of the capacity constraints, the introduction of waiting times, and the revision of the charging amounts. Our results show that there exists a serious risk of ending up with no solution if the method disregards the CS capacity constraints or if the routes cannot be modified by the subproblem. Delaying the starting time and revising the amount of energy charged at the visited CSs lessens this risk. Using more complex strategies to solve capacity violation issues at CSs does not significantly increase the computation time and yields better solutions. The matheuristic is also able to find all optimal solutions of small-size instances.

While our work focuses on CSs that are privately operated, it also applies if reserving chargers at CSs is possible (which is very rare in practice). In this case, there may be time windows when chargers cannot be reserved. We can deal with these periods of unavailability in our matheuristic by inserting dummy operations (fixed in time) in the capacity management subproblem, and by slightly adapting the labeling algorithm used to insert charging decisions in the ILS.

Acknowledgments— This research was partly funded by the French Agence Nationale de la Recherche through project e-VRO (ANR-15-CE22-0005-01) and by the Canadian Natural Sciences and Engineering Research Council under grants 436014-2013 and 2015-06189. This support is gratefully acknowledged.

A Experiments on the feasibility of solutions from the literature when considering capacitated CSs

We verified the feasibility of the 120 best-known solutions (BKSs) for the E-VRP-NL reported in Montoya et al. (2017), by limiting the number of chargers per CS to one, two, three, and four. Table 11 presents the results of our experiments. We observe that if there exists only one charger at each CS, then almost half of the solutions are infeasible (i.e., they violate the capacity constraints). This proportion drops to 11% when allowing two chargers per station, and four chargers at each CS are needed to ensure the feasibility of all solutions. In practice, however, there are usually only one or two chargers available at each CS.

Table 11: Results of the feasibility tests performed on the best solutions obtained in (Montoya et al., 2017) (All CSs have the same number of chargers).

Experiment	# Infeasible solutions	Average duration with capacity violation	Average #EVs during capacity violation
$\mathcal{C}_1$	55/120	32 min	2.1
$\mathcal{C}_2$	23/120	23 min	3.2
$\mathcal{C}_3$	3/120	33 min	4.0
$\mathcal{C}_4$	0/120	0 min	0.0

B Implementation details for the ILS

Our implementation of the ILS is based on the static move descriptor (SMD) concept. An SMD is static information that describes a move independently of the current solution. A cost tag is associated with every SMD to store its evaluation and it is updated throughout the LS. By storing SMDs in special data structures, one can access and update the moves in an efficient way in order to reduce computation time. Specifically, SMDs are stored in a priority queue (PQ) and organized according to their dynamic cost tag. The exploration of a neighborhood consists in looping over the PQ until a feasible SMD can be applied. Then, to avoid unnecessary reevaluations of moves, only the cost tag of the SMDs impacted by the previously applied SMD is updated. One major difference with the previous use of the SMD framework lies in the complexity of the evaluation of the moves. To our knowledge, until now there has been an emphasis on feasibility issues when using the SMD framework. Indeed, since checking the feasibility of each move can be time consuming, this is only done during the exploration phase of the neighborhood. In our case, not only the feasibility of a move can be computationally expensive to check, but also its evaluation.

Our SMD implementation is inspired by the one proposed by Beek et al. (2018) who implemented the PQ with a binary heap[1]. For every operator, we use a matrix data structure to store all the SMDs. We build these matrices during the initialization phase of the algorithm and we store them during the entire runtime of the algorithm. To avoid a computationally expensive initialization phase of the ILS, the cost tag of each SMD is set equal to the value computed after the first step of the evaluation (see §4.1.1). Since we are only interested in the moves that can potentially improve the current solution, only those SMDs with a cost tag strictly below 0 need consideration (we refer to them as improving SMDs). Throughout the algorithm, we use two binary heaps to store potentially improving SMDs: an "exact" binary heap (EBH) for the SMDs that are exactly evaluated and a "approximate" binary heap (ABH) for the remaining SMDs. In the initialization of the algorithm, we add to EBH all the improving SMDs that are exactly evaluated using the first step, and we add to ABH the other improving SMDs. When we start exploring a neighborhood, we apply the first move in EBH if it is not empty. Otherwise, we iterate over ABH. We apply the second step of the evaluation to compute the exact cost tag of each SMD in ABH until we find an improving move or we reach the end of the heap.

[1] A binary heap is a complete binary tree satisfying the heap ordering property: the value of the key stored in each node is less than or equal to the value of the keys in the node's children.

After selecting a move m, we update the values of the SMDs if they contain at least one node in the routes impacted by m. Although in some cases the cost tag of a certain number of these SMDs may remain identical, we want to keep the SMD framework as simple as possible. To avoid a computationally expensive update phase, each cost tag is set equal to the value computed during the first step evaluation. It can therefore be a lower bound on the exact evaluation of the move. In contrast with Beek et al. (2018), we keep cross-operator effects, meaning that we update the cost tag of SMDs associated with other operators. However, we perform this update for the other operators only when the search moves to them. To this end, we associate with each route the iteration number for which has been lastly updated, and to each operator the iteration number for which it has last been called (we increment the iteration number each time the search accepts a move or changes operator).

C Algorithmic details for the ILS

Algorithm 5 describes the general scheme of the VND search phase. Algorithm 6 describes the move evaluation procedure.

Algorithm 5: The VND algorithm

Input : a solution s^0 to the E-VRP-NL and a maximum route duration limit T
Output: a solution s_R to the E-VRP-NL and a solution s_O (possibly equal to `NULL`) to the E-VRP-NL-C
Procedure `VND`(s^0,T)**:**

```
/* We denote f(s) the value of the objective function for a solution s and we assume f(NULL) = +∞*/
```
$s_R \leftarrow s^0$, $s_O \leftarrow$ `NULL`, $k \leftarrow 0$
$N \leftarrow$ [(1-0,2-0 vertex exchanges), (1-1,2-1,2-2 vertex exchanges), (2-opt intra and inter-routes), separate]
while $k < |N|$ **do**
```
    /* The procedure search(N[k], s, T) searches an improving solution to s in the neighborhood N[k] respecting the
       maximum route duration limit T.  It uses Algorithm 6 for move evaluation */
```
    $s' \leftarrow$ `search`$(N[k], s_R, T)$ `/* s' = NULL if no improving solution is found */`
    if s' is a solution to the E-VRP-NL-C and $f(s') < f(s_O)$ **then** $s_O \leftarrow s'$
    if $f(s') < f(s_R)$ **then** $s_R \leftarrow s'$, $k \leftarrow 0$
    else $k \leftarrow k + 1$
end
return (s_R, s_O)

D Algorithmic details for the branch-and-cut algorithm

Algorithm 7 describes the procedure to check the CS capacity constraints for version N of the subproblem. Algorithm 8 and Algorithm 9 provide a detailed description of the fixed part and energy reasoning based algorithms.

Algorithm 6: Evaluation of a move

Input : a move m and a maximum route duration limit T

Output: a couple (c,b) where c is a lower estimation of the move evaluation and b is a boolean equal to `true` if the move is exactly evaluated or if it is a non-improving move and it is equal to `false` otherwise.

Procedure `EvaluateMove`(m,T)**:**

```
/* The evaluation of an infeasible move is set to ∞ */
b ← true, c ← −CurrentDuration(R_m) /* R_m denotes the routes impacted by move m */
/* Let R'_m denotes the newly created or modified routes obtained after applying move m */
for each route r ∈ R'_m do
    t ← DurationWithoutCharging(r)
    if t > T then return (∞,true) /* The move m is infeasible */ ;
    if EnergyConsumptionWithoutCharging(r) ⩽ Q then
        (c_r,b_r) ← (t,true)
    else
        t ← t + DetourAndChargingDurationLB(r) /* The computation of the lower bound
        if t > T then return (∞,true) /* The move m is infeasible */ ;
        if the duration of r is in the cache memory then (c_r,b_r) ← (DurationFromMemory(r),true) else (c_r,b_r) ←
          (t,false)
    end
    c ← c + c_r
    if b_r = false then b ← false
end
if c ⩾ 0 then b ← true /* The move m is non-improving */
return (c,b)
```

Algorithm 7: Checking the violation of capacity constraints for version N of the subproblem

Input : a list of charging operations L numbered from 1 to n ($L[i]$ denotes the operation at position i in the list L) / an integer number $C \geqslant 1$ representing the maximum number of operations that can be scheduled simultaneously

Output: a set containing all the maximal subsets of charging operations leading to a violation of the CS capacity constraint

Procedure `CheckCapacityCut`(O,C)**:**

```
Sort the operations in L in non-decreasing order of their starting time /* if two charging operations have the same
   starting time, then the charging operation with the minimum duration comes first */
O ← ∅ /* subsets of charging operations leading to a violation of the CS capacity constraint*/
U ← {L[1]} /* subset of charging operations currently executed */
k ← 2, excess ← false
while k ⩽ n do
    for every operation o ∈ U do
        if S(L[k]) ⩾ S(o) + d(o) then
            if excess then O ← O ∪ {U}, excess ← false
            else U ← U\{o}
        end
    end
    U ← U ∪ {L[k]}
    if |U| > C then excess ← true
end
if excess then O ← O ∪ {U}
return U
```

Algorithm 8: The fixed part based algorithm

Input : a list of charging operations O numbered from 1 to n ($O[k]$ denotes the operation at position k in the list O) / an integer number $C \geqslant 1$ representing the maximum number of operations that can be scheduled simultaneously

Output: a set containing all the maximal subsets of charging operations leading to a violation of the CS capacity constraint

Procedure `FixedPartAlgorithm`(O,C)**:**

Remove from O all the charging operations such that $\texttt{LS}(o) \geqslant \texttt{ES}(o) + \texttt{d}(o)$

Sort the operations in O in non-decreasing order of starting time `/* if two charging operations have the same starting time, then the charging operation with the minimum duration comes first */`

$\mathcal{O} \leftarrow \varnothing$ `/* subsets of charging operations leading to a violation of the CS capacity constraint */`

$U \leftarrow \{O[1]\}$ `/* subset of charging operations currently executed */`

$k \leftarrow 2$, excess $\leftarrow$ `false`

while $k \leqslant n$ **do**

  for every operation $o \in U$ **do**

    if $\texttt{LS}_{O[k]} \geqslant \texttt{ES}(o) + \texttt{d}(o)$ **then**

      if excess **then** $\mathcal{O} \leftarrow \mathcal{O} \cup \{U\}$, excess $\leftarrow$ `false`

      else $U \leftarrow U \backslash \{o\}$

    end

  end

  $U \leftarrow U \cup \{O[k]\}$

  if $|U| > C$ **then** excess $\leftarrow$ `true`

  $k \leftarrow k + 1$

end

if excess **then** $\mathcal{O} \leftarrow \mathcal{O} \cup \{U\}$

return $\mathcal{O}$

Algorithm 9: The energy reasoning based algorithm

Input : a set of charging operations O / an integer number $C \geqslant 1$ representing the maximum number of operations that can be scheduled simultaneously

Output: a set containing all subsets of charging operations leading to a violation of the CS capacity constraint

Procedure `EnergyAlgorithm`(O,C)**:**

$T_1 := \bigcup_{o \in O} (\{\texttt{ES}(o)\} \cup \{\texttt{ES}(o) + \texttt{d}(o)\} \cup \{\texttt{LS}(o)\})$

$T_2 := \bigcup_{o \in O} (\{\texttt{LS}(o) + \texttt{d}(o)\} \cup \{\texttt{ES}(o) + \texttt{d}(o)\} \cup \{\texttt{LS}(o)\})$

$T(t) := \bigcup_{o \in O} \{\texttt{ES}(o) + \texttt{LS}(o) + \texttt{d}(o) - t\}$

$T = \{(t_1, t_2) \in T_1 \times T_2\} \cup \{(t_1, t_2) \in T_1 \times T(t_1)\} \cup \{(t_1, t_2) \in T(t_2) \times T_2\}$

$\mathcal{O} \leftarrow \varnothing$ `/* subsets of charging operations leading to a violation of the CS capacity constraint */`

for $(t_1, t_2) \in T$ **do**

  `/*` $W(o, t_1, t_2) := \min\{0, t_2 - t_1, p_o^+(t_1), p_o^-(t_2)\}$ `with` $p_o^+(t_1) := \max\{0, \texttt{d}(o) - \max\{0, t_1 - \texttt{ES}(o)\}\}$ `(duration during which` o `is executed after time` t_1 `if it is scheduled as soon as possible) and` $p_o^-(t_2) := \max\{0, \texttt{d}(o) - \max\{0, \texttt{LS}(o) + \texttt{d}(o) - t_2\}\}$ `(duration during which` o `is executed before time` t_2 `if it is scheduled as late as possible) */`

  if $t_1 < t_2 \wedge \sum_{o \in O} W(o, t_1, t_2) > C(t_2 - t_1)$ **then**

    $\mathcal{O} \leftarrow \mathcal{O} \bigcup \{\{o \in O : W(o, t_1, t_2) > 0\}\}$

  end

end

return $\mathcal{O}$

E Detailed computational results

Based on notation introduced by Montoya et al. (2017), we write each instance using the symbol $tc\gamma_1 c\gamma_2 s\gamma_3 c\gamma_4 \#$ where γ_1 is the method used to place the customers (i.e., 0: randomization, 1: mixture of randomization and clustering, 2: clustering), γ_2 is the number of customers, γ_3 is the number of the CSs, γ_4 is 't' if we use a p-median heuristic to locate the CSs and 'f' otherwise, and # is the number of the instance for each combination of parameters (i.e., # = 0 , 1 , 2 , 3 , 4).

E.1 Detailed results for the E-VRP-NL-C formulation

Table 12 reports the detailed results obtained by a commercial MILP solver (Gurobi) for solving the E-VRP-NL-C using the CS path-based formulation.

E.2 Detailed results for the E-VRP-NL

Tables 13 and 14 report the detailed results obtained by our ILS and two methods from the literature on the E-VRP-NL (Montoya et al., 2017; Koç et al., 2018). The best values (objective function value of the BKS and best average value of the objective function obtained over 10 runs) are indicated in boldface.

E.3 Detailed results for the E-VRP-NL-C

Tables 15, 16, 17, and 18 report the detailed results obtained by our matheuristic on the E-VRP-NL-C instances. The best values (objective function value of the BKS and best average value of the objective function obtained over 10 runs) are indicated in boldface.

Table 12: Detailed results for the path-based MILP formulation on 10-customer and 20-customer instances.

Instance	Capacity= 1		Capacity= 2	
	Obj	Time	Obj	Time
tc0c10s2cf1	19.75	51	19.75	58
tc0c10s2ct1	12.30	43	12.30	45
tc0c10s3cf1	19.75	4838	19.75	3825
tc0c10s3ct1	10.80	91	10.80	96
tc1c10s2cf2	9.03	18	9.03	19
tc1c10s2cf3	16.37	373	16.37	200
tc1c10s2cf4	16.10	80	16.10	64
tc1c10s2ct2	10.75	3198	10.75	3602
tc1c10s2ct3	13.17	20	13.17	16
tc1c10s2ct4	13.83	24	13.83	17
tc1c10s3cf2	9.03	64	9.03	38
tc1c10s3cf3	16.37	730	16.37	1208
tc1c10s3cf4	14.90	350	14.90	270
tc1c10s3ct2	9.20	2173	9.20	1494
tc1c10s3ct3	13.02	264	13.02	201
tc1c10s3ct4	13.21	116	13.21	107
tc2c10s2cf0	21.77	T.L.	21.77	T.L.
tc2c10s2ct0	12.45	2407	12.45	1932
tc2c10s3cf0	21.77	T.L.	21.77	T.L.
tc2c10s3ct0	11.51	T.L.	11.51	T.L.
tc0c20s3cf2	27.94	T.L.	28.33	T.L.
tc0c20s3ct2	17.37	T.L.	17.60	T.L.
tc0c20s4cf2	*	T.L.	*	T.L.
tc0c20s4ct2	18.39	T.L.	18.23	T.L.
tc1c20s3cf1	22.37	T.L.	*	T.L.
tc1c20s3cf3	*	T.L.	*	T.L.
tc1c20s3cf4	*	T.L.	19.29	T.L.
tc1c20s3ct1	*	T.L.	*	T.L.
tc1c20s3ct3	*	T.L.	*	T.L.
tc1c20s3ct4	*	T.L.	*	T.L.
tc1c20s4cf1	*	T.L.	*	T.L.
tc1c20s4cf3	*	T.L.	*	T.L.
tc1c20s4cf4	*	T.L.	*	T.L.
tc1c20s4ct1	*	T.L.	*	T.L.
tc1c20s4ct3	*	T.L.	*	T.L.
tc1c20s4ct4	*	T.L.	*	T.L.
tc2c20s3cf0	*	T.L.	*	T.L.
tc2c20s3ct0	*	T.L.	*	T.L.
tc2c20s4cf0	*	T.L.	*	T.L.
tc2c20s4ct0	*	T.L.	*	T.L.

T.L.: CPU time limit reached

$*$: no feasible solution found within the CPU time limit.

Table 13: Detailed comparison of the results obtained by ILS with the results of Montoya et al. (2017) and Koç et al. (2018) for the E-VRP-NL (instances with 10, 20, or 40 customers)

		ILS + HC		LNS		ILS			ILS + HC		LNS		ILS	
Instance	BKS	Best	#R	Best	#R	Best	#R	BA	Avg	Time	Avg	Time	Avg	Time
tc0c10s2cf1	19.75	**19.75**	3	**19.75**	3	**19.75**	3	19.75	20.12	4	19.77	8	**19.75**	5
tc0c10s2ct1	12.30	**12.30**	2	**12.30**	2	**12.30**	2	12.30	12.34	4	12.31	8	**12.30**	3
tc0c10s3cf1	19.75	**19.75**	3	**19.75**	3	**19.75**	3	19.75	20.12	4	19.76	7	**19.75**	5
tc0c10s3ct1	10.80	**10.80**	2	**10.80**	2	**10.80**	2	10.80	**10.80**	5	10.81	8	**10.80**	5
tc1c10s2cf2	9.03	**9.03**	3	**9.03**	3	**9.03**	3	9.03	9.07	2	9.04	9	**9.03**	4
tc1c10s2cf3	16.37	**16.37**	3	**16.37**	3	**16.37**	3	16.37	**16.37**	6	16.38	9	**16.37**	5
tc1c10s2cf4	16.10	**16.10**	3	**16.10**	3	**16.10**	3	16.10	**16.10**	5	16.11	7	**16.10**	5
tc1c10s2ct2	10.75	**10.75**	3	**10.75**	3	**10.75**	3	10.75	**10.75**	4	10.76	8	**10.75**	3
tc1c10s2ct3	13.17	**13.17**	2	**13.17**	2	**13.17**	2	13.17	13.18	8	13.18	9	**13.17**	6
tc1c10s2ct4	13.83	**13.83**	2	**13.83**	2	**13.83**	2	13.83	**13.83**	5	13.84	9	**13.83**	5
tc1c10s3cf2	9.03	**9.03**	3	**9.03**	3	**9.03**	3	9.03	9.06	2	9.04	10	**9.03**	5
tc1c10s3cf3	16.37	**16.37**	3	**16.37**	3	**16.37**	3	16.37	**16.37**	6	16.39	8	**16.37**	3
tc1c10s3cf4	14.90	**14.90**	3	**14.90**	3	**14.90**	3	14.90	**14.90**	7	14.91	8	**14.90**	2
tc1c10s3ct2	9.20	**9.20**	3	**9.20**	3	**9.20**	3	9.20	9.34	5	9.21	9	**9.20**	6
tc1c10s3ct3	13.02	**13.02**	2	**13.02**	2	**13.02**	2	13.02	**13.02**	10	13.03	7	**13.02**	5
tc1c10s3ct4	13.21	**13.21**	2	**13.21**	2	**13.21**	2	13.21	**13.21**	6	13.22	9	**13.21**	5
tc2c10s2cf0	21.77	**21.77**	3	**21.77**	3	**21.77**	3	21.77	**21.77**	9	21.78	8	**21.77**	6
tc2c10s2ct0	12.45	**12.45**	3	**12.45**	3	**12.45**	3	12.45	**12.45**	5	12.46	8	**12.45**	6
tc2c10s3cf0	21.77	**21.77**	3	**21.77**	3	**21.77**	3	21.77	**21.77**	9	21.79	7	**21.77**	3
tc2c10s3ct0	11.51	**11.51**	3	**11.51**	3	**11.51**	3	11.51	11.54	7	11.52	9	**11.51**	5
tc0c20s3cf2	27.47	27.60	4	**27.47**	4	**27.47**	4	27.47	27.66	12	27.52	12	**27.47**	10
tc0c20s3ct2	17.08	**17.08**	3	**17.08**	3	**17.08**	3	17.08	17.13	8	17.11	18	**17.08**	6
tc0c20s4cf2	27.47	27.48	4	27.60	4	**27.47**	4	27.47	27.61	13	27.65	14	**27.47**	9
tc0c20s4ct2	16.99	**16.99**	3	**16.99**	3	**16.99**	3	16.99	17.10	9	17.02	16	**16.99**	9
tc1c20s3cf1	17.49	17.50	3	17.50	3	**17.49**	3	17.49	17.53	12	17.53	13	**17.49**	10
tc1c20s3cf3	16.44	16.63	4	16.48	3	**16.44**	3	16.44	16.78	8	16.50	17	**16.44**	7
tc1c20s3cf4	17.00	**17.00**	4	**17.00**	4	**17.00**	4	17.00	**17.00**	4	17.03	15	**17.00**	5
tc1c20s3ct1	18.94	18.95	4	18.95	4	**18.94**	4	18.94	19.38	15	18.97	14	**18.94**	9
tc1c20s3ct3	12.60	12.65	3	**12.60**	3	**12.60**	3	12.60	12.72	9	12.62	17	**12.60**	9
tc1c20s3ct4	16.21	**16.21**	4	**16.21**	4	**16.21**	4	16.21	16.25	5	16.24	11	**16.21**	8
tc1c20s4cf1	16.38	16.39	4	16.47	3	**16.38**	4	16.38	16.40	13	16.49	18	**16.38**	6
tc1c20s4cf3	16.44	16.56	3	16.48	3	**16.44**	3	16.44	16.80	9	16.51	11	**16.44**	11
tc1c20s4cf4	17.00	**17.00**	4	**17.00**	4	**17.00**	4	17.00	**17.00**	4	17.03	15	**17.00**	8
tc1c20s4ct1	17.80	18.25	4	18.25	4	**17.80**	4	17.80	18.32	16	18.28	18	**17.80**	11
tc1c20s4ct3	14.43	**14.43**	3	**14.43**	3	**14.43**	3	14.43	14.50	8	14.46	12	**14.43**	7
tc1c20s4ct4	17.00	**17.00**	4	**17.00**	4	**17.00**	4	17.00	**17.00**	6	17.03	11	**17.00**	6
tc2c20s3cf0	24.68	**24.68**	4	**24.68**	4	**24.68**	4	24.68	**24.68**	14	24.70	11	**24.68**	7
tc2c20s3ct0	25.79	**25.79**	4	**25.79**	4	**25.79**	4	25.79	**25.79**	15	25.83	15	**25.79**	10
tc2c20s4cf0	24.67	**24.67**	4	**24.67**	4	**24.67**	4	24.67	24.69	15	24.71	13	**24.67**	11
tc2c20s4ct0	26.02	**26.02**	4	**26.03**	4	**26.02**	4	26.02	**26.02**	15	26.07	16	**26.02**	9
tc0c40s5cf0	32.20	32.67	8	32.67	8	**32.20**	7	32.30	33.25	24	32.75	52	**32.30**	16
tc0c40s5cf4	30.25	30.77	6	30.60	6	**30.25**	6	30.25	31.49	33	30.69	49	**30.25**	22
tc0c40s5ct0	27.91	28.72	7	28.70	7	**27.91**	6	27.91	29.35	25	28.78	46	**27.91**	17
tc0c40s5ct4	28.63	**28.63**	6	**29.17**	5	**28.63**	6	28.63	28.72	33	29.25	59	**28.63**	18
tc0c40s8cf0	30.40	31.28	7	31.23	7	**30.40**	6	30.40	32.02	34	31.31	63	**30.40**	18
tc0c40s8cf4	28.11	29.32	6	28.25	5	**28.11**	5	28.23	29.86	43	28.30	52	**28.23**	25
tc0c40s8ct0	26.22	26.35	6	**26.22**	6	**26.22**	6	26.22	26.89	29	26.27	58	**26.22**	17
tc0c40s8ct4	29.07	29.20	6	29.22	6	**29.07**	5	29.07	29.27	47	29.28	48	**29.07**	22
tc1c40s5cf1	64.51	65.16	10	65.52	10	**64.51**	10	64.51	66.03	44	65.67	33	**64.51**	25
tc1c40s5ct1	52.33	52.68	9	52.60	9	**52.33**	8	52.33	53.36	59	52.72	40	**52.33**	23
tc1c40s8cf1	40.64	40.75	7	41.63	7	**40.64**	7	40.64	42.33	70	41.71	34	**40.64**	21
tc1c40s8ct1	40.18	40.56	7	40.56	7	**40.18**	7	40.18	41.19	71	40.67	49	**40.18**	24
tc2c40s5cf2	27.54	**27.54**	6	**27.54**	6	**27.54**	6	27.54	27.67	32	27.62	42	**27.54**	17
tc2c40s5cf3	19.65	19.74	5	**19.65**	5	**19.65**	5	19.65	20.18	17	19.70	50	**19.65**	21
tc2c40s5ct2	26.91	**26.91**	6	**26.91**	6	**26.91**	6	26.91	27.02	23	26.99	42	**26.91**	14
tc2c40s5ct3	23.39	23.54	6	23.71	6	**23.39**	6	23.39	23.77	26	23.75	51	**23.39**	22
tc2c40s8cf2	27.13	27.15	6	27.14	6	**27.13**	6	27.13	27.31	29	27.20	35	**27.13**	16
tc2c40s8cf3	19.65	19.66	5	**19.65**	5	**19.65**	5	19.65	20.24	19	19.69	36	**19.65**	22
tc2c40s8ct2	26.28	26.33	6	26.29	6	**26.28**	6	26.28	26.71	26	26.34	40	**26.28**	16
tc2c40s8ct3	22.45	22.71	5	**22.45**	5	**22.45**	5	22.45	23.23	25	22.52	30	**22.45**	24

Table 14: Detailed comparison of the results obtained by ILS with the results of Montoya et al. (2017) and Koç et al. (2018) for the E-VRP-NL (instances with 80, 160, or 320 customers)

		ILS + HC		LNS		ILS			ILS + HC		LNS		ILS	
Instance	BKS	Best	#R	Best	#R	Best	#R	BA	Avg	Time	Avg	Time	Avg	Time
tc0c80s12cf0	34.16	34.64	9	35.24	8	**34.16**	8	34.16	35.59	57	35.40	105	**34.16**	66
tc0c80s12cf1	40.91	42.90	10	42.30	9	**40.91**	9	40.94	44.07	75	42.47	85	**40.94**	68
tc0c80s12ct0	37.51	39.31	9	39.27	9	**37.51**	8	38.08	39.83	66	39.41	86	**38.08**	65
tc0c80s12ct1	39.91	41.94	10	41.64	9	**39.91**	9	40.06	43.03	73	41.83	103	**40.06**	59
tc0c80s8cf0	39.08	39.43	9	40.64	10	**39.08**	9	39.16	39.86	56	40.77	88	**39.16**	48
tc0c80s8cf1	43.38	45.23	10	46.65	9	**43.38**	9	43.95	45.73	121	46.80	98	**43.95**	73
tc0c80s8ct0	40.52	41.90	10	41.44	9	**40.52**	9	41.44	42.76	54	41.59	87	**41.44**	61
tc0c80s8ct1	43.85	45.27	10	45.25	10	**43.85**	9	44.07	45.85	130	45.37	100	**44.07**	73
tc1c80s12cf2	28.65	29.54	8	29.54	8	**28.65**	7	28.77	30.73	61	29.66	113	**28.77**	52
tc1c80s12ct2	28.73	29.52	8	29.38	8	**28.73**	8	29.18	30.66	59	29.47	114	**29.18**	54
tc1c80s8cf2	29.15	30.81	8	31.38	8	**29.15**	8	29.15	31.83	51	31.47	94	**29.15**	51
tc1c80s8ct2	30.45	31.74	8	31.72	8	**30.45**	8	30.52	32.36	60	31.82	98	**30.52**	57
tc2c80s12cf3	30.60	31.97	9	31.28	8	**30.60**	8	30.60	32.70	76	31.37	105	**30.60**	57
tc2c80s12cf4	42.10	43.89	9	43.69	9	**42.10**	9	42.14	44.97	131	43.81	86	**42.14**	83
tc2c80s12ct3	29.90	30.83	9	30.31	8	**29.90**	8	29.90	31.59	58	30.39	114	**29.90**	54
tc2c80s12ct4	40.27	42.40	9	42.56	9	**40.27**	9	40.27	42.82	134	44.68	103	**40.27**	74
tc2c80s8cf3	31.70	32.44	9	31.94	8	**31.70**	8	31.93	32.60	64	32.06	87	**31.93**	55
tc2c80s8cf4	46.03	49.29	10	49.67	10	**46.03**	9	46.78	49.69	100	49.84	128	**46.78**	93
tc2c80s8ct3	31.38	32.31	9	32.71	9	**31.38**	8	31.43	32.55	65	32.82	89	**31.43**	65
tc2c80s8ct4	43.83	44.83	10	44.16	10	**43.83**	9	44.00	46.61	111	44.31	103	**44.00**	75
tc0c160s16cf2	57.91	61.20	16	62.09	15	**57.91**	15	58.00	62.99	365	62.55	442	**58.00**	242
tc0c160s16cf4	76.90	82.92	18	82.77	18	**76.90**	16	77.55	83.84	1213	83.41	709	**77.55**	367
tc0c160s16ct2	57.64	59.90	15	59.75	15	**57.64**	15	57.73	62.80	342	60.29	811	**57.73**	247
tc0c160s16ct4	76.14	82.37	18	82.90	18	**76.14**	16	76.90	83.08	945	83.85	983	**76.90**	353
tc0c160s24cf2	56.32	59.27	15	59.26	15	**56.32**	14	56.76	60.92	403	59.79	732	**56.76**	253
tc0c160s24cf4	75.53	81.44	18	81.43	18	**75.53**	16	76.30	82.13	1209	82.33	595	**76.30**	370
tc0c160s24ct2	55.42	59.25	16	59.67	16	**55.42**	14	56.47	60.19	410	60.21	915	**56.47**	253
tc0c160s24ct4	75.05	80.96	18	81.38	18	**75.05**	16	75.87	82.11	957	82.21	436	**75.87**	372
tc1c160s16cf0	74.54	79.80	18	79.76	18	**74.54**	16	75.32	80.75	766	80.52	420	**75.32**	327
tc1c160s16cf3	66.45	71.76	17	71.98	17	**66.45**	15	67.20	72.75	462	72.77	729	**67.20**	307
tc1c160s16ct0	74.20	79.04	17	80.21	17	**74.20**	16	75.31	79.90	643	80.99	472	**75.31**	326
tc1c160s16ct3	65.31	73.29	17	73.24	17	**65.31**	15	66.20	75.11	279	73.82	750	**66.20**	289
tc1c160s24cf0	73.62	78.60	17	79.48	17	**73.62**	16	74.05	79.30	741	80.32	460	**74.05**	331
tc1c160s24cf3	62.90	68.56	17	68.73	17	**62.90**	15	63.64	69.57	483	69.28	522	**63.64**	282
tc1c160s24ct0	73.34	78.21	17	78.32	17	**73.34**	16	74.00	79.35	578	79.05	553	**74.00**	319
tc1c160s24ct3	63.19	68.72	17	69.17	17	**63.19**	15	63.66	69.98	358	69.76	889	**63.66**	280
tc2c160s16cf1	56.65	60.34	16	60.25	15	**56.65**	14	57.39	61.26	274	60.70	716	**57.39**	252
tc2c160s16ct1	55.37	60.27	15	59.86	15	**55.37**	14	55.52	60.62	288	60.40	408	**55.52**	232
tc2c160s24cf1	56.70	59.82	16	60.01	16	**56.70**	14	57.27	61.14	305	60.63	564	**57.27**	260
tc2c160s24ct1	55.03	59.13	16	59.97	15	**55.03**	14	55.15	59.72	340	60.53	531	**55.15**	238
tc1c320s24cf2	133.32	152.13	36	153.12	36	**133.32**	31	133.99	153.99	7106	154.65	4155	**133.99**	1287
tc1c320s24cf3	106.43	117.48	30	117.39	30	**106.43**	28	107.00	118.36	3066	118.43	3258	**107.00**	1060
tc1c320s24ct2	131.63	148.77	36	148.57	36	**131.63**	30	132.49	154.13	6853	149.89	4727	**132.49**	1231
tc1c320s24ct3	105.93	116.64	31	117.50	31	**105.93**	27	106.67	119.17	3274	118.53	5105	**106.67**	1045
tc1c320s38cf2	129.19	141.63	33	142.25	33	**129.19**	30	129.76	147.08	7236	144.17	4249	**129.76**	1178
tc1c320s38cf3	106.01	116.22	30	117.31	30	**106.01**	28	106.36	117.74	3114	118.78	5978	**106.36**	1129
tc1c320s38ct2	128.82	140.96	32	142.75	32	**128.82**	30	129.51	145.09	6974	144.50	6078	**129.51**	1167
tc1c320s38ct3	105.73	116.07	30	117.91	30	**105.73**	27	106.74	117.71	3063	119.40	3157	**106.74**	1186
tc2c320s24cf0	158.80	182.45	38	182.90	38	**158.80**	33	160.55	186.94	6566	185.27	4014	**160.55**	1343
tc2c320s24cf1	87.46	95.51	29	95.71	29	**87.46**	26	87.64	96.42	1456	96.81	5150	**87.64**	890
tc2c320s24cf4	111.16	122.74	32	122.83	32	**111.16**	28	111.62	124.68	3681	124.51	3923	**111.62**	989
tc2c320s24ct0	159.70	181.45	37	182.29	37	**159.70**	33	160.49	186.23	7204	183.80	6191	**160.49**	1309
tc2c320s24ct1	87.25	94.73	27	94.97	27	**87.25**	26	87.83	96.49	1259	95.96	3530	**87.83**	863
tc2c320s24ct4	111.09	121.94	32	122.09	32	**111.09**	28	111.62	123.85	4274	123.45	5196	**111.62**	1041
tc2c320s38cf0	158.70	176.92	37	178.17	37	**158.70**	33	159.53	182.31	6734	179.81	3350	**159.53**	1356
tc2c320s38cf1	86.92	94.29	28	95.73	28	**86.92**	26	87.25	95.07	1602	96.79	5343	**87.25**	890
tc2c320s38cf4	109.80	122.32	32	122.26	32	**109.80**	28	110.66	123.47	2661	123.46	3724	**110.66**	1087
tc2c320s38ct0	158.71	190.97	41	192.23	41	**158.71**	33	159.35	192.15	7637	194.66	4448	**159.35**	1374
tc2c320s38ct1	86.59	94.53	28	94.66	28	**86.59**	26	86.97	95.29	1409	95.87	3973	**86.97**	894
tc2c320s38ct4	110.05	121.66	32	121.64	32	**110.05**	28	110.55	123.15	2785	123.06	5554	**110.55**	1034

Table 15: Detailed comparison of the best results obtained by the matheuristic for the E-VRP-NL-C according to the version of the subproblem it uses if the number of chargers at every CS is equal to 1 (instances with 10, 20, or 40 customers).

Instance	Version N				Version D				Version DW				Version DWR				BS	BA
	Time	#F	Best	Avg	Time	#F	Best	Avg	Time	#F	Best	Avg	Time	#F	Best	Avg		
tc0c10s2cf1	5	10	**19.75**	**19.75**	5	10	**19.75**	**19.75**	6	10	**19.75**	**19.75**	5	10	**19.75**	**19.75**	19.75	19.75
tc0c10s2ct1	4	10	**12.30**	**12.30**	4	10	**12.30**	**12.30**	4	10	**12.30**	**12.30**	4	10	**12.30**	**12.30**	12.30	12.30
tc0c10s3cf1	5	10	**19.75**	**19.75**	6	10	**19.75**	**19.75**	6	10	**19.75**	**19.75**	6	10	**19.75**	**19.75**	19.75	19.75
tc0c10s3ct1	5	10	**10.80**	**10.80**	5	10	**10.80**	**10.80**	5	10	**10.80**	**10.80**	5	10	**10.80**	**10.80**	10.80	10.80
tc1c10s2cf2	5	10	**9.03**	**9.03**	5	10	**9.03**	**9.03**	6	10	**9.03**	**9.03**	5	10	**9.03**	**9.03**	9.03	9.03
tc1c10s2cf3	6	10	**16.37**	**16.37**	6	10	**16.37**	**16.37**	6	10	**16.37**	**16.37**	6	10	**16.37**	**16.37**	16.37	16.37
tc1c10s2cf4	5	10	**16.10**	**16.10**	6	10	**16.10**	**16.10**	5	10	**16.10**	**16.10**	6	10	**16.10**	**16.10**	16.10	16.10
tc1c10s2ct2	4	10	**10.75**	**10.75**	4	10	**10.75**	**10.75**	4	10	**10.75**	**10.75**	4	10	**10.75**	**10.75**	10.75	10.75
tc1c10s2ct3	5	10	13.25	13.25	5	10	**13.17**	**13.17**	5	10	**13.17**	**13.17**	5	10	**13.17**	**13.17**	13.17	13.17
tc1c10s2ct4	5	10	**13.83**	**13.83**	5	10	**13.83**	**13.83**	5	10	**13.83**	**13.83**	5	10	**13.83**	**13.83**	13.83	13.83
tc1c10s3cf2	4	10	**9.03**	**9.03**	5	10	**9.03**	**9.03**	5	10	**9.03**	**9.03**	5	10	**9.03**	**9.03**	9.03	9.03
tc1c10s3cf3	4	10	**16.37**	**16.37**	4	10	**16.37**	**16.37**	4	10	**16.37**	**16.37**	4	10	**16.37**	**16.37**	16.37	16.37
tc1c10s3cf4	3	10	**14.90**	**14.90**	4	10	**14.90**	**14.90**	4	10	**14.90**	**14.90**	4	10	**14.90**	**14.90**	14.90	14.90
tc1c10s3ct2	7	10	**9.20**	**9.20**	7	10	**9.20**	**9.20**	7	10	**9.20**	**9.20**	7	10	**9.20**	**9.20**	9.20	9.20
tc1c10s3ct3	5	10	**13.02**	**13.02**	6	10	**13.02**	**13.02**	6	10	**13.02**	**13.02**	6	10	**13.02**	**13.02**	13.02	13.02
tc1c10s3ct4	5	10	**13.21**	**13.21**	6	10	**13.21**	**13.21**	6	10	**13.21**	**13.21**	5	10	**13.21**	**13.21**	13.21	13.21
tc2c10s2cf0	5	10	21.83	21.83	5	10	**21.77**	**21.77**	6	10	**21.77**	**21.77**	6	10	**21.77**	**21.77**	21.77	21.77
tc2c10s2ct0	5	10	**12.45**	**12.45**	5	10	**12.45**	**12.45**	4	10	**12.45**	**12.45**	5	10	**12.45**	**12.45**	12.45	12.45
tc2c10s3cf0	4	10	21.83	21.83	4	10	**21.77**	**21.77**	4	10	**21.77**	**21.77**	4	10	**21.77**	**21.77**	21.77	21.77
tc2c10s3ct0	6	10	**11.51**	**11.51**	6	10	**11.51**	**11.51**	6	10	**11.51**	**11.51**	6	10	**11.51**	**11.51**	11.51	11.51
tc0c20s3cf2	12	10	27.62	27.62	12	10	**27.47**	**27.47**	11	10	**27.47**	**27.47**	12	10	**27.47**	**27.47**	27.47	27.47
tc0c20s3ct2	8	10	**17.08**	**17.08**	8	10	**17.08**	**17.08**	7	10	**17.08**	**17.08**	8	10	**17.08**	**17.08**	17.08	17.08
tc0c20s4cf2	10	10	27.62	27.62	10	10	**27.47**	**27.47**	10	10	**27.47**	**27.47**	9	10	**27.47**	**27.47**	27.47	27.47
tc0c20s4ct2	10	10	**16.99**	**16.99**	10	10	**16.99**	**16.99**	10	10	**16.99**	**16.99**	11	10	**16.99**	**16.99**	16.99	16.99
tc1c20s3cf1	12	10	**17.49**	**17.49**	12	10	**17.49**	**17.49**	13	10	**17.49**	**17.49**	12	10	**17.49**	**17.49**	17.49	17.49
tc1c20s3cf3	9	10	**16.44**	**16.44**	9	10	**16.44**	**16.44**	9	10	**16.44**	**16.44**	10	10	**16.44**	**16.44**	16.44	16.44
tc1c20s3cf4	6	10	**17.00**	**17.00**	7	10	**17.00**	**17.00**	6	10	**17.00**	**17.00**	6	10	**17.00**	**17.00**	17.00	17.00
tc1c20s3ct1	11	10	**18.94**	**18.94**	10	10	**18.94**	**18.94**	11	10	**18.94**	**18.94**	11	10	**18.94**	**18.94**	18.94	18.94
tc1c20s3ct3	11	10	**12.60**	**12.60**	11	10	**12.60**	**12.60**	11	10	**12.60**	**12.60**	11	10	**12.60**	**12.60**	12.60	12.60
tc1c20s3ct4	9	10	**16.21**	**16.21**	10	10	**16.21**	**16.21**	9	10	**16.21**	**16.21**	9	10	**16.21**	**16.21**	16.21	16.21
tc1c20s4cf1	8	10	**16.38**	**16.38**	8	10	**16.38**	**16.38**	8	10	**16.38**	**16.38**	8	10	**16.38**	**16.38**	16.38	16.38
tc1c20s4cf3	13	10	**16.44**	**16.44**	12	10	**16.44**	**16.44**	12	10	**16.44**	**16.44**	13	10	**16.44**	**16.44**	16.44	16.44
tc1c20s4cf4	10	10	**17.00**	**17.00**	10	10	**17.00**	**17.00**	9	10	**17.00**	**17.00**	10	10	**17.00**	**17.00**	17.00	17.00
tc1c20s4ct1	12	10	**17.80**	**17.80**	13	10	**17.80**	**17.80**	12	10	**17.80**	**17.80**	12	10	**17.80**	**17.80**	17.80	17.80
tc1c20s4ct3	7	10	**14.43**	**14.43**	7	10	**14.43**	**14.43**	7	10	**14.43**	**14.43**	8	10	**14.43**	**14.43**	14.43	14.43
tc1c20s4ct4	8	10	**17.00**	**17.00**	8	10	**17.00**	**17.00**	8	10	**17.00**	**17.00**	8	10	**17.00**	**17.00**	17.00	17.00
tc2c20s3cf0	9	10	**24.68**	**24.68**	9	10	**24.68**	**24.68**	8	10	**24.68**	**24.68**	9	10	**24.68**	**24.68**	24.68	24.68
tc2c20s3ct0	12	10	25.80	25.80	12	10	**25.79**	**25.79**	12	10	**25.79**	**25.79**	12	10	**25.79**	**25.79**	25.79	25.79
tc2c20s4cf0	12	10	24.68	24.68	12	10	**24.67**	**24.67**	12	10	**24.67**	**24.67**	12	10	**24.67**	**24.67**	24.67	24.67
tc2c20s4ct0	11	10	26.17	26.18	11	10	**26.02**	**26.02**	12	10	**26.02**	**26.02**	12	10	**26.02**	**26.02**	26.02	26.02
tc0c40s5cf0	21	10	**32.20**	**32.23**	20	10	**32.20**	**32.23**	20	10	**32.20**	**32.23**	20	10	**32.20**	**32.23**	32.20	32.23
tc0c40s5cf4	26	10	**30.25**	**30.25**	25	10	**30.25**	**30.25**	25	10	**30.25**	**30.25**	26	10	**30.25**	**30.25**	30.25	30.25
tc0c40s5ct0	20	10	**27.91**	**27.91**	20	10	**27.91**	**27.91**	21	10	**27.91**	**27.91**	21	10	**27.91**	**27.91**	27.91	27.91
tc0c40s5ct4	23	10	**28.63**	**28.63**	23	10	**28.63**	**28.63**	22	10	**28.63**	**28.63**	22	10	**28.63**	**28.63**	28.63	28.63
tc0c40s8cf0	22	10	30.46	30.51	21	10	**30.40**	**30.40**	21	10	**30.40**	**30.40**	21	10	**30.40**	**30.40**	30.40	30.40
tc0c40s8cf4	28	10	**28.24**	**28.25**	29	10	**28.24**	**28.25**	29	10	**28.24**	**28.25**	28	10	**28.24**	**28.25**	28.24	28.25
tc0c40s8ct0	20	10	**26.22**	**26.22**	20	10	**26.22**	**26.22**	20	10	**26.22**	**26.22**	20	10	**26.22**	**26.22**	26.22	26.22
tc0c40s8ct4	27	10	**29.07**	**29.07**	26	10	**29.07**	**29.07**	26	10	**29.07**	**29.07**	27	10	**29.07**	**29.07**	29.07	29.07
tc1c40s5cf1	27	0	-	-	14	0	-	-	14	0	-	-	1299	0	-	-	-	-
tc1c40s5ct1	26	10	52.42	52.70	28	10	52.42	52.60	26	10	52.42	52.60	27	10	**52.33**	**52.33**	52.33	52.33
tc1c40s8cf1	25	10	41.67	42.21	24	10	**40.64**	**40.64**	23	10	**40.64**	**40.64**	24	10	**40.64**	**40.64**	40.64	40.64
tc1c40s8ct1	29	10	40.45	40.73	28	10	**40.35**	40.36	28	10	**40.35**	40.36	29	10	**40.35**	**40.35**	40.35	40.35
tc2c40s5cf2	20	10	**27.54**	**27.54**	19	10	**27.54**	**27.54**	20	10	**27.54**	**27.54**	21	10	**27.54**	**27.54**	27.54	27.54
tc2c40s5cf3	26	10	**19.65**	**19.65**	25	10	**19.65**	**19.65**	25	10	**19.65**	**19.65**	26	10	**19.65**	**19.65**	19.65	19.65
tc2c40s5ct2	20	10	**26.91**	**26.91**	19	10	**26.91**	**26.91**	19	10	**26.91**	**26.91**	19	10	**26.91**	**26.91**	26.91	26.91
tc2c40s5ct3	23	10	**23.39**	**23.39**	22	10	**23.39**	**23.39**	22	10	**23.39**	**23.39**	21	10	**23.39**	**23.39**	23.39	23.39
tc2c40s8cf2	20	10	**27.13**	**27.13**	20	10	**27.13**	**27.13**	20	10	**27.13**	**27.13**	20	10	**27.13**	**27.13**	27.13	27.13
tc2c40s8cf3	26	10	**19.65**	**19.65**	25	10	**19.65**	**19.65**	26	10	**19.65**	**19.65**	24	10	**19.65**	**19.65**	19.65	19.65
tc2c40s8ct2	19	10	**26.28**	**26.28**	19	10	**26.28**	**26.28**	18	10	**26.28**	**26.28**	18	10	**26.28**	**26.28**	26.28	26.28
tc2c40s8ct3	27	10	**22.45**	**22.45**	28	10	**22.45**	**22.45**	28	10	**22.45**	**22.45**	28	10	**22.45**	**22.45**	22.45	22.45

Table 16: Detailed comparison of the best results obtained by the matheuristic for the E-VRP-NL-C according to the version of the subproblem it uses if the number of charger at every CS is equal to 1(instances with 80, 160, or 320 customers).

Instance	Version N				Version D				Version DW				Version DWR				BS	BA
	Time	#F	Best	Avg	Time	#F	Best	Avg	Time	#F	Best	Avg	Time	#F	Best	Avg		
tc0c80s12cf0	74	10	**34.06**	**34.20**	75	10	**34.06**	**34.20**	78	10	**34.06**	**34.20**	77	10	**34.06**	**34.20**	34.06	34.20
tc0c80s12cf1	78	10	**40.83**	40.95	77	10	**40.83**	**40.94**	77	10	**40.83**	**40.94**	77	10	**40.83**	**40.94**	40.83	40.94
tc0c80s12ct0	76	10	**37.51**	37.95	77	10	**37.51**	**37.89**	77	10	**37.51**	**37.89**	77	10	**37.51**	37.98	37.51	37.89
tc0c80s12ct1	72	10	**39.72**	40.01	73	10	**39.72**	**40.00**	71	10	**39.72**	**40.00**	72	10	**39.72**	**40.00**	39.72	40.00
tc0c80s8cf0	61	10	**38.59**	**38.79**	61	10	**38.59**	**38.79**	62	10	**38.59**	**38.79**	60	10	**38.59**	**38.79**	38.59	38.79
tc0c80s8cf1	83	10	43.41	44.28	80	10	**43.38**	**44.05**	81	10	**43.38**	**44.05**	85	10	**43.38**	44.13	43.38	44.05
tc0c80s8ct0	71	10	**40.53**	**41.47**	71	10	**40.53**	**41.47**	69	10	**40.53**	**41.47**	69	10	**40.53**	**41.47**	40.53	41.47
tc0c80s8ct1	82	10	43.86	43.99	82	10	**43.85**	**43.98**	80	10	**43.85**	**43.98**	82	10	**43.85**	**43.98**	43.85	43.98
tc1c80s12cf2	58	10	**28.65**	**28.77**	59	10	**28.65**	**28.77**	58	10	**28.65**	**28.77**	59	10	**28.65**	**28.77**	28.65	28.77
tc1c80s12ct2	60	10	**28.73**	29.17	60	10	**28.73**	**29.09**	59	10	**28.73**	**29.09**	62	10	**28.73**	29.14	28.73	29.09
tc1c80s8cf2	63	10	**29.15**	**29.15**	62	10	**29.15**	**29.15**	60	10	**29.15**	**29.15**	62	10	**29.15**	**29.15**	29.15	29.15
tc1c80s8ct2	61	10	**29.88**	**30.42**	63	10	**29.88**	**30.42**	62	10	**29.88**	**30.42**	63	10	**29.88**	**30.42**	29.88	30.42
tc2c80s12cf3	63	10	30.61	30.61	62	10	**30.60**	**30.60**	63	10	**30.60**	**30.60**	62	10	**30.60**	**30.60**	30.60	30.60
tc2c80s12cf4	89	10	42.20	42.62	88	10	**42.12**	**42.25**	89	10	**42.12**	**42.25**	105	10	**42.12**	42.26	42.12	42.25
tc2c80s12ct3	62	10	**29.90**	**29.90**	61	10	**29.90**	**29.90**	62	10	**29.90**	**29.90**	61	10	**29.90**	**29.90**	29.90	29.90
tc2c80s12ct4	87	10	40.28	40.28	86	10	**40.27**	**40.27**	85	10	**40.27**	**40.27**	86	10	**40.27**	**40.27**	40.27	40.27
tc2c80s8cf3	61	10	**31.70**	**31.93**	60	10	**31.70**	**31.93**	61	10	**31.70**	**31.93**	60	10	**31.70**	**31.93**	31.70	31.93
tc2c80s8cf4	100	10	**46.19**	46.85	102	10	**46.19**	**46.84**	104	10	**46.19**	**46.84**	102	10	**46.19**	46.94	46.19	46.84
tc2c80s8ct3	75	10	**31.38**	**31.38**	76	10	**31.38**	**31.38**	76	10	**31.38**	**31.38**	74	10	**31.38**	**31.38**	31.38	31.38
tc2c80s8ct4	85	10	**43.98**	44.03	86	10	**43.98**	**44.02**	84	10	**43.98**	**44.02**	88	10	**43.98**	**44.02**	43.98	44.02
tc0c160s16cf2	285	10	57.93	58.04	272	10	**57.91**	58.02	273	10	**57.91**	58.02	268	10	**57.91**	**58.01**	57.91	58.01
tc0c160s16cf4	428	10	77.56	78.47	429	10	**77.30**	**77.63**	425	10	**77.30**	**77.63**	862	10	**77.30**	77.67	77.30	77.63
tc0c160s16ct2	286	10	**57.16**	**57.63**	288	10	**57.16**	57.64	289	10	**57.16**	57.64	279	10	**57.16**	57.64	57.16	57.63
tc0c160s16ct4	417	10	**76.18**	77.25	404	10	76.30	**76.97**	414	10	76.30	76.98	798	10	**76.18**	**76.97**	76.18	76.97
tc0c160s24cf2	291	10	**56.86**	**56.93**	284	10	**56.86**	56.94	285	10	**56.86**	56.94	286	10	**56.86**	**56.93**	56.86	56.93
tc0c160s24cf4	430	10	76.21	77.04	429	10	**75.63**	76.60	432	10	**75.63**	**76.55**	538	10	75.83	76.67	75.63	76.55
tc0c160s24ct2	298	10	55.53	56.30	303	10	**55.42**	56.23	301	10	**55.42**	56.23	294	10	**55.42**	**56.17**	55.42	56.17
tc0c160s24ct4	420	10	75.13	76.34	432	10	**75.09**	**76.26**	430	10	**75.09**	**76.26**	577	10	75.36	76.29	75.09	76.26
tc1c160s16cf0	382	8	75.81	77.16	372	10	74.69	76.53	379	10	74.69	76.53	413	10	**74.64**	**75.83**	74.64	75.83
tc1c160s16cf3	348	10	66.52	67.80	347	10	66.52	**67.17**	352	10	66.52	67.18	352	10	**66.49**	67.26	66.49	67.17
tc1c160s16ct0	369	10	74.30	74.94	367	10	74.24	74.91	360	10	74.24	74.91	382	10	**74.20**	**74.62**	74.20	74.62
tc1c160s16ct3	323	10	65.45	66.40	316	10	65.24	**66.02**	318	10	65.24	**66.02**	313	10	**65.23**	66.08	65.23	66.02
tc1c160s24cf0	381	10	73.93	74.60	372	10	**73.62**	73.96	374	10	**73.62**	**73.90**	445	10	**73.62**	74.03	73.62	73.90
tc1c160s24cf3	328	10	63.22	63.93	323	10	63.06	**63.60**	331	10	63.06	**63.60**	331	10	**63.05**	63.78	63.05	63.60
tc1c160s24ct0	384	10	73.95	74.61	380	10	**73.32**	73.95	371	10	**73.32**	73.95	375	10	**73.32**	**73.77**	73.32	73.77
tc1c160s24ct3	335	10	62.76	**63.22**	334	10	**62.70**	63.30	335	10	**62.70**	63.30	328	10	**62.70**	63.26	62.70	63.22
tc2c160s16cf1	284	10	56.71	57.44	286	10	**56.59**	**57.33**	285	10	**56.59**	**57.33**	284	10	56.66	57.38	56.59	57.33
tc2c160s16ct1	257	10	**55.37**	55.45	257	10	**55.37**	**55.43**	255	10	**55.37**	**55.43**	261	10	**55.37**	55.44	55.37	55.43
tc2c160s24cf1	289	10	**56.00**	57.24	298	10	56.47	**57.20**	299	10	56.47	**04:48**	297	10	56.47	57.36	56.00	57.20
tc2c160s24ct1	276	10	55.03	**55.07**	286	10	**54.96**	55.10	281	10	**54.96**	55.10	270	10	55.03	55.15	54.96	55.07
tc1c320s24cf2	1600	0	-	-	1582	10	133.89	**135.14**	1570	10	133.91	135.16	3219	10	**133.66**	135.37	133.66	135.14
tc1c320s24cf3	1250	10	105.88	106.73	1252	10	105.70	**106.70**	1268	10	105.70	**106.70**	1262	10	**105.47**	106.77	105.47	106.70
tc1c320s24ct2	1480	0	-	-	2337	10	133.46	135.57	2366	10	133.46	135.23	3461	9	**132.50**	**134.69**	132.50	134.69
tc1c320s24ct3	1166	10	**105.40**	106.31	1162	10	105.68	**106.29**	1156	10	105.68	**106.29**	1186	10	105.89	106.47	105.40	106.29
tc1c320s38cf2	1417	0	-	-	1370	10	129.09	129.61	1375	10	129.09	129.68	1561	10	**128.96**	**129.40**	128.96	129.40
tc1c320s38cf3	1286	10	105.60	**106.21**	1276	10	105.71	106.47	1265	10	105.71	106.47	1232	10	**105.41**	106.23	105.41	106.21
tc1c320s38ct2	1422	0	-	-	1397	10	128.86	129.80	1399	10	128.86	129.79	2603	10	**128.69**	**129.65**	128.69	129.65
tc1c320s38ct3	1342	10	106.42	106.82	1356	10	105.93	**106.60**	1377	10	105.93	**106.60**	1395	10	**105.44**	106.66	105.44	106.60
tc2c320s24cf0	1840	0	-	-	2534	10	163.39	165.89	2587	10	163.39	165.94	3472	10	**162.15**	**165.15**	162.15	165.15
tc2c320s24cf1	1014	10	**87.19**	**87.59**	1004	10	87.27	87.65	1021	10	87.27	87.65	1028	10	87.26	87.68	87.19	87.59
tc2c320s24cf4	1113	4	112.46	114.09	1095	10	111.74	112.45	1083	10	111.74	112.39	1433	10	**111.59**	**112.19**	111.59	112.19
tc2c320s24ct0	1683	0	-	-	2482	10	164.81	166.51	2498	10	164.81	**166.46**	3413	10	**164.18**	166.59	164.18	166.46
tc2c320s24ct1	1016	10	**86.86**	87.57	1026	10	87.26	87.64	1043	10	87.26	87.64	1017	10	87.01	**87.51**	86.86	87.51
tc2c320s24ct4	1214	5	111.66	113.18	1191	10	111.00	111.90	1182	10	111.00	**111.76**	1231	10	**110.95**	111.82	110.95	111.76
tc2c320s38cf0	1697	0	-	-	1875	10	160.69	162.51	1886	10	160.69	162.51	3388	10	**160.59**	**161.69**	160.59	161.69
tc2c320s38cf1	1016	10	**86.65**	**87.17**	1013	10	86.97	87.25	1029	10	86.97	87.25	999	10	86.69	87.21	86.65	87.17
tc2c320s38cf4	1274	10	**109.48**	110.62	1226	10	110.19	110.63	1224	10	110.19	110.63	1253	10	109.91	**110.46**	109.48	110.46
tc2c320s38ct0	1825	0	-	-	2169	10	160.54	162.97	2166	10	**160.36**	162.94	3252	10	161.25	**162.34**	160.36	162.34
tc2c320s38ct1	1076	10	86.59	86.97	1075	10	**86.23**	**86.72**	1068	10	**86.23**	**86.72**	1061	10	86.53	86.88	86.23	86.72
tc2c320s38ct4	1182	10	109.97	110.73	1201	10	109.60	110.29	1189	10	109.60	110.29	1210	10	**109.44**	**110.27**	109.44	110.27

Table 17: Detailed comparison of the best results obtained by the matheuristic for the E-VRP-NL-C according to the version of the subproblem it uses if the number of chargers at every CS is equal to 2 (instances with 10, 20, or 40 customers).

Instance	Version N				Version D				Version DW				Version DWR				BS	BA
	Time	#F	Best	Avg	Time	#F	Best	Avg	Time	#F	Best	Avg	Time	#F	Best	Avg		
tc0c10s2cf1	6	10	**19.75**	**19.75**	6	10	**19.75**	**19.75**	5	10	**19.75**	**19.75**	5	10	**19.75**	**19.75**	19.75	19.75
tc0c10s2ct1	4	10	**12.30**	**12.30**	4	10	**12.30**	**12.30**	4	10	**12.30**	**12.30**	4	10	**12.30**	**12.30**	12.30	12.30
tc0c10s3cf1	6	10	**19.75**	**19.75**	5	10	**19.75**	**19.75**	6	10	**19.75**	**19.75**	6	10	**19.75**	**19.75**	19.75	19.75
tc0c10s3ct1	5	10	**10.80**	**10.80**	5	10	**10.80**	**10.80**	5	10	**10.80**	**10.80**	5	10	**10.80**	**10.80**	10.80	10.80
tc1c10s2cf2	6	10	**9.03**	**9.03**	6	10	**9.03**	**9.03**	5	10	**9.03**	**9.03**	6	10	**9.03**	**9.03**	9.03	9.03
tc1c10s2cf3	6	10	**16.37**	**16.37**	6	10	**16.37**	**16.37**	6	10	**16.37**	**16.37**	6	10	**16.37**	**16.37**	16.37	16.37
tc1c10s2cf4	6	10	**16.10**	**16.10**	5	10	**16.10**	**16.10**	5	10	**16.10**	**16.10**	6	10	**16.10**	**16.10**	16.10	16.10
tc1c10s2ct2	4	10	**10.75**	**10.75**	4	10	**10.75**	**10.75**	4	10	**10.75**	**10.75**	4	10	**10.75**	**10.75**	10.75	10.75
tc1c10s2ct3	5	10	**13.17**	**13.17**	5	10	**13.17**	**13.17**	5	10	**13.17**	**13.17**	5	10	**13.17**	**13.17**	13.17	13.17
tc1c10s2ct4	5	10	**13.83**	**13.83**	5	10	**13.83**	**13.83**	5	10	**13.83**	**13.83**	5	10	**13.83**	**13.83**	13.83	13.83
tc1c10s3cf2	5	10	**9.03**	**9.03**	5	10	**9.03**	**9.03**	5	10	**9.03**	**9.03**	5	10	**9.03**	**9.03**	9.03	9.03
tc1c10s3cf3	4	10	**16.37**	**16.37**	4	10	**16.37**	**16.37**	4	10	**16.37**	**16.37**	4	10	**16.37**	**16.37**	16.37	16.37
tc1c10s3cf4	3	10	**14.90**	**14.90**	4	10	**14.90**	**14.90**	3	10	**14.90**	**14.90**	3	10	**14.90**	**14.90**	14.90	14.90
tc1c10s3ct2	6	10	**9.20**	**9.20**	7	10	**9.20**	**9.20**	7	10	**9.20**	**9.20**	6	10	**9.20**	**9.20**	9.20	9.20
tc1c10s3ct3	6	10	**13.02**	**13.02**	5	10	**13.02**	**13.02**	5	10	**13.02**	**13.02**	5	10	**13.02**	**13.02**	13.02	13.02
tc1c10s3ct4	5	10	**13.21**	**13.21**	6	10	**13.21**	**13.21**	5	10	**13.21**	**13.21**	5	10	**13.21**	**13.21**	13.21	13.21
tc2c10s2cf0	6	10	**21.77**	**21.77**	6	10	**21.77**	**21.77**	6	10	**21.77**	**21.77**	6	10	**21.77**	**21.77**	21.77	21.77
tc2c10s2ct0	5	10	**12.45**	**12.45**	5	10	**12.45**	**12.45**	5	10	**12.45**	**12.45**	5	10	**12.45**	**12.45**	12.45	12.45
tc2c10s3cf0	4	10	**21.77**	**21.77**	4	10	**21.77**	**21.77**	4	10	**21.77**	**21.77**	4	10	**21.77**	**21.77**	21.77	21.77
tc2c10s3ct0	6	10	**11.51**	**11.51**	6	10	**11.51**	**11.51**	7	10	**11.51**	**11.51**	6	10	**11.51**	**11.51**	11.51	11.51
tc0c20s3cf2	11	10	**27.47**	**27.47**	11	10	**27.47**	**27.47**	12	10	**27.47**	**27.47**	12	10	**27.47**	**27.47**	27.47	27.47
tc0c20s3ct2	8	10	**17.08**	**17.08**	8	10	**17.08**	**17.08**	8	10	**17.08**	**17.08**	7	10	**17.08**	**17.08**	17.08	17.08
tc0c20s4cf2	9	10	**27.47**	**27.47**	10	10	**27.47**	**27.47**	10	10	**27.47**	**27.47**	9	10	**27.47**	**27.47**	27.47	27.47
tc0c20s4ct2	11	10	**16.99**	**16.99**	11	10	**16.99**	**16.99**	11	10	**16.99**	**16.99**	10	10	**16.99**	**16.99**	16.99	16.99
tc1c20s3cf1	12	10	**17.49**	**17.49**	12	10	**17.49**	**17.49**	12	10	**17.49**	**17.49**	12	10	**17.49**	**17.49**	17.49	17.49
tc1c20s3cf3	10	10	**16.44**	**16.44**	9	10	**16.44**	**16.44**	9	10	**16.44**	**16.44**	9	10	**16.44**	**16.44**	16.44	16.44
tc1c20s3cf4	7	10	**17.00**	**17.00**	7	10	**17.00**	**17.00**	6	10	**17.00**	**17.00**	7	10	**17.00**	**17.00**	17.00	17.00
tc1c20s3ct1	11	10	**18.94**	**18.94**	11	10	**18.94**	**18.94**	11	10	**18.94**	**18.94**	10	10	**18.94**	**18.94**	18.94	18.94
tc1c20s3ct3	11	10	**12.60**	**12.60**	11	10	**12.60**	**12.60**	10	10	**12.60**	**12.60**	10	10	**12.60**	**12.60**	12.60	12.60
tc1c20s3ct4	9	10	**16.21**	**16.21**	9	10	**16.21**	**16.21**	10	10	**16.21**	**16.21**	10	10	**16.21**	**16.21**	16.21	16.21
tc1c20s4cf1	9	10	**16.38**	**16.38**	8	10	**16.38**	**16.38**	8	10	**16.38**	**16.38**	8	10	**16.38**	**16.38**	16.38	16.38
tc1c20s4cf3	13	10	**16.44**	**16.44**	13	10	**16.44**	**16.44**	13	10	**16.44**	**16.44**	13	10	**16.44**	**16.44**	16.44	16.44
tc1c20s4cf4	10	10	**17.00**	**17.00**	9	10	**17.00**	**17.00**	10	10	**17.00**	**17.00**	9	10	**17.00**	**17.00**	17.00	17.00
tc1c20s4ct1	12	10	**17.80**	**17.80**	13	10	**17.80**	**17.80**	13	10	**17.80**	**17.80**	13	10	**17.80**	**17.80**	17.80	17.80
tc1c20s4ct3	7	10	**14.43**	**14.43**	8	10	**14.43**	**14.43**	7	10	**14.43**	**14.43**	8	10	**14.43**	**14.43**	14.43	14.43
tc1c20s4ct4	8	10	**17.00**	**17.00**	8	10	**17.00**	**17.00**	8	10	**17.00**	**17.00**	8	10	**17.00**	**17.00**	17.00	17.00
tc2c20s3cf0	9	10	**24.68**	**24.68**	9	10	**24.68**	**24.68**	8	10	**24.68**	**24.68**	8	10	**24.68**	**24.68**	24.68	24.68
tc2c20s3ct0	12	10	**25.79**	**25.79**	12	10	**25.79**	**25.79**	12	10	**25.79**	**25.79**	11	10	**25.79**	**25.79**	25.79	25.79
tc2c20s4cf0	12	10	**24.67**	**24.67**	12	10	**24.67**	**24.67**	11	10	**24.67**	**24.67**	11	10	**24.67**	**24.67**	24.67	24.67
tc2c20s4ct0	12	10	**26.02**	**26.02**	12	10	**26.02**	**26.02**	11	10	**26.02**	**26.02**	11	10	**26.02**	**26.02**	26.02	26.02
tc0c40s5cf0	20	10	**32.20**	**32.23**	20	10	**32.20**	**32.23**	21	10	**32.20**	**32.23**	20	10	**32.20**	**32.23**	32.20	32.23
tc0c40s5cf4	26	10	**30.25**	**30.25**	26	10	**30.25**	**30.25**	25	10	**30.25**	**30.25**	26	10	**30.25**	**30.25**	30.25	30.25
tc0c40s5ct0	20	10	**27.91**	**27.91**	20	10	**27.91**	**27.91**	19	10	**27.91**	**27.91**	19	10	**27.91**	**27.91**	27.91	27.91
tc0c40s5ct4	23	10	**28.63**	**28.63**	23	10	**28.63**	**28.63**	23	10	**28.63**	**28.63**	22	10	**28.63**	**28.63**	28.63	28.63
tc0c40s8cf0	20	10	**30.40**	**30.40**	21	10	**30.40**	**30.40**	21	10	**30.40**	**30.40**	20	10	**30.40**	**30.40**	30.40	30.40
tc0c40s8cf4	28	10	**28.24**	**28.25**	28	10	**28.24**	**28.25**	29	10	**28.24**	**28.25**	29	10	**28.24**	**28.25**	28.24	28.25
tc0c40s8ct0	21	10	**26.22**	**26.22**	20	10	**26.22**	**26.22**	20	10	**26.22**	**26.22**	21	10	**26.22**	**26.22**	26.22	26.22
tc0c40s8ct4	26	10	**29.07**	**29.07**	26	10	**29.07**	**29.07**	26	10	**29.07**	**29.07**	26	10	**29.07**	**29.07**	29.07	29.07
tc1c40s5cf1	28	2	66.55	66.55	28	10	**65.38**	67.50	28	10	**65.38**	67.50	454	10	**65.38**	**66.51**	65.38	66.51
tc1c40s5ct1	27	10	**52.33**	**52.33**	27	10	**52.33**	**52.33**	26	10	**52.33**	**52.33**	26	10	**52.33**	**52.33**	52.33	52.33
tc1c40s8cf1	23	10	**40.64**	**40.64**	24	10	**40.64**	**40.64**	24	10	**40.64**	**40.64**	23	10	**40.64**	**40.64**	40.64	40.64
tc1c40s8ct1	28	10	**40.18**	**40.18**	27	10	**40.18**	**40.18**	27	10	**40.18**	**40.18**	28	10	**40.18**	**40.18**	40.18	40.18
tc2c40s5cf2	19	10	**27.54**	**27.54**	19	10	**27.54**	**27.54**	20	10	**27.54**	**27.54**	20	10	**27.54**	**27.54**	27.54	27.54
tc2c40s5cf3	25	10	**19.65**	**19.65**	25	10	**19.65**	**19.65**	25	10	**19.65**	**19.65**	26	10	**19.65**	**19.65**	19.65	19.65
tc2c40s5ct2	18	10	**26.91**	**26.91**	19	10	**26.91**	**26.91**	19	10	**26.91**	**26.91**	19	10	**26.91**	**26.91**	26.91	26.91
tc2c40s5ct3	22	10	**23.39**	**23.39**	22	10	**23.39**	**23.39**	23	10	**23.39**	**23.39**	22	10	**23.39**	**23.39**	23.39	23.39
tc2c40s8cf2	20	10	**27.13**	**27.13**	20	10	**27.13**	**27.13**	20	10	**27.13**	**27.13**	19	10	**27.13**	**27.13**	27.13	27.13
tc2c40s8cf3	25	10	**19.65**	**19.65**	26	10	**19.65**	**19.65**	25	10	**19.65**	**19.65**	25	10	**19.65**	**19.65**	19.65	19.65
tc2c40s8ct2	19	10	**26.28**	**26.28**	19	10	**26.28**	**26.28**	18	10	**26.28**	**26.28**	19	10	**26.28**	**26.28**	26.28	26.28
tc2c40s8ct3	29	10	**22.45**	**22.45**	28	10	**22.45**	**22.45**	29	10	**22.45**	**22.45**	29	10	**22.45**	**22.45**	22.45	22.45

Table 18: Detailed comparison of the best results obtained by the matheuristic for the E-VRP-NL-C according to the version of the subproblem it uses if the number of chargers at every CS is equal to 2 (instances with 80, 160, or 320 customers).

Instance	Version N				Version D				Version DW				Version DWR				BS	BA
	Time	#F	Best	Avg	Time	#F	Best	Avg	Time	#F	Best	Avg	Time	#F	Best	Avg		
tc0c80s12cf0	76	10	**34.06**	**34.20**	74	10	**34.06**	**34.20**	75	10	**34.06**	**34.20**	77	10	**34.06**	**34.20**	34.06	34.20
tc0c80s12cf1	77	10	**40.83**	**40.94**	77	10	**40.83**	**40.94**	77	10	**40.83**	**40.94**	76	10	**40.83**	**40.94**	40.83	40.94
tc0c80s12ct0	75	10	**37.51**	37.95	76	10	**37.51**	**37.89**	74	10	**37.51**	**37.89**	75	10	**37.51**	37.98	37.51	37.89
tc0c80s12ct1	71	10	**39.72**	**40.00**	70	10	**39.72**	**40.00**	72	10	**39.72**	**40.00**	71	10	**39.72**	**40.00**	39.72	40.00
tc0c80s8cf0	61	10	**38.59**	**38.84**	62	10	**38.59**	**38.84**	62	10	**38.59**	**38.84**	61	10	**38.59**	**38.84**	38.59	38.84
tc0c80s8cf1	83	10	**43.38**	**44.09**	82	10	**43.38**	44.12	85	10	**43.38**	44.12	87	10	**43.38**	44.10	43.38	44.09
tc0c80s8ct0	67	10	**40.53**	**41.47**	69	10	**40.53**	**41.47**	70	10	**40.53**	**41.47**	71	10	**40.53**	**41.47**	40.53	41.47
tc0c80s8ct1	78	10	**43.85**	**43.98**	80	10	**43.85**	**43.98**	79	10	**43.85**	**43.98**	81	10	**43.85**	**43.98**	43.85	43.98
tc1c80s12cf2	59	10	**28.65**	**28.77**	57	10	**28.65**	**28.77**	60	10	**28.65**	**28.77**	59	10	**28.65**	**28.77**	28.65	28.77
tc1c80s12ct2	60	10	**28.73**	29.17	62	10	**28.73**	**29.09**	63	10	**28.73**	**29.09**	62	10	**28.73**	29.14	28.73	29.09
tc1c80s8cf2	62	10	**29.15**	**29.15**	63	10	**29.15**	**29.15**	62	10	**29.15**	**29.15**	62	10	**29.15**	**29.15**	29.15	29.15
tc1c80s8ct2	64	10	**29.88**	**30.42**	64	10	**29.88**	**30.42**	64	10	**29.88**	**30.42**	62	10	**29.88**	**30.42**	29.88	30.42
tc2c80s12cf3	63	10	**30.60**	**30.60**	62	10	**30.60**	**30.60**	61	10	**30.60**	**30.60**	62	10	**30.60**	**30.60**	30.60	30.60
tc2c80s12cf4	87	10	42.12	42.20	89	10	**42.11**	**42.18**	88	10	**42.11**	**42.18**	88	10	**42.11**	**42.18**	42.11	42.18
tc2c80s12ct3	63	10	**29.90**	**29.90**	64	10	**29.90**	**29.90**	64	10	**29.90**	**29.90**	62	10	**29.90**	**29.90**	29.90	29.90
tc2c80s12ct4	88	10	**40.27**	**40.27**	86	10	**40.27**	**40.27**	88	10	**40.27**	**40.27**	85	10	**40.27**	**40.27**	40.27	40.27
tc2c80s8cf3	59	10	**31.70**	**31.93**	60	10	**31.70**	**31.93**	59	10	**31.70**	**31.93**	59	10	**31.70**	**31.93**	31.70	31.93
tc2c80s8cf4	101	10	46.18	46.65	101	10	**46.06**	**46.64**	103	10	**46.06**	**46.64**	102	10	46.18	46.65	46.06	46.64
tc2c80s8ct3	75	10	**31.38**	**31.38**	78	10	**31.38**	**31.38**	76	10	**31.38**	**31.38**	76	10	**31.38**	**31.38**	31.38	31.38
tc2c80s8ct4	87	10	**43.72**	**43.97**	85	10	**43.72**	**43.97**	84	10	**43.72**	**43.97**	87	10	**43.72**	**43.97**	43.72	43.97
tc0c160s16cf2	277	10	**57.91**	**58.01**	279	10	**57.91**	**58.01**	276	10	**57.91**	**58.01**	279	10	**57.91**	**58.01**	57.91	58.01
tc0c160s16cf4	416	10	77.04	**77.59**	418	10	77.04	77.62	424	10	77.04	77.62	427	10	**77.03**	77.60	77.03	77.59
tc0c160s16ct2	286	10	**57.16**	**57.63**	280	10	**57.16**	**57.63**	285	10	**57.16**	**57.63**	286	10	**57.16**	57.64	57.16	57.63
tc0c160s16ct4	403	10	76.66	76.93	395	10	76.37	**76.86**	408	10	76.37	**76.86**	399	10	**76.14**	76.87	76.14	76.86
tc0c160s24cf2	284	10	**56.86**	56.93	283	10	**56.86**	**56.92**	283	10	**56.86**	**56.92**	285	10	**56.86**	**56.92**	56.86	56.92
tc0c160s24cf4	432	10	**75.50**	**76.46**	415	10	76.00	76.61	410	10	76.00	76.61	430	10	75.59	76.48	75.50	76.46
tc0c160s24ct2	289	10	55.43	56.18	291	10	**55.42**	**56.16**	302	10	**55.42**	**56.16**	296	10	**55.42**	56.17	55.42	56.16
tc0c160s24ct4	418	10	**74.99**	76.16	428	10	75.06	**76.01**	429	10	75.06	**76.01**	436	10	**74.99**	76.19	74.99	76.01
tc1c160s16cf0	365	10	**74.55**	75.63	370	10	**74.55**	**75.51**	368	10	**74.55**	**75.51**	374	10	**74.55**	**75.51**	74.55	75.51
tc1c160s16cf3	348	10	**66.14**	**66.92**	352	10	66.32	67.02	351	10	66.32	67.02	344	10	66.32	66.98	66.14	66.92
tc1c160s16ct0	372	10	**74.19**	74.70	372	10	74.20	75.16	368	10	74.20	75.16	379	10	**74.19**	**74.66**	74.19	74.66
tc1c160s16ct3	307	10	**65.22**	**66.08**	309	10	66.03	66.42	311	10	66.03	66.42	319	10	65.26	66.27	65.22	66.08
tc1c160s24cf0	377	10	73.67	74.05	367	10	**73.62**	**73.85**	371	10	**73.62**	**73.85**	372	10	**73.62**	74.06	73.62	73.85
tc1c160s24cf3	323	10	**63.17**	63.86	334	10	63.22	63.83	325	10	63.22	63.83	333	10	**63.17**	**63.74**	63.17	63.74
tc1c160s24ct0	371	10	73.34	**73.81**	367	10	**73.32**	**73.81**	361	10	**73.32**	**73.81**	366	10	73.34	74.12	73.32	73.81
tc1c160s24ct3	330	10	**62.70**	63.28	326	10	**62.70**	**63.27**	337	10	**62.70**	**63.27**	336	10	**62.70**	63.32	62.70	63.27
tc2c160s16cf1	285	10	57.04	57.65	279	10	**57.01**	**57.62**	284	10	**57.01**	**57.62**	280	10	57.04	57.65	57.01	57.62
tc2c160s16ct1	260	10	**55.37**	**55.44**	258	10	**55.37**	**55.44**	262	10	**55.37**	**55.44**	258	10	**55.37**	55.46	55.37	55.44
tc2c160s24cf1	298	10	**56.00**	57.27	300	10	56.47	**57.18**	294	10	56.47	**57.18**	288	10	56.67	57.38	56.00	57.18
tc2c160s24ct1	281	10	55.03	55.16	291	10	55.03	55.14	283	10	55.03	55.14	274	10	**54.96**	**55.10**	54.96	55.10
tc1c320s24cf2	1561	10	133.66	134.28	1464	10	**132.45**	**133.57**	1458	10	**132.45**	**133.57**	1461	10	132.82	133.75	132.45	133.57
tc1c320s24cf3	1237	10	**105.29**	**106.53**	1233	10	105.47	106.60	1224	10	105.47	106.60	1264	10	106.21	106.98	105.29	106.53
tc1c320s24ct2	1601	10	132.16	133.09	1380	10	131.73	132.30	1379	10	131.73	132.30	1386	10	**131.69**	**132.28**	131.69	132.28
tc1c320s24ct3	1199	10	105.92	106.40	1142	10	**105.25**	**106.06**	1150	10	**105.25**	**106.06**	1156	10	105.37	106.27	105.25	106.06
tc1c320s38cf2	1359	10	129.01	129.40	1360	10	**128.87**	**129.21**	1375	10	**128.87**	**129.21**	1371	10	**128.87**	129.28	128.87	129.21
tc1c320s38cf3	1231	10	105.55	106.24	1243	10	**104.90**	**105.88**	1227	10	**104.90**	**105.88**	1253	10	105.72	106.24	104.90	105.88
tc1c320s38ct2	1395	10	128.84	129.44	1366	10	**128.71**	129.43	1349	10	**128.71**	129.43	1418	10	128.79	**129.34**	128.71	129.34
tc1c320s38ct3	1350	10	105.83	106.50	1385	10	105.39	106.64	1379	10	105.39	106.64	1339	10	**105.23**	**106.42**	105.23	106.42
tc2c320s24cf0	1880	10	160.36	161.73	1539	10	159.66	**160.55**	1555	10	159.66	**160.55**	1703	10	**158.96**	160.60	158.96	160.55
tc2c320s24cf1	1012	10	**87.26**	87.73	1032	10	87.35	87.71	1047	10	87.35	87.71	1039	10	87.31	**87.66**	87.26	87.66
tc2c320s24cf4	1065	10	110.98	111.52	1088	10	**110.58**	**111.12**	1074	10	**110.58**	**111.12**	1085	10	110.73	111.37	110.58	111.12
tc2c320s24ct0	1826	10	160.19	161.36	1505	10	**159.58**	160.22	1485	10	**159.58**	160.30	2184	10	159.62	**160.01**	159.58	160.01
tc2c320s24ct1	996	10	87.26	87.68	1030	10	**87.02**	**87.52**	1024	10	**87.02**	**87.52**	1014	10	87.28	87.70	87.02	87.52
tc2c320s24ct4	1175	10	111.07	111.63	1177	10	111.04	**111.54**	1155	10	111.04	**111.54**	1171	10	**110.99**	111.63	110.99	111.54
tc2c320s38cf0	1571	10	159.35	159.93	1588	10	**158.29**	**159.33**	1580	10	**158.29**	**159.33**	1589	10	158.44	159.48	158.29	159.33
tc2c320s38cf1	1008	10	86.88	87.22	1024	10	86.78	**87.19**	1015	10	86.78	**87.19**	1021	10	**86.77**	87.30	86.77	87.19
tc2c320s38cf4	1264	10	109.93	110.56	1264	10	109.94	**110.39**	1255	10	109.94	**110.39**	1246	10	**109.80**	110.57	109.80	110.39
tc2c320s38ct0	1782	9	158.59	159.53	1617	10	158.30	159.06	1580	10	158.30	159.06	1586	10	**157.81**	**158.88**	157.81	158.88
tc2c320s38ct1	1093	10	**85.90**	**86.81**	1038	10	86.52	86.95	1049	10	86.52	86.95	1074	10	86.26	86.85	85.90	86.81
tc2c320s38ct4	1204	10	**109.50**	**110.12**	1191	10	109.75	110.13	1164	10	109.75	110.13	1187	10	109.85	110.18	109.50	110.12

References

Alvarenga, G. B., Mateus, G., and De Tomi, G. (2007). A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows. *Computers & Operations Research*, 34(6):1561–1584.

Andelmin, J. and Bartolini, E. (2017). An exact algorithm for the green vehicle routing problem. *Transportation Science*, 51(4):1288–1303.

Andelmin, J. and Bartolini, E. (2019). A multi-start local search heuristic for the Green Vehicle Routing Problem based on a multigraph reformulation. *Computers & Operations Research*, 109:43–63.

Artigues, C., Michelon, P., and Reusser, S. (2003). Insertion techniques for static and dynamic resource-constrained project scheduling. *European Journal of Operational Research*, 149(2):249–267.

Baptiste, P., Le Pape, C., and Nuijten, W. (2001). *Constraint-based scheduling: applying constraint programming to scheduling problems*, volume 39. Springer, Boston, MA.

Baum, M., Dibbelt, J., Gemsa, A., Wagner, D., and Zündorf, T. (2019). Shortest Feasible Paths with Charging Stops for Battery Electric Vehicles. *Transportation Science*, 53(6):1627–1655.

Baum, M., Dibbelt, J., Wagner, D., and Zündorf, T. (2020). Modeling and engineering constrained shortest path algorithms for battery electric vehicles. *Transportation Science*, 54(6):1571–1600.

Beek, O., Raa, B., Dullaert, W., and Vigo, D. (2018). An efficient implementation of a static move descriptor-based local search heuristic. *Computers & Operations Research*, 94:1–10.

Brandstätter, G., Leitner, M., and Ljubić, I. (2020). Location of charging stations in electric car sharing systems. *Transportation Science*, 54(5):1408–1438.

Bruglieri, M., Mancini, S., Pezzella, F., and Pisacane, O. (2019a). A path-based solution approach for the Green Vehicle Routing Problem. *Computers & Operations Research*, 103:109–122.

Bruglieri, M., Mancini, S., and Pisacane, O. (2019b). The green vehicle routing problem with capacitated alternative fuel stations. *Computers & Operations Research*, 112:104759.

Codato, G. and Fischetti, M. (2006). Combinatorial Benders' cuts for mixed-integer linear programming. *Operations Research*, 54(4):756–766.

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

Desaulniers, G., Gschwind, T., and Irnich, S. (2020). Variable Fixing for Two-Arc Sequences in Branch-Price-and-Cut Algorithms on Path-Based Models. *Transportation Science*, 54(5):1170–1188.

Drexl, M. (2012). Synchronization in vehicle routing – a survey of vrps with multiple synchronization constraints. *Transportation Science*, 46(3):297–316.

El Hachemi, N., Gendreau, M., and Rousseau, L.-M. (2013). A heuristic to solve the synchronized log-truck scheduling problem. *Computers & Operations Research*, 40(3):666–673.

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

Felipe, A., Ortuño, M. T., Righini, G., and 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.

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

Gnann, T., Funke, S., Jakobsson, N., Plötz, P., Sprei, F., and Bennehag, A. (2018). Fast charging infrastructure for electric vehicles: Today's situation and future needs. *Transportation Research Part D: Transport and Environment*, 62:314–329.

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

Grangier, P., Gendreau, M., Lehuédé, F., and Rousseau, L.-M. (2017). A matheuristic based on large neighborhood search for the vehicle routing problem with cross-docking. *Computers & Operations Research*, 84:116–126.

Grimault, A., Bostel, N., and Lehuédé, F. (2017). An adaptive large neighborhood search for the full truckload pickup and delivery problem with resource synchronization. *Computers & Operations Research*, 88:1–14.

Hempsch, C. and Irnich, S. (2008). Vehicle routing problems with inter-tour resource constraints. In Golden, B., Raghavan, S., and Wasil, E., editors, *The Vehicle Routing Problem: Latest Advances and New Challenges*, pages 421–444. Springer, Boston, MA.

Hiermann, G., Hartl, R. F., Puchinger, J., and Vidal, T. (2019). Routing a mix of conventional, plug-in hybrid, and electric vehicles. *European Journal of Operational Research*, 272(1):235–248.

Hiermann, G., Puchinger, J., Ropke, S., and 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.

Juan, A. A., Mendez, C. A., Faulin, J., de Armas, J., and Grasman, S. E. (2016). Electric vehicles in logistics and transportation: A survey on emerging environmental, strategic, and operational challenges. *Energies*, 9(2):86.

Keskin, M. and Çatay, B. (2018). A matheuristic method for the electric vehicle routing problem with time windows and fast chargers. *Computers & Operations Research*, 100:172–188.

Keskin, M. and Çatay, B. (2016). Partial recharge strategies for the electric vehicle routing problem with time windows. *Transportation Research Part C: Emerging Technologies*, 65:111–127.

Keskin, M., Laporte, G., and Çatay, B. (2019). Electric vehicle routing problem with time-dependent waiting times at recharging stations. *Computers & Operations Research*, 107:77–94.

Koç, Ç. and Karaoglan, I. (2016). The green vehicle routing problem: A heuristic based exact solution approach. *Applied Soft Computing*, 39:154–164.

Koç, Ç., Jabali, O., Mendoza, J. E., and Laporte, G. (2018). The electric vehicle routing problem with shared charging stations. *International Transactions in Operational Research*, 26(4):1211–1243.

Kullman, N., Goodson, J., and Mendoza, J. E. (2020). Electric Vehicle Routing with Public Charging Stations. Working paper available at `https://hal.archives-ouvertes.fr/hal-01928730` (accepted for publication in Transportation Science).

Lam, E. and Van Hentenryck, P. (2016). A branch-and-price-and-check model for the vehicle routing problem with location congestion. *Constraints*, 21(3):1–19.

Lee, C. (2020). An exact algorithm for the electric-vehicle routing problem with nonlinear charging time. *Journal of the Operational Research Society*, page doi: `10.1080/01605682.2020.1730250`.

Leggieri, V. and Haouari, M. (2017). A practical solution approach for the green vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review*, 104:97–112.

Lourenço, H. R., Martin, O. C., and Stützle, T. (2003). Iterated local search. In Glover, F. and Kochenberger, G. A., editors, *Handbook of metaheuristics*, volume 57, pages 320–353. Springer, Boston, MA.

Macrina, G., Laporte, G., Guerriero, F., and Di Puglia Pugliese, L. (2019). 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., and Villegas, J. (2016). A multi-space sampling heuristic for the green vehicle routing problem. *Transportation Research Part C: Emerging Technologies*, 70:113–128.

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

Morais, V. W. C., Mateus, G. R., and Noronha, T. F. (2014). Iterated local search heuristics for the vehicle routing problem with cross-docking. *Expert Systems with Applications*, 41(16):7495–7506.

Pelletier, S., Jabali, O., and Laporte, G. (2016). 50th anniversary invited article—goods distribution with electric vehicles: Review and research perspectives. *Transportation Science*, 50(1):3–22.

Pelletier, S., Jabali, O., and Laporte, G. (2018). Charge scheduling for electric freight vehicles. *Transportation Research Part B: Methodological*, 115:246–269.

Pelletier, S., Jabali, O., Laporte, G., and Veneroni, M. (2017). Battery degradation and behaviour for electric vehicles: Review and numerical analyses of several models. *Transportation Research Part B: Methodological*, 103:158–187.

Rix, G., Rousseau, L.-M., and Pesant, G. (2015). A column generation algorithm for tactical timber transportation planning. *Journal of the Operational Research Society*, 66(2):278–287.

Roberti, R. and Wen, M. (2016). The electric traveling salesman problem with time windows. *Transportation Research Part E: Logistics and Transportation Review*, 89:32–52.

Sassi, O. and Oulamara, A. (2014). Joint scheduling and optimal charging of electric vehicles problem. In Murgante, B., Misra, S., Rocha, A. M. A. C., Torre, C., Rocha, J. G., Falcão, M. I., Taniar, D., Apduhan, B. O., and Gervasi, O., editors, *Computational Science and Its Applications – ICCSA 2014*, volume 8580, pages 76–91, Cham. Springer.

Schiffer, M., Schneider, M., and Laporte, G. (2018a). Designing sustainable mid-haul logistics networks with intra-route multi-resource facilities. *European Journal of Operational Research*, 265(2):517–532.

Schiffer, M., Stütz, S., and Walther, G. (2018b). Electric commercial vehicles in mid-haul logistics networks. In Pistoia, G. and Liaw, B., editors, *Behaviour of Lithium-Ion Batteries in Electric Vehicles*, pages 153–173. Springer, Cham.

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

Schneider, M., Stenger, A., and Hof, J. (2015). An adaptive VNS algorithm for vehicle routing problems with intermediate stops. *OR Spectrum*, 37(2):353–387.

Subramanian, A., Uchoa, E., and Ochi, L. S. (2013). A hybrid algorithm for a class of vehicle routing problems. *Computers & Operations Research*, 40(10):2519–2531.

Villegas, J. G., Guéret, C., Mendoza, J. E., and Montoya, A. (2018). The technician routing and scheduling problem with conventional and electric vehicle. Working paper available at `https://hal.archives-ouvertes.fr/hal-01813887`.

Villegas, J. G., Prins, C., Prodhon, C., Medaglia, A. L., and Velasco, N. (2013). A matheuristic for the truck and trailer routing problem. *European Journal of Operational Research*, 230(2):231–244.

Zachariadis, E. E. and Kiranoudis, C. T. (2010). A strategy for reducing the computational complexity of local search-based methods for the vehicle routing problem. *Computers & Operations Research*, 37(12):2089–2105.