Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

Revisiting Large Neighborhood Search With On-The-Fly Charging Station Insertion for the Electric Autonomous Dial-A-Ride Problem

Maria Bresich, Guenther Raidl, Steffen Limmer

2025

Preprint:

Copyright ACM 2025. This is the author's version of the work. It is posted here for your personal use. Not for redistribution. The definitive Version of Record was published in ACM Transactions on Evolutionary Learning and Optimization, https://doi.org/10.1145/3779426.

Revisiting Large Neighborhood Search with On-the-Fly Charging Station Insertion for the Electric Autonomous Dial-a-Ride Problem

MARIA BRESICH*, Institute of Logic and Computation, TU Wien, Austria
GÜNTHER R. RAIDL, Institute of Logic and Computation, TU Wien, Austria
STEFFEN LIMMER, Honda Research Institute Europe GmbH, Germany

We address the electric autonomous dial-a-ride problem (E-ADARP), a challenging extension of the dial-a-ride problem with the goal of finding minimum cost routes serving given transportation requests with a fleet of electric and autonomous vehicles (EAVs). Special emphasis lies on the minimization of user excess ride time under consideration of the charging requirements of the EAVs while operational constraints have to be satisfied. We propose a novel large neighborhood search (LNS) approach for the E-ADARP together with two alternatives for handling the charging of the EAVs, the scheduling, and route evaluation. One deals with these challenges separately using dedicated LNS operators and a forward labeling algorithm and the other provides a combined approach with on-the-fly charging stop insertion during route evaluation. The performance of the algorithms is evaluated on various configurations of two common benchmark sets as well as some very large-scale instances. Results show that especially the approach with the on-the-fly insertion almost consistently outperforms former state-of-the-art techniques on the common benchmark instances, finding many new best known solutions. For this best performing approach, multiple variants of a more advanced destroy operator for the underlying LNS are investigated. This enhancement can yield significantly improved performance especially on the very large instances as illustrated by further empirical results.

CCS Concepts: • **Theory of computation** → *Random search heuristics*; **Randomized local search**; • **Mathematics of computing** → **Combinatorial optimization**; • **Applied computing** → *Transportation.*

Additional Key Words and Phrases: Dial-a-Ride Problem, Electric Autonomous Vehicles, Large Neighborhood Search

ACM Reference Format:
Maria Bresich, Günther R. Raidl, and Steffen Limmer. 2025. Revisiting Large Neighborhood Search with On-the-Fly Charging Station Insertion for the Electric Autonomous Dial-a-Ride Problem. 1, 1 (November 2025), 31 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Increasing mobility and transportation demands especially in urban areas lead to a growing interest in on-demand transportation as well as ride-sharing services [Shaheen and Cohen 2020]. Such services are gaining popularity as they provide more flexibility than public transportation with its fixed schedules, and sharing rides with other users usually decreases the costs and environmental

*We acknowledge the financial support from Honda Research Institute Europe GmbH.

Authors' Contact Information: Maria Bresich, Institute of Logic and Computation, TU Wien, Vienna, Austria, mbresich@ac.tuwien.ac.at; Günther R. Raidl, Institute of Logic and Computation, TU Wien, Vienna, Austria, raidl@ac.tuwien.ac.at; Steffen Limmer, Honda Research Institute Europe GmbH, Offenbach/Main, Germany, steffen.limmer@honda-ri.de.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM XXXX-XXXX/2025/11-ART
https://doi.org/10.1145/nnnnnnn.nnnnnnn

impact compared to privately owned cars and classic taxi services. They fall into the class of dial-a-ride (DAR) systems, which have been around since 1970 [Oxley 1980].

The dial-a-ride problem (DARP) is concerned with the planning and scheduling of such DAR services where a fleet of vehicles is employed to fulfill transportation requests of users. Each request specifies a pickup and a drop-off location as well as the desired time window, when service should be provided at one of these locations. The standard DARP as defined by Cordeau and Laporte [2003] aims at finding minimum cost routes serving all requests while satisfying a set of constraints. Typical constraints consider operational aspects including time windows, vehicle capacities, and maximum route durations, as well as user experience by establishing a maximum user ride time. The literature on DARPs encompasses diverse other objectives and constraints, and comprehensive reviews are given by Molenbruch et al. [2017b] and Ho et al. [2018].

These reviews also provide excellent overviews of established solving approaches, most of which can be classified as either exact methods or (meta-)heuristics. Exact algorithms often make use of mixed integer linear programming (MILP) techniques like branch-and-cut [Braekers et al. 2014; Braekers and Kovacs 2016; Cordeau 2006; Parragh 2011; Ropke et al. 2007], branch-and-price [Garaix et al. 2011; Parragh et al. 2012; Parragh and Schmid 2013; Ropke and Cordeau 2009], or branch-and-price-and-cut [Gschwind and Irnich 2015; Qu and Bard 2015]. Unfortunately, these exact methods often do not scale well enough in practice and therefore diverse (meta-)heuristics have been proposed for DARPs, including tabu search [Cordeau and Laporte 2003], deterministic annealing (DA) [Braekers et al. 2014], large neighborhood search (LNS) [Braekers and Kovacs 2016; Molenbruch et al. 2017a], and genetic algorithms [Cubillos et al. 2009; Jorgensen et al. 2007].

Aside from ride-sharing services, another approach to environment-friendlier mobility is the employment of electric vehicles (EVs), whose adoption is quickly progressing. Additionally, the development and introduction of self-driving cars, or autonomous vehicles (AVs), are advancing and offer new opportunities. As AVs do not require human drivers, they provide greater flexibility and more operating time without having to return to a depot. Thus, the provision of ride-sharing services by means of electric autonomous vehicles (EAVs) is a highly promising advancement [Narayanan et al. 2020]. The *electric autonomous dial-a-ride problem* (E-ADARP) as introduced by Bongiovanni et al. [2019] considers EAVs within the DARP, which requires to additionally deal with the battery management. Employing EVs for a DAR service requires to plan in recharging of the vehicles, which affects the routing as well as the scheduling. Detours to charging stations and the vehicles' battery levels have to be taken into account when planning the routes, and charging durations must be scheduled in a way that prevents battery depletion as well as time window violations. In the E-ADARP, (partial) recharging en route is allowed whenever no passenger is on board, and a minimum battery level, or state of charge (SOC), for the end of a route is specified. These and other additional constraints further increase the practical complexity of the standard DARP, which is already NP-hard, and call for new solution approaches.

This article is an extended version of our conference paper [Bresich et al. 2024] on a large neighborhood search for the E-ADARP. We propose two approaches to deal with the main challenges which are the charging of the vehicles as well as the scheduling and evaluation of routes. The first approach handles these concepts individually by employing dedicated LNS operators to deal with charging stops together with a modified version of the labeling algorithm by Su et al. [2023] for an efficient evaluation. For the second approach, we introduce a novel route evaluation procedure that inserts charging stops on-the-fly as needed while also determining the respective charging durations. We formulate the underlying charging station insertion subproblem first as a MILP, before proposing a highly effective time-efficient heuristic, which both can be used irrespective of the employed metaheuristic. The most important original features that distinguish our work from the literature are (a) a structurally rather simple LNS framework with only few destroy and repair

operators, (b) the consideration of battery-restricted fragments and their caching for efficient route evaluation, and (c) the highly effective on-the-fly insertion of charging stops. Results are compared to exact and heuristic approaches from the literature [Bongiovanni et al. 2019; Limmer 2023; Su et al. 2024, 2023]. These show that especially the approach with the heuristic on-the-fly insertion of charging stops almost consistently outperforms all former methods on the common benchmark instances, matching or finding new best known solutions on 97 out of 144 instances in total.

This work extends our original conference paper [Bresich et al. 2024] in several ways. (a) In addition to a simple uniform random destroy within the LNS, we consider new destroy operator variants that preferably select requests inducing high costs in the objective function. As a detailed experimental comparison shows, these destroy operators can significantly improve the performance of the LNS, especially on very large instances. (b) We perform computational experiments not only on various additional configurations of two sets of common benchmark instances but also consider instances that are magnitudes larger and come closer to real-world scenarios. (c) Experimental results are augmented with rigorous statistical tests, which clearly illustrate the superiority of some configurations over others.

To sum up, the main contributions of this work are as follows:

- On-demand evaluation and caching only of relevant battery-restricted fragments for efficient route evaluation.
- Development of a large neighborhood search for the E-ADARP using a combination of dedicated operators for customer requests as well as charging station visits, and modification of a forward labeling algorithm for efficient route evaluation. For the latter, we propose a fragment-based route representation for increased performance and resolve the issue of incorrect infeasibility declarations of the original algorithm.
- Development of a highly effective on-the-fly (OTF) insertion approach for charging stops during route evaluation, removing the need for dedicated operators and making the procedure independent of the metaheuristic framework. We first introduce a MILP formulation for the charging station insertion and evaluation subproblem and then propose a time-efficient heuristic.
- Investigation of more advanced and targeted destroy operator variants for the E-ADARP to better guide the search process.
- Extensive experiments and evaluation of the proposed approaches on various configurations of common and very large-scale instances from the literature and comparison to existing approaches. On the common benchmark instances, the heuristic OTF approach mostly outperforms all other heuristics and constitutes the new state-of-the-art.

The remainder of this article is organized as follows. The next section reviews related work. A concrete definition of the E-ADARP is provided in Section 3. Section 4 details the LNS framework including the newly adopted destroy operator and its variants, and our two proposed approaches for charging stop insertion and route evaluation are covered in depth in Section 5. Section 6 describes the experiments and discusses the results. Concluding remarks and an outlook on promising future research lines are given in Section 7.

2 Related Work

So far only few solving approaches have been proposed for the E-ADARP as considered here, which we discuss in the following.

In 2019, Bongiovanni et al. [2019] introduced the E-ADARP and model it by three- and two-indexed MILP formulations. The latter is more efficient but less expressive as it cannot handle vehicle-dependent battery consumptions and recharge rates. They further strengthen the models with new

valid inequalities and solve them in a branch-and-cut algorithm. A performance evaluation on two sets of new benchmark instances indicates that the two-indexed formulation mostly outperforms the three-indexed one.

In [Bongiovanni et al. 2024], the same authors propose a scheduling algorithm which aims to minimize user ride time and can be applied to DARPs in general. To deal with charging decisions for EVs as used in the E-ADARP, they also introduce a battery management heuristic.

Su et al. [2023] proposed a *deterministic annealing* (DA) metaheuristic for the (static) E-ADARP. It employs various local search operators for intra- and inter-route modification. The authors also present the first exact user excess ride time optimization involving a route evaluation scheme of linear time complexity that is based on a forward labeling algorithm. They introduce the notion of *battery-restricted fragments* for a new path representation of E-ADARP routes, which allows for minimization of user excess ride time. Each fragment represents a subsequence of pickup and drop-off locations in between which no charging is possible; for these the user excess ride time can be independently optimized by solving a linear program. To enable an efficient route evaluation, all feasible fragments are enumerated and evaluated in a preprocessing phase. The performance of the DA algorithm is tested on three sets of benchmark instances, two of which come from [Bongiovanni et al. 2019]. Compared to the MILP-based approaches from [Bongiovanni et al. 2019], the DA algorithm performs better on several instances.

Su et al. [2024] further build upon the concept of battery-restricted fragments and introduce a path-based MILP formulation of the E-ADARP. A problem-tailored forward labeling algorithm is used for column generation within a branch-and-price (B&P) algorithm for solving the problem in an exact manner. This approach outperforms the exact methods from [Bongiovanni et al. 2019] on the instances used in that earlier work and improves solutions over the DA algorithm on the larger instances from [Su et al. 2023].

Limmer [2023] proposed a *bilevel large neighborhood search* (BI-LNS). In this approach, an outer level employs variation operators to only schedule the charging of the EVs. In the inner level, the charging stops are considered fixed, and destroy and repair operators are used for optimizing the insertion of pickup and drop-off stops for fulfilling user requests. While the fixation of charging stops in the outer level may appear rather crude and naive, the big advantage of this approach is that it enables a fast and simple realization of the inner level operations. The actual route evaluation can be done in a classical way based on forward time slacks. This improves the overall scalability in comparison to more expensive operators as used, for example, in [Su et al. 2023]. Experimental results show that BI-LNS is competitive on the smaller instances from [Bongiovanni et al. 2019] and outperforms DA on the larger instances from [Su et al. 2023]. Moreover, also some new very large instances with hundreds of vehicles and thousands of requests are considered, illustrating the excellent scalability of the approach.

Besides our conference paper [Bresich et al. 2024] of which the current work is an extended version, the short paper [Bresich et al. 2025] gives a rough overview of the LNS with heuristic on-the-fly charging stop insertion together with a strongly restricted evaluation.

Aside from the static E-ADARP, some works also consider the dynamic problem variant where not all customer requests are known from the start and new ones arrive over time. Bongiovanni et al. [2022] tackle this dynamic version of the E-ADARP by sequentially solving static E-ADARP subproblems with a two-phase metaheuristic. In the first phase, a greedy insertion heuristic assigns requests to vehicles and vehicle schedules are computed. A machine learning-based large neighborhood search is employed in the second phase for re-optimization where routes from the first phase are modified and improved by means of destroy and repair operators, which are selected by an offline trained machine learning model. Huang et al. [2025] approach a similar problem variant by means of a genetic programming hyper-heuristic (GP-HH) that learns policies for vehicle

dispatching and charging decisions in an offline fashion. The resulting best performing policies are then employed to make real-time decisions in an event-driven simulation of the dynamic setting. The same problem is investigated by Tomandl et al. [2025] who utilize large neighborhood search to repeatedly solve static E-ADARP subproblems whenever new requests arrive. The authors propose to employ reinforcement learning to learn policies for charging and vehicle dispatching decisions in an offline manner which are then applied to guide the LNS in the dynamic scenario. This approach is shown to scale better and outperform the earlier GP-HH.

3 Formal Definitions

This section first specifies the E-ADARP following the original definition from [Bongiovanni et al. 2019]. Then, the concept of battery-restricted fragments and how they are used for user excess ride time optimization of routes is explained.

The E-ADARP is defined on a complete directed graph $G = (V, A)$ with vertex set V that represents all relevant geographic locations and arc set $A = \{(i, j) : i, j \in V, i \neq j\}$. A fleet of n_K EAVs, denoted by $K = \{1, \ldots, n_k\}$, is considered for providing service for n customer requests within a planning horizon of T^{plan} time units. The set $V = N \cup O \cup F \cup S$ is composed of multiple subsets, each denoting a different type of location. Let $N = P \cup D$ be the set of all customer locations, where $P = \{1, \ldots, n\}$ denotes the pickup locations and $D = \{n + 1, \ldots, 2n\}$ denotes the drop-off locations. Thus, a pair $(i, i + n)$ represents the i-th request with pickup location $i \in P$ and drop-off location $i + n \in D$; moreover, a maximum user ride time u_i is given for each request $i \in P$. As in [Limmer 2023], each vehicle $k \in K$ is associated with an origin depot location $o_k \in O$ and a destination depot location $f_k \in F$, with $|O| = |F| = n_K$. Note that [Bongiovanni et al. 2019] allows vehicles to select from multiple destination depots. However, on the benchmark instances used here including those adopted from [Bongiovanni et al. 2019], this makes no difference since all of them actually consider only a unique destination depot location. Finally, the set S denotes the available charging stations (CSs), and each station $s \in S$ has a charging rate α_s, which specifies the amount of energy charged per time unit. Vehicles are allowed to either partially or fully recharge when visiting a charging station.

Each vertex $i \in V$ is associated with a time window $[w_i^{\text{start}}, w_i^{\text{end}}]$ with w_i^{start} being the earliest possible and w_i^{end} being the latest possible service start time for request locations $i \in N$. For all other locations $i \in V \setminus N$, time windows represent opening hours within which they can be visited. Service at a pickup or drop-off location $i \in N$ is assumed to take a certain duration $d_i \geq 0$, whereas $d_i = 0$ for other locations $i \in V \setminus N$. When a vehicle serves a location $i \in V$, its load changes by an amount l_i depending on the number of customers to be picked up or dropped off; i.e., $l_i > 0$ at pickup locations $i \in P$, $l_i = -l_{i-n}$ at drop-off locations $i \in D$, and $l_i = 0$ at all other locations $i \in V \setminus N$. Vehicles $k \in K$ are heterogeneous in terms of their maximum (load) capacities C_k and homogeneous in terms of their battery capacities Q. The initial battery level of a vehicle $k \in K$ at its origin depot is given as $B_{k,1}$. A minimum end battery level ratio $\gamma \in [0, 1]$ determines the minimum battery level γQ that a vehicle needs to satisfy when arriving at its destination depot f_k.

Each arc $(i, j) \in A$ is associated with a travel time $t_{i,j}$ and a battery consumption $\beta_{i,j}$. We assume that the triangle inequality holds for travel times as well as battery consumptions.

A path in graph G starting at the respective origin depot, potentially passing through pickup, drop-off, and charging station locations, and ending at a destination depot is called the *route* of a vehicle. An E-ADARP solution is then composed of n_K vehicle routes together with their respective schedules, specifying the service start time t_i^{serv} at each location/node i of the route.

3.1 Further Constraints

A solution of the E-ADARP is feasible if its routes satisfy the following further constraints.

- All requests are served exactly once, meaning that there is for each request $i = 1, \ldots, n$ one route that visits pickup location i and drop-off location $i + n$ in this order.
- The maximum EAV capacity C_k is not exceeded at any time.
- The maximum user ride time u_i of each request is respected.
- Service starts at each location $i \in V$ within its time window $[w_i^{\text{start}}, w_i^{\text{end}}]$. While it is not allowed to arrive later than w_i^{end}, vehicles may arrive earlier than w_i^{start}, which induces some waiting time.
- Each vehicle k starts its route with the initial battery levels $B_{k,1}$, may never run out of energy or be charged beyond its battery capacity, and has to meet the minimum required battery level γQ when reaching its destination depot.
- Only EAVs not carrying a load may visit charging stations.
- Over all routes, each charging station can only be visited at most once. Note that this constraint can also be relaxed or removed to allow multiple visits of each CS, as considered in [Bongiovanni et al. 2019; Limmer 2023; Su et al. 2024, 2023].

In contrast to the standard DARP from [Cordeau and Laporte 2003], there are no constraints limiting the total duration of routes as there are no human drivers when employing AVs.

3.2 Objective

The goal of the E-ADARP is to find a feasible solution that minimizes as costs a weighted sum of the total travel time over all routes and the total user excess ride time over all requests $i \in P$, i.e.,

$$\min w^{\text{routing}} \sum_{k \in K} \sum_{(i,j) \in A} t_{i,j} x_{i,j}^k + w^{\text{excess}} \sum_{i \in P} t_i^{\text{excess}} \tag{1}$$

where w^{routing} and w^{excess} are the weight factors for the two objectives and $x_{i,j}^k$ is a binary decision variable indicating whether vehicle k travels from location i directly to j. The user excess ride time t_i^{excess} of request $i \in P$ is calculated as the difference between the actual ride time of i and the direct travel time $t_{i,i+n}$ from pickup to drop-off, i.e., $t_i^{\text{excess}} = t_{i+n}^{\text{serv}} - t_i^{\text{serv}} - d_i - t_{i,i+n}$.

3.3 Battery-Restricted Fragments

The consideration of user excess ride time in the objective function is an important additional feature of the E-ADARP compared to the standard DARP. Appropriately dealing with this aspect in the scheduling and evaluation of routes is thus crucial. To this end, Su et al. [2023] introduce the concept of a battery-restricted fragment, which is an extension of the fragments from Rist and Forbes [2021] for the DARP, by also taking battery-related matters into account. A battery-restricted fragment $\mathcal{F}$ is defined as a subsequence of a feasible route consisting only of pickup and drop-off locations such that the vehicle's load is zero on arrival at the start and on departure at the end location and positive at all other stops. A route can then be represented as a combination of battery-restricted fragments and charging stations, embedded within the origin and destination depots. Let $P_{\mathcal{F}}$ be the set of pickup locations in a battery-restricted fragment $\mathcal{F}$, the user excess

ride time of $\mathcal{F}$ can be minimized by solving the following linear program (LP)

$$\min \sum_{i \in P_{\mathcal{F}}} (t_{i+n}^{\text{serv}} - t_i^{\text{serv}} - d_i - t_{i,i+n}) \tag{2}$$

s.t.

$$t_i^{\text{serv}} + t_{i,j} + d_i \leq t_j^{\text{serv}} \qquad \forall i \in \mathcal{F},\ idx_i \neq |\mathcal{F}|,\ idx_j = idx_i + 1 \tag{3}$$

$$t_{i+n}^{\text{serv}} - t_i^{\text{serv}} - d_i \leq u_i \qquad \forall i \in P_{\mathcal{F}} \tag{4}$$

$$w_i^{\text{start}} \leq t_i^{\text{serv}} \leq w_i^{\text{end}} \qquad \forall i \in \mathcal{F} \tag{5}$$

with idx_i denoting the index of location i in $\mathcal{F}$. The minimum user excess ride time of a route is then equal to the sum of minimum user excess ride times of all contained battery-restricted fragments (simply called "fragments" from now on) [Su et al. 2023]. In [Su et al. 2023], this property is exploited by enumerating and evaluating all feasible fragments in a preprocessing phase and then using the precalculated user excess ride times during route evaluation when needed.

This approach, however, may be inefficient for instances with a large number of requests, as possibly also fragments are evaluated and stored that never appear in any candidate solution of the metaheuristic. For very large instances, this preprocessing may also already be too time-consuming or too memory demanding.

In our work, we refrain from initially enumerating all possible fragments. Instead, we evaluate fragments only at their first encounter during the LNS and store them with their minimum user excess ride times in a cache realized by a hash table. For the benchmark instances used in our experiments, the memory consumption of this cache never exceeded 3 GB. Thus, it was not necessary to discard least recently used entries, which would be the solution when some memory limit is reached when dealing with larger instances.

4 Large Neighborhood Search Framework

In this work, we propose two variants of an LNS to solve the E-ADARP. Large neighborhood search is a popular category of local search metaheuristics, and our LNS framework follows a rather simple and classical destroy and repair scheme similar to [Pisinger and Ropke 2010; Shaw 1998] as outlined in Algorithm 1. Starting with an initial solution x (cf. Section 4.1), the key idea is to consider much larger neighborhoods than in classical local search, which cannot be efficiently searched in their entirety and are only implicitly defined by a pair of a destroy and a repair operator. Generally, a destroy operator determines which of the problem's decision variables are to be re-optimized while leaving the remaining variables fixed to their values in the incumbent solution. The repair method then tries to restore a valid solution by heuristically solving the induced subproblem. In the standard DARP, this is realized by removing some requests with their pickup and drop-off stops from the tours in the destroy operation and then reinserting them again by some heuristics as repair. Some popular operators for removal and insertion are random removal, worst removal, greedy insertion, and regret insertion [Braekers and Kovacs 2016; Ropke and Pisinger 2006]. In case of multiple available operators, one pair is chosen in each LNS iteration according to some selection mechanism. Using probabilistic selection with dynamically adjusted operator weights extends the LNS framework to an adaptive large neighborhood search [Ropke and Pisinger 2006]. Applying the selected destroy and repair operators results in a single candidate solution and the update of the current solution is done according to the employed acceptance criterion. Commonly used criteria include the acceptance of only strictly better solutions or also accepting worse solutions as in simulated annealing with the Metropolis criterion [Kirkpatrick et al. 1983]. Such LNS approaches have been successfully applied not only to the DARP [Braekers and Kovacs 2016; Masmoudi et al. 2016; Molenbruch et al. 2017a] and the E-ADARP [Limmer 2023] but also to a multitude of

Algorithm 1: Large Neighborhood Search (LNS)

Input: Requests $reqs$
Output: Best found solution x^b

```
1  x ← initial_solution(reqs);
2  x^b ← x;
3  while stop criterion not met do
4      remove_requests(), insert_requests() ← select_operators();
5      x' ← remove_requests(x);
6      x'' ← insert_requests(x');
7      if accept(x'', x) then
8          x ← x'';
9      if objective(x'') < objective(x^b) then
10         x^b ← x'';
11 return x^b;
```

other challenging combinatorial optimization problems including routing, scheduling, and location problems [Jatschka et al. 2022; Ma et al. 2021; Shaw 1998].

In contrast to many former LNS approaches, we utilize only few destroy and repair operators and select the operators to be applied simply uniformly at random without any adaption of selection probabilities. We also only accept non-deteriorating solutions as new incumbents. Our main focus lies on the handling of some intricacies of the E-ADARP regarding the charging requirements of EVs and an efficient route evaluation, for which we propose two approaches as described in Section 5. In the following, we cover the main components of the LNS framework, starting with the generation of an initial solution and our acceptance criterion and then detailing the employed destroy and repair operators for requests.

4.1 Initial Solution Construction

An initial solution is constructed by starting for each vehicle with an empty route leading from the vehicle's origin depot directly to its destination depot. If our direct approach (Section 5.1) is employed, an available charging station nearest to the respective origin depot is next inserted into each route. Then, we try to insert all customer requests into the routes by the time window order based repair operator described below in Section 4.4. As this is a heuristic, not all requests can always be inserted, i.e., the resulting solution may be infeasible in this respect. The following LNS tries to resolve this and come up with a solution covering all requests as quickly as possible.

4.2 Acceptance Criterion and Handling of Infeasibility

We only accept a candidate solution as the new current incumbent solution if it is better or equally good in terms of its objective value, with the latter case allowing for further exploration during the search process. While in general, we only consider feasible solutions during the LNS by assuming the objective value of infeasible solutions to be a very high value, we might have to deal with infeasible solutions that do not serve all customer requests but satisfy all other constraints at the beginning of the LNS as described before in Section 4.1. Our employed acceptance criterion is thus as outlined in Algorithm 2. Among two infeasible solutions, one that satisfies more or equally many requests is considered better, and a feasible solution serving all requests is always accepted over infeasible ones. Hence, once feasibility is reached, no solutions with any constraint violations are

Algorithm 2: Acceptance function (accept)

Input: Candidate solution x'', incumbent x

if x is infeasible, i.e., not all requests fulfilled, **then**
  return num_fulfilled_requests(x'') ≥ num_fulfilled_requests(x)
else if x'' is infeasible, i.e., not all requests fulfilled, **then**
  return false
else
  return objective(x'') ≤ objective(x)

accepted anymore. Two feasible solutions are compared solely based on their objective values as defined in Expression (1).

4.3 Destroy Operators for Requests

For the deletion of customer requests from routes, we first employ the popular *random removal* (RR) [Ropke and Pisinger 2006] where κ^{rr} served requests are selected uniformly at random from the routes and are removed, with κ^{rr} being the degree of destruction. While this method is efficient, it does not consider any request or route characteristics and is therefore in general not as effective as other approaches. In the context of other VRPs, more targeted operators have been successfully applied to better guide the search towards promising areas in the search space, see, e.g., [Keskin and Çatay 2016; Ropke and Pisinger 2006; Shaw 1998]. For example, taking the contribution to the objective function value into account when choosing the elements to be removed can be advantageous and facilitate improvements in the following repair step.

Following this lead, we later adopt another common operator for our best performing approach, which is called the *worst-distance removal* (WDR) [Demir et al. 2012; Ropke and Pisinger 2006]. It iteratively removes the request inducing the highest distance-based cost until κ^{wdr} requests have been selected. The distance-based cost is calculated as the sum of distances from the preceding and to the succeeding nodes of the pickup and drop-off locations of the request within the route it is contained in. For the E-ADARP, this corresponds to the induced travel times of these arcs as the objective function considers the travel time instead of the distance of routes. Let i, k, i', and k' be the direct neighbors of a request's pickup $j \in P$ and drop-off $j + n$ in a route, the distance-based cost is then given by

$$c(j) = t_{i,j} + t_{j,k} + t_{i',j+n} + t_{j+n,k'}. \tag{6}$$

A slightly different cost calculation found in the literature [Kancharla and Ramadurai 2020] computes instead the additional distance that a request introduces in a route; or in other words, the distances between the respective pre- and succeeding nodes are subtracted from $c(j)$. We call this the *detour distance* (DD) cost which is in our case

$$c^{\text{DD}}(j) = t_{i,j} + t_{j,k} - t_{i,k} + t_{i',j+n} + t_{j+n,k'} - t_{i',k'}. \tag{7}$$

Whether or not c^{DD} is employed in our LNS instead of c is determined by a parameter $p^{\text{DD}} \in \{0, 1\}$, where value one denotes its usage. The respective parameter value to be used is considered an externally provided input to the LNS, resulting in the following employed cost function:

$$c'(j) = t_{i,j} + t_{j,k} + t_{i',j+n} + t_{j+n,k'} - p^{\text{DD}}(t_{i,k} + t_{i',k'}). \tag{8}$$

In each iteration, all requests $j \in P'$, with P' denoting the remaining served requests, are sorted in non-increasing order of their cost $c'(j)$ and the top one is selected for removal. However, such a deterministic selection may reduce the diversification of the search and even lead to cycling if

always the same requests are picked. A common approach to mitigate this effect is the introduction of randomness into the selection by not always removing the top request but the request at position $\lfloor y^{p^{\mathrm{df}}} |P'| \rfloor$. Here, $y \in [0, 1)$ is chosen uniformly at random and $p^{\mathrm{df}} \geq 1$ is a determinism factor used to control the degree of randomness. Lower values for p^{df} imply stronger randomization and vice versa. Like before, we use a prespecified strategy parameter $p^{\mathrm{R}} \in \{0, 1\}$ to (de-)activate the randomization of the operator. A value of zero stands for the deterministic selection of the top request and value one for the randomized approach. Another variability is derived from the occurrence of charging stops within routes. When determining the neighboring stops of a request within a route, we may take charging stations into consideration or not. The idea behind this distinction is that a request can have low distance-based cost and seem to be well embedded within a route, when in reality it is only close to a CS inserted as stop next to it due to the large distance to any other request or depot node. On the other hand, such a combination of charging stop and request may fit well into the route as a whole, e.g., for meeting battery requirements, and neglecting CSs could be detrimental. Both are valid points of view and we enable switching between them by means of a strategy parameter $p^{\mathrm{CS}} \in \{0, 1\}$ where a value of one indicates the neglect of charging stops during the cost computation. Overall, this threefold parameterization allows for eight different versions of the WDR operator, which we will experimentally evaluate in Section 6.1.4.

4.4 Repair Operators for Requests

For repair and request insertion, we employ three different methods. (a) We utilize the greedy heuristic from [Ropke and Pisinger 2006] which considers all possible insertion positions in all routes for the pickup and drop-off stops of every unserved request and realizes the overall cheapest insertion. This process is repeated until either all requests are served or no further feasible request insertion is possible. To speed up the process, we cache the insertion cost for each request-route combination in the first iteration and only recalculate it for modified routes in the following steps. (b) Our second repair method works similarly but reduces the computational effort by first sorting the unserved requests in non-decreasing order of the start of their pickup time windows. Then, these requests are tried for insertion in this order. (c) In our third repair method, the unserved requests are considered for greedy insertion in a random order to reduce the danger of getting trapped in a local optimum. Similarly to [Limmer 2023; Ropke and Pisinger 2006], random noise is added to the increases of the objective function values associated with the different position combinations when searching for a request's cheapest insertion for further diversification. The noise is randomly sampled from $[-\eta t^{\max}, \eta t^{\max}]$ with $t^{\max} = \max_{i,j \in V} t_{i,j}$, where the noise rate η is a parameter regulating the amount of noise used.

5 Charging Stop Insertion and Route Evaluation

Within our solution framework, we face two specific challenges of the E-ADARP: (a) the inclusion of appropriate charging stops and (b) the route evaluation including the determination of best service start and possible waiting times. While the common neighborhood operators mentioned in Section 4 are applicable to diverse VRP variants, they generally only consider customer requests and do not cover the additional charging requirements of EVs. Besides, being able to efficiently evaluate routes is another crucial aspect when solving such a highly constrained optimization problem with LNS. Generally, a large number of candidate solutions have to be evaluated for feasibility, and feasibility checking of a route in the context of the E-ADARP is challenging. It involves verifying whether a valid setting of service start and charging times exists as well as checking battery and charging requirements and time window constraints. Thus, employing an efficient evaluation procedure is essential for the overall performance of the LNS. In the context of DARPs, commonly applied

heuristics are the eight-step procedure based on forward time slacks by Cordeau and Laporte [2003], its modified version by Parragh et al. [2010], and the procedure from Molenbruch et al. [2017c] as used in [Braekers et al. 2014; Braekers and Kovacs 2016; Limmer 2023; Masmoudi et al. 2016, 2018]. These approaches usually come with a trade-off between solution quality regarding excess ride times and correctness of (in-)feasibility declarations. Due to the fragment-based optimization of user excess ride times in our framework, we do not have to take them into account during route evaluation, but these procedures also do not consider charging times and requirements, leaving this an open issue.

We therefore propose two ways to deal with the issues of charging stop insertion and route evaluation where the first one handles them separately and the second one provides a combined approach. First, we introduce the *direct* approach where the LNS is directly concerned with the charging stops and the optimization of their insertion into routes by employing dedicated destroy and repair operators. For efficient evaluation of candidate routes, we then use a modified version of the labeling algorithm from [Su et al. 2023]. The second approach, called *on-the-fly* approach, is more indirect and LNS independent as it does not involve additional LNS operators but instead introduces a novel route evaluation procedure that inserts charging stops on-the-fly as needed, hereby also efficiently determining the respective charging times. In the following, we provide more details on our proposed approaches including a description of the operators for CSs, the labeling algorithm, as well as the MILP formulation and heuristic for the on-the-fly insertion.

5.1 Direct Approach

In this approach, we have dedicated LNS operators for removing and reinserting charging stations to meet the battery requirements of the EVs. Similar approaches have been proposed for electric vehicle routing problem (EVRP) variants in [Kancharla and Ramadurai 2020; Keskin and Çatay 2016]. In contrast to these, we employ an exact labeling algorithm utilizing a fragment-based route representation for efficient solution evaluation during repair.

5.1.1 Destroy and Repair Operators for Charging Stations. Similarly to the random removal of requests, the destroy operator for charging stations selects κ^s visited charging stations uniformly at random and removes them from the routes. In the repair operator, we first have to determine in which state each vehicle route is after the destroy. If a route is still feasible, either because it has not been modified or because the removed charging station(s) were actually redundant, it is left unchanged. If a route now violates time window constraints due to potentially necessary longer charging durations at remaining CSs in the route or if it is battery infeasible and there are no available charging stations left, feasibility is restored by removing as many stops as necessary, starting from the first visited location after the origin depot. Otherwise the route violates battery constraints with at least one charging station available, and in this case, the following greedy heuristic is applied. It identifies the first customer in the route at which the vehicle arrives with negative battery level and also which fragment this customer belongs to. Then, an available charging station which increases the route distance the least is identified and inserted on the arc between this and the preceding fragment. If there is no feasible charging station or if the insertion leads to time window violations, the step is reverted and an insertion before the previous fragments is tried in the same manner. Similarly, if the route is still not battery feasible after a successful insertion, the procedure is repeated considering the whole route again. In case of remaining constraint violations but no feasible insertion possibility, the heuristic reverts to removing stops from the route.

In the direct approach, this heuristic is also employed within the repair operators for requests in order to provide more and better insertion possibilities. Whenever a candidate route violates only battery-related constraints after request insertion, charging stations are greedily inserted

to potentially restore the route's feasibility, leading to an increased number of feasible request insertion possibilities.

5.1.2 Labeling Algorithm. Considering the need for an efficient route evaluation procedure, we propose a modified version of the labeling algorithms from Su et al. [2023] for the E-ADARP and Desaulniers et al. [2016] for the EVRP with time windows, which determines the feasibility of a route in an exact manner and is of linear time complexity in the length of the considered route. In the following, we present the main idea of the algorithm as well as our adjustments and refer the interested reader to the earlier works for a more detailed description.

In the algorithm, each stop i in a route is assigned a label L_i composed of four resource attributes, i.e., $L_i = (\{T_i^{\text{rch}_s} | s \in S\}, T_i^{\text{tMin}}, T_i^{\text{tMax}}, T_i^{\text{rtMax}})$. Let $\mathcal{P}_i$ denote the partial path from the start of the route up to stop i, the resource attributes are then defined as follows:

(1) $T_i^{\text{rch}_s}$: The number of times each charging station $s \in S$ is visited up to stop i.
(2) T_i^{tMin}: The earliest service start time at stop i assuming that battery feasibility is maintained up to i by charging as little as necessary if a charging station is visited along $\mathcal{P}_i$ before i.
(3) T_i^{tMax}: The earliest service start time at stop i assuming that time-window feasibility is maintained up to i despite charging as much as possible if a charging station is visited along $\mathcal{P}_i$ before i.
(4) T_i^{rtMax}: The maximum charging time required to fully recharge at stop i again assuming that battery feasibility is maintained up to i by charging as little as necessary if a charging station is visited along $\mathcal{P}_i$ before i. If a stop is not at a charging station, this is only used hypothetically to forward the information along the path.

Su et al. [2023] set the label for the first stop, the origin depot, to $L_0 = (\{0, \dots, 0\}, 0, 0, 0)$, and the labels of the succeeding nodes are computed one by one using resource extension functions while iterating over the route once. This involves computing the available slack time at the current node i, which is the difference between the start of the corresponding time window and the earliest arrival time at i. This slack time can be used for additional charging beforehand and thus can reduce the required maximum charging time T_i^{rtMax}. Besides, a minimum charging time necessary at preceding charging stations to maintain battery feasibility up to i already taking into account the respective slack time is considered when computing T_i^{tMin}. The travel time as well as the battery consumption of the traveled arc also play a role during label computation. For the route to be feasible, the label of each stop i in the route has to satisfy the following conditions regarding time window and battery requirements, where i' denotes the location visited at stop i and $H = \frac{Q}{\alpha}$ is the charging time of an empty battery to its full capacity:

$$T_i^{\text{rch}_s} \leq 1,\ T_i^{\text{tMin}} \leq w_{i'}^{\text{end}},\ T_i^{\text{tMin}} \leq T_i^{\text{tMax}},\ \text{and}\ T_i^{\text{rtMax}} \leq \begin{cases} (1-\gamma)H, & \text{if } i' \in F, \\ H, & \text{otherwise.} \end{cases}$$

As mentioned before, the above procedure runs in linear time with regards to the length of the considered route. To further reduce the computational effort, we propose a fragment-based route representation and assign labels to fragments instead of individual stops, making the route evaluation even more efficient. This is possible because charging stops can only be inserted between fragments but not within, and each fragment can be independently optimized and results are cached, cf. LP (2–5) and Section 3.3. A route stop is then either a fragment, a depot, or a charging station. Hence, we consider not only the battery consumption for traveling in between two stops but also the battery requirements b_i of the current stop i when computing the labels. The battery requirement of a fragment $\mathcal{F}$ is the sum of battery consumptions for traveling all the arcs within the fragment, i.e., $b_{\mathcal{F}} = \sum_{(i,j)\in\mathcal{F}} b_{i,j}$. For origin depots and charging stations, b_i is set to zero and

we use the minimum battery level γQ at destination depots. The latter is necessary to prevent incorrect infeasibility declarations that can occur with the labeling algorithm in [Su et al. 2023]. These are due to the original algorithm's goal of charging as much as possible at visited CSs while still arriving as early as possible within the time window of each location. This can lead to zero charging time being scheduled despite having time available to charge without violating time windows along the route. This issue is resolved by our adjusted approach. Another modification concerns the initial label for the origin depot o_k of a route of vehicle k. We additionally consider the respective time window start as well as the possibility of the vehicle not being fully charged, resulting in $L_0 = (\{0, \ldots, 0\}, w_{o_k}^{\text{start}}, w_{o_k}^{\text{start}}, \frac{Q - B_{k,1}}{\alpha})$, which represents a more realistic and practical view.

An interesting property of the labeling algorithm is that the route evaluation can be done without calculation of actual service start and charging times. An explicit schedule is not needed when pairing the evaluation with the fragment-based optimization of the user excess ride times as charging costs are not considered in the objective function. However, for the final solution, it is possible to extract necessary charging durations from the labeling process and together with the inner-fragment waiting times derived from the LP, service start times can be determined in a single linear-time forward pass.

5.2 On-The-Fly (OTF) Approach

Contrary to the direct approach, the LNS framework is here not concerned with charging station visits at all and thus does not require corresponding operators. Instead, we only apply the operators for requests described in Sections 4.3 and 4.4 and introduce a novel route evaluation scheme to be used in place of the labeling algorithm. As this new procedure is fully independent of the LNS, it can be used within diverse metaheuristic frameworks. It inserts charging stops on-the-fly as needed into a route while also checking its feasibility and computing the respective charging and service start times. The actual route and its length are here only known after applying this evaluation algorithm. The algorithm is developed with the following assumptions in mind. (a) At most one charging stop needs to be inserted between any pair of successive fragments or depots, which is in contrast to the direct approach where there is no such limitation. For practical applications in urban areas, this will be no serious restriction. (b) For the sake of simplicity, we further assume that charging stations have no time windows, although considering them would be straightforward. One major advantage of the OTF approach is its ability to handle vehicle/station-individual charging speeds, which is not possible with the labeling algorithm. In the following, we first consider a MILP formulation of the charging station insertion and evaluation subproblem before we propose a time-efficient heuristic procedure.

5.2.1 MILP Formulation. For a route r, R represents its abstract equivalent where sequences of pickup and drop-off locations are replaced by corresponding fragments. Let $R = R_1, \ldots, R_{n_R}$ denote the abstract route after removal of all possibly so far existing charging stops and B_1 the vehicle's initial battery level when leaving the origin depot. In the following, we utilize notations such as $t_{i,j}$ and d_i by using indices i and j into the current route R instead of the actually corresponding locations. Binary decision variables $x_{i,s}$ for $i = 2, \ldots, |R|$ indicate with value one that a charging stop is included at station s between stops $i-1$ and i of the route. If a charging station s is visited before the i-th stop, the charging duration is assigned to $d_{i,s}^{\text{ch}}$, otherwise it is set to zero. The battery level on departure from the i-th stop of the route is given by B_i, whereas B_i^{ch} represents the battery level on arrival at the charging station before the i-th stop; if there is no charging station visit before the i-th stop, $B_i^{\text{ch}} = B_{i-1}$. Variables t_i^{serv} denote the time at which service starts at the i-th

stop. The MILP model is then defined as follows.

$$\min \quad \sum_{i=2}^{|R|} \sum_{s \in S} (t_{i-1,s} + t_{s,i} - t_{i-1,i}) \cdot x_{i,s} \tag{9}$$

$$\text{s.t.} \quad \sum_{s \in S} x_{i,s} \leq 1 \qquad i = 2, \ldots, |R| \tag{10}$$

$$t_i^{\text{serv}} \geq t_{i-1}^{\text{serv}} + d_{i-1} + t_{i-1,i} + \sum_{s \in S} \left((t_{i-1,s} + t_{s,i} - t_{i-1,i}) \cdot x_{i,s} + d_{i,s}^{\text{ch}} \right) \qquad i = 2, \ldots, |R| \tag{11}$$

$$B_i^{\text{ch}} = B_{i-1} - \sum_{s \in S} \beta_{i-1,s} x_{i,s} \qquad i = 2, \ldots, |R| \tag{12}$$

$$B_i^{\text{ch}} + \sum_{s \in S} \alpha_s d_{i,s}^{\text{ch}} \leq Q \qquad i = 2, \ldots, |R| \tag{13}$$

$$B_i = B_i^{\text{ch}} + \sum_{s \in S} \alpha_s d_{i,s}^{\text{ch}} - \beta_{i-1,i} - \sum_{s \in S} (\beta_{s,i} - \beta_{i-1,i}) \cdot x_{i,s} - b_i \qquad i = 2, \ldots, |R| \tag{14}$$

$$d_{i,s}^{\text{ch}} \leq \frac{Q}{\alpha_s} x_{i,s} \qquad i = 2, \ldots, |R|,\ s \in S \tag{15}$$

$$x_{i,s} \in \{0, 1\} \qquad i = 2, \ldots, |R|,\ s \in S \tag{16}$$

$$d_{i,s}^{\text{ch}} \geq 0 \qquad i = 2, \ldots, |R|,\ s \in S \tag{17}$$

$$0 \leq B_i \leq Q \qquad i = 2, \ldots, |R| \tag{18}$$

$$0 \leq B_i^{\text{ch}} \leq Q \qquad i = 2, \ldots, |R| \tag{19}$$

$$w_i^{\text{start}} \leq t_i^{\text{serv}} \leq w_i^{\text{stop}} \qquad i = 1, \ldots, |R| \tag{20}$$

The objective function (9) minimizes the total detour length caused by all newly inserted charging visits in order to increase the objective value of the route as little as necessary. Inequalities (10) limit the number of charging visits between successive stops to one, and Constraints (11) ensure feasible service start times. Equations (12) set the battery levels on arrival at a charging station, whereas Equations (14) determine the battery levels on departure from all other route stops taking a potential preceding charging visit into account. Constraints (13) prevent overcharging of vehicles, and Constraints (15) restrict the charging durations. Lastly, (16)–(20) state the domains of the decision variables. Solving this MILP model for a route tells whether the route is feasible or not as well as where and how long the vehicle has to charge and when service starts at all stops.

5.2.2 Time-Efficient Heuristic. In the LNS, many candidate routes have to be evaluated, and the above MILP approach can therefore be expected to be a serious bottleneck. We propose as an alternative a fast heuristic procedure for simultaneously evaluating a route and inserting charging stations on-the-fly, starting again with the fragment-based representation R of the route after removal of all charging stops.

Algorithm 3 shows the main steps of this procedure in pseudo-code. In a forward-pass on R, it determines for all stops $i \in R$ the earliest possible service start time t_i^{early}, waiting duration d_i^{wait} (which is induced by a vehicle's arrival before the start of the time window), and battery level B_i on departure. Similarly, the latest possible service start times t_i^{late} as well as "backward-waiting" durations $d_i^{\text{wait_back}}$ when waiting for the end of time windows are determined in a backward-pass on R. If any time window violation is detected at this step, the route is already proven infeasible

Algorithm 3: OTF charging stop insertion and evaluation.

Input: route r
Output: feasibility status of r

$r \leftarrow$ remove_charging_stops(r);
$R \leftarrow$ get_fragments_and_depots(r);
if contains_infeasible_fragment(R) **then**
    return false;
$i^{\text{ch}} \leftarrow none,\ S' \leftarrow$ available_stations(S);
$t^{\text{early}},\ d^{\text{wait}},\ B,\ i^{\text{ch}} \leftarrow$ forward_pass(R);
$t^{\text{late}},\ d^{\text{wait_back}} \leftarrow$ backward_pass(R);
if violates_time_windows(t^{early}) **or** violates_time_windows(t^{late}) **then**
    return false;
while $i^{\text{ch}} \neq none$ **do**
    $min_detour \leftarrow \infty,\ max_energy \leftarrow 0,\ \hat{i} \leftarrow none$;
    for i in $i^{\text{ch}}, \dots, 2,\ s \in S'$ **do**
        $feasible,\ detour,\ energy \leftarrow$ check_stop(R, i, s);
        if $feasible$ **and** better_stop($detour, energy, min_detour, max_energy$) **then**
            $min_detour \leftarrow detour,\ max_energy \leftarrow energy$;
            $\hat{i} \leftarrow i,\ \hat{s} \leftarrow s$;
    if $\hat{i} = none$ **then**
        return false;
    insert_charging_stop($r, R, \hat{i}, \hat{s}$);
    update($t^{\text{early}}, d^{\text{wait}}, t^{\text{late}}, d^{\text{wait_back}}, B, i^{\text{ch}}, S'$);
return true;

and the procedure stops. Charging is necessary if at least one B_i assumes a negative value; in this case, it identifies the latest stop i^{ch} before which this value turns negative. Otherwise, if all B_i are non-negative and no charging is required, the route is declared feasible as it is and the algorithm returns. For the insertion of a charging stop, it considers all stops $i \in \{2, \dots, i^{\text{ch}}\}$ in reverse order and all available charging stations $s \in S'$ and checks for all combinations whether a charging stop at s before i would be feasible and how much energy could be charged. If no feasible charging stop is found, the route's feasibility cannot be restored and the procedure stops. Otherwise, from all feasible insertion options, it selects the one that introduces the smallest detour length and provides a long enough charging duration to satisfy all battery requirements. If no such charging stop exists, the one with the highest amount of charged energy after deduction of the battery consumption required for the incurred detour is chosen. The selected charging stop is then inserted into the route and S' as well as t_i^{early}, t_i^{late}, d_i^{wait}, $d_i^{\text{wait_back}}$, and B_i are updated for all $i \in R$ affected by the insertion. We determine again whether and where charging is necessary at the latest and repeat the required steps for finding, selecting, and inserting a best suited charging stop until the route becomes feasible or the procedure terminates due to no further feasible insertion options.

A property of this algorithm is that when feasibility of a route can be restored by a single charging stop insertion, it will do this in an optimal way. Otherwise, it will always find suitable charging station insertions if charging rates are homogeneous and there is a way to make the solution feasible,

but optimality in respect to the tour length is not guaranteed anymore. Excluding the evaluation of fragments appearing the first time via the LP, the algorithm runs in time $O(|R| \cdot |S'| \cdot n_{\text{ch}})$ with n_{ch} being the number of inserted charging stops. Considering existing benchmark instances, the number of available CSs $|S'|$ is at most three and only very few charging stops are typically required per route (≤ 3). We may therefore neglect these factors and say the algorithm runs in amortized linear time in the number of stops.

6 Experimental Analysis

All experiments were performed on single cores of 2.4 GHz Intel Xeon E5-2640 v4 processors and with a memory limit of 20 GB. Our proposed approaches are implemented in Julia 1.10.0, and we used Gurobi 10.0.3[1] as MILP solver in single-threaded mode. We consider two sets of common benchmark instances based on instances for the standard DARP, i.e., 14 Cordeau instances[2] [Cordeau 2006] and ten Ropke instances[3] [Ropke et al. 2007], as well as five very large-scale instances[3] from [Limmer 2023]. All instances are enhanced with E-ADARP-specific features as detailed in [Bongiovanni et al. 2019], in particular by adding three charging stations and incorporating battery capacities and requirements of the electric vehicles. Instances are named using the pattern an_K-n, where n_K is the number of vehicles and n the number of requests.

Like in [Bongiovanni et al. 2019; Limmer 2023; Su et al. 2024, 2023], $w^{\text{routing}} = 0.75$ and $w^{\text{excess}} = 0.25$ are used as weight factors in the objective function (1). The battery consumption $\beta_{i,j}$ is assumed to be proportional to the travel time $t_{i,j}$ considering a homogeneous battery drain rate $\dot{\beta} = 0.055$ kWh per unit of travel time, i.e., $\beta_{i,j} = \dot{\beta} t_{i,j}$. As in the literature, we consider three different minimum end battery level ratios $\gamma \in \{0.1, 0.4, 0.7\}$ of a vehicle's battery capacity required on arrival at the destination depot. Besides, we investigate two modes for charging station visits: a limitation to one visit per station over all routes as in [Bongiovanni et al. 2019], denoted by $n_{\text{s}} = 1$, and an unlimited number of allowed visits as in [Limmer 2023; Su et al. 2024, 2023] with $n_{\text{s}} = \infty$. In total, this results in six different configurations for the benchmark instances. For each instance and configuration, we perform 30 runs of our solving approaches with a time limit of 300 s per run.

To reduce the complexity of the instances, we employ time window tightening and arc elimination as proposed by Dumas et al. [1991] and Cordeau [2006] for other problems. For the former, we use the rules for the E-ADARP as stated in [Su et al. 2023] with some modifications. Regarding arc elimination, we adjust the rules from Goeke [2019] to consider multiple depots and add a rule based on vehicle loads. The application of further path-based elimination rules [Cordeau 2006; Dumas et al. 1991] allows not only for arc removal but also identification of incompatible request pairs that can never be served by the same route. We exploit this information to reduce the number of insertion positions of a request that have to be checked for feasibility in the LNS repair operators described in Section 4.4, thereby improving their efficiency. Robust settings for the noise rate η as well as the degrees of destruction κ^{rr} and κ^{s} are identified through parameter tuning with SMAC3[4] [Lindauer et al. 2022] by running 10000 trials on ten representative instances. The resulting configuration further used is $\eta = 0.014$, $\kappa^{\text{rr}} = 15$, and $\kappa^{\text{s}} = 2$ for $n_{\text{s}} = 1$ and $\kappa^{\text{s}} = 0.54$ for $n_{\text{s}} = \infty$.

6.1 Results

First, the results of our approaches employing only the random removal operator are compared to the two-indexed MILP formulation e-ADARP2 from Bongiovanni et al. [2019], the B&P algorithm

[1] https://www.gurobi.com
[2] https://luts.epfl.ch/wpcontent/uploads/2019/03/e_ADARP_archive.zip
[3] Available under https://github.com/HRI-EU/e_adarp_material
[4] https://github.com/automl/SMAC3

from Su et al. [2024], the DA approach from Su et al. [2023], and the BI-LNS from Limmer [2023] using the data reported in these works. In the following tables, we report run times (RT) and objective values (obj) for e-ADARP2 and B&P and the mean run times ($\overline{\text{RT}}$) as well as the minimum and mean objective values (obj_{min} and $\overline{\text{obj}}$) over 50 trials for DA, where "NA" means that some trials did not find a feasible solution. Note that the run times of B&P on Cordeau instances are omitted but are of the same order of magnitude as for e-ADARP2. The minimum and average objective values and the number of runs yielding a feasible solution are shown for BI-LNS, our direct approach, and our OTF approaches employing the MILP formulation and the heuristic for inserting charging stations. For these, the run time was fixed to five minutes per trial and mean values are computed over all obtained feasible solutions. For our proposed approaches, the standard deviation σ_{obj} over all feasible runs is given as well. Value "NA" denotes that no feasible solution was found and bold values mark best known objective values. For some instances, the supposedly optimal results by e-ADARP2 are not globally optimal due to an error in the model as pointed out in [Su et al. 2023] and thus, better objective values can be achieved.

Secondly, we investigate the integration of the worst-distance removal operator into the OTF heuristic-based approach. In this context, suitable parameterization and parameter tuning are discussed and different versions of the operator are compared to each other as well as the OTF heuristic and the direct approach as baselines.

6.1.1 Cordeau Instances. The results on the smaller Cordeau instances under $\gamma = 0.1$ and for both modes regarding the allowed number of charging station visits are shown in Table 1. For the restricted charging case of $n_s = 1$, efficient computation of optimal solutions is mostly possible as illustrated by the results of e-ADARP2 and B&P. Except for the largest of these instances, DA is able to match these solutions whereas the ones found by BI-LNS are slightly worse. Our proposed direct approach as well as the OTF insertion based one outperform the other heuristics by finding best known solutions for all instances and having an overall better mean performance. For three instances, i.e., a3-18, a4-32, and a5-40, the computed objective values of our approaches are 0.01 less than the claimed optimal ones from e-ADARP2 and B&P, which we assume is due to negligible numerical imprecisions in the floating point value calculations. Please note that such minor deviations also occur in some other cases in the following. The MILP-based OTF procedure is inferior to our other approaches and to DA but able to beat BI-LNS in terms of best found solutions. Its average performance is slightly worse but still exhibits limited mean gaps with an average of 0.64% over all instances and at most 2.20% (on a5-50).

Keeping $\gamma = 0.1$ but relaxing the limitation on charging station visits has almost no impact as illustrated by the results in Table 1. For this configuration, [Bongiovanni et al. 2019; Su et al. 2024, 2023] do not report results for e-ADARP2, B&P, and DA respectively. The results of BI-LNS as well as our approaches are similar to the restricted charging mode with the most differences being observed for the OTF MILP approach and some minor changes in the mean performances overall. However, the best found objective values are identical, which indicates that on these instances a minimum end battery level ratio of 0.1 can be satisfied by charging at most once with some vehicles not having to charge at all.

Most of the above observations regarding the results for $\gamma = 0.1$ also apply to the case of an increased minimum end battery level ratio of $\gamma = 0.4$, for which detailed results are given in Table 12 in Appendix A.1. One main difference though is that our direct approach and the OTF heuristic do not find the best known solutions for all instances for $n_s = 1$ anymore, and we see a slightly higher performance decrease for the direct approach than for the OTF heuristic. Nevertheless, the heuristic OTF procedure remains the best performing heuristic approach. Another interesting observation is that, while still being outperformed in general by our top approaches, BI-LNS shows improved

Table 1. Results on Cordeau instances with $\gamma = 0.1$ for limited CS visits ($n_s = 1$) and unlimited CS visits ($n_s = \infty$).

Inst	e-ADARP2		B&P	DA			BI-LNS			Direct Approach				OTF Approach MILP				OTF Approach Heuristic			
	RT [min]	obj	obj	$\overline{\text{RT}}$ [min]	$\text{obj}_{\min}$	$\overline{\text{obj}}$	$\text{obj}_{\min}$	$\overline{\text{obj}}$	Feas	$\text{obj}_{\min}$	$\overline{\text{obj}}$	σ_{obj}	Feas	$\text{obj}_{\min}$	$\overline{\text{obj}}$	σ_{obj}	Feas	$\text{obj}_{\min}$	$\overline{\text{obj}}$	σ_{obj}	Feas
$n_s = 1$																					
a2-16	0.02	**237.38**	**237.38**	0.65	**237.38**	237.38	238.20	238.20	10/10	**237.38**	237.38	0.00	30/30	**237.38**	237.38	0.00	30/30	**237.38**	237.38	0.00	30/30
a2-20	0.07	**279.08**	**279.08**	1.23	**279.08**	279.08	281.00	281.00	10/10	**279.08**	279.08	0.00	30/30	**279.08**	279.08	0.00	30/30	**279.08**	279.08	0.00	30/30
a2-24	0.15	**346.21**	**346.21**	2.68	**346.21**	346.21	**346.21**	346.59	10/10	**346.21**	346.21	0.00	30/30	**346.21**	346.66	1.03	30/30	**346.21**	346.21	0.00	30/30
a3-18	0.08	**236.82**	**236.82**	0.42	**236.82**	236.82	238.73	238.75	10/10	**236.81**	236.81	0.00	30/30	**236.81**	236.81	0.00	30/30	**236.81**	236.81	0.00	30/30
a3-24	0.23	**274.80**	**274.80**	0.97	**274.80**	274.80	275.18	275.18	10/10	**274.80**	274.80	0.00	30/30	**274.80**	275.50	1.01	30/30	**274.80**	274.80	0.00	30/30
a3-30	1.70	**413.27**	**413.27**	0.90	**413.27**	413.27	414.88	414.88	10/10	**413.27**	413.27	0.00	30/30	**413.27**	414.33	1.27	30/30	**413.27**	413.27	0.00	30/30
a3-36	1.78	**481.17**	**481.17**	2.54	**481.17**	481.17	483.86	484.05	10/10	**481.17**	483.88	2.74	30/30	481.38	486.09	3.25	30/30	**481.17**	482.18	1.37	30/30
a4-16	0.06	**222.49**	**222.49**	0.32	**222.49**	222.49	**222.49**	222.49	10/10	**222.49**	222.49	0.00	30/30	**222.49**	222.61	0.28	30/30	**222.49**	222.49	0.00	30/30
a4-24	0.52	**310.84**	**310.84**	0.49	**310.84**	310.84	311.48	311.48	10/10	**310.84**	310.84	0.00	30/30	**310.84**	310.95	0.22	30/30	**310.84**	310.84	0.00	30/30
a4-32	10.20	**393.96**	**393.96**	0.87	**393.96**	395.12	394.66	394.79	10/10	**393.95**	393.95	0.00	30/30	**393.95**	394.89	1.12	30/30	**393.95**	393.95	0.00	30/30
a4-40	8.62	**453.84**	**453.84**	1.53	**453.84**	459.42	456.93	457.08	10/10	**453.84**	455.17	2.35	30/30	**453.84**	462.55	4.48	30/30	**453.84**	454.46	1.95	30/30
a4-48	120.00	**554.54**	**554.54**	2.36	555.93	561.26	557.24	557.94	10/10	**554.54**	556.74	1.30	30/30	556.36	560.02	1.60	30/30	**554.54**	555.38	0.73	30/30
a5-40	19.03	**414.51**	**414.51**	1.08	414.80	420.35	415.62	415.65	10/10	**414.50**	415.30	1.10	30/30	415.80	422.04	4.35	30/30	**414.50**	414.99	0.91	30/30
a5-50	120.00	**559.17**	**559.17**	2.29	561.41	570.58	560.07	560.66	10/10	**559.17**	562.14	2.24	30/30	563.50	571.47	3.89	30/30	**559.17**	562.18	2.40	30/30
$n_s = \infty$																					
a2-16							238.20	238.20	10/10	**237.38**	237.38	0.00	30/30	**237.38**	237.38	0.00	30/30	**237.38**	237.38	0.00	30/30
a2-20							281.00	281.00	10/10	**279.08**	279.08	0.00	30/30	**279.08**	279.08	0.00	30/30	**279.08**	279.08	0.00	30/30
a2-24							**346.21**	346.22	10/10	**346.21**	346.21	0.00	30/30	**346.21**	346.39	0.69	30/30	**346.21**	346.21	0.00	30/30
a3-18							238.73	238.73	10/10	**236.81**	236.81	0.00	30/30	**236.81**	236.81	0.00	30/30	**236.81**	236.81	0.00	30/30
a3-24							275.18	275.18	10/10	**274.80**	274.80	0.00	30/30	**274.80**	275.60	0.84	30/30	**274.80**	274.80	0.00	30/30
a3-30							414.88	414.88	10/10	**413.27**	413.27	0.00	30/30	**413.27**	414.92	2.26	30/30	**413.27**	413.27	0.00	30/30
a3-36							483.86	483.86	10/10	**481.17**	483.80	2.92	30/30	**481.17**	486.63	2.91	30/30	**481.17**	482.26	1.77	30/30
a4-16							**222.49**	222.49	10/10	**222.49**	222.49	0.00	30/30	**222.49**	222.78	0.33	30/30	**222.49**	222.49	0.00	30/30
a4-24							311.48	311.48	10/10	**310.84**	310.84	0.00	30/30	**310.84**	310.96	0.23	30/30	**310.84**	310.84	0.00	30/30
a4-32							394.66	394.79	10/10	**393.95**	393.95	0.00	30/30	**393.95**	394.64	1.11	30/30	**393.95**	393.95	0.00	30/30
a4-40							456.93	457.37	10/10	**453.84**	454.88	2.02	30/30	455.26	462.06	3.82	30/30	**453.84**	453.84	0.00	30/30
a4-48							557.25	557.82	10/10	**554.54**	556.22	1.30	30/30	555.70	560.01	1.87	30/30	**554.54**	555.43	0.82	30/30
a5-40							415.62	415.62	10/10	**414.50**	415.30	1.10	30/30	416.45	422.27	3.63	30/30	**414.50**	415.13	0.98	30/30
a5-50							560.07	560.95	10/10	559.51	563.32	2.80	30/30	563.87	571.30	3.76	30/30	**559.17**	561.81	2.01	30/30

performance on the larger instances and overtakes DA. The influence of the larger γ-value is small but observable by higher best objective values than for $\gamma = 0.1$ and the possibility of finding better solutions for $n_s = \infty$ than for $n_s = 1$ as the charging of EVs gains importance.

Moving on to the largest tested value of $\gamma = 0.7$, Table 2 shows the results for both modes regarding the allowed number of charging station visits, which exhibit clear differences compared to the previous configurations. The exact e-ADARP2 struggles to find feasible solutions within its time limit of two hours and is not able to solve five of the 14 instances at all, whereas this is only once the case for B&P. For $n_s = 1$, the heuristic approaches DA and BI-LNS cannot provide feasible solutions in all runs anymore and while DA still finds best known solutions for the smaller instances, it is again inferior to BI-LNS on the larger ones. On this instance configuration, all approaches from the literature are consistently outperformed by the OTF insertion heuristic, which is our best performing approach, matching all best known objective values so far and providing a new best solution while having average gaps in between −1.5% (on a4-48) and 1.44% (on a4-40) compared to the approaches from literature. The direct approach obtains two new best objective values as well but performs overall slightly worse with mean gaps in the range from −0.98% (on a4-48) to 2.82% (on a4-40). In contrast to the results from literature, both approaches yield feasible solutions in all runs. This is not achieved by the MILP-based OTF insertion procedure for three instances but it is still competitive on most instances with mean gaps between 0.07% (on a2-16) and 4.21% (on a4-40).

Regarding the mode $n_s = \infty$, no results are reported for e-ADARP2 but all other approaches obtain better results than with $n_s = 1$ for most instances, highlighting the increased importance and impact of charging stops regarding solution quality. Both OTF approaches find optimal solutions on the majority of instances and are now clearly superior to the direct approach as well as DA and

Table 2. Results on Cordeau instances with $\gamma = 0.7$ for limited CS visits ($n_s = 1$) and unlimited CS visits ($n_s = \infty$).

Inst	e-ADARP2		B&P	DA			BI-LNS			Direct Approach				OTF Approach MILP				OTF Approach Heuristic			
	RT [min]	obj	obj	$\overline{\text{RT}}$ [min]	$\text{obj}_{\min}$	$\overline{\text{obj}}$	$\text{obj}_{\min}$	$\overline{\text{obj}}$	Feas	$\text{obj}_{\min}$	$\overline{\text{obj}}$	σ_{obj}	Feas	$\text{obj}_{\min}$	$\overline{\text{obj}}$	σ_{obj}	Feas	$\text{obj}_{\min}$	$\overline{\text{obj}}$	σ_{obj}	Feas
$n_s = 1$																					
a2-16	0.09	**240.66**	**240.66**	1.60	**240.66**	240.66	242.83	245.50	10/10	**240.66**	242.19	1.03	30/30	**240.66**	240.84	0.40	30/30	**240.66**	240.66	0.00	30/30
a2-20	120.00	NA	**293.27**	2.88	**293.27**	294.11	NA	NA	0/10	**293.27**	297.60	6.44	30/30	**293.27**	294.98	3.30	22/30	**293.27**	293.27	0.00	30/30
a2-24	16.02	358.21	**353.18**	3.44	**353.18**	NA	356.99	363.04	10/10	**353.18**	358.85	5.43	30/30	**353.18**	353.45	1.45	30/30	**353.18**	353.18	0.00	30/30
a3-18	0.80	**240.58**	**240.58**	0.97	**240.58**	240.58	242.49	246.13	10/10	244.35	244.39	0.08	30/30	**240.58**	240.98	0.95	30/30	**240.58**	240.58	0.00	30/30
a3-24	2.54	277.72	**275.97**	2.06	**275.97**	277.43	277.52	277.52	10/10	277.43	278.42	1.18	30/30	**275.97**	277.46	1.76	30/30	**275.97**	275.97	0.00	30/30
a3-30	120.00	NA	**424.93**	1.30	**424.93**	436.20	432.27	436.56	10/10	**424.93**	430.07	4.05	30/30	**424.93**	430.32	3.59	30/30	**424.93**	426.12	1.59	30/30
a3-36	120.00	**494.04**	**494.04**	2.09	**494.04**	502.27	496.75	500.84	10/10	**494.04**	497.86	3.83	30/30	**494.04**	502.35	4.27	30/30	**494.04**	497.18	2.61	30/30
a4-16	1.12	**223.13**	**223.13**	0.52	**223.13**	223.13	**223.13**	223.95	10/10	**223.13**	223.13	0.00	30/30	**223.13**	223.37	0.27	30/30	**223.13**	223.13	0.00	30/30
a4-24	30.58	318.21	**316.65**	0.90	**316.65**	318.31	319.37	321.10	10/10	318.31	319.89	0.67	30/30	**316.65**	319.17	1.88	30/30	**316.65**	316.65	0.00	30/30
a4-32	120.00	430.07	**397.87**	1.19	**397.87**	405.85	401.97	402.59	10/10	**397.87**	399.26	2.18	30/30	**397.87**	399.98	2.35	30/30	**397.87**	397.87	0.00	30/30
a4-40	120.00	NA	**467.72**	1.91	479.02	NA	471.72	478.93	10/10	**467.72**	480.89	7.48	30/30	471.78	487.42	7.65	30/30	**467.72**	474.47	6.23	30/30
a4-48	120.00	NA	NA	2.74	582.22	NA	579.71	588.48	10/10	**575.35**	582.71	4.80	30/30	580.49	591.36	6.69	29/30	575.62	579.63	2.40	30/30
a5-40	120.00	447.63	**418.75**	1.63	424.26	436.94	420.20	423.88	10/10	420.01	425.32	3.20	30/30	425.93	433.02	5.60	30/30	**418.75**	421.16	3.26	30/30
a5-50	120.00	NA	690.79	2.64	603.24	NA	593.71	602.30	9/10	**589.61**	597.55	5.14	30/30	596.54	612.00	9.81	27/30	**589.61**	596.09	3.87	30/30
$n_s = \infty$																					
a2-16			**240.66**	1.99	**240.66**	240.66	242.44	242.44	10/10	241.32	241.34	0.06	30/30	**240.66**	240.66	0.00	30/30	**240.66**	240.66	0.00	30/30
a2-20			**285.86**	5.27	286.32	288.89	290.33	291.23	10/10	288.39	290.50	1.14	30/30	**285.86**	285.95	0.21	30/30	**285.86**	285.86	0.00	30/30
a2-24			**350.32**	5.96	354.38	374.68	354.53	356.89	10/10	**350.32**	355.40	3.02	30/30	**350.32**	350.33	0.05	30/30	**350.32**	350.32	0.00	30/30
a3-18			**238.82**	1.10	**238.82**	238.82	241.95	242.46	10/10	241.13	241.13	0.00	30/30	**238.82**	239.55	0.76	30/30	240.03	240.03	0.00	30/30
a3-24			**275.20**	2.50	**275.20**	275.20	277.52	278.02	10/10	276.27	276.62	0.58	30/30	**275.20**	276.05	0.96	30/30	**275.20**	275.20	0.00	30/30
a3-30			**413.35**	2.85	415.71	417.07	419.16	426.30	10/10	416.89	421.54	3.55	30/30	413.45	415.85	1.87	30/30	413.45	414.08	1.07	30/30
a3-36			**483.08**	5.72	484.85	487.91	490.26	492.79	10/10	484.07	492.96	4.49	30/30	**483.08**	490.60	3.64	30/30	484.49	486.98	3.11	30/30
a4-16			**222.49**	0.52	**222.49**	222.49	**222.49**	223.57	10/10	**222.49**	222.49	0.00	30/30	**222.49**	223.09	0.54	30/30	**222.49**	222.49	0.00	30/30
a4-24			**315.40**	1.18	315.98	317.99	316.51	318.38	10/10	**315.40**	316.48	1.21	30/30	**315.40**	316.00	0.47	30/30	315.98	315.98	0.00	30/30
a4-32			**394.94**	2.06	**394.94**	401.82	396.64	397.98	10/10	**394.94**	394.94	0.00	30/30	**394.94**	395.84	1.19	30/30	**394.94**	394.94	0.00	30/30
a4-40			**457.76**	3.77	458.52	467.60	461.16	461.91	10/10	458.63	462.73	3.34	30/30	**457.76**	465.13	3.85	30/30	457.88	458.67	1.20	30/30
a4-48			570.31	6.72	568.08	575.96	568.01	570.80	10/10	565.04	569.97	3.00	30/30	**561.15**	565.94	3.26	30/30	561.38	564.54	1.85	30/30
a5-40			**415.88**	2.50	419.33	425.29	418.79	421.06	10/10	415.96	417.24	1.68	30/30	**415.88**	425.07	3.79	30/30	**415.88**	416.43	1.05	30/30
a5-50			580.00	5.88	579.15	588.98	571.37	575.49	10/10	571.13	576.96	4.07	30/30	573.37	586.47	7.02	30/30	**567.61**	571.48	2.62	30/30

BI-LNS in terms of best found solutions. This demonstrates the advantages of our proposed OTF insertion procedure for charging station visits in the face of more restrictive battery requirements. Even though employing the MILP model yields more best known solutions, the heuristic performs better in terms of average objective values and gaps, having mean gaps in the range from −1.01% to 0.81%. This illustrates the overall better performance of the OTF heuristic. The average performance of the direct approach remains almost the same compared to setting $n_s = 1$, with a highest average gap of 2.05% for a3-36.

6.1.2 Ropke Instances. For the larger Ropke instances, we focus on the larger γ-values here but for completeness, we report the results for $\gamma = 0.1$ and both modes for charging visits in Appendix A.2 in Table 13. For $\gamma = 0.4$ and $\gamma = 0.7$, the respective results are given in Tables 3 and 4 in the same way as before except that for these instances, e-ADARP2 has not been tested. Instead, we also show run times for B&P and want to point out that this exact approach had a time limit of five hours on these instances. This is substantially more than for the considered heuristic approaches, which is why we include the B&P results only for completeness and focus on the heuristics in the following discussion. Compared to the smaller Cordeau instances, some similar trends are also visible for this benchmark set.

For example, considering $\gamma = 0.4$, BI-LNS now outperforms DA on all instances regardless of the mode for charging stops, despite the run time of DA exceeding the five minute limit employed by BI-LNS and our approaches on half of the instances. The increased size of the instances also substantially deteriorates the performance of the MILP-based OTF approach, which is competitive with DA for $n_s = 1$ but otherwise worse than all other approaches for limited as well as unlimited charging station visits. This illustrates that the underlying exact solving procedure for the charging

Table 3. Results on Ropke instances with $\gamma = 0.4$ for limited CS visits ($n_s = 1$) and unlimited CS visits ($n_s = \infty$).

Inst	B&P		DA			BI-LNS			Direct Approach				OTF Approach MILP				OTF Approach Heuristic			
	RT [min]	obj	$\overline{\text{RT}}$ [min]	$\text{obj}_{\min}$	$\overline{\text{obj}}$	$\text{obj}_{\min}$	$\overline{\text{obj}}$	Feas	$\text{obj}_{\min}$	$\overline{\text{obj}}$	σ_{obj}	Feas	$\text{obj}_{\min}$	$\overline{\text{obj}}$	σ_{obj}	Feas	$\text{obj}_{\min}$	$\overline{\text{obj}}$	σ_{obj}	Feas
$n_s = 1$																				
a5-60	288.02	**684.49**	4.89	697.97	718.44	688.16	693.43	10/10	685.32	695.79	4.77	30/30	691.49	703.42	5.68	30/30	685.51	690.63	2.94	30/30
a6-48	19.93	**506.45**	4.29	506.91	514.46	506.85	507.43	10/10	**506.45**	506.83	0.45	30/30	508.70	515.80	4.66	30/30	**506.45**	506.84	0.30	30/30
a6-60	42.02	**689.46**	2.89	694.78	706.07	692.69	695.13	10/10	691.71	695.93	3.19	30/30	698.53	705.90	5.77	30/30	690.29	693.16	2.31	30/30
a6-72	300.00	NA	5.83	799.60	821.17	771.97	778.30	10/10	772.35	784.49	5.57	30/30	783.18	795.86	8.00	30/30	**765.64**	776.00	4.47	30/30
a7-56	13.38	**612.02**	1.67	613.66	624.40	614.58	616.02	10/10	**612.01**	615.69	3.62	30/30	618.27	628.57	5.52	30/30	612.78	615.27	2.51	30/30
a7-70	227.74	**754.28**	4.56	766.05	784.54	761.62	765.71	10/10	755.60	764.08	4.37	30/30	763.06	778.86	8.93	30/30	756.46	760.21	2.03	30/30
a7-84	300.00	1081.76	9.74	932.12	NA	886.19	891.04	10/10	881.94	896.73	7.84	30/30	893.14	914.92	11.51	30/30	**878.99**	890.18	7.23	30/30
a8-64	50.50	**632.22**	10.69	638.36	652.30	637.95	640.41	10/10	632.80	639.80	3.11	30/30	646.28	655.15	5.41	30/30	632.95	639.02	3.05	30/30
a8-80	300.00	**788.99**	7.47	811.19	833.05	793.17	798.41	10/10	795.63	804.04	5.00	30/30	811.62	823.98	7.00	30/30	794.04	800.85	5.13	30/30
a8-96	NA	NA	10.29	NA	NA	1048.72	1057.47	10/10	1042.83	1058.75	9.51	30/30	1055.03	1080.91	10.41	30/30	**1036.22**	1047.47	5.62	30/30
$n_s = \infty$																				
a5-60	236.75	**683.58**	4.75	691.72	709.78	685.68	689.42	10/10	684.23	688.67	3.23	30/30	691.64	698.91	4.29	30/30	683.85	686.91	2.53	30/30
a6-48	20.43	**506.45**	4.26	507.25	514.64	506.85	507.30	10/10	**506.45**	506.89	0.36	30/30	507.82	518.25	5.12	30/30	**506.45**	506.87	0.22	30/30
a6-60	85.55	**689.46**	2.90	692.83	701.86	692.25	693.68	10/10	690.54	694.56	3.03	30/30	698.82	705.81	4.47	30/30	690.40	693.72	2.62	30/30
a6-72	300.00	762.65	5.71	781.22	801.86	774.38	778.84	10/10	765.60	773.88	5.29	30/30	784.50	797.62	9.52	30/30	**762.05**	772.91	4.45	30/30
a7-56	14.10	**612.02**	1.65	615.74	623.51	614.58	615.40	10/10	**612.01**	614.84	2.31	30/30	619.52	629.99	4.88	30/30	612.53	614.78	2.15	30/30
a7-70	177.18	**754.28**	4.56	761.58	778.04	762.78	764.51	10/10	757.23	761.79	3.26	30/30	767.89	784.46	9.43	30/30	757.01	760.15	2.23	30/30
a7-84	300.00	891.44	7.61	896.91	916.23	884.94	890.81	10/10	880.51	890.60	5.56	30/30	901.72	916.28	11.20	30/30	**877.98**	886.22	5.82	30/30
a8-64	95.95	**632.22**	11.99	637.84	652.17	637.95	640.30	10/10	633.73	640.31	4.10	30/30	649.89	662.91	8.31	29/29	**632.21**	639.48	3.15	30/30
a8-80	195.96	**788.99**	7.52	813.16	829.92	794.38	797.08	10/10	793.46	803.55	4.82	30/30	825.43	835.17	6.71	29/29	793.39	802.36	5.25	30/30
a8-96	300.00	1144.15	9.41	1058.41	1090.04	1048.45	1053.35	10/10	1039.49	1049.83	5.97	30/30	1068.79	1082.10	8.35	29/29	**1034.17**	1043.42	6.12	30/30

station insertion and evaluation subproblem does not scale well for larger instances. This is in stark contrast to the performance of the heuristic-based OTF approach, which outperforms all other heuristic approaches. In case of limited charging stops, it is superior in terms of average solution quality with mean gaps between −0.95% (on a8-96) and 1.5% (on a8-80), and it achieves three new best objective values and matches the optimal solution once despite its much lower run time than B&P. The direct approach finds optimal solutions for two instances and is on average competitive with BI-LNS with mean gaps of up to 1.91% (on a8-80). It performs better when allowing unlimited CS visits, exhibiting average gaps ranging from −0.33% to 1.85%, but is still dominated by the OTF heuristic. The latter finds again new best objective values for three instances and matches best known ones for two, having mean gaps as low as −0.94%.

For the more restrictive configuration of $\gamma = 0.7$, B&P and DA are not able to find any feasible solutions in case of limited CS visits. For this setting and on instances with an average of twelve requests per vehicle, none of the other compared approaches yield any feasible solutions either. However, the OTF heuristic and the direct approach obtain the first reported feasible results for instance a7-70, with the OTF heuristic being able to solve it in every trial. Regarding the other instances previously solved by BI-LNS, the OTF heuristic again consistently achieves better mean objective values with average gaps ranging from −3.04% to −0.12%, and the direct approach still exhibits a competitive performance with 1.10% being the highest mean gap. The performance of the MILP-based OTF approach is similar to before, rendering it noncompetitive for both charging modes. Considering $n_s = \infty$, the direct approach yields better average objective values than DA and BI-LNS on all instances but is still significantly outperformed by the OTF heuristic. The OTF heuristic dominates on this configuration as it not only has the best mean objective values over all heuristic approaches but also provides new best objective values for half of the instances. These results impressively highlight its ability to optimize charging station visits within a reasonable time.

6.1.3 Very Large Instances. As the very large instances with up to 260 vehicles and 5200 requests are orders of magnitude larger than the other considered benchmark instances, they are utilized to investigate the scalability of our proposed approaches. Preliminary tests indicated that a restriction

Table 4. Results on Ropke instances with $\gamma = 0.7$ for limited CS visits ($n_s = 1$) and unlimited CS visits ($n_s = \infty$).

Inst	B&P		DA			BI-LNS			Direct Approach				OTF Approach MILP				OTF Approach Heuristic			
	RT [min]	obj	$\overline{\text{RT}}$ [min]	$\text{obj}_{\min}$	$\overline{\text{obj}}$	$\text{obj}_{\min}$	$\overline{\text{obj}}$	Feas	$\text{obj}_{\min}$	$\overline{\text{obj}}$	σ_{obj}	Feas	$\text{obj}_{\min}$	$\overline{\text{obj}}$	σ_{obj}	Feas	$\text{obj}_{\min}$	$\overline{\text{obj}}$	σ_{obj}	Feas
$n_s = 1$																				
a5-60			8.46	NA	NA	NA	NA	0/10	NA	NA	NA	0/30	NA	NA	NA	0/30	NA	NA	NA	0/30
a6-48			8.37	NA	NA	519.55	522.50	10/10	517.60	528.24	6.81	30/30	523.49	537.45	9.06	30/30	**517.12**	521.62	2.95	30/30
a6-60			5.45	NA	NA	733.45	742.02	9/10	719.78	737.77	10.75	30/30	736.52	750.14	11.54	10/30	**714.16**	731.45	10.02	30/30
a6-72			9.84	NA	NA	NA	NA	0/10	NA	NA	NA	0/30	NA	NA	NA	0/30	NA	NA	NA	0/30
a7-56			3.69	NA	NA	649.11	669.71	10/10	**633.65**	656.45	12.31	30/30	656.86	675.48	15.31	23/30	636.56	649.37	10.43	30/30
a7-70			8.51	NA	NA	NA	NA	0/10	820.90	840.02	9.75	26/30	NA	NA	NA	0/30	**816.64**	840.59	16.29	30/30
a7-84			13.18	NA	NA	NA	NA	0/10	NA	NA	NA	0/30	NA	NA	NA	0/30	NA	NA	NA	0/30
a8-64			20.12	NA	NA	646.82	652.38	10/10	645.69	656.21	8.63	30/30	652.65	674.25	15.43	30/30	**639.06**	651.33	6.63	30/30
a8-80			14.47	NA	NA	854.85	863.74	10/10	842.23	863.08	10.24	30/30	869.19	888.37	15.02	8/30	**837.79**	862.75	15.14	30/30
a8-96			14.35	NA	NA	NA	NA	0/10	NA	NA	NA	0/30	NA	NA	NA	0/30	NA	NA	NA	0/30
$n_s = \infty$																				
a5-60	300.00	688.84	8.21	708.54	723.73	697.87	709.11	10/10	693.79	703.04	4.83	30/30	695.81	704.15	5.31	30/30	**686.36**	692.75	4.39	30/30
a6-48	33.80	**508.10**	8.07	509.76	525.10	511.04	514.53	10/10	508.22	510.85	2.47	30/30	509.87	519.44	5.22	30/30	**508.10**	509.46	1.88	30/30
a6-60	110.08	**689.55**	4.83	697.57	711.52	699.70	705.56	10/10	694.19	700.72	3.36	30/30	699.25	708.04	5.09	30/30	689.95	695.13	3.43	30/30
a6-72	300.00	788.83	9.57	796.19	826.48	788.34	801.80	10/10	780.68	796.83	8.00	30/30	782.32	799.84	11.52	30/30	**769.12**	779.49	4.46	30/30
a7-56	237.46	**616.16**	3.53	625.91	641.82	627.34	633.38	10/10	619.63	627.64	4.69	30/30	623.45	632.08	3.91	30/30	617.12	621.51	3.02	30/30
a7-70	300.00	765.49	8.00	781.56	800.35	777.69	785.49	10/10	763.22	773.36	5.76	30/30	769.84	783.03	8.58	30/30	**757.66**	765.36	3.57	30/30
a7-84	300.00	1004.81	11.75	915.61	938.49	900.98	916.93	10/10	892.31	906.41	6.19	30/30	900.77	920.03	9.99	30/30	**888.40**	897.33	6.44	30/30
a8-64	82.70	**632.61**	21.50	649.93	668.48	645.62	648.60	10/10	634.36	646.57	5.03	30/30	654.75	663.89	6.35	30/30	632.95	641.17	3.77	30/30
a8-80	250.86	**794.41**	12.41	843.26	865.90	815.06	825.74	10/10	813.44	823.82	6.09	30/30	821.29	837.30	7.16	30/30	801.08	814.87	6.25	30/30
a8-96	300.00	1174.49	13.45	1097.76	1136.43	1072.77	1091.06	10/10	1058.08	1073.35	9.09	30/30	1074.02	1091.18	11.36	30/30	**1048.87**	1060.22	6.23	30/30

Table 5. Results and statistics on large instances with unlimited CS visits, $\gamma = 0.7$, and a time limit of five minutes: I^{total} denotes the total number of iterations and I^{last} the iteration in which the last improvement was found, both averaged over all runs resulting in a feasible solution.

Inst	BI-LNS					Direct Approach						OTF Approach Heuristic					
	$\text{obj}_{\min}$	$\overline{\text{obj}}$	I^{total}	I^{last}	Feas	$\text{obj}_{\min}$	$\overline{\text{obj}}$	σ_{obj}	I^{total}	I^{last}	Feas	$\text{obj}_{\min}$	$\overline{\text{obj}}$	σ_{obj}	I^{total}	I^{last}	Feas
a180-3600	**29793.51**	29977.80	270	269	10/10	37442.76	37963.02	213.60	237	230	30/30	37891.94	38338.53	191.57	77	75	30/30
a200-4000	**32803.99**	33161.17	289	289	10/10	41840.86	42399.92	231.89	203	190	30/30	42373.76	42776.22	183.55	64	60	30/30
a220-4400	**36365.84**	37523.84	311	311	10/10	46316.63	46736.61	171.41	174	150	30/30	46683.47	47073.41	194.30	52	47	30/30
a240-4800	**41357.57**	42150.46	334	334	2/10	50830.44	51288.25	203.03	152	123	30/30	51330.36	51678.20	238.18	40	35	30/30
a260-5200	NA	NA	NA	NA	0/10	**55195.55**	55680.75	208.25	135	110	30/30	55761.89	56170.95	231.20	30	25	30/30

to $n_s = 1$ renders these instances infeasible, thus in the following, we investigate the mode with unlimited CS visits and the largest minimum end battery level ratio of $\gamma = 0.7$, as seen in [Limmer 2023]. Concerning the preprocessing, the path-based elimination rules and the corresponding evaluation of all occurring fragments turned out to be too expensive regarding computation time as well as the required memory; we therefore disabled them. Moreover, in order to trade quality for higher speed within the LNS repair operators, we limit the number of considered routes for each request insertion to a maximum amount $r^{\max} = 50$. These candidate routes are randomly selected. Only if no feasible insertion position is found within this limit for a request, further routes are considered until either a feasible insertion is found or all routes have been tried.

Tables 5 and 6 show the results of BI-LNS, our direct approach, and the OTF-based heuristic when using the usual time limit of five minutes and an extended run time of 15 minutes, respectively. In addition to the previously shown metrics, we also report mean values of the total number of iterations I^{total} that the LNS-based approaches performed as well as the iteration I^{last} in which the last improvement was achieved, both averaged over all runs yielding a feasible solution. For BI-LNS, these numbers refer to the outer-level iterations as the number of inner-level iterations, that are executed within each outer level, is fixed. The longer run time together with the iteration numbers are used to gain insights on the convergence of the approaches. No results are given for the MILP-based OTF approach as its run time already exceeded 15 minutes for the preprocessing and

Table 6. Results and statistics on large instances with unlimited CS visits, $\gamma = 0.7$, and a time limit of 15 minutes: I^{total} denotes the total number of iterations and I^{last} the iteration in which the last improvement was found, both averaged over all runs resulting in a feasible solution.

Inst	BI-LNS					Direct Approach						OTF Approach Heuristic					
	obj_{min}	$\overline{\text{obj}}$	I^{total}	I^{last}	Feas	obj_{min}	$\overline{\text{obj}}$	σ_{obj}	I^{total}	I^{last}	Feas	obj_{min}	$\overline{\text{obj}}$	σ_{obj}	I^{total}	I^{last}	Feas
a180-3600	**29177.96**	29312.55	296	295	10/10	35928.50	36314.79	156.92	1381	1375	30/30	35962.66	36520.18	321.60	567	564	30/30
a200-4000	**32013.56**	32263.01	311	311	10/10	40844.17	41163.38	180.11	1351	1340	30/30	41070.58	41509.23	304.53	515	512	30/30
a220-4400	**35259.06**	35428.04	332	331	10/10	45624.53	46061.85	226.02	1316	1300	30/30	46040.15	46367.00	170.38	485	478	30/30
a240-4800	**38270.98**	38460.95	386	385	10/10	50684.67	50993.61	151.75	1276	1255	30/30	50776.06	51247.17	237.25	447	439	30/30
a260-5200	**41472.11**	41745.98	407	407	10/10	55186.42	55515.55	205.59	1219	1159	30/30	55344.22	55802.20	221.39	402	396	30/30

initial solution construction alone, while not yielding better solutions than the other approaches. This confirms once more that this approach is not suitable for solving larger instances.

As can be seen from the results, BI-LNS clearly outperforms our approaches on these instances in terms of solution quality. However, for a run time limit of five minutes, BI-LNS struggles to solve the largest two instances, while both the direct and the OTF heuristic-based approaches manage to find valid solutions to all instances in all runs, which illustrates their capability of quickly reaching feasibility. This seems to be due to the more sophisticated procedures for charging stop insertion compared to BI-LNS, which needs most of its iterations to construct feasible routes [Limmer 2023]. Nevertheless, as soon as feasibility is established, BI-LNS achieves better results, which may indicate that the infeasibility stems from some outer-level charging decisions while the inner-level request optimization has already constructed high quality sequences for most routes. In contrast to the results on the other benchmark sets, the direct approach now performs significantly better than the OTF heuristic. One of the reasons seems to be the higher number of iterations that the direct approach is able to execute, providing it more opportunities for improvement. We observe that the average length of a route increases substantially on these larger instances, which has a larger impact on the run time of the OTF CS insertions than on the labeling algorithm employed by the direct approach, thus explaining the performance drop. For BI-LNS, on the other hand, longer routes barely influence charging related operations. Overall, this shows that our proposed approaches for charging stop optimization and route evaluation together with the employed greedy insertion mechanism for requests, which necessitates many evaluations of candidate routes, are too time-consuming to scale well for such very large instances.

Considering a time limit of 15 minutes, all approaches yield improved solutions as shown in Table 6 but their performance in relation to each other is the same as for the shorter time frame, except that BI-LNS now also feasibly solves all instances in all trials. An interesting observation concerns the executed iterations, as the number only slightly increases for BI-LNS, with only between 21 to 52 more iterations, whereas the increase is substantially higher for the direct approach and the OTF heuristic. The direct approach executes between five to nine times more iterations and the OTF heuristic achieves up to 13 times as many as before. For BI-LNS, this is explained by an increased computational effort for the inner-level iterations after finding a first feasible solution [Limmer 2023]. The jump in iteration numbers for our proposed approaches is linked to the initial solution construction, which is done by applying one of the repair operators to empty routes (cf. Section 4.1). Due to the large number of requests in these instances, this results in many executions of the greedy request insertion and charging stop optimization, which is costly in terms of time as mentioned above. Thus, a considerable portion of the first five minutes is spent on constructing a first solution, which is why relatively few LNS iterations are executed, but with an increased limit of 15 minutes, much more time is available for LNS iterations. Despite this increase

Table 7. Parameter configurations of WDR operator variants.

Parameter	WDR 000	WDR 010	WDR 100	WDR 110	WDR 001	WDR 011	WDR 101	WDR 111
p^{df}	-	-	-	-	2	2	2	2
κ^{wdr}	40	40	26	32	15	15	15	15

in iterations, all approaches are able to find improvements up until the time limit is reached as shown by the recorded values of I^{last} in comparison to I^{total}. This indicates not only that further improvements in solution quality are possible but also that in principle, our proposed approaches have the potential of closing the gap to BI-LNS. A first step towards this is the application of more sophisticated LNS operators for request optimization such as the worst-distance removal, which is discussed below.

6.1.4 Integration of Worst-Distance Removal. So far, the underlying LNS of all proposed approaches has employed uniform random removal as destroy method for requests. We now enhance our OTF heuristic-based approach with variants of the WDR operator described in Section 4.3, as the OTF heuristic performs best on the Cordeau and Ropke benchmark sets and also has a promising performance on the very large instances. To distinguish between the different WDR variants, we use the following naming scheme based on the operator's parameterization: $p^{\mathrm{DD}}p^{\mathrm{CS}}p^{\mathrm{R}}_p^{\mathrm{df}}_\kappa^{\mathrm{wdr}}$. Thus, for example, WDR(101_2_15) refers to the randomized variant using the detour distance cost and considering charging stops during the cost computation with a determinism factor $p^{\mathrm{df}} = 2$ and degree of destruction $\kappa^{\mathrm{wdr}} = 15$. The respective parameter values are prespecified, and κ^{wdr} and p^{df} are determined as detailed below.

Parameter Tuning. First, we applied the same degree of destruction as for the RR operator; i.e., $\kappa^{\mathrm{rr}} = \kappa^{\mathrm{wdr}} = 15$, and tentatively set the determinism factor $p^{\mathrm{df}} = 2$ for the randomized variants according to results from preliminary tests. Further tests made clear that this operator could benefit from a different parameterization, which we found by repeating the same tuning procedure as for the RR operator (cf. Section 6) for each of the WDR variants. We then tested all variants with their original parameter values as well as with the tuned values on the benchmark instances and compared the performance of the original parameterization to the tuned one using pairwise one-sided Wilcoxon signed-rank tests with a threshold $p \leq 0.05$. This showed that the tuning works well for the deterministic operator variants, as they yield significantly better results compared to when using the original parameter values, but did not lead to improvements for the randomized versions. The final configurations further used are given in Table 7.

Results on Cordeau and Ropke Instances. To gain insights into the performance of the different WDR variants, we separately integrate each one into the OTF heuristic-based approach by replacing the RR operator and test the resulting eight approaches on all configurations of the Cordeau and Ropke instances. We compare them to each other and also to the original OTF heuristic and the direct approach for each instance configuration using again pairwise one-sided Wilcoxon signed-rank tests with $p \leq 0.05$ as exemplarily shown in Table 8 for the Ropke instances with a minimum end battery level ratio $\gamma = 0.7$ and unlimited visits to charging stations. For the other configurations as well as the Cordeau benchmark set, similar values are mostly observed. The results illustrate that there is a significant difference in the performance of the deterministic and the randomized WDR variants. The deterministic variants perform the worst out of all approaches including the OTF heuristic with random removal, with the detour distance-based ones yielding

Table 8. Obtained p-values of pairwise one-sided Wilcoxon signed-rank tests on final solution qualities of the OTF heuristic, the direct approach, and the OTF heuristic with different variants of the WDR operator on the Ropke instances with $\gamma = 0.7$ and $n_s = \infty$. The WDR variants follow the naming scheme $p^{\mathrm{DD}}p^{\mathrm{CS}}p^{\mathrm{R}}_p^{\mathrm{df}}_\kappa^{\mathrm{wdr}}$. Values ≤ 0.05 shown with shaded background indicate that the configuration corresponding to the row performs significantly better than the method corresponding to the column.

p-value	OTF Heuristic	Direct Approach	WDR 000_-_40	WDR 010_-_40	WDR 100_-_26	WDR 110_-_32	WDR 001_2_15	WDR 011_2_15	WDR 101_2_15	WDR 111_2_15
OTF Heuristic	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.9993	0.9775	0.9994	0.9997
Direct Approach	1.0000	1.0000	0.0000	0.0000	0.0000	0.0000	1.0000	1.0000	1.0000	1.0000
WDR 000_-_40	1.0000	1.0000	1.0000	0.0657	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
WDR 010_-_40	1.0000	1.0000	0.9343	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
WDR 100_-_26	1.0000	1.0000	0.0000	0.0000	1.0000	0.9967	1.0000	1.0000	1.0000	1.0000
WDR 110_-_32	1.0000	1.0000	0.0000	0.0000	0.0033	1.0000	1.0000	1.0000	1.0000	1.0000
WDR 001_2_15	0.0007	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.3489	0.3871	0.7081
WDR 011_2_15	0.0225	0.0000	0.0000	0.0000	0.0000	0.0000	0.6511	1.0000	0.7572	0.9484
WDR 101_2_15	0.0006	0.0000	0.0000	0.0000	0.0000	0.0000	0.6129	0.2428	1.0000	0.8147
WDR 111_2_15	0.0003	0.0000	0.0000	0.0000	0.0000	0.0000	0.2919	0.0516	0.1853	1.0000

slightly better results than when using the base cost computation. We found that the deterministic element selection resulted too often in the same requests being chosen in successive LNS iterations leading to few solution improvements overall, thus confirming the need for some randomness to preserve diversification capabilities of the search. In contrast, the randomized operator versions significantly outperform not only the deterministic ones but also our original approaches while having no clear winner among each other. This shows that on these instances, randomization parameter p^{R} is the most impactful one for WDR.

The corresponding results of the WDR-based approaches with $p^{\mathrm{R}} = 1$ on the Ropke instances with $\gamma = 0.7$ are shown in Table 9 together with the best known solutions (BKSs) for this instance configuration taken from the results reported above (cf. Section 6.1.2). Regarding the case of at most one visit per charging station, all approaches are able to solve the same instances as the base OTF heuristic in all trials with two exceptions. First, WDR(001_2_15) did not yield a feasible solution for a7-70 in one run, although achieving an unmatched new best objective value for this instance. Secondly, instance a6-72 is solved for the first time regarding this instance configuration by WDR(101_2_15), which was able to find a valid solution in one of its runs. Three instances remain unsolved, but for the six previously solved ones, five new best solutions are achieved by different WDR variants. This demonstrates that despite the base OTF heuristic's excellent performance on these instances, it can still be improved upon by better guidance of the search process. All new approaches have mean gaps in the range from −1.10% to 0.48% and overall, none of them performs significantly better than the others. These observations also hold true when allowing unlimited charging stops, as, for example, the BKS is improved upon for two out of ten instances and matched for another four, again by different WDR parameterizations. Although WDR(001_2_15) finds the most (new) best solutions, it does not perform better than the other approaches on average.

Results on Very Large Instances. Analogous to the tests on the other benchmark sets, we investigate the performance of the WDR operator variants on the very large instances, and like in Section 6.1.3, we focus on the configuration with unlimited CS visits and $\gamma = 0.7$. The p-values of the corresponding pairwise one-sided Wilcoxon signed-rank tests are given in Table 10 and show contrasting results to the ones observed for the Ropke instances. Again, we have a significant performance difference between the deterministic and randomized variants but this time, the deterministic WDR versions outperform the others as well as the original OTF heuristic and also the direct approach. Despite the deterministic element selection still choosing often the same requests during

Table 9. Results of OTF heuristic with randomized WDR operator variants on Ropke instances with $\gamma = 0.7$ for limited CS visits ($n_s = 1$) and unlimited CS visits ($n_s = \infty$).

Inst	BKS	WDR 001_2_15				WDR 011_2_15				WDR 101_2_15				WDR 111_2_15			
		obj_{min}	$\overline{obj}$	σ_{obj}	Feas	obj_{min}	$\overline{obj}$	σ_{obj}	Feas	obj_{min}	$\overline{obj}$	σ_{obj}	Feas	obj_{min}	$\overline{obj}$	σ_{obj}	Feas
$n_s = 1$																	
a5-60	NA	NA	NA	NA	0/30	NA	NA	NA	0/30	NA	NA	NA	0/30	NA	NA	NA	0/30
a6-48	517.12	516.19	521.04	3.69	30/30	517.12	521.38	3.94	30/30	**515.14**	520.22	2.29	30/30	515.20	521.28	4.14	30/30
a6-60	**714.16**	718.81	734.93	8.02	30/30	720.67	733.48	7.24	30/30	718.12	733.03	8.20	30/30	717.63	730.56	6.30	30/30
a6-72	NA	NA	NA	NA	0/30	NA	NA	NA	0/30	**885.13**	885.13	NA	1/30	NA	NA	NA	0/30
a7-56	633.65	**632.99**	648.50	8.88	30/30	**632.99**	651.12	8.59	30/30	**632.99**	646.83	6.85	30/30	636.56	649.81	7.77	30/30
a7-70	816.64	**811.16**	834.95	14.70	29/30	811.71	831.88	11.98	30/30	815.48	830.82	10.27	30/30	814.58	834.75	10.30	30/30
a7-84	NA	NA	NA	NA	0/30	NA	NA	NA	0/30	NA	NA	NA	0/30	NA	NA	NA	0/30
a8-64	639.06	639.12	650.26	6.11	30/30	640.30	650.89	6.31	30/30	637.66	649.11	5.52	30/30	**637.29**	646.59	4.04	30/30
a8-80	837.79	837.76	858.47	8.86	30/30	842.86	856.66	8.83	30/30	845.14	860.72	11.51	30/30	**836.36**	855.81	11.24	30/30
a8-96	NA	NA	NA	NA	0/30	NA	NA	NA	0/30	NA	NA	NA	0/30	NA	NA	NA	0/30
$n_s = \infty$																	
a5-60	**686.36**	686.40	693.07	4.19	30/30	687.68	695.08	5.23	30/30	**686.36**	692.70	3.83	30/30	686.85	693.68	4.19	30/30
a6-48	**508.10**	**508.10**	508.78	0.77	30/30	**508.10**	508.97	1.88	30/30	**508.10**	509.07	1.20	30/30	**508.10**	508.90	1.34	30/30
a6-60	**689.55**	**689.55**	694.97	2.54	30/30	691.19	694.95	2.60	30/30	690.58	694.55	3.33	30/30	690.55	694.83	2.56	30/30
a6-72	**769.12**	771.47	780.42	4.82	30/30	771.65	781.79	5.15	30/30	770.23	780.68	5.67	30/30	770.75	779.19	4.41	30/30
a7-56	**616.16**	617.35	620.64	2.42	30/30	616.51	620.61	2.21	30/30	616.51	620.39	2.21	30/30	617.03	620.70	2.09	30/30
a7-70	**757.66**	758.06	765.60	3.74	30/30	757.89	764.95	3.37	30/30	759.08	765.88	4.45	30/30	758.26	764.63	3.62	30/30
a7-84	888.40	**881.00**	892.56	5.44	30/30	881.95	892.56	5.25	30/30	883.46	893.34	5.41	30/30	885.14	894.17	4.57	30/30
a8-64	**632.61**	**632.61**	641.16	4.77	30/30	634.28	640.27	3.04	30/30	**632.61**	638.45	3.99	30/30	**632.61**	638.65	3.83	30/30
a8-80	**794.41**	799.27	810.29	5.40	30/30	804.08	811.02	4.19	30/30	802.73	812.76	4.79	30/30	803.25	811.64	5.17	30/30
a8-96	1048.87	**1041.54**	1055.81	8.87	30/30	1045.41	1057.15	6.38	30/30	1044.49	1056.76	6.54	30/30	1045.42	1054.49	6.51	30/30

Table 10. Obtained p-values of pairwise one-sided Wilcoxon signed-rank tests on final solution qualities of the OTF heuristic, the direct approach, and the OTF heuristic with different variants of the WDR operator on the very large instances with $\gamma = 0.7$ and $n_s = \infty$. The WDR variants follow the naming scheme $p^{DD}p^{CS}p^{R}_p^{df}_\kappa^{wdr}$. Values ≤ 0.05 shown with shaded background indicate that the configuration corresponding to the row performs significantly better than the method corresponding to the column.

p-value	OTF Heuristic	Direct Approach	WDR 000_-_40	WDR 010_-_40	WDR 100_-_26	WDR 110_-_32	WDR 001_2_15	WDR 011_2_15	WDR 101_2_15	WDR 111_2_15
OTF Heuristic	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Direct Approach	0.00	1.00	1.00	1.00	1.00	1.00	0.00	0.00	1.00	0.97
WDR 000_-_40	0.00	0.00	1.00	0.91	1.00	1.00	0.00	0.00	0.63	0.14
WDR 010_-_40	0.00	0.00	0.09	1.00	1.00	1.00	0.00	0.00	0.27	0.03
WDR 100_-_26	0.00	0.00	0.00	0.00	1.00	1.00	0.00	0.00	0.00	0.00
WDR 110_-_32	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
WDR 001_2_15	0.00	1.00	1.00	1.00	1.00	1.00	1.00	0.73	1.00	1.00
WDR 011_2_15	0.00	1.00	1.00	1.00	1.00	1.00	0.27	1.00	1.00	1.00
WDR 101_2_15	0.00	0.00	0.37	0.73	1.00	1.00	0.00	0.00	1.00	0.03
WDR 111_2_15	0.00	0.03	0.86	0.97	1.00	1.00	0.00	0.00	0.98	1.00

the search, this appears to be beneficial for the LNS on such very large-scale instances. Regardless of the used (in-)determinism, the variants with $p^{DD} = 1$ employing the detour distance-based cost computation perform significantly better than the base cost computation, whereas for the parameter p^{CS} controlling the consideration of charging stops no statistically significant influence can be observed in general, although a trend towards better neglecting them for better results can be seen. Overall, both variants with deterministic selection and detour distance cost metric exhibit significant differences in performance compared to all other approaches.

Table 11 shows the results of the original RR-based and all WDR-based variants of the OTF heuristic approach given a time limit of five minutes on the very large instances. The best known solutions and mean objective values as reported in Section 6.1.3 are also repeated for comparison.

Table 11. Results of original OTF heuristic and with WDR operator variants on the very large instances with unlimited CS visits, $\gamma = 0.7$, and a time limit of five minutes.

Solving Approach	a180-3600			a200-4000			a220-4400			a240-4800			a260-5200		
	obj_{min}	$\overline{obj}$	σ_{obj}	obj_{min}	$\overline{obj}$	σ_{obj}	obj_{min}	$\overline{obj}$	σ_{obj}	obj_{min}	$\overline{obj}$	σ_{obj}	obj_{min}	$\overline{obj}$	σ_{obj}
BKS	**29177.96**	29312.55		**32013.56**	32263.01		**35259.06**	35428.04		**38270.98**	38460.95		**41472.11**	41745.98	
OTF Heuristic	37891.94	38338.53	191.57	42373.76	42776.22	183.55	46683.47	47073.41	194.30	51330.36	51678.20	238.18	55761.89	56170.95	231.20
WDR 000_-_40	37620.29	38038.99	237.82	41907.25	42331.24	225.46	46041.21	46606.51	284.39	49769.68	50891.76	357.29	54884.52	55455.86	310.78
WDR 010_-_40	37455.37	37912.11	221.64	41649.03	42268.74	269.27	45804.56	46482.76	219.99	50562.39	50980.24	237.29	54843.76	55517.45	383.01
WDR 100_-_26	36812.85	37457.72	310.51	40895.80	41611.34	266.32	44879.98	45796.64	349.67	49479.46	50107.04	446.13	53403.21	54376.72	373.60
WDR 110_-_32	36822.21	37321.46	244.62	*40693.01	41422.04	277.09	*44833.27	45503.79	329.55	*49353.91	49981.45	328.48	*52954.50	54066.66	545.15
WDR 001_2_15	37418.99	37882.12	230.87	42028.14	42503.93	189.32	46568.87	46969.30	194.31	51206.03	51622.70	190.61	55677.59	56071.09	200.80
WDR 011_2_15	37458.14	37883.58	225.84	42139.97	42492.14	275.55	46447.32	46893.10	226.77	51058.64	51654.92	256.95	55746.83	56066.15	203.40
WDR 101_2_15	*36613.00	37300.50	320.07	41435.36	41951.16	281.94	46048.55	46616.36	248.49	51014.22	51452.69	233.48	55238.55	55941.29	285.86
WDR 111_2_15	36623.89	37302.76	379.39	41431.03	42057.49	266.72	46140.99	46663.25	227.57	50728.60	51492.90	352.96	55415.08	56118.24	317.76

The number of feasible runs is not listed as every approach yields a valid solution in all 30 trials for each instance. The BKS is marked in bold like before and an asterisk (*) indicates the best objective value found by one of our proposed approaches. This illustrates that while our enhanced approaches still do not achieve a comparable solution quality, they consistently outperform the base OTF heuristic in terms of both minimum and average objective values. The results further confirm our observation above that the combination of deterministic selection and detour distance-based cost computation performs best with WDR(110_-_32) being slightly better than WDR(100_-_26). The only exception occurs for the smallest of these instances, a180-3600, where the corresponding randomized variants come out on top. Overall, the WDR-based solving approaches have mean gaps of up to 34.43% and are far from convergence, which is a reason why the deterministic operator variants perform so well. At this stage in the search, there are still lots of opportunities for improvement which are further amplified by our approach of only considering a limited amount of randomly selected routes in the repair methods for the very large-scale instances (cf. Section 6.1.3). Thus, it is to be expected that at a later stage, when incumbent routes are already of high quality, the randomized variants will perform better. As of now, we have shown that employing more targeted destroy operators than random removal within our OTF heuristic approach significantly improves solution quality.

7 Conclusion and Future Work

This work proposes a structurally simple large neighborhood search framework for the electric autonomous dial-a-ride problem where battery-restricted fragments are evaluated on demand and cached, thus enabling an efficient optimization of user excess ride times. We introduce two approaches to handle the insertion of appropriate charging stops and the route evaluation and scheduling, two challenges specific to the E-ADARP. The OTF approach provides a highly efficient, novel route evaluation procedure that inserts and schedules charging stops on-the-fly by applying either a MILP formulation or a time-efficient heuristic. This procedure is generally applicable and can be used irrespective of the metaheuristic framework. The direct approach employs dedicated LNS operators for charging stop optimization and an exact, fragment-based labeling algorithm with linear run time for evaluation. For request optimization, the underlying LNS of the proposed approaches utilizes a simple random removal procedure and three repair operators.

In computational experiments, the performance of the proposed approaches is first evaluated on two sets of common benchmark instances and compared to exact as well as heuristic approaches from the literature; i.e., the mixed integer linear programming based e-ADARP2 of Bongiovanni et al. [2019], the branch-and-price (B&P) algorithm from Su et al. [2024], the deterministic annealing (DA) algorithm by Su et al. [2023], and the bilevel large neighborhood search (BI-LNS) from Limmer [2023]. We consider six different instance configurations based on three options for the employed

minimum end battery level ratio γ and two alternative modes for charging station visits, which together influence not only the importance of charging stops but also the practical hardness of the instances. On the smaller instances with up to five vehicles and 50 requests and for the two smaller γ-values, the direct approach and OTF heuristic outperform the established heuristic approaches and mostly match the optimal solutions of e-ADARP2 and B&P, whereas the MILP-based OTF procedure performs slightly worse. For the largest considered γ, both OTF approaches but especially the OTF heuristic perform best on most instances, while the direct approach remains competitive. Considering the set of larger instances with at most eight vehicles and 96 requests, the direct approach beats BI-LNS and DA but is still clearly outperformed by the OTF heuristic. The latter performs better on almost all instances regardless of which configuration is used, finding new best known solutions for 19 out of 60 instances and matching optimal ones on another eight instances. The OTF MILP approach struggles to find high quality solutions for this instance size and thus, it is not further considered for the very large-scale instances with up to 260 vehicles and 5200 requests that are utilized to investigate the scalability of the other two proposed approaches in comparison to BI-LNS. In terms of solution quality, BI-LNS outperforms the OTF heuristic and the direct approach on these instances, with the direct approach yielding better results than the OTF heuristic. In contrast to BI-LNS, both proposed approaches are able to find feasible solutions for all instances within a time limit of five minutes. Tests with longer run times indicate that further improvements in solution quality are possible and a first step towards closing the gap is the enhancement of the OTF heuristic-based approach with different variants of the more targeted worst-distance removal operator. For the two smaller benchmark sets, this better guidance of the search process leads to a further increase in solution quality yielding multiple new best known solutions and even a newly solved instance. While the randomized operator variants are preferable on these instances, the deterministic ones perform in general better on the very large instances. Although the WDR-based OTF heuristic approaches still do not reach the solution quality of BI-LNS on these large-scale instances, they consistently outperform the base OTF heuristic.

Regarding future work, enhancing our basic LNS framework with different acceptance criteria such as the Metropolis criterion or towards adaptive LNS with more advanced techniques for operator selection has the potential to further improve its performance. The algorithmic approaches presented in this paper, such as the on-the-fly charging stop insertion and route evaluation, the fragment caching and fragment-based labeling, could also be of interest in the context of other electric vehicle routing problems. Similarly, investigation of these concepts as well as the employed operators within different metaheuristic frameworks like variable neighborhood search would deepen the insights on their applicability. Further improving the scalability of our proposed approaches by tackling the bottleneck of greedy request insertion and route evaluation is also interesting future work. One improvement opportunity can be a reduction of the search space and possible insertion positions, e.g., by graph sparsification and pruning as a preprocessing step or by biasing also the insertion operators to look at most promising insertion options first. For this, the consideration of machine learning techniques for learning and quickly predicting how likely an edge is to be part of a high quality solution is an interesting research direction. A further step can be to learn a model for dynamically deciding which operator is most promising to apply with which probability and which configuration at which state of the whole optimization process; this relates to the field of hyper-heuristics. In respect to real-world applications, the consideration of non-linear charging rates and a more realistic modeling of charging stations with different maximum charging rates and visitor constraints constitute further highly relevant research lines. Last but not least, an extension to the dynamic (online) problem variant, where customer requests only become available over time or changes in the environment such as varying travel times need to be considered, are further challenging research directions.

References

Claudia Bongiovanni, Nikolas Geroliminis, and Mor Kaspi. 2024. A ride time-oriented scheduling algorithm for dial-a-ride problems. *Computers & Operations Research* 165 (2024), 106588. https://doi.org/10.1016/j.cor.2024.106588

Claudia Bongiovanni, Mor Kaspi, Jean-François Cordeau, and Nikolas Geroliminis. 2022. A machine learning-driven two-phase metaheuristic for autonomous ridesharing operations. *Transportation Research Part E: Logistics and Transportation Review* 165 (2022), 102835. https://doi.org/10.1016/j.tre.2022.102835

Claudia Bongiovanni, Mor Kaspi, and Nikolas Geroliminis. 2019. The electric autonomous dial-a-ride problem. *Transportation Research Part B: Methodological* 122 (2019), 436–456. https://doi.org/10.1016/j.trb.2019.03.004

Kris Braekers, An Caris, and Gerrit K. Janssens. 2014. Exact and meta-heuristic approach for a general heterogeneous dial-a-ride problem with multiple depots. *Transportation Research Part B: Methodological* 67 (2014), 166–186. https://doi.org/10.1016/j.trb.2014.05.007

Kris Braekers and Attila A. Kovacs. 2016. A multi-period dial-a-ride problem with driver consistency. *Transportation Research Part B: Methodological* 94 (2016), 355–377. https://doi.org/10.1016/j.trb.2016.09.010

Maria Bresich, Günther R. Raidl, and Steffen Limmer. 2024. Letting a Large Neighborhood Search for an Electric Dial-A-Ride Problem Fly: On-The-Fly Charging Station Insertion. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Melbourne, VIC, Australia) *(GECCO '24)*. Association for Computing Machinery, New York, NY, USA, 142–150. https://doi.org/10.1145/3638529.3654057

Maria Bresich, Günther R. Raidl, and Steffen Limmer. 2025. Improvements in Large Neighborhood Search for the Electric Autonomous Dial-A-Ride Problem. In *Computer Aided Systems Theory – EUROCAST 2024 (Lecture Notes in Computer Science (LNCS), Vol. 15172)*, Alexis Quesada-Arencibia, Michael Affenzeller, and Roberto Moreno-Díaz (Eds.). Springer, Cham, 211–220. https://doi.org/10.1007/978-3-031-82949-9_20

Jean-François Cordeau. 2006. A Branch-and-Cut Algorithm for the Dial-a-Ride Problem. *Operations Research* 54, 3 (2006), 573–586. https://doi.org/10.1287/opre.1060.0283

Jean-François Cordeau and Gilbert Laporte. 2003. A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research Part B: Methodological* 37, 6 (2003), 579–594. https://doi.org/10.1016/S0191-2615(02)00045-0

Claudio Cubillos, Enrique Urra, and Nibaldo Rodríguez. 2009. Application of Genetic Algorithms for the DARPTW Problem. *International Journal of Computers, Communications & Control* 4, 22 (2009), 127–136. https://doi.org/10.15837/ijccc.2009.2.2420

Emrah Demir, Tolga Bektaş, and Gilbert Laporte. 2012. An adaptive large neighborhood search heuristic for the Pollution-Routing Problem. *European Journal of Operational Research* 223, 2 (2012), 346–359. https://doi.org/10.1016/j.ejor.2012.06.044

Guy Desaulniers, Fausto Errico, Stefan Irnich, and Michael Schneider. 2016. Exact Algorithms for Electric Vehicle-Routing Problems with Time Windows. *Operations Research* 64, 6 (2016), 1388–1405. https://doi.org/10.1287/opre.2016.1535

Yvan Dumas, Jacques Desrosiers, and François Soumis. 1991. The pickup and delivery problem with time windows. *European Journal of Operational Research* 54, 1 (1991), 7–22. https://doi.org/10.1016/0377-2217(91)90319-Q

Thierry Garaix, Christian Artigues, Dominique Feillet, and Didier Josselin. 2011. Optimization of occupancy rate in dial-a-ride problems via linear fractional column generation. *Computers & Operations Research* 38, 10 (2011), 1435–1442. https://doi.org/10.1016/j.cor.2010.12.014

Dominik Goeke. 2019. Granular tabu search for the pickup and delivery problem with time windows and electric vehicles. *European Journal of Operational Research* 278, 3 (2019), 821–836. https://doi.org/10.1016/j.ejor.2019.05.010

Timo Gschwind and Stefan Irnich. 2015. Effective Handling of Dynamic Time Windows and Its Application to Solving the Dial-a-Ride Problem. *Transportation Science* 49, 2 (2015), 335–354. https://doi.org/10.1287/trsc.2014.0531

Sin C. Ho, Wai Y. Szeto, Yong-Hong Kuo, Janny M. Y. Leung, Matthew Petering, and Terence W. H. Tou. 2018. A survey of dial-a-ride problems: Literature review and recent developments. *Transportation Research Part B: Methodological* 111 (2018), 395–421. https://doi.org/10.1016/j.trb.2018.02.001

William Huang, Yi Mei, Günther Raidl, Fangfang Zhang, Laurenz Tomandl, Steffen Limmer, Mengjie Zhang, and Tobias Rodemann. 2025. Genetic Programming Hyper-Heuristic for the Dynamic Electric Dial-a-Ride Problem. In *2025 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 1–8. https://doi.org/10.1109/CEC65147.2025.11042942

Thomas Jatschka, Tobias Rodemann, and Günther R. Raidl. 2022. A Large Neighborhood Search for a Cooperative Optimization Approach to Distribute Service Points in Mobility Applications. In *Metaheuristics and Nature Inspired Computing (Communications in Computer and Information Science (CCIS), Vol. 1541)*, Bernabé Dorronsoro, Farouk Yalaoui, El-Ghazali Talbi, and Grégoire Danoy (Eds.). Springer International Publishing, Cham, 3–17. https://doi.org/10.1007/978-3-030-94216-8_1

Rene M. Jorgensen, Jesper Larsen, and Kristin B. Bergvinsdottir. 2007. Solving the Dial-a-Ride problem using genetic algorithms. *Journal of the Operational Research Society* 58, 10 (2007), 1321–1331. https://doi.org/10.1057/palgrave.jors.2602287

Surendra R. Kancharla and Gitakrishnan Ramadurai. 2020. Electric vehicle routing problem with non-linear charging and load-dependent discharging. *Expert Systems with Applications* 160 (2020), 113714. https://doi.org/10.1016/j.eswa.2020.113714

Merve Keskin and Bülent Çatay. 2016. Partial recharge strategies for the electric vehicle routing problem with time windows. *Transportation Research Part C: Emerging Technologies* 65 (2016), 111–127. https://doi.org/10.1016/j.trc.2016.01.013

Scott Kirkpatrick, C. Daniel Gelatt, Jr., and Mario P. Vecchi. 1983. Optimization by Simulated Annealing. *Science* 220, 4598 (1983), 671–680. https://doi.org/10.1126/science.220.4598.671

Steffen Limmer. 2023. Bilevel large neighborhood search for the electric autonomous dial-a-ride problem. *Transportation Research Interdisciplinary Perspectives* 21 (2023), 100876. https://doi.org/10.1016/j.trip.2023.100876

Marius Lindauer, Katharina Eggensperger, Matthias Feurer, André Biedenkapp, Difan Deng, Carolin Benjamins, Tim Ruhkopf, René Sass, and Frank Hutter. 2022. SMAC3: A Versatile Bayesian Optimization Package for Hyperparameter Optimization. *Journal of Machine Learning Research* 23, 54 (2022), 1–9. http://jmlr.org/papers/v23/21-0888.html

Bingshan Ma, Dawei Hu, Xiqiong Chen, Yin Wang, and Xue Wu. 2021. The vehicle routing problem with speed optimization for shared autonomous electric vehicles service. *Computers & Industrial Engineering* 161 (2021), 107614. https://doi.org/10.1016/j.cie.2021.107614

Mohamed A. Masmoudi, Manar Hosny, Kris Braekers, and Abdelaziz Dammak. 2016. Three effective metaheuristics to solve the multi-depot multi-trip heterogeneous dial-a-ride problem. *Transportation Research Part E: Logistics and Transportation Review* 96 (2016), 60–80. https://doi.org/10.1016/j.tre.2016.10.002

Mohamed A. Masmoudi, Manar Hosny, Emrah Demir, Konstantinos N. Genikomsakis, and Naoufel Cheikhrouhou. 2018. The dial-a-ride problem with electric vehicles and battery swapping stations. *Transportation Research Part E: Logistics and Transportation Review* 118 (2018), 392–420. https://doi.org/10.1016/j.tre.2018.08.005

Yves Molenbruch, Kris Braekers, and An Caris. 2017a. Benefits of horizontal cooperation in dial-a-ride services. *Transportation Research Part E: Logistics and Transportation Review* 107 (2017), 97–119. https://doi.org/10.1016/j.tre.2017.09.001

Yves Molenbruch, Kris Braekers, and An Caris. 2017b. Typology and literature review for dial-a-ride problems. *Annals of Operations Research* 259, 1-2 (2017), 295–325. https://doi.org/10.1007/s10479-017-2525-0

Yves Molenbruch, Kris Braekers, An Caris, and Greet Vanden Berghe. 2017c. Multi-directional local search for a bi-objective dial-a-ride problem in patient transportation. *Computers & Operations Research* 77 (2017), 58–71. https://doi.org/10.1016/j.cor.2016.07.020

Santhanakrishnan Narayanan, Emmanouil Chaniotakis, and Constantinos Antoniou. 2020. Shared autonomous vehicle services: A comprehensive review. *Transportation Research Part C: Emerging Technologies* 111 (2020), 255–293. https://doi.org/10.1016/j.trc.2019.12.008

Philip Oxley. 1980. Dial/a/ride: a review. *Transportation Planning and Technology* 6, 3 (1980), 141–148. https://doi.org/10.1080/03081068008717184

Sophie N. Parragh. 2011. Introducing heterogeneous users and vehicles into models and algorithms for the dial-a-ride problem. *Transportation Research Part C: Emerging Technologies* 19, 5 (2011), 912–930. https://doi.org/10.1016/j.trc.2010.06.002

Sophie N. Parragh, Jean-François Cordeau, Karl F. Doerner, and Richard F. Hartl. 2012. Models and algorithms for the heterogeneous dial-a-ride problem with driver-related constraints. *OR Spectrum* 34, 3 (2012), 593–633. https://doi.org/10.1007/s00291-010-0229-9

Sophie N. Parragh, Karl F. Doerner, and Richard F. Hartl. 2010. Variable neighborhood search for the dial-a-ride problem. *Computers & Operations Research* 37, 6 (2010), 1129–1138. https://doi.org/10.1016/j.cor.2009.10.003

Sophie N. Parragh and Verena Schmid. 2013. Hybrid column generation and large neighborhood search for the dial-a-ride problem. *Computers & Operations Research* 40, 1 (2013), 490–497. https://doi.org/10.1016/j.cor.2012.08.004

David Pisinger and Stefan Ropke. 2010. Large Neighborhood Search. In *Handbook of Metaheuristics*, Michel Gendreau and Jean-Yves Potvin (Eds.). Springer US, Boston, MA, 399–419. https://doi.org/10.1007/978-1-4419-1665-5_13

Yuan Qu and Jonathan F. Bard. 2015. A Branch-and-Price-and-Cut Algorithm for Heterogeneous Pickup and Delivery Problems with Configurable Vehicle Capacity. *Transportation Science* 49, 2 (2015), 254–270. https://doi.org/10.1287/trsc.2014.0524

Yannik Rist and Michael A. Forbes. 2021. A New Formulation for the Dial-a-Ride Problem. *Transportation Science* 55, 5 (2021), 1113–1135. https://doi.org/10.1287/trsc.2021.1044

Stefan Ropke and Jean-François Cordeau. 2009. Branch and Cut and Price for the Pickup and Delivery Problem with Time Windows. *Transportation Science* 43, 3 (2009), 267–286. https://doi.org/10.1287/trsc.1090.0272

Stefan Ropke, Jean-François Cordeau, and Gilbert Laporte. 2007. Models and branch-and-cut algorithms for pickup and delivery problems with time windows. *Networks* 49, 4 (2007), 258–272. https://doi.org/10.1002/net.20177

Stefan Ropke and David Pisinger. 2006. An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows. *Transportation Science* 40, 4 (2006), 455–472. https://doi.org/10.1287/trsc.1050.0135

Susan Shaheen and Adam Cohen. 2020. Chapter 3 - Mobility on demand (MOD) and mobility as a service (MaaS): early understanding of shared mobility impacts and public transit partnerships. In *Demand for Emerging Transportation Systems*, Constantinos Antoniou, Dimitrios Efthymiou, and Emmanouil Chaniotakis (Eds.). Elsevier, 37–59. https://doi.org/10.1016/B978-0-12-815018-4.00003-6

Paul Shaw. 1998. Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. In *Principles and Practice of Constraint Programming*, Michael Maher and Jean-Francois Puget (Eds.). Springer, Heidelberg, 417–431. https://doi.org/10.1007/3-540-49481-2_30

Yue Su, Nicolas Dupin, Sophie N. Parragh, and Jakob Puchinger. 2024. A Branch-and-Price algorithm for the electric autonomous Dial-A-Ride Problem. *Transportation Research Part B: Methodological* 186 (2024), 103011. https://doi.org/10.1016/j.trb.2024.103011

Yue Su, Nicolas Dupin, and Jakob Puchinger. 2023. A deterministic annealing local search for the electric autonomous dial-a-ride problem. *European Journal of Operational Research* 309, 3 (2023), 1091–1111. https://doi.org/10.1016/j.ejor.2023.02.012

Laurenz Tomandl, Maria Bresich, Günther R. Raidl, Yi Mei, Steffen Limmer, and Tobias Rodemann. to appear 2025. A Reinforcement Learning Guided Large Neighborhood Search for the Dynamic Electric Autonomous Dial-a-Ride Problem. In *11th International Conference on Machine Learning, Optimization, and Data Science (LOD 2025)*. Springer.

A Further Experimental Results

A.1 Cordeau Instances

Table 12 shows the results of e-ADARP2, B&P, DA, BI-LNS, and all our proposed approaches on the Cordeau instances with a minimum end battery level ratio $\gamma = 0.4$ for limited ($n_s = 1$) as well as unlimited CS visits ($n_s = \infty$). For e-ADARP2, no results for the case of $n_s = \infty$ are reported in [Bongiovanni et al. 2019]. For instance a2-20 considering $n_s = \infty$, the OTF MILP approach achieves a lower objective value than the supposedly optimal one from B&P, but we want to point out that our solutions were independently checked for feasibility and refer to the impact of numerical imprecisions in floating point value calculations also observed in other cases.

Table 12. Results on Cordeau instances with $\gamma = 0.4$ for limited CS visits ($n_s = 1$) and unlimited CS visits ($n_s = \infty$).

Inst	e-ADARP2		B&P	DA			BI-LNS				Direct Approach				OTF Approach MILP				OTF Approach Heuristic			
	RT [min]	obj	obj	$\overline{\text{RT}}$ [min]	$\text{obj}_{\min}$	$\overline{\text{obj}}$	$\text{obj}_{\min}$	$\overline{\text{obj}}$	Feas	$\text{obj}_{\min}$	$\overline{\text{obj}}$	σ_{obj}	Feas	$\text{obj}_{\min}$	$\overline{\text{obj}}$	σ_{obj}	Feas	$\text{obj}_{\min}$	$\overline{\text{obj}}$	σ_{obj}	Feas
$n_s = 1$																						
a2-16	0.03	**237.38**	**237.38**	0.88	**237.38**	237.38	238.20	238.20	10/10	**237.38**	237.38	0.00	30/30	**237.38**	237.38	0.00	30/30	**237.38**	237.38	0.00	30/30
a2-20	0.83	**280.70**	**280.70**	2.35	**280.70**	280.70	282.90	283.24	10/10	**280.70**	280.70	0.00	30/30	**280.70**	280.70	0.00	30/30	**280.70**	280.70	0.00	30/30
a2-24	0.42	348.04	**347.04**	3.85	**347.04**	347.04	**347.04**	349.67	10/10	349.68	350.66	1.07	30/30	**347.04**	348.14	1.12	30/30	349.20	349.20	0.00	30/30
a3-18	0.07	**236.82**	**236.82**	0.44	**236.82**	236.82	238.73	238.73	10/10	**236.81**	236.81	0.00	30/30	**236.81**	236.94	0.43	30/30	**236.81**	236.81	0.00	30/30
a3-24	0.28	**274.80**	**274.80**	1.13	**274.80**	274.80	275.58	275.58	10/10	**274.80**	274.80	0.00	30/30	**274.80**	276.42	1.34	30/30	**274.80**	274.80	0.00	30/30
a3-30	1.65	413.37	**413.34**	1.48	**413.34**	413.34	415.51	415.75	10/10	413.64	414.09	1.00	30/30	**413.34**	415.36	2.46	30/30	413.37	413.37	0.00	30/30
a3-36	5.11	484.14	**482.75**	2.63	483.06	483.86	485.98	487.42	10/10	483.28	489.27	3.06	30/30	483.06	490.65	5.00	30/30	483.06	485.92	2.78	30/30
a4-16	0.09	**222.49**	**222.49**	0.32	**222.49**	222.49	**222.49**	222.49	10/10	**222.49**	222.49	0.00	30/30	**222.49**	222.91	0.46	30/30	**222.49**	222.49	0.00	30/30
a4-24	0.66	**311.03**	**311.03**	0.53	**311.03**	311.65	311.48	311.48	10/10	**311.03**	311.03	0.00	30/30	**311.03**	311.35	0.30	30/30	**311.03**	311.03	0.00	30/30
a4-32	11.36	**394.26**	**394.26**	1.05	**394.26**	397.21	394.96	395.45	10/10	**394.26**	394.26	0.00	30/30	**394.26**	395.12	1.10	30/30	**394.26**	394.26	0.00	30/30
a4-40	6.96	**453.84**	**453.84**	1.94	**453.84**	459.46	457.01	457.39	10/10	**453.84**	455.63	2.40	30/30	**453.84**	462.97	4.77	30/30	**453.84**	454.84	1.87	30/30
a4-48	120.00	**554.60**	**554.60**	2.96	558.11	563.47	557.56	560.09	10/10	555.29	560.36	2.43	30/30	556.83	563.07	2.89	30/30	**554.60**	556.98	1.49	30/30
a5-40	20.35	**414.51**	**414.51**	1.21	416.25	420.32	415.63	415.63	10/10	**414.50**	415.17	0.96	30/30	415.79	421.75	3.91	30/30	**414.50**	415.12	1.05	30/30
a5-50	120.00	560.50	**559.51**	2.71	567.54	574.56	560.41	562.55	10/10	561.55	566.48	3.42	30/30	566.50	576.18	4.69	30/30	**559.51**	564.41	3.44	30/30
$n_s = \infty$																						
a2-16			**237.38**	0.84	**237.38**	237.38	238.20	238.20	10/10	**237.38**	237.38	0.00	30/30	**237.38**	237.38	0.00	30/30	**237.38**	237.38	0.00	30/30
a2-20			280.70	2.42	280.70	280.70	282.62	283.10	10/10	280.70	280.70	0.00	30/30	**280.58**	280.58	0.00	30/30	280.70	280.70	0.00	30/30
a2-24			**346.28**	4.43	**346.28**	346.28	347.93	349.07	10/10	348.92	348.92	0.00	30/30	**346.28**	346.56	0.85	30/30	348.92	348.92	0.00	30/30
a3-18			**236.82**	0.42	**236.82**	236.82	238.73	238.73	10/10	**236.81**	236.81	0.00	30/30	**236.81**	236.91	0.40	30/30	**236.81**	236.81	0.00	30/30
a3-24			**274.80**	1.11	**274.80**	274.80	275.18	275.46	10/10	**274.80**	274.80	0.00	30/30	**274.80**	275.49	1.25	30/30	**274.80**	274.80	0.00	30/30
a3-30			**413.28**	1.73	413.34	413.34	415.51	415.81	10/10	413.54	413.75	0.53	30/30	**413.28**	414.51	1.73	30/30	413.37	413.37	0.00	30/30
a3-36			**481.17**	4.15	**481.17**	481.17	484.07	485.37	10/10	**481.17**	484.00	2.54	30/30	**481.17**	487.67	3.53	30/30	**481.17**	482.68	2.26	30/30
a4-16			**222.49**	0.30	**222.49**	222.49	**222.49**	222.49	10/10	**222.49**	222.49	0.00	30/30	**222.49**	223.14	0.42	30/30	**222.49**	222.49	0.00	30/30
a4-24			**311.03**	0.49	**311.03**	311.65	311.48	311.48	10/10	**311.03**	311.03	0.00	30/30	**311.03**	311.28	0.20	30/30	**311.03**	311.03	0.00	30/30
a4-32			**394.26**	1.03	**394.26**	397.27	394.96	395.15	10/10	**394.26**	394.37	0.44	30/30	**394.26**	395.58	1.90	30/30	**394.26**	394.26	0.00	30/30
a4-40			**453.84**	2.02	**453.84**	458.74	456.93	457.50	10/10	**453.84**	454.47	1.74	30/30	455.22	463.28	4.08	30/30	**453.84**	454.53	1.84	30/30
a4-48			**554.54**	3.86	558.96	564.86	557.63	559.63	10/10	556.02	558.06	1.37	30/30	556.09	561.83	2.58	30/30	555.25	556.40	1.24	30/30
a5-40			**414.51**	1.18	415.79	419.82	415.62	415.62	10/10	**414.50**	415.86	1.10	30/30	416.98	423.21	3.44	30/30	**414.50**	414.91	0.82	30/30
a5-50			**559.48**	3.07	567.13	574.28	560.41	562.64	10/10	560.49	566.02	3.14	30/30	562.56	574.19	4.83	30/30	**559.48**	562.81	2.78	30/30

Table 13. Results on Ropke instances with $\gamma = 0.1$ for limited CS visits ($n_s = 1$) and unlimited CS visits ($n_s = \infty$).

Inst	B&P		DA			Direct Approach				OTF Approach MILP				OTF Approach Heuristic			
	RT [min]	obj	$\overline{\text{RT}}$ [min]	obj_{min}	$\overline{\text{obj}}$	obj_{min}	$\overline{\text{obj}}$	σ_{obj}	Feas	obj_{min}	$\overline{\text{obj}}$	σ_{obj}	Feas	obj_{min}	$\overline{\text{obj}}$	σ_{obj}	Feas
$n_s = 1$																	
a5-60	130.28	**683.39**	2.97	691.83	706.20	683.93	688.35	3.02	30/30	689.73	697.36	5.37	30/30	683.87	687.42	2.50	30/30
a6-48	18.49	**506.45**	3.82	506.72	512.69	**506.45**	507.23	1.76	30/30	507.51	515.37	5.47	30/30	**506.45**	506.77	0.21	30/30
a6-60	61.60	**689.45**	2.12	692.00	700.15	690.50	694.35	2.95	30/30	695.27	705.03	5.08	30/30	690.29	693.54	2.14	30/30
a6-72	300.00	**761.34**	3.47	777.44	794.69	765.38	773.17	4.32	30/30	780.66	793.95	9.51	30/30	762.16	770.69	3.77	30/30
a7-56	13.51	**612.02**	1.47	613.10	624.51	612.37	615.13	3.23	30/30	616.18	629.56	6.62	30/30	612.53	614.70	2.30	30/30
a7-70	300.00	**754.28**	3.50	760.90	778.84	755.75	762.51	3.30	30/30	763.73	779.19	9.43	30/30	756.27	761.16	3.23	30/30
a7-84	300.00	975.26	5.38	889.38	904.88	878.71	886.67	4.50	30/30	900.10	914.76	11.19	30/30	**874.57**	883.45	4.79	30/30
a8-64	54.11	**632.22**	10.20	641.99	652.59	633.82	641.74	4.12	30/30	647.01	660.08	6.69	30/30	**632.21**	637.93	3.89	30/30
a8-80	242.73	**788.99**	5.96	803.52	828.67	794.94	804.74	6.44	30/30	810.67	828.02	7.56	30/30	793.64	802.86	4.42	30/30
a8-96	300.00	1329.75	6.06	1053.11	1080.80	1036.71	1046.42	5.18	30/30	1055.38	1077.99	10.71	30/30	**1032.76**	1041.59	4.74	30/30
$n_s = \infty$																	
a5-60	232.36	**683.39**	2.86	687.68	705.59	684.37	688.55	3.15	30/30	686.47	698.19	5.15	30/30	683.81	686.03	2.09	30/30
a6-48	27.92	**506.45**	4.03	506.91	514.15	**506.45**	507.33	1.69	30/30	508.63	517.20	6.41	30/30	**506.45**	506.91	0.34	30/30
a6-60	55.09	**689.45**	2.14	691.07	702.09	**689.45**	693.33	2.37	30/30	694.85	703.87	4.73	30/30	689.86	692.31	1.58	30/30
a6-72	300.00	**761.91**	3.51	777.46	795.14	766.06	773.65	5.04	30/30	776.58	793.20	8.35	30/30	763.39	771.62	4.29	30/30
a7-56	40.88	**612.02**	1.46	614.18	622.69	**612.01**	615.33	2.59	30/30	618.99	628.00	6.41	30/30	**612.01**	615.23	3.16	30/30
a7-70	170.07	**754.28**	3.37	760.10	777.10	755.98	761.17	3.08	30/30	763.75	779.40	9.77	30/30	756.81	760.44	2.27	30/30
a7-84	300.00	**870.21**	5.00	885.89	905.13	872.61	887.62	6.96	30/30	893.10	910.68	9.73	30/30	874.69	884.33	5.40	30/30
a8-64	66.76	**632.22**	10.80	640.24	653.81	632.95	640.24	4.12	30/30	646.60	658.10	7.73	30/30	634.04	639.14	2.40	30/30
a8-80	81.38	**788.99**	6.20	804.02	826.92	796.24	804.88	5.87	30/30	813.97	827.17	7.30	30/30	792.92	802.74	4.97	30/30
a8-96	300.00	1098.12	6.11	1049.98	1077.21	1033.43	1046.79	8.88	30/30	1056.50	1080.17	11.05	30/30	**1031.01**	1040.36	4.49	30/30

A.2 Ropke Instances

Table 13 shows the results of B&P, DA, and all our proposed approaches on the Ropke instances with a minimum end battery level ratio $\gamma = 0.1$ for limited ($n_s = 1$) as well as unlimited CS visits ($n_s = \infty$). For these instance configurations, no results are reported in [Limmer 2023] for BI-LNS.