systems

MDPI

Article

Electric Vehicle Routing Problem with Time Windows and Flexible Service Locations

Xinlong Duan, Xuanyi Chen and Rui Xu *

School of Business, Hohai University, Nanjing 211100, China; 231313030006@hhu.edu.cn (X.D.); 2308080113@hhu.edu.cn (X.C.)
* Correspondence: rxu@hhu.edu.cn; Tel.: +86-25-6851-4618

Highlights

Please indicate how your work links to systems science via your contributions to systems practice, theory, and/or methodology.

- Formulates electric last-mile delivery with flexible service locations as a coupled logistics service system, integrating service location selection, routing, and charging decisions under operational constraints.
- Develops an integrated optimization methodology combining a unified mathematical formulation and a MALNS-FO algorithm to coordinate interdependent decisions in sustainable logistics systems.

What are the main findings and/or the implications of the main findings?

- Reveals that flexible service locations consolidate spatially dispersed demand, reduce routing costs, and balance operational efficiency with customer service convenience.
- Demonstrates that the MALNS-FO algorithm effectively solves the coupled location-routing-charging problem and provides high-quality solutions for sustainable logistics decision-making.

Abstract

The rapid development of shared delivery, parcel lockers, and community pickup services has enabled customers to receive orders at multiple alternative service locations. In such scenarios, the fixed-location assumption adopted in traditional electric vehicle routing problems is no longer appropriate. This paper investigates the Electric Vehicle Routing Problem with Time Windows and Flexible Service Locations (EVRPTW-FSL), in which customers can be served at one selected location from a candidate set, while each service location may accommodate multiple customers subject to capacity limits. A mixed-integer optimization model is developed to jointly determine service location assignments, vehicle routing, and charging decisions under vehicle capacity, battery range, partial recharging, and customer time-window constraints. To balance operational efficiency and customer convenience, the objective minimizes the total travel cost and the customer deviation cost incurred when a customer is assigned to an alternative service location rather than the original service location. To solve this NP-hard problem, a Modified Adaptive Large Neighborhood Search with Fix-and-Optimize mechanism (MALNS-FO) is proposed, incorporating specialized operators such as location association destroy, location similarity destroy, and route reconstruction repair, as well as a fix-and-optimize mechanism. Computational experiments demonstrate that the proposed method consistently outperforms benchmark approaches in solution quality and computational efficiency. Results further show that introducing flexible service locations can significantly reduce fleet usage and routing cost by consolidating

Academic Editor: Emilio Larrodé

Received: 8 May 2026
Revised: 1 August 2026
Accepted: 10 August 2026
Published: 13 August 2026

spatially dispersed demand. Moreover, moderate customer flexibility provides substantial operational benefits while maintaining acceptable service deviation levels.

Keywords: electric vehicle routing problem; adaptive large neighborhood search algorithm; flexible service locations; deviation cost; fix-and-optimize mechanism

1. Introduction

In the context of global energy constraints and increasingly severe environmental challenges, promoting a green and low-carbon transition of economic and social systems has become a common international objective. With the rapid development of e-commerce and urbanization, the logistics and distribution industry plays an increasingly important role in economic activities, while also contributing substantially to energy consumption and carbon emissions. In particular, emissions generated by urban last-mile delivery continue to place pressure on environmental sustainability [1]. Compared with traditional fuel vehicles, electric vehicles (EVs) offer significant advantages, such as zero tailpipe emissions and lower environmental impact [2]. Driven by these advantages, EVs have been increasingly adopted in urban logistics systems, attracting extensive research attention to the Electric Vehicle Routing Problem (EVRP) [3–5].

In recent years, new business models such as shared delivery and community group buying have gradually emerged, allowing customers to choose among multiple service locations [6]. For example, in scenarios such as parcel pickup and fresh-food cold-chain distribution, customers are no longer limited to receiving deliveries only at their own addresses, but may choose among smart lockers, community pickup stations, or shared service points. This trend challenges the fixed-location assumption embedded in traditional routing models [7]. In this study, we define such a service mode as "flexible service locations", in which customers may choose from a set of candidate locations and one service location can serve multiple customers.

In general, the relationship between customers and service locations can be divided into three types: one-to-one, many-to-one, and many-to-many, as shown in Figure 1. Traditional logistics distribution typically follows a one-to-one relationship, where each customer corresponds to a fixed service location. Many-to-one relationships arise in scenarios such as community pickup stations, where multiple customers are served by one common location. The many-to-many relationship further extends flexibility by allowing customers to choose from multiple candidate locations while each service location can simultaneously serve multiple customers. The flexible service locations studied in this paper correspond to this many-to-many relationship. Compared with fixed service modes, this setting introduces bidirectional choice and resource-sharing characteristics between customers and service locations.

Although EVRP has been widely studied, most existing studies focus on routing, charging strategies, battery constraints, and time windows under fixed customer service locations. Meanwhile, several studies on flexible delivery locations mainly consider conventional vehicle routing settings and rarely incorporate EV-specific charging constraints. As a result, service location assignment decisions and EV charging-routing decisions are usually studied separately. However, in practice, these decisions are highly interdependent. This research gap motivates the present study.

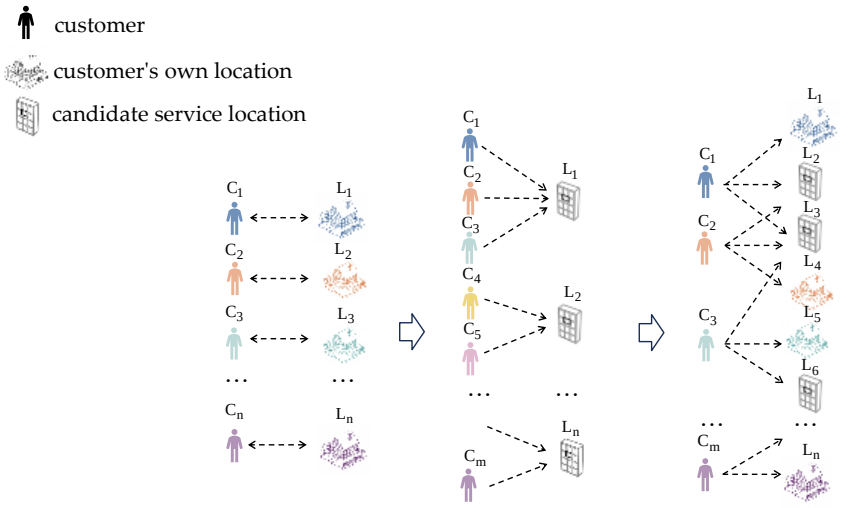

Figure 1. The relationship between customers and service locations across different logistics delivery models.

Specifically, the selection of customer service locations directly changes the set of nodes to be visited, route distances, service sequences, and battery consumption patterns, which further affects charging feasibility and route design. Conversely, limited battery range, charging infrastructure availability, and time-window constraints may restrict feasible service location assignments. Therefore, the problem extends from the traditional routing-charging decisions to a strongly coupled joint optimization problem involving service location assignment, routing, and charging strategies.

In addition, managing customer satisfaction has become increasingly important in logistics and distribution [8]. In scenarios involving flexible service locations, the degree of deviation between the actual service location and the customer's own location directly determines the last-mile delivery experience. If only travel cost is minimized, customers may be excessively concentrated at a few service locations, leading to significant deviations from their preferred locations and reduced service quality. In this study, customer dissatisfaction cost is defined as the deviation cost resulting from the distance between the actual service location and the customer's own location.

To address the above challenges, this study extends the classic EVRP by introducing flexible service locations while considering customer time windows. We refer to this problem as the Electric Vehicle Routing Problem with Time Windows and Flexible Service Locations (EVRPTW-FSL), which aims to minimize the sum of travel cost and deviation cost under constraints such as battery range, service location capacity, vehicle capacity, and customer time windows.

The main contributions of this paper are threefold: First, we define an EVRPTW-FSL problem by extending the EVRPTW to flexible service location scenarios and integrating service location assignment with EV-specific charging and partial recharging decisions. Second, we formulate the problem as a mixed-integer optimization model and develop a Modified Adaptive Large Neighborhood Search with Fix-and-Optimize mechanism (MALNS-FO) to solve it efficiently. Third, we validate the algorithm's effectiveness through extensive numerical experiments, analyzing the effectiveness of the designed operators, the effectiveness of the fix-and-optimize mechanism, and the impact of deviation costs.

The rest of the paper is organized as follows. Section 2 reviews the related literature. Section 3 presents the problem description. Section 4 gives the mathematical formulation. Section 5 introduces the proposed solution methodology. Section 6 reports computational experiments and discusses managerial insights. Finally, Section 7 concludes the paper and outlines future research directions.

2. Literature Review

Since Dantzig and Ramser [9] first proposed the vehicle routing problem (VRP) in 1959, the problem and its various extended variants and solution algorithms have been extensively and deeply studied. The objective of the problem is to determine a set of routes that depart from a depot, serve a set of customers, and then return to the depot. Typically, the objective function aims to minimize the total cost, total travel time, or total distance. Toth [10], Braekers [11], and Tan [12] provided comprehensive summaries of VRP.

In recent years, an increasing number of scholars have been working in the field of EVRP [13–15]. EVRP is one of the key variants of VRP, characterized by the integration of electric vehicles into logistics and delivery systems. Compared with the traditional VRP, EVRP focuses on minimizing operational costs by optimizing routes for a fleet of electric vehicles that can recharge at recharging stations. As a result, charging decisions become a critical constraint. Subsequent research on the EVRP has incorporated a variety of features, leading to the development of different variants, such as customer time windows, partial recharging strategies, and recharging station siting. Preis et al. [16] constructed an urban delivery model for electric vehicles with the goal of reducing energy consumption and designed an adaptive tabu search algorithm to solve it. Schneider et al. [17] were the first to establish a standard model for the Electric Vehicle Routing Problem with Time Windows (EVRPTW), integrating time window constraints with recharging station considerations. Keskin and Çatay [18] proposed a partial charging strategy based on EVRPTW and demonstrated that flexible charging levels can effectively reduce operating costs. To address the dual uncertainties of driving time and energy consumption in electric vehicles, Han et al. [19] constructed a path-dependent uncertainty set and proposed a two-stage adaptive robust model. The model first determines core decisions such as routing and payload, and then adjusts the charging volume and service time based on the actual scenario. Bányai et al. [20] developed a mathematical optimization model that integrates vehicle capacity, delivery time windows, and greenhouse gas emission constraints to optimize last-mile logistics routing for small-scale urban food producers using green micromobility solutions. Guo et al. [21] proposed an integrated optimization framework combining multi-agent simulation and a genetic algorithm to optimize charging strategies for shared autonomous electric vehicles while balancing operational costs, empty vehicle miles traveled, and passenger waiting times. Davatgari et al. [22] proposed an innovative framework that integrates integer programming, clustering simplification strategies, and genetic algorithms to solve the problem of charging facility site selection and capacity allocation for electric freight vehicles operating on fixed routes.

Despite these advances, the aforementioned studies generally assume fixed service locations, meaning that each node corresponds to exactly one customer, and each customer is visited exactly once. However, the EVRP with fixed service locations fails to adequately capture the flexible service locations that are widespread in real-world logistics scenarios. The consideration of flexible service locations represents an extension of the classical EVRP on the demand side. Customers are no longer fixed at their own locations; instead, service must be provided at exactly one capable location selected from a set of alternatives. Compared to the classical EVRP, this adds an additional layer of service location assignment decisions, significantly increasing the complexity of the solution. To date, limited research has simultaneously addressed both service location allocation and charging decisions. Nevertheless, some studies have already examined VRP problems related to flexible service locations. The first works considering both location and routing aspects date back to the 1960s [23–25]. Cavagnini et al. [26] summarized the literature on location and routing, noting that integrating the site selection and routing problems can, to some extent, prevent excessive system costs or suboptimal solutions. Janihoff et al. [27] also classified and

summarized existing literature and identified key characteristics across dimensions such as facility location, vehicle routing, and site-to-route planning. Dumez et al. [28] proposed a new vehicle routing problem with flexible customer delivery choices and developed a hybrid optimization algorithm to reduce delivery costs and improve service quality in urban last-mile logistics. Yang et al. [29] developed a neighborhood-based matheuristic algorithm for the vehicle routing problem with delivery options, combining local search, column generation, and a tailored initial solution method to optimize routing under service level and capacity constraints. Hu et al. [30] proposed a model and hybrid metaheuristic for the workforce scheduling and routing problem with sequence constraints and alternative locations and demonstrated the effectiveness and managerial implications through computational experiments.

In studies on flexible service locations, the objective functions are primarily defined from the perspective of logistics companies, and most fail to measure performance in terms of customer satisfaction. For example, if a customer must travel a long distance to receive a service, this can negatively impact service quality to some extent. This paper defines the additional cost incurred by the customer as the deviation cost. Some literature has also considered these additional costs from the customer's perspective. For example, Frey et al. [31] proposed a vehicle routing problem with time windows and flexible delivery locations, and designed an adaptive neighborhood search algorithm to solve it; their objective function balances both the vehicle routing cost and the cost of customer location preferences. Zhang et al. [32] formulated a vehicle routing problem that balances flexible delivery locations and time windows. They developed an optimization model that incorporates total travel costs, fixed vehicle operating costs, and customer satisfaction penalty costs, and designed a multi-strategy adaptive large neighborhood search algorithm to achieve the joint optimization of costs and satisfaction. Zang et al. [33] addressed the problem of time-slot conflicts in last-mile delivery by proposing an urban delivery model that balances door-to-door delivery with parcel locker pickup. The model aims to minimize the sum of vehicle travel costs and customer pickup costs, and employs an exact algorithm to achieve efficient solutions.

To further clarify the relationship between this study and existing VRP variants, Table 1 summarizes the main problem features considered in the related literature.

Table 1. Comparison of related literature.

Document	Electric Vehicles	Time Windows	Partial Recharging	Flexible Service Locations	Service Location Capacity	Deviation Cost
[16]	√ [1]					
[17]	√	√				
[18]	√	√	√			
[19]	√	√	√			
[20]	√	√				
[21]	√		√			
[22]	√	√	√			
[26]		√		√	√	
[27]		√		√	√	
[28]		√		√	√	
[29]		√		√	√	
[30]		√		√	√	
[31]		√		√	√	√
[32]		√		√		√
[33]		√		√		√
This article	√	√	√	√	√	√

[1] The √ symbol indicates that the research question in the corresponding literature involves a specific variant of VRP.

As shown in Table 1, flexible service locations and deviation costs have been considered in some VRP studies, while EVRP and EVRPTW studies have mainly focused on battery constraints and recharging decisions under fixed service locations. This study incorporates charging decisions and service location decisions into the EVRP framework, recognizing that the two are interdependent. The selection of service locations directly determines the driving range, energy consumption along the route, and remaining battery level, which in turn influence the optimal scheduling of charging times, locations, and durations. Conversely, the limited range of electric vehicles, the distribution of charging infrastructure, and charging capacity constraints, in turn, limit the range of available service locations and the feasibility of allocation schemes. Optimizing service location allocation or developing charging strategies in isolation is likely to result in suboptimal solutions characterized by unreasonable energy consumption, infeasible routes, or high overall costs. Therefore, these decisions must be optimized jointly and in a coordinated manner.

3. Description of the Problem

EVRPTW-FSL can be described as follows: under the constraints of electric vehicle load, battery capacity, and customer time windows, this problem involves comprehensively optimizing vehicle routes, charging strategies, and specific service location selections for customers with multiple service location options.

In this problem, each customer has an original service location, which represents the customer's preferred receiving location in the delivery system. This location may correspond to the customer's address or another predefined preferred pickup point, depending on the practical service scenario. The original service location is included in the candidate service-location set of the customer. Therefore, a customer may either be served at the original service location or at another feasible alternative service location. However, each customer can be served at only one location in a feasible solution. In other words, "flexible service locations" means that one service location is selected from the candidate set for each customer, rather than that a customer is served repeatedly at multiple locations.

The goal is to balance delivery costs, including travel cost, against customer deviation cost, which reflects the inconvenience or compensation associated with being served away from a preferred location. To illustrate the differences between electric vehicle routing problems under flexible and fixed service location settings, this study designs a small-scale comparative example involving 10 customers, as shown in Figure 2.

Under fixed service locations, three vehicles are required to complete the delivery task due to battery capacity constraints. In route 1, with the visiting sequence $D_0 \rightarrow C_1 \rightarrow C_2 \rightarrow C_3 \rightarrow C_4 \rightarrow D_0$, customer C_4 is located relatively far from the depot; after visiting C_4, the remaining battery is only sufficient for the vehicle to return to the depot. In route 2, denoted by the sequence $D_0 \rightarrow C_5 \rightarrow S_3 \rightarrow C_6 \rightarrow C_7 \rightarrow D_0$, customer C_5 is far from the depot, therefore, the vehicle must recharge at station S_3 during the trip, and battery constraints prevent the two routes from being merged. In route 3, with the visiting sequence $D_0 \rightarrow C_8 \rightarrow C_9 \rightarrow C_{10} \rightarrow S_4 \rightarrow D_0$, the vehicle must recharge at S_4 after visiting customer C_{10}; otherwise, it cannot return to the depot. Under fixed service location constraints, complex geographical distributions reduce vehicle utilization efficiency and limit recharging flexibility, thereby hindering effective route integration. Under flexible service locations, as shown in Figure 2b, one location can be selected from a customer's set of candidate service locations for service delivery. The selected location must belong to the corresponding customer's candidate service-location set, and the final assignment is determined by the total cost, including both travel cost and deviation cost. By assigning C_4 to L_1 and C_5 to L_2 and adjusting the choice of recharging stations, routes 1 and 2 can be restructured into a single route: $D_0 \rightarrow C_1 \rightarrow C_2 \rightarrow C_3 \rightarrow C_4 \rightarrow S_2 \rightarrow C_5 \rightarrow C_6 \rightarrow C_7 \rightarrow D_0$.

In this case, only one vehicle is required to serve customers who originally required two vehicles. Meanwhile, by assigning C_9 and C_{10} to L_3, the remaining customers can be served without recharging. As shown in Figure 2, allowing flexible service locations reduces the total cost by 17%. This reduction is mainly due to the use of fewer vehicles, fewer charging operations, and shorter travel distances. This example clearly illustrates the key characteristics that distinguish the route structure of the EVRPTW-FSL from that of the traditional EVRP.

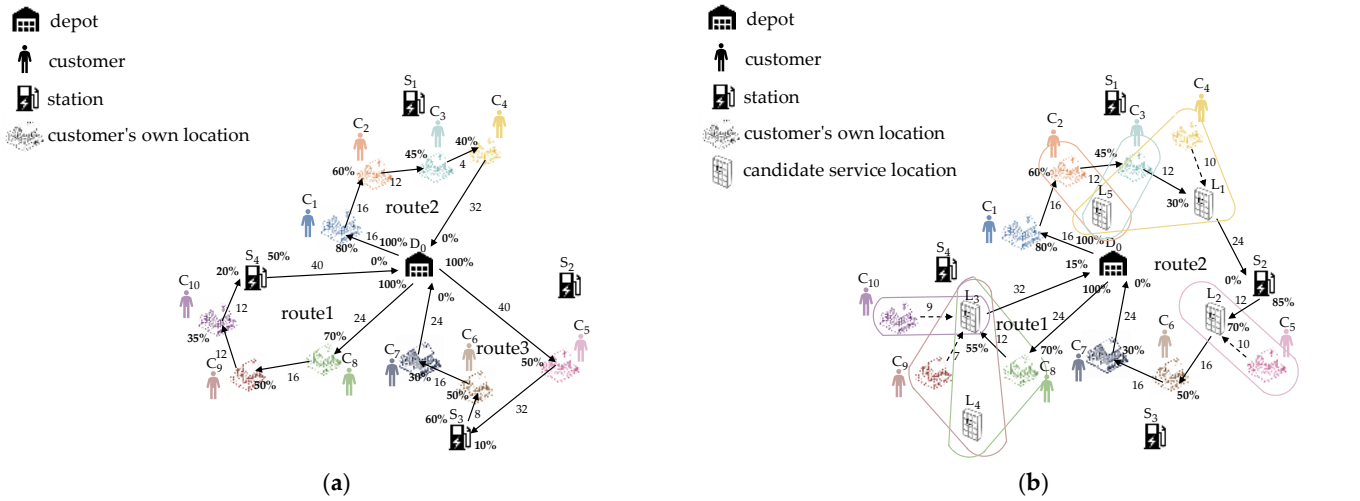

Figure 2. Schematic illustration of electric vehicle routing under fixed and flexible service location settings. (**a**) EVRP with fixed service locations; (**b**) EVRP with flexible service locations. Percentages indicate the remaining battery levels of vehicles along the routes, and the numbers beside the arcs or assignment links indicate the corresponding travel costs or deviation costs.

4. Mathematical Model

EVRPTW-FSL concerns a set of customers and a set of service locations. As previously established, a many-to-many relationship exists between the customers and the service locations. The subset of candidate service locations where each customer can be served is known. The service process is performed by a homogeneous fleet of electric vehicles with fixed loading capacities and limited cruising ranges. While a vehicle is traveling, its battery charge level decreases proportionally to the distance traversed, and the vehicle may need to visit a recharging station in order to continue its route.

In the EVRPTW-FSL, each customer may be served at one of several candidate service locations. Therefore, the model needs to jointly determine the route sequence of customers and the specific service location selected for each customer. To represent these two decisions in a unified way, this study formulates the EVRPTW-FSL on a directed graph by introducing customer-location tuples as route vertices. With this representation, service-location selection and route construction can be described within the same route-vertex framework, and the subsequent flow, time, load, and battery constraints can be imposed directly on the actually visited service locations.

Let $C = \left\{C_1, C_2, \ldots, C_{|C|}\right\}$ be the set of physical customers, and $L = \left\{L_1, L_2, \ldots, L_{|L|}\right\}$ be the set of service locations. For each customer $i \in C$, its original service location is also included in L and belongs to its candidate service-location set. This original service location represents the customer's preferred receiving location, while the other feasible locations in the candidate set are regarded as alternative service locations. A customer-location tuple $< i, l >$ represents the event that customer i is served at service location l. Therefore, when l is the original service location of customer i, visiting tuple $< i, l >$ means that customer i is served at its own original location; otherwise, customer i is served at an alternative

location and a deviation cost is incurred. In this way, both original-location service and alternative-location service are represented within the same tuple-based structure. Let $V = \{< i, l > | i \in C, l \in L_i\}$ denote the set of customer-location tuples, where $L_i \subseteq L$ is the candidate service location set of customer i. In a feasible solution, each customer must be associated with exactly one selected tuple from its candidate tuple set. Thus, flexible service locations mean that one service location is selected for each customer from L_i, rather than that the same customer is served repeatedly at multiple locations.

To keep a unified tuple-based representation for all route vertices, the depot and each recharging station are also represented by tuple-like vertices. Since the depot and recharging stations do not correspond to real customers, they are paired with virtual customers whose demand and service time are both set to zero. For all virtual customers, the left time-window bound is set to 0 and the right time-window bound is set sufficiently large, so that these auxiliary vertices do not impose additional customer time-window restrictions. These virtual customers are auxiliary modeling devices only. They are not involved in customer service assignments and do not represent additional delivery requests. Their role is to allow depot vertices, recharging-station vertices, and customer-location vertices to be represented within the same route-vertex framework. The set DD contains the departure-depot tuple and its copies, and the set AD contains the arrival-depot tuple and its copies. Every route starts at a vertex in set DD and ends at a vertex in set AD. The set S denotes the recharging stations, and the set S' contains the recharging-station tuples and their copies. The set S' is introduced to allow multiple visits to each recharging station in S. Let V' be the set of vertices with $V' = V \cup S'$. To distinguish sets containing the depot vertices, the subscript DD or AD is added to the set. Specifically, ${V'}_{DD} = V' \cup DD$, ${V'}_{AD} = V' \cup AD$ and ${V'}_{DD,AD} = V' \cup DD \cup AD$.

Thus, EVRPTW-FSL can be defined in the directed graph $G = ({V'}_{DD,AD}, A)$, with the set of arcs is $A = \{(< i, l >, < j, r >) | < i, l > \in {V'}_{DD}, < j, r > \in {V'}_{AD}, i \neq j\}$. Each arc corresponds to a travel distance $d_{l,r}$, a travel time $t_{l,r}$, and a travel cost $c_{l,r}^{travel}$, which equals $d_{l,r}$ by default. Each traveled arc consumes the amount $hd_{l,r}$ of the remaining battery charge of the vehicle traveling the arc, where h denotes the constant charge consumption rate. Each vertex $< i, l > \in {V'}_{DD,AD}$ is associated with a demand q_i, a time window $[a_i, b_i]$ and a service time s_i. The demand and service time for all virtual customers are set to 0. Service cannot begin before a_i, which may result in waiting time, and is not allowed to start after b_i, but might end later. Each customer can only be served at one of multiple candidate locations, meaning that only one of the corresponding vertices may be selected to serve the customer. The delivery routes must cover all customers but need not cover all vertices. Consequently, each vertex in set V is either not visited or visited exactly once, implying that its in-degree and out-degree must simultaneously equal zero or one. For each service location $l \in L$, the number of assigned customers cannot exceed its maximum capacity MC_l. For each selected customer-location tuple $< j, r >$, $c_{j,r}^{location}$ denotes the deviation cost of assigning customer j to service location r. If r is the original service location of customer j, this cost is zero. Otherwise, r is an alternative service location, and $c_{j,r}^{location}$ reflects the inconvenience or compensation associated with serving customer j at this alternative location. At a recharging station, vehicles are charged at a recharging rate of g. The vehicles depart from the depot fully charged but may depart from a station with any state of charge, meaning that partial recharging is permitted. The recharging time depends on the amount of energy replenished.

We use decision variables associated with vertices to keep track of vehicle states. The decision variables $\tau_{i,l}$, $u_{i,l}$, $y_{i,l}$ keep track of the service starting time, the remaining charge level on arrival at vertex $< i, l > \in {V'}_{DD,AD}$. EVRPTW-FSL allows partial recharges by defining the decision variable $Y_{i,l}$ which represents the battery state of charge on departure from vertex $< i, l > \in {V'}_{DD,AD}$. The decision variables $x_{i,l,j,r}$ are binary and equal 1 if

customer i is served in location l immediately before serving customer j in location r, and 0 otherwise. The objective is to determine a set of vertices $V_t \in V'_{DD,AD}$ that minimizes the sum of total travel cost and total deviation cost, subject to constraints on vehicle capacity, battery levels, service location capacity, and customer time windows.

4.1. Notation

The sets, parameters, and decision variables are shown in Table 2.

Table 2. List of notations.

	Physical Sets
C	Set of physical customers, $C = \{C_1, C_2, \ldots, C_{\|C\|}\}$
L	Set of service locations, $L = \{L_1, L_2, \ldots, L_{\|L\|}\}$
L_i	Candidate service location set of customer i, including its original service location, $i \in C$
S	Set of recharging stations, $S = \{S_1, S_2, \ldots, S_{\|S\|}\}$
	Basic tuple-vertex sets
V	Set of customer-location tuple vertices, $V = \{< i, l > \| i \in C, l \in L_i\}$
S'	Set of recharging-station tuple vertices, each recharging station and each of its copies are paired with a virtual customer to form a tuple vertex for unified route representation, $S' = \{< C_{S1}, S_1 >, < C_{S1}\prime, S_1\prime >, < C_{S1}\prime\prime, S_1\prime\prime >, \ldots, < C_{S2}, S_2 >, < C_{S2}\prime, S_2\prime >, \ldots\}$
DD	Set of departure-depot tuple vertices; the departure depot and each of its copies are paired with a virtual customer to form a tuple vertex for unified route representation, $DD = \{< C_{D0}, D_0 >, < C_{D0}\prime, D_0\prime >, < C_{D0}\prime\prime, D_0\prime\prime >, \ldots\}$
AD	Set of arrival-depot tuple vertices; the arrival depot and each of its copies are paired with a virtual customer to form a tuple vertex for unified route representation, $AD = \{< C_{D0\_end}, D_{0\_end} >, < C_{D0\_end}\prime, D_{0\_end}\prime >, < C_{D0\_end}\prime\prime, D_{0\_end}\prime\prime >, \ldots\}$
	Composite route-vertex sets for constraint indexing
V'	$V' = V \cup S'$
V_{DD}	$V_{DD} = V \cup DD$
V'_{DD}	$V'_{DD} = V\prime \cup DD$
V_{AD}	$V_{AD} = V \cup AD$
V'_{AD}	$V'_{AD} = V\prime \cup AD$
$V'_{DD,AD}$	$V'_{DD,AD} = V\prime \cup DD \cup AD$
S'_{DD}	$S'_{DD} = S' \cup DD$
	Parameters
MC_l	Service location capacity of l, measured by the maximum number of customers that can be assigned to this location
$d_{l,r}$	The distance between locations l and r
$t_{l,r}$	The travel time between locations l and r
$c^{travel}_{l,r}$	The travel cost between locations l and r
$c^{location}_{j,r}$	The deviation cost of assigning customer j to service location r; it is zero if r is the original service location of customer j
q_i	The demand of customer i
s_i	The service time of customer i
$[a_i, b_i]$	Time window of customer i
g	Recharging time coefficient, defined as the time required to recharge one unit of battery energy
h	Battery energy consumption coefficient, defined as the battery energy consumed per unit of travel distance
Q	Battery capacity
W	The maximum load allowed on a route
	Decision variables
$x_{i,l,j,r}$	Binary variable indicating whether arc $(< i, l >, < j, r >)$ is traveled
$u_{i,l}$	Load of vehicle upon at vertex $< i, l >$
$y_{i,l}$	Remaining battery level of the vehicle upon arrival at vertex $< i, l >$
$Y_{i,l}$	Remaining battery level of the vehicle upon departure from vertex $< i, l >$
$\tau_{i,l}$	The start time of the service of the vehicle at vertex $< i, l >$

Note: Virtual customers are auxiliary dummy customers used only to express depot and recharging-station vertices in the same tuple-based form as customer-location vertices. They have zero demand and zero service time, are not included in the physical customer set C, and are not involved in customer service location assignment. For the depot, $[a_0, b_0]$ denotes the planning horizon. The parameter a_0 is usually set to 0, while b_0 is set sufficiently large.

4.2. Model Assumptions

1. Electric vehicles are homogeneous, start from the depot with fully charged standard batteries, and return to the depot [17].

2. Each customer is associated with a candidate service location set that includes the customer's original service location. In a feasible solution, exactly one location is selected from this set to serve the customer. If the selected service location differs from the customer's original service location, a deviation cost is incurred, which depends on the distance between the selected location and the original service location [31].
3. Each service location can serve multiple customers, subject to a maximum capacity constraint [31].
4. Vehicles travel at a constant speed, with power consumption proportional to the travel distance [17].
5. The energy replenished at recharging stations increases linearly with time, allowing for partial recharging [18].

4.3. Objective Function and Constraints

The mathematical formulation of the proposed EVRPTW-FSL model is presented as follows.

$$\min z = \sum_{<i,l>\in V'_{DD}} \sum_{<j,r>\in V'_{AD}} (c_{l,r}^{travel} + c_{j,r}^{location}) x_{i,l,j,r} \tag{1}$$

Subject to :

$$\sum_{<i,l>\in V} \sum_{<j,r>\in V'_{AD}, i\neq j} x_{i,l,j,r} = 1, \forall i \in C \tag{2}$$

$$\sum_{<j,r>\in V'_{AD}} x_{i,l,j,r} \leq 1, \forall <i,l>\in S' \tag{3}$$

$$\sum_{<i,l>\in V'_{DD}} x_{i,l,j,r} = \sum_{<i,l>\in V'_{AD}} x_{j,r,i,l}, \forall <j,r>\in V' \tag{4}$$

$$\sum_{<j,r>\in V'} x_{i,l,j,r} \leq 1, \forall <i,l>\in DD \tag{5}$$

$$\sum_{<i,l>\in V'} x_{i,l,j,r} \leq 1, \forall <j,r>\in AD \tag{6}$$

$$\sum_{<i,l>\in DD} \sum_{<j,r>\in V'} x_{i,l,j,r} = \sum_{<j,r>\in V'} \sum_{<i,l>\in AD} x_{j,r,i,l} \tag{7}$$

$$\sum_{<i,l>\in V} \sum_{<j,r>\in V'_{AD}} x_{i,l,j,r} \leq MC_l, \forall l \in L \tag{8}$$

$$\tau_{i,l} + (t_{l,r} + s_i) x_{i,l,j,r} - b_0(1 - x_{i,l,j,r}) \leq \tau_{j,r}, \forall <i,l>\in V_{DD}, \forall <j,r>\in V'_{AD} \tag{9}$$

$$\tau_{i,l} + t_{l,r} x_{i,l,j,r} + g(Y_{i,l} - y_{i,l}) - (b_0 + gQ)(1 - x_{i,l,j,r}) \leq \tau_{j,r}, \forall <i,l>\in S', \forall <j,r>\in V'_{AD} \tag{10}$$

$$a_i \leq \tau_{i,l} \leq b_i, \forall <i,l>\in V'_{DD,AD} \tag{11}$$

$$0 \leq u_{j,r} \leq u_{i,l} - q_i x_{i,l,j,r} + W(1 - x_{i,l,j,r}), \forall <i,l>\in V'_{DD}, \forall <j,r>\in V'_{AD} \tag{12}$$

$$0 \leq u_{i,l} \leq W, \forall <i,l>\in DD \tag{13}$$

$$0 \leq y_{j,r} \leq y_{i,l} - (hd_{l,r}) x_{i,l,j,r} + Q(1 - x_{i,l,j,r}), \forall <i,l>\in V, \forall <j,r>\in V'_{AD} \tag{14}$$

$$0 \leq y_{j,r} \leq Y_{i,l} - (hd_{l,r}) x_{i,l,j,r} + Q(1 - x_{i,l,j,r}), \forall <i,l>\in S'_{DD}, \forall <j,r>\in V'_{AD} \tag{15}$$

$$y_{i,l} \leq Y_{i,l} \leq Q, \forall <i,l>\in S'_{DD} \tag{16}$$

$$x_{i,l,j,r} \in \{0,1\}, \forall x_{i,l,j,r} \in A \tag{17}$$

The objective function and constraints are explained as follows: Equation (1) is the objective function, which minimizes the total cost consisting of travel cost and deviation cost.

The deviation cost is zero when a customer is assigned to the original service location and positive only when the customer is assigned to an alternative service location. Constraint (2) ensures that exactly one customer-location tuple is selected for each customer. Thus, each customer is assigned to only one service location in a feasible solution, which may be either the original service location or an alternative service location. The virtual customers used for depot and recharging-station tuple vertices are not included in C and are not involved in customer service-location assignments. Constraint (3) ensures that each recharging-station tuple vertex, including each individual copy, is visited at most once. Multiple visits to the same physical recharging station can still be represented by selecting different copies of that station. Constraint (4) imposes flow conservation on customer-location tuple vertices and recharging-station tuple vertices. Constraints (5) and (6) restrict the outgoing and incoming arcs associated with departure and arrival-depot tuple vertices, respectively. Depot copies are introduced to represent multiple route starts and route terminations separately, with each used depot copy associated with at most one route. Constraint (5) ensures that each departure-depot tuple vertex can have at most one selected outgoing arc, so each used departure-depot copy can start at most one route. Constraint (6) ensures that each arrival-depot tuple vertex can have at most one selected incoming arc, so each used arrival-depot copy can terminate at most one route. Constraint (7) ensures that the number of selected route departures equals the number of selected route arrivals, so that each started route is properly terminated at an arrival-depot tuple vertex. Constraint (8) ensures that the number of customers assigned to each service location does not exceed its service-location capacity MC_l. The total number of customers served at a location is determined by the total number of active outgoing arcs originating from all of its associated customer-location tuple vertices. Constraints (9) and (10) update the service start time along selected arcs and ensure that the service times of consecutive vertices are consistent. Constraint (9) applies to arcs departing from departure-depot tuple vertices and customer-location tuple vertices. For such an arc from vertex $< i, l >$ to vertex $< j, r >$, the service start time at the next vertex $\tau_{j,r}$ must be no earlier than the current service start time $\tau_{i,l}$ plus the service time s_i at the current vertex and the travel time $t_{l,r}$ between the two locations. Constraint (10) applies to arcs departing from recharging-station tuple vertices. For a selected arc from vertex $< i, l >$ to vertex $< j, r >$, the service start time at the next vertex $\tau_{j,r}$ must be no earlier than the current service start time $\tau_{i,l}$, plus the recharging duration $g(Y_{i,l} - y_{i,l})$ at the current station and the travel time $t_{l,r}$ between the two locations. In Constraint (10), $b_0 + gQ$ serves as a sufficiently large time-related constant, where gQ represents the maximum possible full-recharging time. Constraint (11) enforces the time-window requirements of the corresponding vertices. Constraint (12) propagates the route load along selected arcs and updates the remaining load according to the demand of the physical customer associated with the predecessor tuple. Constraint (13) bounds the load at each departure-depot tuple vertex by the maximum route load. For depot and recharging-station tuple vertices, the paired virtual customers have zero demand; therefore, visits to these vertices do not affect the route load. Constraint (14) describes the battery propagation when the predecessor vertex is a customer-location tuple vertex. Since no recharging operation is allowed at the vertex, the battery level at the next vertex is calculated based on the arrival battery level $y_{i,l}$ of the current vertex minus the energy consumption $hd_{l,r}$ along the selected arc. Constraint (15) describes the battery propagation when the predecessor vertex is a departure-depot vertex or a recharging-station vertex. In this case, the vehicle may depart with a different battery level after charging or initialization, so the departure battery level $Y_{i,l}$ is used to update the battery level at the next vertex. Constraint (16) links the arrival and departure battery levels at recharging-station and departure-depot tuple vertices and

ensures that the departure battery level lies between the arrival battery level and the battery capacity. Constraint (17) defines the binary decision variables.

4.4. Complexity Analysis

Theorem 1. *EVRPTW-FSL is NP-hard.*

Proof of Theorem 1. Assuming that each customer can only receive service at original service locations, the flexible service location decision constraint simplifies to fixed service locations. Under the premise of keeping the other assumptions unchanged, the problem degenerates into the standard EVRPTW problem. EVRPTW has been proven to be NP-hard [17]. EVRPTW-FSL, as an extension of EVRPTW, introduces an additional layer of service location selection decisions; therefore, EVRPTW-FSL is also an NP-hard problem. Theorem 1 is proved. □

5. Solution Methodology

The challenges in solving this problem lie in the coupling between the charging strategy and service location assignment, as well as the correlation in location selection among customers. Specifically: (1) The coupling between the charging strategy and service location assignment. The assignment of service locations directly affects the spatial direction and travel distances of the route, which further affects the vehicle's battery consumption and charging requirements. Assigning service locations far from charging stations increases range pressure, while charging decisions, in turn, restrict the feasible set of service location assignments. (2) Since assigning multiple customers to the same service location reduces the travel cost of vehicles shuttling between different locations, subsequent customers tend to concentrate on service locations already serving other customers, which can easily lead to a local optimum characterized by "irrational clustering."

Given the NP-hard nature of the problem, we develop an MALNS-FO algorithm based on the adaptive large neighborhood search framework proposed by Ropke et al. [34]. The main challenge in algorithm design lies in developing efficient operators tailored to the problem characteristics and avoiding entrapment in local optima. To this end, we design specialized operators, including location association destroy, location similarity destroy, and route reconstruction repair for the characteristics of flexible service locations. Moreover, a fix-and-optimize mechanism is designed to overcome the "irrational clustering" phenomenon during the solution process and escape local optima.

The overall framework of the algorithm is shown in Figure 3. The algorithm consists of three main parts. First, an initial solution is constructed by encoding each solution as a sequence of customer-location tuples, where the service location assignment and vehicle routing are represented simultaneously. During this stage, the cost evaluation function and path feasibility assessment are used to guide the insertion of tuples. Second, the destroy-repair phase iteratively improves the current solution. Several destroy operators are used to remove selected customer-location tuples, and repair operators are then applied to reconstruct feasible routes. Third, the fix-and-optimize phase is triggered to further improve local route structures. It identifies a cluster center, removes the related central customer, and performs local reconstruction to reduce irrational clustering. Through these three stages, MALNS-FO jointly optimizes service location assignment, vehicle routing, and charging decisions.

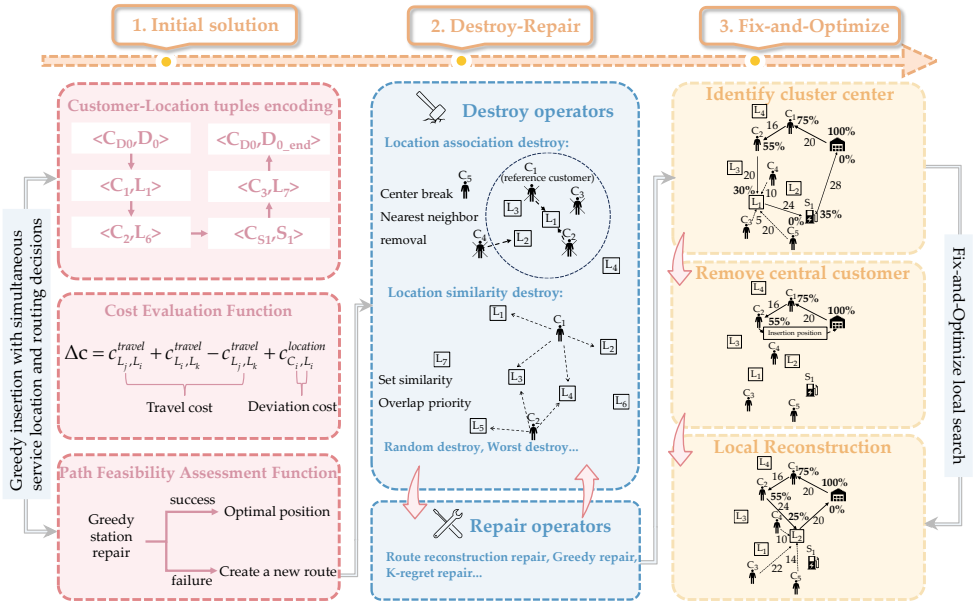

Figure 3. The structure of the MALNS-FO algorithm.

To further clarify the solution procedure, the pseudocode of the MALNS-FO algorithm is described as shown in Algorithm 1.

Algorithm 1 MALNS-FO algorithm

$S_{ini} = GenerateInitialSolution()$
$S_{best} = S_{ini}; S_{cur} = S_{ini}; T = T_{ini}$
while termination conditions are not met **do:**
select destroy and repair operators adaptively
  $S_{des} = Destory(S_{cur})$
  $S_{new} = Repair(S_{des})$
  if $f(S_{new}) < f(S_{best})$ **then**:
    $S_{best} = S_{new}; S_{cur} = S_{new}$
  end if
  if $f(S_{new}) < f(S_{cur})$ **then:**
    $S_{cur} = S_{new}$
  else then:
    accepted S_{new} as S_{cur} according to the simulated annealing criterion
  end if
update the scores of operators and penalty coefficient
  $T = T * K$
  $S_{cur}, S_{best} = FixAndOptimize(S_{cur}, S_{best})$
end while
return S_{best}

5.1. Solution Encoding

To capture the feature of the EVRPTW-FSL problem, where service location assignment and vehicle routing must be optimized simultaneously, we develop a solution encoding based on a list of customer-location tuples. Unlike traditional methods that only encode the sequence of customer visits, this encoding scheme represents each customer visit as a customer-location tuples and organizes the visit sequence of a single vehicle into a list of

sub-paths. In this way, both routing and service location assignment decisions are captured within a unified data type.

Specifically, a complete solution is denoted as $routes = [route_1, route_2, \ldots, route_k]$, where each $route_i$ corresponds to a sequence of a single vehicle. Each route is an ordered sequence of customer-location tuples. For instance, $route_i = [< C_{D0}, D_0 >, < C_1, L_1 >, < C_2, L_6 >, < C_{S1}, S_1 >, < C_3, L_7 >, < C_{D0}, D_{0\_end} >]$ indicates that the vehicle departs from depot D_0, serves C_1 at location L_1, then serves C_2 at L_6, visits recharging station S_1, serves C_3 at L_7, and finally returns to the depot D_{0_end}. By explicitly recording service locations, this encoding scheme directly supports flexible selection among different service locations for the same customer, while the sequential structure maintains route feasibility. In subsequent optimization processes, destroy and repair operators can be directly applied to this representation to jointly adjust service locations and reconstruct routes.

The route encoding is consistent with the mathematical model. In the MALNS-FO algorithm, each route is represented as an ordered sequence of tuple vertices, starting from a departure-depot tuple vertex and ending at an arrival-depot tuple vertex. If two consecutive tuples $< i, l >$ and $< j, r >$ appear in a route, the corresponding arc variable $x_{i,l,j,r}$ is equal to 1, otherwise, it is equal to 0. Moreover, when tuple $< i, l >$ appears in a route, customer i is assigned to service location l. Therefore, the encoded solution simultaneously represents vehicle routing and service location assignment, which is consistent with the tuple-based mathematical model.

5.2. Initial Solution Construction

To efficiently generate a high-quality and feasible initial solution, this study proposes a fast greedy insertion algorithm with simultaneous service location and routing decisions. By simultaneously optimizing the assignment of service locations and the construction of route sequences, this method gradually constructs feasible solutions. The procedure begins with an empty set of routes, and at each iteration selects an unserved customer along with its candidate service locations, evaluating the total incremental cost for all feasible insertion positions. This cost jointly considers travel cost and location deviation cost. Specifically, when inserting node $< C_i, L_i >$ between nodes $< C_j, L_j >$ and $< C_k, L_k >$, the incremental cost Δc is defined as $\Delta c = c^{travel}_{L_j,L_i} + c^{travel}_{L_i,L_k} - c^{travel}_{L_j,L_k} + c^{location}_{C_i,L_i}$, where the first three terms account for the variation in travel cost and the last term denotes the deviation cost. The algorithm prioritizes the insertion operation with the minimum incremental cost. If the current route cannot accommodate further customer insertions due to battery limitations, the basic greedy recharging station insertion operator described in Section 5.3.2 is invoked to insert a charging station at the optimal position to ensure energy feasibility. If the constraints still cannot be satisfied, a new route is created, and the above process is repeated until all customers are served. By coupling location assignments with route construction in a unified decision process, the proposed algorithm effectively balances travel cost and deviation cost, thereby providing high-quality initial solutions for subsequent optimization.

5.3. Operator Design

Based on the initial feasible solution, the destroy and repair operators collaboratively drive the iterative optimization process in the solution space. The destroy operators designed in this study consist of two types: customer destroy operators and recharging station destroy operators, while the repair operators include customer repair operators and recharging station repair operators.

5.3.1. Destroy Operator

The customer destroy operator selects γ customers from the current routes according to different rules, then removes the corresponding customer-location tuples and adds

the removed customers into a removal list. The value of γ depends on the total number of customers *nc* and is determined randomly between *removal_lower* and *removal_upper* using a uniform distribution.

We design location association destroy and location similarity destroy for flexible service locations. In addition, several standard operators are also adopted, including random destroy, worst destroy, Shaw destroy, and zone-based destroy.

1. Location association destroy: A customer is first randomly selected as the reference, and all customers served at the same location are removed. If the number of customers removed is less than γ, customers whose service locations are closer to that of the reference customer are preferentially removed. Figure 4 illustrates an example of the execution process of this operator. Specifically, Customer C_1 is first randomly selected as the reference customer. Customer C_2, who is being served at the same service location 2 as the reference customer, is removed. Since the service locations of customers C_3 and C_4 are close to location L_1, they are subsequently removed in sequence.
2. Location similarity destroy: A customer is first randomly selected as the reference, and the similarity between this customer and the others is then calculated. The similarity is defined as the number of common possible service locations divided by the number of locations available for the customer with fewer location flexibility, and customers with higher similarity are removed first. Figure 5 illustrates an example of similarity calculation. Customer C_1 has 5 candidate locations (including its original service location and $L_1 - L_4$), while customer C_2 has 4 (including its original service location and $L_3 - L_5$); since they share locations L_3 and L_4, the similarity between them is $2/\min(5, 4) = 0.5$.
3. Random destroy: γ customers are randomly selected from the current solution and removed.
4. Worst destroy: Remove customers with high cost, where cost is the sum of travel cost of the customer from the preceding and succeeding vertices in the route, plus the deviation cost.
5. Shaw destroy: A customer is first randomly selected as the reference. Calculate the similarity between other customers and the reference customer based on factors such as location, time window, and cargo demand, and prioritize removing customers with higher similarity to the reference customer.
6. Zone destroy: The Cartesian coordinate system in which the vertices are located is divided into smaller areas that are called zones. A zone is randomly selected and all the customers in that zone are removed from the solution.

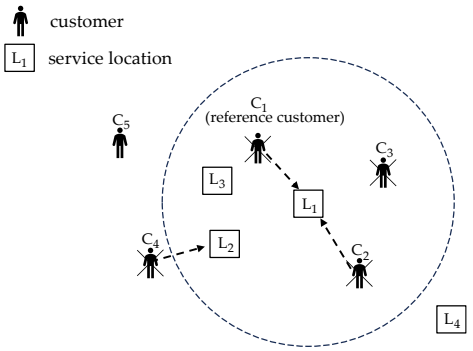

Figure 4. Location association destroy operator.

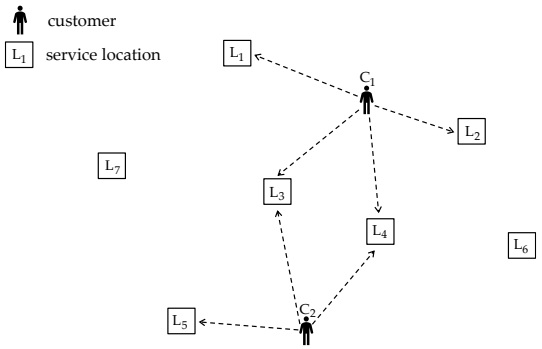

Figure 5. Location similarity destroy operator.

Changes in customer service sequences and service location assignments may lead to different charging requirements. Therefore, after a fixed number of iterations, recharging stations are removed and reinserted to improve the current solution by identifying more efficient charging strategies. To this end, four charging station destroy operators are employed in this study: random station destroy, worst distance station destroy, worst charge usage station destroy, and full charge station destroy.

1. Random station destroy: A set of recharging stations is randomly selected from the current solution and removed.
2. Worst distance station destroy: Recharging stations that contribute to higher travel cost are removed first.
3. Worst charge usage station destroy: Recharging stations are sorted in non-increasing order of charging amount, and those with smaller charging amounts are removed first.
4. Full charge station destroy: Recharging stations where vehicles leave with a full battery are removed first.

5.3.2. Repair Operator

This study employs the classic greedy repair operator and the k-regret repair operator. However, due to the flexibility in customer service location selection, exhaustively evaluating all possible insertion options in large-scale instances would incur substantial computational overhead. Therefore, this study significantly reduces the computational load by performing reasonable pruning to avoid unreasonable insertions in advance. In addition, a route reconstruction repair operator is developed to prevent excessive concentration of customers on existing routes.

1. Greedy repair: Determine the best insertion position for customer by calculating the cost of inserting it between all feasible pairs and selecting the position with the minimum cost. The procedure is repeated for all customers and the customer who has the minimum insertion cost is inserted to its designated position. To improve efficiency, several heuristic pruning strategies are applied during operator execution, including distance-based pruning, capacity-based pruning, and time-window violation pruning. Distance-based pruning avoids insertions into routes that are excessively far from the customer, which would otherwise incur high travel cost. Capacity-load pruning removes insertions that would violate vehicle capacity. Time-window violation pruning preemptively filters out insertions that violate the time window. By discarding infeasible insertions in advance rather than checking route feasibility after insertion, the computational load can be significantly reduced.
2. K-regret repair: Calculate the difference between the cost of the first and kth best insertions of the customers and inserts the one with the highest difference into its best position. Similar to the greedy insertion operator, the k-regret insertion operator also incorporates multiple heuristic pruning strategies, including distance-based pruning,

capacity-based pruning, and time-window violation pruning. In this study, we utilize 2-regret and 3-regret repair operators.

3. Route reconstruction repair: Create a new route, and unserved customers are sequentially inserted into the new route in a greedy manner. Unlike traditional repair operators that are restricted by existing route structures, whereas the route-reconstruction insertion operator achieves structural improvement by constructing new routes. Similar to the greedy insertion operator, the route-reconstruction insertion operator also incorporates the aforementioned heuristic pruning strategies to reduce computational time.

These pruning strategies only reduce candidate insertion positions for efficiency, while the final acceptance of a move is still based on the complete feasibility check consistent with the mathematical model.

This study employs three recharging station repair operators: the basic greedy station repair, the greedy station repair with comparison, and the best station repair.

1. Basic greedy station repair: Determine the first customer in the route at which the vehicle arrives with a negative battery level and insert the station that increases the travel cost least on the arc between that customer and the previous customer. If this insertion is not feasible, then the previous arcs are attempted in the same manner.
2. Greedy station repair with comparison: Determine the best charging station on the arc leading to the customer at which the battery level becomes negative and compare this outcome with that obtained by inserting the corresponding best station on the immediate predecessor arc. The insertion which increases the route distance the least is performed.
3. Best station repair: Determine the best charging station insertion between the customer at which the vehicle arrives with a negative battery level and the depot or a previously visited station by examining all preceding arcs along the route. Then, select the best feasible insertion and perform it.

5.3.3. Feasibility Check

To ensure the consistency between the MALNS-FO algorithm and the mathematical model, a feasibility check is performed after each repair operation. A candidate solution is accepted only if it satisfies all feasibility requirements defined in the mathematical model. The feasibility check includes four specific verification functions.

1. Customer Uniqueness Check: This function scans all routes and records the customer index of each visited customer-location tuple. If a customer appears more than once or does not appear in any route, the solution is regarded as infeasible.
2. Vehicle Capacity Check: By traversing each route sequentially, this function dynamically tracks the vehicle's payload after each service. If the total demand of a route exceeds W, the route is infeasible.
3. Battery and Time-Window Check: This function is conducted by a forward evaluation along each route. For each selected route, it computes the arrival time, service start time, and state of charge at each vertex. If a vehicle arrives at a recharging station, the function dynamically calculates the required charging duration based on the actual power replenished. A move is deemed infeasible if the arrival time violates a customer's time window or if the state of charge drops below zero before reaching a station, ensuring alignment with the time continuous and charging mechanics.
4. Service Location Capacity Check: This function aggregates the total number of customers assigned to each service location across all generated routes. If the number of assigned customers is larger than MC_l, the solution is infeasible.

5.4. Fix-and-Optimize Local Search Mechanism

Traditional repair operators typically follow a "single insertion" strategy, in which customers are inserted individually and sequentially into routes according to a given order. This means that earlier insertions may affect subsequent insertion processes, a phenomenon that becomes more pronounced in EVRPTW-FSL. Figure 6a illustrates a route to be repaired, where customers C_3, C_4, and C_5 are to be inserted. Figure 6b shows the result of inserting customers using the greedy repair operator. Specifically, location L_1 is first selected as the service location for customer C_3 based on the criterion of minimizing the sum of travel cost and deviation cost. Then, location L_1 is also selected for customer C_4, since choosing the same service location incurs no additional travel cost, and customer C_4 is relatively close to L_1, resulting in a low deviation cost. The same applies to customer C_5, thereby illustrating a phenomenon of L_1 serving as a "clustering center" for multiple customers.

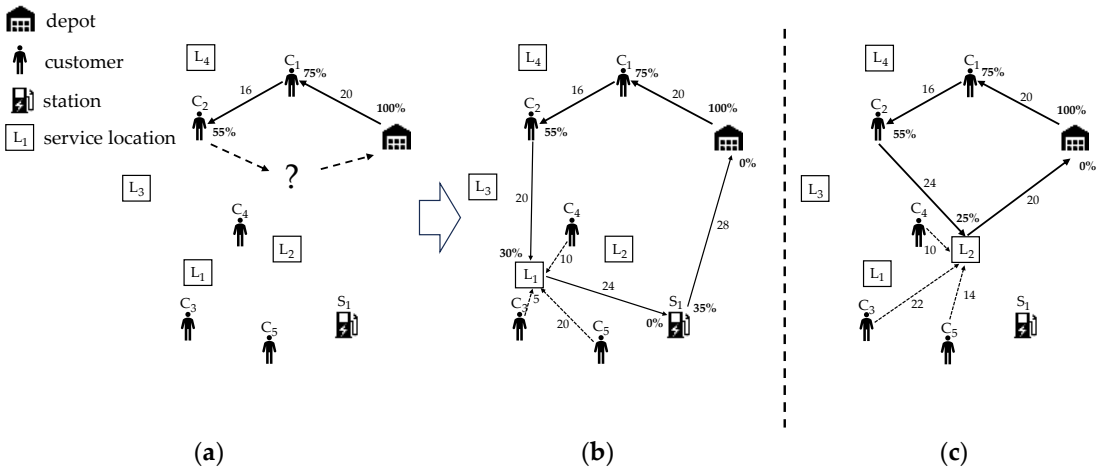

Figure 6. Comparison between local optimum and global optimum. (**a**) Customers to be inserted; (**b**) local optimal state; (**c**) global optimal state.

The routing solution shown in Figure 6c represents the optimal solution to this instance, where L_2 acts as the clustering center and the vehicle does not need to visit recharging station S_1. It is easy to see that the traditional single insertion operator may get trapped in the aforementioned "irrational gathering" local optimum.

Therefore, to address the aforementioned local optimum phenomenon, this paper proposes the FO local search mechanism. The idea of FO is to fix the values of a subset of decision variables while optimizing the remaining ones, thereby reducing the problem size and improving computational efficiency [35]. The FO local search consists of three steps: First, identify cluster centers along the route. If multiple consecutive customers are served at the same location, record that location and the number of customers served. A roulette wheel selection is then performed based on the number of customers served at each recorded location; locations serving more customers are more likely to be selected. The selected location is the cluster center to be disrupted, such as point L_1 in Figure 7a. Second, all customers served at the selected clustering center are removed from the route. Meanwhile, if the clustering center is adjacent to a recharging station, the corresponding recharging station is also removed, as illustrated in Figure 7b. This removal creates "insertion positions" in the route, while the remaining parts are kept unchanged, corresponding to the fixed variables in the model. Third, a commercial solver is invoked to determine the service locations and visiting sequence of the removed customers, with the requirement that their insertion must be confined to the insertion positions identified in the second step.

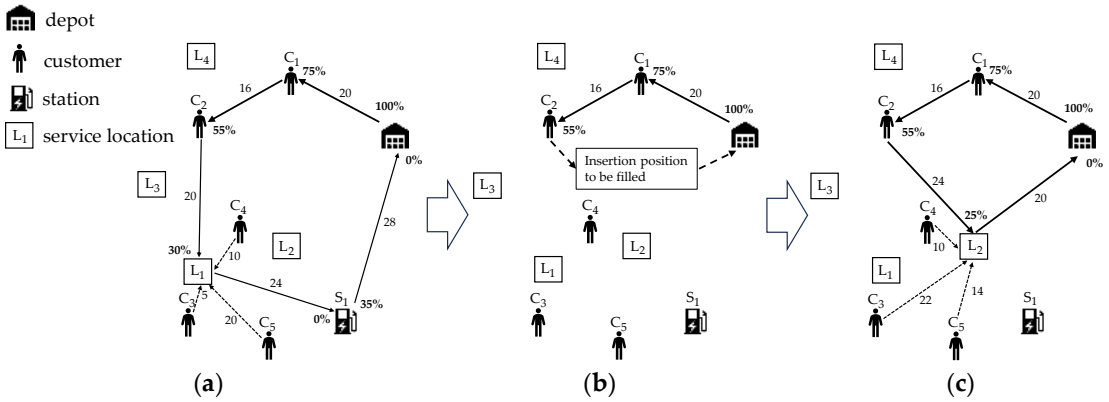

Figure 7. Schematic diagram of the FO local search process. (**a**) Local optimal state; (**b**) customers at the gathering center are removed; (**c**) global optimal state.

To make the FO procedure reproducible, we further specify the corresponding subproblem. After the cluster center is selected, all customers served at this location are removed from the current route, and the remaining route segments are fixed. The insertion position is defined as the gaps between two consecutive fixed vertices after the removal operation. The FO subproblem only optimizes the removed customers within these insertion positions, including their service location choices, visiting sequence, and the related charging-station decisions inside the local area. The load, time, and battery states are then updated according to the same feasibility rules as in the mathematical model. If the commercial solver obtains an optimal or feasible solution within 60 s, the local route segment is replaced by the solver solution only when the complete route passes the feasibility check and improves the objective value. If no feasible solution is found within the time limit, the original route is kept unchanged.

5.5. Simulated Annealing Criterion

The solution update mechanism follows the simulated annealing criterion. A new solution is accepted directly if it improves the current solution; otherwise, it is accepted with probability $e^{\frac{-(f(S_{new})-f(S_{cur}))}{T_{ini}K^{\beta}}}$, where $f(S_{new})$ and $f(S_{cur})$ denote the objective values of the new and current solutions, respectively. T_{ini} is the initial temperature, K is the cooling factor with $0 < K < 1$, and β is the current iteration number. By regulating the acceptance of inferior solutions through a temperature-controlled mechanism, the simulated annealing criterion dynamically balances exploration and exploitation, thereby enhancing the algorithm's ability to approach global optimality.

5.6. Adaptive Weight Adjustment

During the iteration process of the MALNS-FO algorithm, operator weights are updated adaptively based on their historical performance. Each operator is scored according to the quality of the solutions it produces (e.g., improvement, acceptance, or rejection). A reaction-based weight updating strategy is then periodically applied to adjust the selection probabilities, allowing more effective operators to receive higher weights while preserving the exploration potential of less frequently selected ones, thereby guiding the search process adaptively. All operators are initialized with equal weights. At the end of each iteration cycle, the weight of operator i, denoted as ω_i, is updated as $\omega_i' = (1-r)\omega_i + r(\pi_i/\theta_i)$, where r controls how quickly the weight adjustment algorithm reacts to changes in the effectiveness of the heuristics, θ_i is the number of times operator i is applied within the cycle, and π_i is its accumulated score. The score is determined by solution quality: obtaining a new global best, improving the current solution, or producing a worse solution

that is still accepted under the simulated annealing criterion corresponding to scores σ_1, σ_2, and σ_3, respectively.

6. Experimental Tests

To evaluate the performance of the MALNS-FO algorithm, this paper assesses its effectiveness from two dimensions: solution quality and computational efficiency. In addition, we analyze the effectiveness of the operators, validate the FO mechanism, and examine the impact of deviation cost. All experiments are performed on a Windows 10 system equipped with 16 GB of memory and an Intel(R)Core(TM) i7-8550UCPU@1.80 GHz. The algorithm is implemented in Python 3.8 using Anaconda, and Gurobi 9.5.1 is employed to solve the mathematical model.

6.1. Test Instances and Parameter Settings

Since EVRPTW-FSL is a new variant of the EVRP, there are currently no standard benchmark instances for this problem. Therefore, the test instances are generated by extending the classical EVRPTW benchmark instances of Schneider et al. [17], including three classes where customers and recharging stations are clustered (C), randomly distributed (R), and both clustered and randomly distributed (RC). The original information in these instances, including the depot, customers, charging stations, customer demands, service times, time windows, vehicle capacity, battery capacity, and travel-related parameters, is retained. To construct the EVRPTW-FSL instances, additional service locations are randomly generated on the same map using fixed random seeds. For each customer, a service location is considered as a candidate location if its distance from the customer does not exceed 10% of the maximum spatial span of the instance map. The customer-location tuples are then generated based on these candidate service locations. The capacity of each service location is set to 5% of the number of customers in the corresponding instance. The generated instances and the related random-seed information are publicly available in the GitHub repository described in the Data Availability Statement.

To make the scale of the test instances explicit, the number of customers in each instance group is summarized in Table 3. The small-size instances are used for exact-solver validation, and their instance names indicate the number of selected customers. Specifically, instances ending with C5, C10, and C15 contain 5, 10, and 15 customers, respectively. The large-size instances retain the original 100-customer structure of the Schneider et al. benchmark and are used for comparative experiments among metaheuristic algorithms. The number of used vehicles is not fixed in advance in the test instances; instead, it is determined by the solution through the number of constructed routes.

Table 3. Number of customers in the test instances.

Instance Scale	Instance Group/ Naming Rule	Distribution Type	No. of Customers Per Instance	No. of Locations per Instance
Small size	Instances ending with C5	C, R, RC	5	15
Small size	Instances ending with C10	C, R, RC	10	30
Small size	Instances ending with C15	C, R, RC	15	45
Large size	c101–c109, c201–c208	C	100	300
Large size	r101–r112, r201–r211	R	100	300
Large size	rc101–rc108, rc201–rc208	RC	100	300

Note: C, R, and RC denote clustered, randomly distributed, and mixed customer distributions, respectively.

In the numerical experiments, the parameter settings follow Ropke [34], with $\omega = 0.5$, $K = 0.9995$, $\sigma_1 = 33$, $\sigma_2 = 20$, $\sigma_3 = 12$, and $r = 0.85$. The FO local search is applied when the iteration count reaches a multiple of $N_{FO} = 100$. The maximum running time for

Gurobi is set to 7200 s. In the large-scale instance experiments, the MALNS-FO algorithm terminates after reaching 2500 iterations.

For each physical recharging station $s \in S$, $2|C|$ recharging-station tuple vertices are created in the exact Gurobi model, where $|C|$ is the number of physical customers. This setting follows the worst-case treatment used in related CEVRP formulations, where a sufficiently large number of charging-station copies is introduced to allow each physical charging station to be visited zero, one, or multiple times [36].

6.2. Experimental Results and Comparative Analysis

6.2.1. Numerical Results for Small-Size Instances

Table 4 compares the performance of Gurobi and the MALNS-FO algorithm on 32 small-size instances, where *TC* denotes the total cost and *RT* denotes the computation time in seconds (s). The symbol "-" indicates that Gurobi did not obtain a feasible incumbent solution within the maximum running time. For these cases, no objective-value comparison with Gurobi is made, and the results of MALNS-FO are reported only as feasible solutions.

Table 4. Comparison of solution results for small-scale instances.

Instance Name	Gurobi		MALNS-FO		Gap
	TC	***RT***	***TC***	***RT***	
c101C5	207.55	156.38	207.55	12.75	0.00%
c103C5	136.36	289.71	136.36	26.38	0.00%
c206C5	192.21	421.05	192.21	18.91	0.00%
r104C5	121.26	367.92	121.26	23.05	0.00%
r105C5	140.23	198.46	140.23	15.64	0.00%
r202C5	121.99	512.83	121.99	28.21	0.00%
r203C5	175.40	234.59	175.40	17.87	0.00%
rc105C5	199.75	478.20	199.75	21.52	0.00%
rc108C5	252.24	310.17	252.24	19.93	0.00%
rc204C5	167.08	265.64	167.08	25.14	0.00%
rc208C5	165.61	559.32	165.61	16.25	0.00%
c101C10	343.39	3562.78	343.39	45.28	0.00%
c104C10	268.92	4891.05	268.92	59.71	0.00%
c202C10	227.38	6217.39	227.38	63.05	0.00%
c205C10	216.11	5103.64	216.11	48.92	0.00%
r102C10	214.17	3985.21	214.17	52.46	0.00%
r103C10	-	7200.00	152.46	67.83	-
r201C10	174.72	7129.87	174.72	43.59	0.00%
r203C10	214.18	4346.52	214.18	56.20	0.00%
rc102C10	339.50	5780.93	339.50	49.17	0.00%
rc108C10	-	7200.00	329.91	58.64	-
rc201C10	276.86	6512.14	276.86	65.32	0.00%
rc205C10	297.74	6832.46	297.74	47.98	0.00%
c103C15	-	7200.00	315.88	192.68	-
c106C15	-	7200.00	247.91	218.45	-
c202C15	-	7200.00	356.84	235.12	-
r102C15	-	7200.00	279.47	187.90	-
r202C15	-	7200.00	314.14	205.37	-
rc103C15	-	7200.00	370.48	241.79	-
rc108C15	-	7200.00	366.50	198.23	-
rc202C15	-	7200.00	358.51	226.51	-
rc204C15	-	7200.00	286.90	183.84	-

Note: $Gap = \frac{100\% * (TC(ALNS-FO) - TC(Gurobi))}{TC(Gurobi)}$.

As shown in Table 4, among the 21 instances where Gurobi is able to obtain optimal solutions, the MALNS-FO achieves exactly the same optimal solutions, with all gaps equal to 0.00%. For the remaining 11 instances, Gurobi does not obtain a feasible incumbent solution within the time limit, and therefore no objective-value comparison is made for these cases. In contrast, MALNS-FO is able to generate feasible solutions for all instances within a much shorter computational time. In terms of computational efficiency, as the problem size increases from C5 to C15, the computation time of Gurobi exhibits a typical exponential growth trend, reaching the maximum time limit of 7200 s for several instances. In contrast, the computation time of MALNS-FO increases approximately linearly, with significantly lower average computational time. For example, for the C15 instances, MALNS-FO requires on average only 210 s, which is substantially lower than the 7200 s required by Gurobi. Overall, the small-size experiments mainly verify the solution correctness and feasibility-finding ability of MALNS-FO, and show clear advantages on larger instances that are difficult for Gurobi to handle within the time limit.

6.2.2. Numerical Results for Large-Size Instances

To further evaluate the performance of the MALNS-FO algorithm on large-size instances, Adaptive Large Neighborhood Search (ALNS), Variable Neighborhood Search (VNS), Memetic Algorithm (MA), and Ant Colony Optimization (ACO) are selected as comparison algorithms. Jia [3], Mara [4], Souza [5] and other scholars have proven the effectiveness of these algorithms in solving routing optimization problems. For each large-size instance, each algorithm was independently run 20 times, and the reported objective value and runtime are the average values over the 20 runs. The parameter settings of the comparison algorithms are adopted from the original studies or widely recognized benchmark references in the literature, with default values corresponding to those validated through parameter tuning. All comparison algorithms were adapted to the EVRPTW-FSL structure and implemented under the same computational environment, stopping criterion, route encoding logic, station repair mechanism, and feasibility checks. The results comparing the total cost and runtime of each algorithm under large-size instances are shown in Table 5. The relative percentage difference (*RPD*) is also reported to compare MALNS-FO with the best-performing comparison algorithm.

As shown in Table 5, in terms of solution quality, the results obtained by the MALNS-FO algorithm are superior to or equal to those of the four comparison algorithms across all 56 large-size instances. For type-C instances with clustered customer distributions, MALNS-FO demonstrates the most significant advantage, with an average *TC* of 832.60 and an average *RPD* of −7.29%. The negative *RPD* value indicates that MALNS-FO obtains a lower total cost than the best-performing comparison algorithm. This improvement is mainly attributed to the FO mechanism, which effectively identifies and restructures the "irrational clustering" phenomenon. For type-R instances with randomly distributed customers, the performance gap among the algorithms becomes smaller; however, MALNS-FO still maintains its lead, with an average *TC* of 1105.11 and an average *RPD* of −2.02%. For type-RC instances with mixed distributions, MALNS-FO achieves an average *TC* of 1260.11 and an average *RPD* of −2.14%. In terms of computational efficiency, the average computation time of MALNS-FO is lower than that of the comparison algorithms in all three types of instances. For example, compared with VNS, the average computation time is reduced by approximately 24%, mainly due to the significant reduction in the computational effort of the insertion operators enabled by multiple heuristic pruning strategies. Overall, the experimental results on large-size instances demonstrate the effectiveness of the MALNS-FO algorithm for solving the EVRPTW-FSL. The algorithm outperforms the comparison algorithms in both solution quality and computational efficiency, with particularly pronounced advantages observed in type-C instances.

Table 5. Comparison of solution results for large-scale instances.

Instance Name	MALNS-FO		ALNS		VNS		MA		ACO		*RPD*
	TC	*RT*	*TC*	*RT*	*TC*	*RT*	*TC*	*RT*	*TC*	*RT*	
c101	1051.28	3126.48	1146.18	5862.37	1112.04	4926.37	1112.99	5626.38	1111.83	6026.37	−5.45%
c102	1025.08	2879.35	1159.37	4315.72	1096.02	4189.75	1096.02	4391.75	1090.89	4689.75	−6.03%
c103	995.93	3562.71	1031.58	5298.15	1074.11	4963.12	1074.11	5168.24	1067.54	5412.89	−3.46%
c104	939.53	3781.29	1092.11	5739.46	1021.36	4678.54	1021.36	3587.59	1013.28	7097.54	−7.28%
c105	1029.78	2745.63	1232.70	5371.90	1129.05	4712.96	1129.05	5193.12	1115.35	4938.12	−7.67%
c106	1021.53	3308.92	1209.08	6153.28	1129.61	4350.28	1129.61	5206.87	1113.16	5236.87	−8.23%
c107	1039.51	3015.47	1220.54	5353.81	1158.33	5041.63	1101.15	5241.63	1137.43	5541.63	−5.60%
c108	1027.49	3842.16	1251.05	5620.83	1088.42	5298.41	1101.26	3712.94	1128.70	5012.94	−5.60%
c109	934.67	2903.85	1082.70	5391.05	1001.78	4687.25	1011.69	4678.35	1030.57	5178.35	−6.70%
c201	633.21	3476.52	708.73	5535.61	685.39	5073.89	691.28	4982.71	701.03	6892.71	−7.61%
c202	625.27	3259.74	649.47	5784.29	682.61	4895.32	649.47	5305.46	662.35	5305.46	−3.73%
c203	629.54	2689.03	739.16	5289.34	694.32	4126.71	670.15	5097.28	672.22	4607.28	−6.06%
c204	676.41	3614.28	775.89	5316.7	756.70	4660.94	728.43	4759.61	727.41	5059.61	−7.01%
c205	622.43	3092.65	728.66	5412.48	662.58	4332.58	677.39	4563.98	673.22	6163.98	−6.06%
c206	641.64	2817.91	740.74	5378.92	690.98	4917.65	704.91	5189.32	700.16	5489.32	−7.14%
c207	619.25	3508.37	753.442	6220.55	673.93	4789.31	686.25	5496.75	684.21	6596.75	−8.11%
c208	641.68	3382.54	722.38	4994.13	704.95	4392.87	681.40	4831.49	681.79	5231.49	−5.83%
Avg . (C)	832.60	3235.70	955.52	5472.86	903.66	4708.09	898.03	4884.32	900.66	5557.71	−7.29%
r101	1709.43	3726.89	1709.43	4690.76	1709.43	4576.43	1709.43	4472.86	1709.43	4872.86	0.00%
r102	1566.89	2956.12	1629.38	5165.30	1594.94	4054.19	1600.58	4710.53	1611.86	5510.53	−1.76%
r103	1299.21	3187.40	1389.09	5947.59	1327.14	4638.72	1337.28	4608.17	1338.84	3008.17	−2.10%
r104	1095.40	3679.23	1166.03	4787.24	1123.66	3215.96	1113.80	3459.62	1124.32	5159.62	−1.65%
r105	1422.16	2705.68	1422.16	5031.98	1422.16	3340.25	1422.16	3854.38	1422.16	4954.38	0.00%
r106	1309.93	3421.95	1365.07	4535.60	1352.76	3187.63	1351.72	4212.89	1336.26	5312.89	−1.97%
r107	1208.97	3054.76	1283.21	4168.42	1226.62	4509.18	1226.14	4363.25	1229.28	4663.25	−1.40%
r108	1082.29	3810.32	1144.79	5383.17	1103.07	3563.54	1111.62	5228.71	1106.86	5028.71	−1.88%
r109	1224.66	2893.51	1224.66	3510.59	1253.68	2732.87	1266.05	4876.54	1262.50	4876.54	0.00%
r110	1190.06	3296.87	1246.55	5359.23	1222.31	4756.21	1212.08	4310.96	1229.09	5210.96	−1.82%
r111	1124.32	3587.19	1197.58	5527.85	1158.61	4320.79	1150.07	4532.47	1156.59	4732.47	−2.24%
r112	1222.68	3142.63	1297.13	5642.01	1264.37	3518.46	1259.85	4589.31	1258.38	5489.31	−2.84%
r201	1139.84	2650.94	1173.35	2349.67	1153.29	3389.32	1152.72	4927.68	1173.35	4927.68	−1.12%
r202	983.03	3753.26	983.03	5403.94	999.25	4671.58	1008.29	3693.25	1005.34	5193.25	0.00%
r203	880.93	3350.78	931.56	5316.38	899.34	4295.84	909.65	4578.94	895.82	5578.94	−1.66%
r204	735.89	2981.45	783.67	4768.51	753.85	4863.17	750.17	4405.72	753.48	4605.72	−1.90%
r205	1080.38	3638.91	1104.79	5382.79	1111.28	3954.69	1110.63	4816.39	1109.01	5016.39	−2.21%
r206	951.49	3027.53	977.79	5101.43	981.84	4538.25	966.33	4967.58	978.42	4867.58	−1.54%
r207	826.28	2846.17	856.26	5457.86	838.92	3620.73	845.95	5042.13	845.53	5342.13	−1.51%
r208	739.13	3529.64	778.36	5689.25	752.95	4187.51	764.33	4768.95	756.50	4968.95	−1.84%
r209	949.79	3218.30	973.72	5494.10	971.26	3826.94	966.22	4521.67	974.96	5221.67	−1.70%
r210	861.58	3876.59	1136.44	5275.32	884.76	4378.36	884.15	4519.42	888.63	3719.42	−2.55%
r211	813.17	2768.24	864.11	4764.98	840.17	3752.18	822.03	4299.83	839.52	5499.83	−1.08%
Avg . (R)	1105.11	3265.41	1158.18	4989.30	1128.07	3995.34	1127.88	4511.36	1130.70	4946.14	−2.02%
rc101	1709.19	3491.80	1806.50	5583.54	1758.24	3236.82	1730.90	4905.26	1763.37	5905.26	−1.25%
rc102	1637.21	3165.72	1774.36	4878.06	1697.62	4615.47	1672.41	4638.71	1716.12	5138.71	−2.10%
rc103	1412.61	3698.41	1526.59	6062.71	1476.46	3550.93	1454.00	5203.59	1492.99	5503.59	−2.85%
rc104	1316.05	2927.36	1354.74	4915.89	1386.85	4817.65	1338.16	4487.34	1354.74	4687.34	−1.65%
rc105	1583.70	3325.97	1618.38	4954.21	1680.94	3968.29	1622.66	4852.96	1649.42	5052.96	−2.14%
rc106	1381.12	3079.15	1504.53	5039.68	1421.86	4589.71	1425.18	4330.18	1454.04	5330.18	−2.87%
rc107	1353.30	3792.68	1468.07	5828.95	1404.86	4193.54	1372.52	5156.72	1402.42	5556.72	−1.40%
rc108	1215.81	2698.42	1215.81	5015.73	1215.81	4956.82	1215.81	4593.41	1215.81	4993.41	0.00%
rc201	1331.89	3551.83	1371.18	5742.07	1406.88	4763.27	1376.91	4720.85	1413.40	5220.85	−2.87%
rc202	1188.64	3274.59	1280.12	5141.50	1264.36	4340.91	1210.63	5018.63	1221.68	4818.63	−1.82%
rc203	980.13	3830.27	1060.11	5619.36	1006.99	5032.64	1002.57	4456.29	1018.94	5156.29	−2.24%
rc204	896.26	2863.70	994.19	5005.49	928.35	4075.38	923.51	4897.54	944.75	5397.54	−2.95%
rc205	1126.25	3405.21	1197.24	5237.82	1175.69	4659.12	1138.98	4582.76	1175.69	5482.76	−1.12%
rc206	1099.62	3109.86	1186.68	4708.65	1156.47	4287.65	1127.88	5234.19	1169.45	4734.19	−2.51%
rc207	948.39	3654.39	1032.54	5652.14	1005.10	4890.37	979.31	4671.38	979.50	6071.38	−3.16%
rc208	981.65	2789.53	1081.55	5568.03	1043.00	3942.81	1010.51	4976.82	1031.03	5376.82	−2.86%
Avg . (RC)	1260.11	3291.18	1342.04	5309.61	1314.34	4370.09	1287.62	4795.41	1312.71	5276.66	−2.14%

Note: The rows labelled Avg. (C), Avg. (R), and Avg. (RC) report the average results over all large-size instances of the corresponding distribution type. All large-size instances in this table contain 100 customers. *RPD* denotes the relative percentage difference between MALNS-FO and the best-performing comparison algorithm. $RPD = \frac{100\% * (TC(MALNS-FO) - \min(TC(ALNS), TC(VNS), TC(MA), TC(ACO)))}{\min(TC(ALNS), TC(VNS), TC(MA), TC(ACO))}$. Since the objective is to minimize the total cost, a negative *RPD* value indicates that MALNS-FO obtains a lower total cost than all comparison algorithms, a zero value indicates the same total cost as the best comparison algorithm, and a positive value indicates that at least one comparison algorithm obtains a lower total cost than MALNS-FO. A smaller *RPD* value indicates better performance of MALNS-FO relative to the comparison algorithms.

Table 6 reports the results of the Wilcoxon rank-sum test comparing MALNS-FO with other comparison algorithms in terms of total cost and computation time, where $p < 0.01$ indicates a statistically significant difference. At the 99% confidence level, MALNS-

FO performs significantly better than the comparison algorithms in both total cost and computation time.

Table 6. Wilcoxon signed-rank test of MALNS-FO and four comparison algorithms.

Test Statistics	Algorithm Pair	Difference	Z	*p*	Cohen's d
TC	MALNS-FO vs. ALNS	82.52 ± 58.00	6.2146	<0.0001	1.4227
	MALNS-FO vs. VNS	46.50 ± 27.48	6.3342	<0.0001	1.6924
	MALNS-FO vs. MA	37.07 ± 24.10	6.3342	<0.0001	1.5382
	MALNS-FO vs. ACO	46.20 ± 25.35	6.3342	<0.0001	1.8222
RT	MALNS-FO vs. ALNS	1963.86 ± 579.01	6.5012	<0.0001	3.3918
	MALNS-FO vs. VNS	1055.03 ± 637.41	6.3136	<0.0001	1.6552
	MALNS-FO vs. MA	1441.99 ± 642.55	6.4278	<0.0001	2.2442
	MALNS-FO vs. ACO	1962.48 ± 680.90	6.4849	<0.0001	2.8822

6.3. Operator Effectiveness Analysis

To analyze the behavior of adaptive operator selection, we recorded the selection probabilities of different destroy and repair operators over iterations. Since the selection probabilities are updated according to the historical scores of operators, this analysis helps show which operators are more often rewarded and selected during the search process. Figures 8 and 9 show how the probability of selecting the destroy and repair operators changes with the number of iterations. Black thin lines represent classic operators. Blue lines represent our operators. Gray lines represent the operators that are not the focus of the subplot. Above average performance is displayed by solid line segments, while dotted lines represent below average performance.

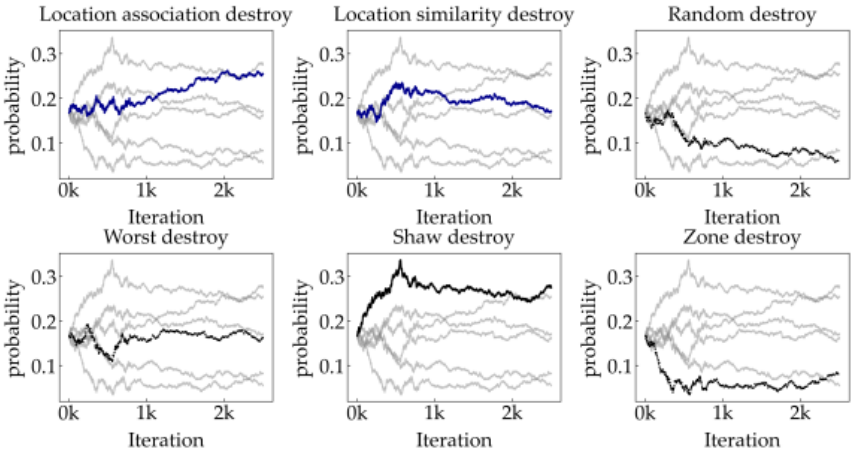

Figure 8. The probability of the destroy operator varies with the number of iterations.

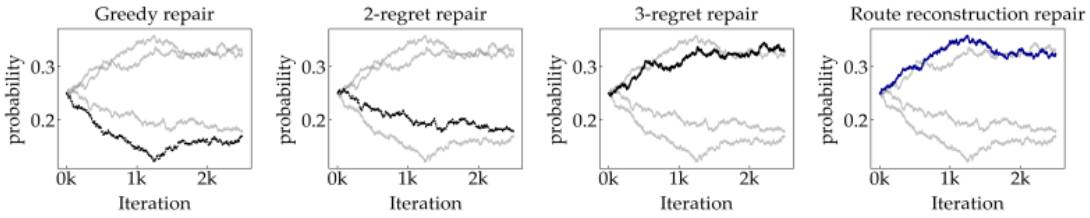

Figure 9. The probability of the repair operator varies with the number of iterations.

As shown in Figure 8, the location association destroy operator performs well among all removal operators. Its selection probability increases from about 0.18 at the beginning to around 0.25, and remains above this level in later iterations, indicating stable performance

above average. This can be attributed to its ability to remove batches of location-related customers, thereby selectively disrupting clustered customer structures and more effectively guiding the search toward improved solutions. The location similarity destroy operator also performs above the average level, but overall it is inferior to the Shaw removal operator. A possible explanation is that, although it follows the same similarity-based removal principle as the Shaw operator, the latter considers multiple attributes, including customer time windows, demand, and route information, resulting in a more comprehensive measure of similarity. As shown in Figure 9, the selection probability of the route reconstruction repair operator exhibits a steadily increasing trend during the iterative process, rising from approximately 0.22 in the initial stage to around 0.32 and eventually stabilizing above 0.30. The superior performance of this operator mainly stems from its distinctive mechanism. Unlike traditional insertion operators that are restricted to local adjustments within the existing route structure, the route reconstruction repair operator constructs entirely new route sequences, thereby overcoming structural limitations and enabling exploration of more promising regions of the solution space. Overall, the proposed location association destroy operator and route reconstruction repair operator show relatively high selection probabilities during the search process, indicating that they are frequently rewarded by the adaptive scoring rule. This result suggests that these operators are useful for guiding the search of the MALNS-FO algorithm.

6.4. Effectiveness Analysis of the FO Mechanism

To evaluate the effectiveness of the FO local search mechanism, the performance of the modified adaptive large neighborhood search without the FO mechanism (Modified Adaptive Large Neighborhood Search, MALNS) is compared with that of the MALNS-FO algorithm, as reported in Table 7. In terms of solution quality, MALNS-FO achieves total costs that are superior to or equal to those of MALNS for the vast majority of test instances. Specifically, out of 56 instances, MALNS-FO obtains better solutions in 49 instances and achieves the same results as MALNS in the remaining 7 instances. For the type-C instances, the algorithm incorporating the FO mechanism demonstrates significant advantages, with the average objective value reduced by 9% compared with the variant without FO, at the expense of an approximately 11% increase in computational time. For the type-RC instances, the average objective value is reduced by 5%, while the computational time increases by about 8%. In these instances, characterized by clustered customer distributions, the improvement brought by the FO mechanism is more pronounced, as the concentration of customer locations tends to induce locally optimal "irrational clustering" patterns, which can be effectively mitigated through targeted local reconstruction by the FO mechanism. For the type-R instances, the FO mechanism leads to an increase of approximately 6% in computational time, while multiple instances exhibit identical objective values, and the average performance gap is smaller than that observed in the type-C and type-RC instances. This is because randomly distributed customers are less likely to form "irrational clustering" patterns. Overall, the FO mechanism demonstrates good adaptability across most instances, particularly for type-C and type-RC instances, where it effectively improves solution quality with a relatively modest increase in computational time. Table 8 presents the results of the Wilcoxon signed-rank test at the 99% confidence level, where $p < 0.01$ indicates a statistically significant difference. At this confidence level, the inclusion of the FO mechanism leads to significantly better performance in terms of total cost compared with the comparison algorithm.

Table 7. Comparison of solution results between MALNS-FO and MALNS.

Instance Name	MALNS-FO		MALNS		Gap
	TC	*RT*	*TC*	*RT*	
c101	1051.28	3126.48	1127.35	2869.17	7.24%
c102	1025.08	2879.35	1126.26	2605.52	9.87%
c103	995.93	3562.71	995.93	3179.36	0.00%
c104	939.53	3781.29	1047.99	3349.47	11.54%
c105	1029.78	2745.63	1165.89	2505.11	13.22%
c106	1021.53	3308.92	1131.14	3005.16	10.73%
c107	1039.51	3015.47	1136.23	2706.38	9.30%
c108	1027.49	3842.16	1152.40	3385.33	12.16%
c109	934.67	2903.85	1068.91	2667.19	14.36%
c201	633.21	3476.52	691.38	3133.39	9.19%
c202	625.27	3259.74	625.27	2903.45	0.00%
c203	629.54	2689.03	704.24	2386.78	11.87%
c204	676.41	3614.28	736.42	3291.89	8.87%
c205	622.43	3092.65	686.25	2803.18	10.25%
c206	641.64	2817.91	686.44	2522.59	6.98%
c207	619.25	3508.37	696.08	3098.94	12.41%
c208	641.68	3382.54	690.41	3091.98	7.59%
Avg . (C)	832.60	3235.70	909.92	2912.05	9.15%
r101	1709.43	3726.89	1709.43	3605.02	0.00%
r102	1566.89	2956.12	1608.95	2820.43	2.68%
r103	1299.21	3187.40	1350.08	3021.97	3.92%
r104	1095.40	3679.23	1147.19	3427.94	4.73%
r105	1422.16	2705.68	1422.16	2504.92	0.00%
r106	1309.93	3421.95	1339.49	3117.05	2.26%
r107	1208.97	3054.76	1250.94	2937.15	3.47%
r108	1082.29	3810.32	1128.76	3651.05	4.29%
r109	1224.66	2893.51	1224.66	2727.71	0.00%
r110	1190.06	3296.87	1224.99	3090.49	2.94%
r111	1124.32	3587.19	1166.89	3302.01	3.79%
r112	1222.68	3142.63	1282.00	2880.53	4.85%
r201	1139.84	2650.94	1139.84	2557.89	0.00%
r202	983.03	3753.26	983.03	3570.48	0.00%
r203	880.93	3350.78	909.64	3167.83	3.26%
r204	735.89	2981.45	769.28	2784.97	4.54%
r205	1080.38	3638.91	1080.38	3377.64	0.00%
r206	951.49	3027.53	977.79	2765.04	2.76%
r207	826.28	2846.17	856.26	2755.38	3.63%
r208	739.13	3529.64	769.97	3377.16	4.17%
r209	949.79	3218.30	949.79	3025.85	0.00%
r210	861.58	3876.59	861.58	3607.55	0.00%
r211	813.17	2768.24	846.75	2543.46	4.13%
Avg . (R)	1105.11	3265.41	1141.43	3070.41	2.41%
rc101	1709.19	3491.80	1764.68	3307.43	3.25%
rc102	1637.21	3165.72	1709.07	2969.76	4.39%
rc103	1412.61	3698.41	1486.31	3426.58	5.22%
rc104	1316.05	2927.36	1406.00	2679.71	6.84%
rc105	1583.70	3325.97	1583.70	3002.69	0.00%
rc106	1381.12	3079.15	1451.13	2902.71	5.07%
rc107	1353.30	3792.68	1431.70	3533.26	5.79%
rc108	1215.81	2698.42	1291.90	2484.97	6.26%
rc201	1331.89	3551.83	1331.89	3234.65	0.00%
rc202	1188.64	3274.59	1238.13	2971.69	4.16%
rc203	980.13	3830.27	1034.46	3632.63	5.54%
rc204	896.26	2863.70	958.26	2678.42	6.92%
rc205	1126.25	3405.21	1170.78	3160.72	3.95%
rc206	1099.62	3109.86	1151.22	2841.79	4.69%
rc207	948.39	3654.39	1004.12	3304.30	5.88%
rc208	981.65	2789.53	1044.77	2623.83	6.43%
Avg . (RC)	1260.11	3291.18	1316.13	3047.20	4.65%

Note: $Gap = \frac{100\% * (TC(MALNS) - TC(MALNS-FO))}{TC(MALNS-FO)}$.

Table 8. Wilcoxon signed-rank test of MALNS-FO and MALNS.

Test Statistics	Algorithm Pair	Difference	Z	*p*	Cohen's d
TC	MALNS-FO vs. MALNS	49.88 ± 36.31	5.7767	<0.0001	1.3737

To further analyze the FO mechanism, we recorded several FO-related indicators over repeated runs, as shown in Table 9. The runtime of FO is included in the total runtime of MALNS-FO, and both MALNS-FO and MALNS are tested under the same stopping criterion. Therefore, the comparison does not give MALNS-FO extra computational resources outside the algorithm.

Table 9. Statistical results of the FO mechanism.

Type	Instances	Avg. Obj. Reduction	Improved Solutions	Avg. Subproblem Size	Avg. FO Runtime (s)	Success Rate
C	17	77.32	15	5	32.37	88.24%
R	23	25.32	15	2	19.49	65.22%
RC	16	56.02	14	3	24.39	87.50%
All	56	49.88	44	3	24.81	78.57%

The results show that the FO mechanism is called regularly during the search, but each FO subproblem remains small because only customers around the selected cluster center are reoptimized. The average FO runtime is also limited compared with the total runtime. Moreover, a certain proportion of FO calls successfully generate feasible local solutions, and some of them are accepted as improved solutions. This indicates that the FO mechanism can help reconstruct local route structures and escape the irrational clustering pattern, especially in type-C and type-RC instances where customer distributions are more clustered.

6.5. Impact of Deviation Costs

To evaluate the impact of different deviation cost settings, four cases are considered following [31]: (1) no deviation costs; (2) fixed deviation costs of 1; (3) costs equal to the distance $d_{l,r}$, where l and r denote the customer location and the assigned service location, respectively; and (4) costs equal to this distance squared $(d_{l,r})^2$. As shown in Figure 10, for large-size instances, both the total cost and the proportion of customers served at their original service locations increase across all three instance types as the deviation cost increases from 0 and 1 to $d_{l,r}$ and $(d_{l,r})^2$. This clearly indicates that improvements in customer service satisfaction are generally accompanied by higher operational costs. Table 10 reports the average total cost and the proportion of customers served at their original service locations under different deviation cost settings (where ρ denotes the proportion of customers assigned to their original service locations). Increasing the deviation cost from 0 to 1 has only a limited effect on customer assignment. For instance, in type-R instances, ρ rises slightly from 0.42 to 0.49, meaning most customers are still assigned to alternative service locations despite the slight penalty. When the deviation cost is set to $d_{l,r}$, the effect becomes much more pronounced: ρ increases from 0.45 to 0.65 in type-C instances, from 0.49 to 0.81 in type-R instances, and from 0.47 to 0.76 in type-RC instances. When the deviation cost is further raised to $(d_{l,r})^2$, ρ remains at a high level across all instance types, with most customers are assigned to their original service locations. However, the TC continues to rise, suggesting that an overly strong penalty may push the model toward minimizing deviation at the expense of higher travel cost. Overall, $d_{l,r}$ serves as an effective benchmark for modeling deviation cost, as it achieves a reasonable balance between controlling total cost and accommodating customer location preferences.

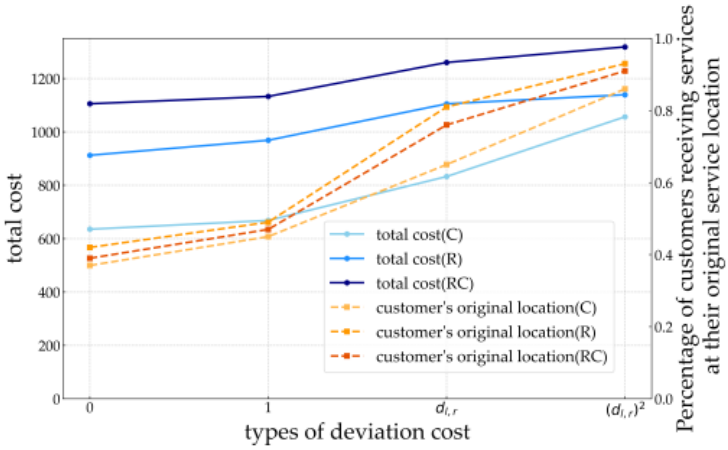

Figure 10. Impact of different deviation costs.

Table 10. Statistical comparison of the impact of different deviation costs.

Type	*TC*				*ρ*			
	0	**1**	$d_{l,r}$	$(d_{l,r})^2$	**0**	**1**	$d_{l,r}$	$(d_{l,r})^2$
C	635.32	668.15	832.60	956.47	0.37	0.45	0.65	0.89
R	912.37	968.55	1105.11	1139.82	0.42	0.49	0.81	0.93
RC	1105.38	1132.75	1260.11	1318.42	0.39	0.47	0.76	0.91

7. Conclusions

This paper studies the electric vehicle routing problem with flexible service locations, breaking away from the assumption of fixed customer service locations in traditional route optimization. A mixed-integer programming model is developed that jointly considers service location assignment, routing, and charging strategies. By introducing deviation costs and location capacity constraints, the model captures the trade-off between service flexibility and operating cost. To address the NP-hard nature of the problem and the presence of "irrational clustering" in the solution space, a MALNS-FO algorithm is proposed. Specialized operators, including location association destroy, location similarity destroy, and route reconstruction repair, are designed, and an FO mechanism is incorporated to effectively escape local optima. The computational results show that, on small-size instances, MALNS-FO obtains the same optimal solutions as Gurobi in the instances where Gurobi proves optimality. For instances where Gurobi does not find a feasible incumbent solution within the time limit, MALNS-FO is still able to generate feasible solutions. In large-size instances, MALNS-FO demonstrates significant advantages over algorithms such as ACO and VNS in terms of both total cost and computational efficiency, enabling the attainment of higher-quality solutions in less time.

The findings of this study provide the following managerial implications for logistics enterprises: (1) Introducing flexible service locations can provide logistics companies with more routing flexibility and help consolidate spatially dispersed demand, which may reduce the overall operating cost. (2) Managers should appropriately set deviation costs to avoid an excessive focus on service consolidation that may compromise service quality, thereby achieving a balance between operational cost and service level.

At the modeling level, this study does not yet consider some complex operational factors, such as stochastic customer preferences, stochastic customer demand, dynamic service location availability, and heterogeneous electric vehicle fleets. Future research may extend the proposed EVRPTW-FSL model by incorporating these factors into more realistic delivery settings. On the algorithmic side, combining reinforcement learning with

metaheuristic methods may be a promising direction to further improve solution efficiency and adaptability under uncertainty.

Author Contributions: Conceptualization, X.D. and R.X.; methodology, X.D. and R.X.; software, X.D. and X.C.; validation, X.C. and R.X.; formal analysis, X.D. and R.X.; investigation, X.D. and X.C.; resources, R.X.; data curation, X.D. and X.C.; writing—original draft preparation, X.D. and X.C.; writing—review and editing, X.D. and R.X.; supervision, R.X.; funding acquisition, R.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Fundamental Research Funds for the Central Universities, grant number B250207085; the Philosophy and Social Science Fund of Education Department of Jiangsu Province, grant number 2024SJYB0054; and the Jiangsu Provincial Social Science Applied Research Excellent Project, grant number 25SYB-076.

Data Availability Statement: The data that support the findings of this study are openly available at https://github.com/Duanduan2001/EVRPTW-FSL (accessed on 9 August 2026).

Acknowledgments: The authors would like to acknowledge the support and inspiration provided by the Business Data Laboratory at the School of Business, Hohai University. The authors are grateful to the anonymous reviewers for greatly improving the quality of this paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhao, P.; Li, Z.; He, Z.; Chen, Y.; Xiao, Z. Reducing the road freight emissions through integrated strategy in the port cities. *Nat. Commun.* **2025**, *16*, 2563. [CrossRef] [PubMed]
2. Basso, R.; Kulcsár, B.; Sanchez-Diaz, I.; Qu, X. Dynamic stochastic electric vehicle routing with safe reinforcement learning. *Transp. Res. Part E Logist. Transp. Rev.* **2022**, *157*, 102496.
3. Jia, Y.H.; Mei, Y.; Zhang, M. Confidence-based ant colony optimization for capacitated electric vehicle routing problem with comparison of different encoding schemes. *IEEE Trans. Evol. Comput.* **2022**, *26*, 1394–1408. [CrossRef]
4. Mara, S.T.W.; Sarker, R.; Essam, D.; Elsayed, S. Solving electric vehicle drone routing problem using memetic algorithm. *Swarm Evol. Comput.* **2023**, *79*, 101295. [CrossRef]
5. Souza, A.L.; Papini, M.; Penna, P.H.; Souza, M.J. A flexible variable neighbourhood search algorithm for different variants of the electric vehicle routing problem. *Comput. Oper. Res.* **2024**, *168*, 106713. [CrossRef]
6. Liu, X.; Pang, Q.; Yuen, K.F.; Wang, X. Transforming last-mile delivery into marketplaces of logistics services: An investigation on consumer participation motives, resources, and contextual differences. *Bus. Res.* **2025**, *200*, 115624.
7. Fajardo-Calderin, J.; Masegosa, A.D.; Cantero, X.; Moreno, A.; Elejoste, P. Intelligent vehicle routing problem solver for simulating urban last-mile logistics. *Eur. Transp. Res. Rev.* **2026**, *18*, 22. [CrossRef]
8. Lu, F.; Gao, Z.; Bi, H. Two-echelon vehicle routing problem with time windows and intermediate facilities for e-commerce logistics in crowdsourcing model. *Systems* **2026**, *14*, 382. [CrossRef]
9. Dantzig, G.B.; Ramser, J.H. The truck dispatching problem. *Manag. Sci.* **1959**, *6*, 80–91.
10. Toth, P.; Vigo, D. (Eds.) *The Vehicle Routing Problem*; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 2002; pp. 1–367.
11. Braekers, K.; Ramaekers, K.; Van Nieuwenhuyse, I. The vehicle routing problem: State of the art classification and review. *Comput. Ind. Eng.* **2016**, *99*, 300–313. [CrossRef]
12. Tan, S.Y.; Yeh, W.C. The vehicle routing problem: State-of-the-art classification and review. *Appl. Sci.* **2021**, *11*, 10295. [CrossRef]
13. Shen, Y.D.; Peng, L.W.; Li, J.P. An improved estimation of distribution algorithm for multi-compartment electric vehicle routing problem. *J. Syst. Eng. Electron.* **2021**, *32*, 365–379.
14. Tang, M.; Zhuang, W.; Li, B.; Liu, H.; Song, Z.; Yin, G. Energy-optimal routing for electric vehicles using deep reinforcement learning with transformer. *Appl. Energy* **2023**, *350*, 121711.
15. Çatay, B.; Sadati, İ. A note on battery swapping policies in the electric vehicle routing problem with time windows and battery swapping vehicles. *Comput. Oper. Res.* **2025**, *185*, 107277.
16. Preis, H.; Frank, S.; Nachtigall, K. Energy-Optimized Routing of Electric Vehicles in Urban Delivery Systems. In *OperationsResearch Proceedings 2012. Operations Research Proceedings*; Springer: Cham, Switzerland, 2014; pp. 583–588. Available online: https://link.springer.com/chapter/10.1007/978-3-319-00795-3_87 (accessed on 9 August 2026).

17. Schneider, M.; Stenger, A.; Goeke, D. The electric vehicle-routing problem with time windows and recharging stations. *Transp. Sci.* **2014**, *48*, 500–520. [CrossRef]
18. Keskin, M.; Çatay, B. Partial recharge strategies for the electric vehicle routing problem with time windows. *Transp. Res. Part C Emerg. Technol.* **2016**, *65*, 111–127. [CrossRef]
19. Han, B.; Yan, Y.; Chi, T.; Park, Y. The electric vehicle routing problem with travel time and energy consumption uncertainty. *Transp. Res. Part E Logist. Transp. Rev.* **2025**, *202*, 104211. [CrossRef]
20. Bányai, Á.; Kaczmar, I.; Bányai, T. Green Micromobility-Based Last-Mile Logistics from Small-Scale Urban Food Producers. *Systems* **2025**, *13*, 785. [CrossRef]
21. Guo, S.; Ye, X.; Pei, S.; Yan, X.; Wang, T.; Chen, J.; Cheng, R. Charging Decision Optimization Strategy for Shared Autonomous Electric Vehicles Considering Multi-Objective Conflicts: An Integrated Solution Process Combining Multi-Agent Simulation Model and Genetic Algorithm. *Systems* **2025**, *13*, 921. [CrossRef]
22. Davatgari, A.; Cokyasar, T.; Subramanyam, A.; Larson, J.; Mohammadian, A.K. Electric vehicle supply equipment location and capacity allocation for fixed-route networks. *Eur. J. Oper. Res.* **2024**, *317*, 953–966. [CrossRef]
23. Von Boventer, E. The relationship between transportation costs and location rent in transportation problems. *J. Reg. Sci.* **1961**, *3*, 27–40. [CrossRef]
24. Maranzana, F.E. On the location of supply points to minimize transport costs. *J. Oper. Res. Soc.* **1964**, *15*, 261–270. [CrossRef]
25. Webb, M.H.J. Cost functions in the location of depots for multiple-delivery journeys. *J. Oper. Res. Soc.* **1968**, *19*, 311–320. [CrossRef]
26. Cavagnini, R.; Santini, A.; Schneider, M. Recent developments in location-routing problems: Deterministic, single-echelon, single-objective, single-period problems. *Eur. J. Oper. Res.* **2026**, *332*, 711–729.
27. Janinhoff, L.; Klein, R.; Sailer, D.; Schoppa, J.M. Out-of-home delivery in last-mile logistics: A review. *Comput. Oper. Res.* **2024**, *168*, 106686. [CrossRef]
28. Dumez, D.; Lehuédé, F.; Péton, O. A large neighborhood search approach to the vehicle routing problem with delivery options. *Transp. Res. Part B Methodol.* **2021**, *144*, 103–132. [CrossRef]
29. Yang, K.; Zhang, Z.; Zhao, J.; Liang, Z. A neighborhood-based matheuristic for the vehicle routing problem with delivery options. *Transp. Res. Part E Logist. Transp. Rev.* **2025**, *203*, 104366. [CrossRef]
30. Hu, X.; Xiang, K.; Wang, K. The workforce scheduling and routing problem with sequence constraints and alternative locations. *Comput. Ind. Eng.* **2025**, *209*, 111487. [CrossRef]
31. Frey, C.M.; Jungwirth, A.; Frey, M.; Kolisch, R. The vehicle routing problem with time windows and flexible delivery locations. *Eur. J. Oper. Res.* **2023**, *308*, 1142–1159. [CrossRef]
32. Zhang, X.; Dai, X.; Lou, P.; Hu, J. A multi-strategy ALNS for the VRP with flexible time windows and delivery locations. *Appl. Sci.* **2025**, *15*, 4995. [CrossRef]
33. Zang, X.; Jiang, L.; Liang, C.; Fang, X. Coordinated home and locker deliveries: An exact approach for the urban delivery problem with conflicting time windows. *Transp. Res. Part E Logist. Transp. Rev.* **2023**, *177*, 103228. [CrossRef]
34. Ropke, S.; Pisinger, D. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transp. Sci.* **2006**, *40*, 455–472. [CrossRef]
35. Fonseca, G.H.; Figueiroa, G.B.; Toffolo, T.A. A fix-and-optimize heuristic for the unrelated parallel machine scheduling problem. *Comput. Oper. Res.* **2024**, *163*, 106504. [CrossRef]
36. Wang, C.; Qin, F.; Xiang, X.; Jiang, H.; Zhang, X. A Dual-Population-Based Co-Evolutionary Algorithm for Capacitated Electric Vehicle Routing Problems. *IEEE Trans. Transp. Electrif.* **2024**, *10*, 2663–2676. [CrossRef]