applied sciences

MDPI

Article

The Electric Vehicle Routing Problem with Time Windows, Partial Recharges, and Parcel Lockers

Vincent F. Yu [1,2], Pham Tuan Anh [1,*] and Yu-Wei Chen [1]

[1] Department of Industrial Management, National Taiwan University of Science and Technology, Taipei 106, Taiwan
[2] Center for Cyber-Physical System Innovation, National Taiwan University of Science and Technology, Taipei 106, Taiwan
* Correspondence: phamtuananhise@gmail.com

Abstract: This paper presents an extension of the Electric Vehicle Routing Problem with Time Windows and Partial Recharges (EVRPTW-PR), which incorporates the use of parcel lockers as a delivery method (i.e., self-pickup method). This variant, named the electric vehicle routing problem with time windows, partial recharges, and parcel lockers (EVRPTW-PR-PL), focuses on minimizing delivery costs by employing a homogeneous fleet of electric vehicles (EVs) and providing two delivery methods for serving customers: home delivery and self-pickup methods. We derive a mathematical formulation and propose an adaptive large neighborhood search (ALNS) algorithm to address EVRPTW-PR-PL. Moreover, in ALNS, the solution representation is constructed to handle the assignment of delivery methods. The performance of our proposed ALNS algorithm is evaluated by solving EVRPTW-PR benchmark instances. Finally, the results of EVRPTW-PR-PL obtained by using the GUROBI solver and our ALNS algorithm are provided, accompanied by managerial insights on the implementation of parcel lockers.

Keywords: electric vehicle; routing problem; partial recharging; parcel locker; adaptive large neighborhood search

Citation: Yu, V.F.; Anh, P.T.; Chen, Y.-W. The Electric Vehicle Routing Problem with Time Windows, Partial Recharges, and Parcel Lockers. *Appl. Sci.* **2023**, *13*, 9190. https://doi.org/10.3390/app13169190

Academic Editors: Muhammad Waseem, Shah Fahad and Arman Goudarzi

Received: 13 July 2023
Revised: 8 August 2023
Accepted: 9 August 2023
Published: 12 August 2023

1. Introduction

The COVID-19 pandemic has stimulated the growth of the e-commerce market, which is projected to exceed US$6.388 trillion in 2024 with an annual growth rate of approximately 13.5% [1]. In addition, the number of people who have used online shopping services has significantly increased, reaching 3 billion people in 2018, and this growth has further accelerated due to the challenges of pandemics [2]. Last-mile logistics are a key point for addressing these bottlenecks. However, last-mile activities are costly and environmentally polluting, accounting for 13–75% of overall logistics costs [3]. According to a report by Forum [4], the number of delivery vehicles used is expected to increase by 36%, but effective solutions can lead to a 30% reduction in waste emissions and a nearly 25% decrease in total logistics costs. Logistics companies aim to offer diverse services along with faster deliveries to enhance customer satisfaction and reduce operating costs. Environmental concerns, such as noise and air pollution, prompt them to invest in and implement alternative systems [5].

To effectively address environmental concerns, there is a growing trend to replace fuel-consuming vehicles with those powered by sustainable and renewable energy sources. Electric vehicles (EVs) are widely recognized as one of the cleanest transportation means for both private and commercial purposes that generate no local greenhouse gas (GHG) emissions and help reduce noise pollution. Therefore, logistics companies such as DHL, Amazon, and FedEx have already implemented fleets of EVs into their logistics networks. From the perspective of operation research, the use of EVs in last-mile delivery networks to serve customers is studied by ref. [6], which establishes a more sustainable delivery network. Variants of the electric vehicle routing problem (EVRP) have been developed and

extended over the past decade. Keskin and Çatay [7] investigate recharge policies to tackle the limitations of battery capacities with shorter recharging durations, named the electric vehicle routing problem with time windows and partial recharges (EVRPTW-PR).

The conventional last-mile delivery method (i.e., home delivery method) faces various challenges in serving customers within the limitations of time windows and dealing with the possibility of customers being unavailable. Moreover, the problem of urban traffic congestion further causes delays in delivery. These factors contribute to a significant risk of delivery failures, resulting in increased air pollution and logistics costs and decreased customer satisfaction levels. Consequently, logistics companies investigate alternative delivery methods to overcome these challenges. One of the potential alternatives is the self-pickup method, wherein couriers drop customers' parcels at assigned parcel lockers, allowing customers to conveniently retrieve their packages at any time. In practice, parcel locker systems used for delivery have been successfully installed and implemented in more than 20 countries, such as the U.S., UK, Germany, and Canada, which ensure reliable, responsive, and professional delivery experiences [8]. In addition, this delivery method proves particularly suitable during a pandemic as it eliminates the need for direct customer interactions [9].

From the perspective of operations research, we study the integration of parcel lockers into sustainable last-mile delivery networks so as to provide routing plans while minimizing logistics costs, called the electric vehicle routing problem with time windows, partial recharges, and parcel lockers (EVRPTW-PR-PL). While the concept of our study has also been explored by ref. [10], our proposed network differs from them in two features: (1) the locations of parcel lockers and recharging stations (CSs) are distinguished, and (2) each customer is associated only with a designated parcel locker if the self-pickup method is offered. To sum up, our main contributions are highlighted as follows.

- We study the implementation of parcel lockers based on the concepts of EVRPTW-PR, called the Electric Vehicle Routing Problem with Time Windows, Partial Recharges, and Parcel Lockers (EVRPTW-PR-PL).
- We formulate a mixed-integer programming (MIP) model and design an adaptive large neighborhood search (ALNS) for solving both EVRPTW-PR and EVRPTW-PR-PL.
- The performance of the proposed ALNS is shown by solving the EVRPTW-PR benchmark instances.
- We provide some managerial insights from the implementation of parcel lockers.

The remaining parts of this study are structured as follows. Section 2 reviews papers related to EVRPTW-PR and the implementation of parcel lockers. Section 3 formulates MILP to solve EVRPTW-PR-PL along with a formal description. Section 4 presents an ALNS algorithm that incorporates operators designed for addressing parcel lockers in the network. Section 5 conducts numerical experiments related to ALNS performance and the implementation of parcel lockers. Section 6 discusses the effects of parcel lockers on the problems. Finally, we give some conclusions and potential research in Section 7.

2. Related Work

The literature on variants of vehicle routing problems (VRPs) related to electric vehicles (EVs) and delivery methods is presented as follows. Section 2.1 begins with papers that focus on electric vehicle routing problems (EVRPs). Section 2.2 covers the literature on the utilization of parcel lockers in VRPs.

2.1. Electric Vehicle Routing Problems

The utilization of EVs is first investigated by ref. [6], where they implement a fleet of EVs in a classical VRP. The EVs are assumed to recharge to a minimum of 80% battery capacity with a constant charging rate. The authors devise a mixed-integer non-linear programming model to present the proposed problem. Schneider et al. [11] expand upon the initial concept of integrating EVs into VRPs by introducing electric vehicle routing with time windows (EVRPTW). This model assumes that the batteries of EVs are fully charged when they visit recharging stations (CSs), while also considering the time required for recharging. An efficient

mixed integer linear programming (MILP) model is provided that can track the battery status of EVs at each visited node. In addition, they generate well-known benchmark instances for EVRPTW, which are widely used in subsequent studies related to EVRPs.

To analyze the effects of different battery strategies in EVRPTW, Keskin and Çatay [7] investigate a relaxation of the full recharge policy by allowing partial recharges. This relaxation results in the formulation of a new problem known as the electric vehicle routing problem with time windows and partial recharges (EVRPTW-PR). The research highlights the benefits of the partial recharge policy. To effectively solve EVRPTW with both full and partial recharge policies, Desaulniers et al. [12] propose branch-and-price-and-cut algorithms, which attempt to obtain optimal solutions for these problems. Hiermann et al. [13] consider a mixed fleet consisting of three vehicle types: EVs, internal combustion vehicles (ICVs), and hybrid vehicles. The research shows that utilizing a mixed fleet can reduce total transportation costs. Furthermore, Cortés-Murcia et al. [14] investigate the use of alternative means such as walking, bikes, and drones for customer service to leverage the time for recharging EVs' batteries.

To enhance the practicality of routing plans for EVs, many studies consider realistic energy consumption models by providing more accurate estimations of battery charging and discharging rates. Keskin and Çatay [15] investigate three recharging specifications with varying recharging speeds and costs, and their results reveal tradeoffs between the speeds and costs, which highlight the significant benefits of rapid recharges. Rastani and Çatay [16] focus on the impacts of load-dependent energy consumption on the EVs' routing plans, where fleet sizes and the current loading of vehicles are load-dependent factors when minimizing total energy consumption. Montoya et al. [17] propose a more accurate charging time estimation (i.e., a piecewise linear approximation) to capture charging behavior accurately. Froger et al. [18] offer an arc-based formulation and an exact labeling algorithm to efficiently solve the [17] problem.

A combination of both electrical and fossil engine vehicles is utilized by refs. [19,20]. In addition, Mancini [21] and Hiermann et al. [13] investigate the use of hybrid vehicles that switch to consuming fossil fuels when their batteries are fully discharged. Other concepts of EVRPs are extended, such as time-dependent waiting times at public CSs [22,23], time synchronization at CSs with recharging policies [22,24,25], the implementation of battery swapping stations [26–28], and the impact of traffic on EVs' driving speed [29]. Other applications of electric vehicles (EVs) in goods distribution appear in the survey conducted by ref. [30].

2.2. Delivery Options in Last-Mile Delivery

Logistic companies face challenges in the sustainability and efficiency of their delivery networks due to factors such as traditional vehicles, traffic congestion, and urbanization [31]. A comprehensive review of recent and future last-mile delivery concepts is presented by ref. [9], who emphasize the significant benefits of alternative delivery options and address key decision problems using established operations research methods.

Regarding alternative delivery options, the concept of considering multiple time windows for each customer was first introduced by ref. [32], called VRP with multiple time windows (VRPMTW). Later, several variants of VRPMTW are studied, such as VRP with multiple interdependent time windows [33] and VRP with multiple prioritized time windows [34]. In addition, Belhaiza et al. [35] develop a hybrid variable neighborhood tabu search heuristic to effectively solve VRPMTW. Reyes et al. [36] further extend VRPMTW to VRP with roaming delivery locations (VRPRDL), where multiple delivery locations are provided with different non-overlapping time slots for customers. Ozbaygin et al. [37] offer an exact algorithm to solve both VRPRDL and the dynamic version of VRPRDL proposed by ref. [38]. In the dynamic version, routing plans are rescheduled in response to changes during the transportation execution. Another extension of VRPRDL is proposed by ref. [39], where parcel lockers are used as shared delivery locations with limited service capacities, replacing the trunks of cars. In addition, Dumez et al. [40] extend VRP with delivery

options (VRPDO), where customers' parcels can be delivered to various locations such as their private addresses, lockers, or car trunks.

Parcel lockers are especially used for proposing a delivery method (i.e., the self-pickup method where customers pick up the packages by themselves), which provides more flexibility and efficiency in serving customers. The use of parcel lockers in last-mile delivery can reduce up to 66% of total delivery costs compared with traditional methods [8]. Applications for parcel lockers for current and future logistics networks have been presented in a survey conducted by ref. [9]. Zhou et al. [41] investigate parcel lockers into a two-echelon VRP, where each customer can be served at either their home or a designated shared location (i.e., parcel locker), but time windows are not considered. Sitek and Wikarek [42] address alternative delivery methods to accommodate customer preferences where the capacity of shared locations (i.e., parcel lockers and post offices) is limited. Enthoven et al. [43] use parcel lockers as covering locations where nearby customers can pick up their parcels. This problem provides two delivery options: self-pickup and home delivery. Carotenuto et al. [44] compare the advantages of parcel lockers in reducing pollution by evaluating the results obtained from home delivery systems and parcel locker systems. Their findings contribute to understanding the positive environmental impact of parcel lockers as an alternative delivery method. Yu et al. [45] address the use of parcel lockers in VRPTW, which provides three delivery methods for serving customers. The concept of delivery methods in this paper is investigated in our problem. Other studies also investigate the applications of parcel lockers in last-mile delivery, such as automated parcel locker systems [46] and heterogeneous locker boxes [47].

3. Problem Description and Mathematical Model

With the presence of parcel lockers, there are two delivery methods offered, including (1) home delivery and (2) self-pickup at parcel lockers. With these delivery methods, we propose three types of customers for our problem. The first type is home delivery customers (H), who are required to directly receive parcels at their home locations. The second type is self-pickup customers (S), who come to the designated parcel lockers to pick up parcels. Each self-pickup customer is allowed to select a preferred parcel locker, which is convenient for collecting parcels. The last type is flexible customers (F), who are willing to be served by either home delivery or self-pickup methods. Note that the preferred delivery method for each customer is known in advance.

Let $C = \{C_H \cup C_S \cup C_F\}$ be a set of all customers (i.e., consisting of three subsets of home delivery, self-pickup, and flexible customers, respectively), P denotes a set of parcel lockers, F denotes the set of recharging stations (CSs), and $\{0, 0^-\}$ is the origin and destination depots. We define EVRPTW-PR-PL as a directed graph $G = (N, A)$, where the set of nodes N can be separated into subsets as $N = \{0, 0^{'}\} \cup C \cup P \cup F^{'}$. Note that set $F^{'}$ includes set F and its copies of each CS to allow multiple visits to CSs. For later formal expressions, we denote subsets $N^{'} = N \setminus \{0, 0^{'}\}$, $N^{(o)} = N \setminus \{0^{'}\}$, $N^{(d)} = N \setminus \{0\}$, and $F^{+} = F^{'} \cup \{0\}$. The set of arcs A is defined by $A = \{(i, j) | i \in N^{(s)}, j \in N^{(d)}, i \neq j\}$. For each arc $(i, j) \in A$, travel cost and travel time are defined by c_{ij} and t_{ij}, respectively.

All customers are served by a homogeneous fleet of EVs (denoted by set K) with a load capacity of Q, battery capacity of B, and electric consumption rate of r. Each utilized EV starts at the depot with a fully charged battery and ends at the depot. During transportation, each EV is allowed to recharge its battery at CSs with a charging rate of g. Note that each CS may serve EVs multiple times. Furthermore, each parcel locker with a capacity limited by the number of customers (denoted by $\mathcal{Q}$) may serve several self-pickup and flexible customers and can be visited by one or more vehicles. Let ξ denote the designated parcel locker of customer $i \in C_S \cup C_F$. Consequently, each parcel locker $i \in P$ has a subset of associated customers $\mathcal{C}_i \subseteq C_S \cup C_F$, which represents the possible connection between the parcel locker and customers in the subset. Each customer $i \in C$ has a predefined time window $[e_i, l_i]$, a non-negative demand d_i, and a service time s_i. Demand and service time

of the remaining vertices (i.e., parcel lockers and CSs) are set to 0, while their time windows are denoted by $[0, T_{max}]$, with T_{max} as the maximum duration of the routing plan.

Figure 1 presents an illustrative network including 4 home delivery customers (1–4), 4 self-pickup customers (5–8), 4 flexible customers (9–12), 4 parcel lockers (P1–P4), 3 recharging stations (CS1–CS3), and depot 0. Regarding the self-pickup method, the sets of possible connections for parcel lockers are $\mathscr{C}_1 = \{5, 8, 10\}$, $\mathscr{C}_2 = \{9, 11\}$, $\mathscr{C}_3 = \{6, 12\}$, and $\mathscr{C}_4 = \{7\}$ (see hash lines in Figure 1). The figure depicts a routing plan for serving all customers along with their corresponding delivery methods by using three EVs. All home delivery customers have their parcels delivered to their homes, while self-pickup customers receive their parcels at parcel lockers (see red hashed lines in Figure 1). Flexible customers 9 and 11 are not assigned to the self-pickup method (see black hashed lines in Figure 1), while customers 10 and 12 take their orders from the parcel lockers. In particular, route 1 visits parcel locker P4 to drop off parcels for customer 7. Similarly, route 2 delivers parcels from customers 6 and 12 to parcel locker P3. Finally, route 3 involves a visit to parcel locker P1, where customers 5, 8, and 10 retrieve their parcels. Some CSs are also visited by EVs to recharge their batteries.

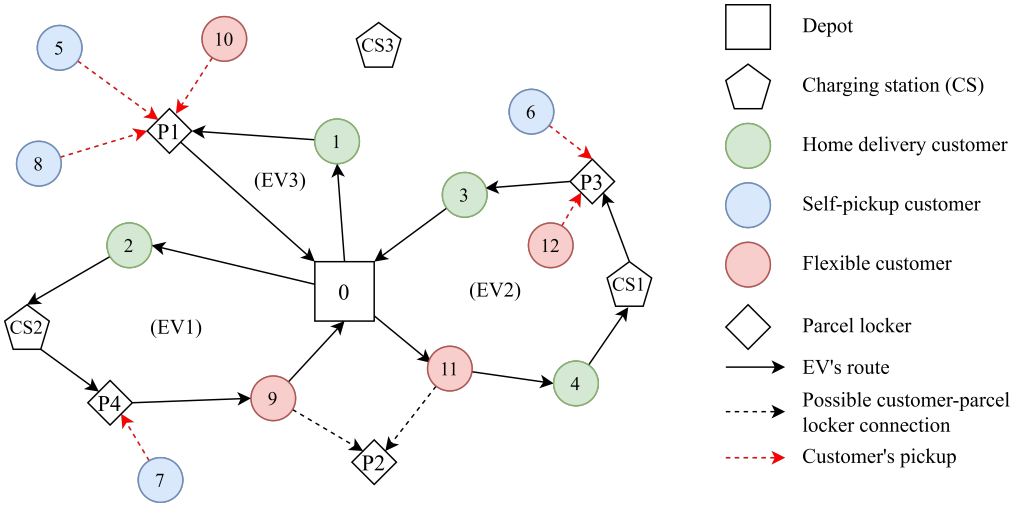


Figure 1. An illustrative example of EVRPTW-PR-PL.

For seeking the routing plans on EVRPTW-PR-PL, the model uses the following decision variables: binary variable x_{ij} is equal to 1 if arc $(i, j) \in A$ is traversed and 0 otherwise. Let o_i for all $i \in C_S \cup C_F$ be a binary variable equal to 1 if the self-pickup method is chosen for customer i and 0 otherwise (indicating the home delivery method). The non-negative variable w_i for all $i \in P$ then determines the total parcels delivered to the parcel locker i. Regarding the battery level, non-negative variables y_i and Y_i present the battery state when coming to and departing from vertex $i \in N$, respectively. Finally, non-negative variables τ_i and u_i track the service starting time and current load when the EV arrives at vertex $i \in N$.

Objective function:

$$\text{Minimize} \sum_{(i,j)\in A} c_{ij} x_{ij} \tag{1}$$

The objective function (1) minimizes the total travel distance of the EVs.

Constraints:

$$\sum_{(i,j)\in A} x_{ij} = 1, \quad \forall\, i \in C_H \tag{2}$$

$$\sum_{(i,j)\in A} x_{ij} = 1 - o_i, \quad \forall\, i \in C_S \cup C_F \tag{3}$$

$$\sum_{(i,j)\in A} x_{ij} \leq 1, \quad \forall\, i \in F' \tag{4}$$

$$Q \sum_{(i,j)\in A} x_{ij} \geq w_i, \quad \forall\, i \in P \tag{5}$$

$$\sum_{(j,i)\in A} x_{ji} - \sum_{(i,j)\in A} x_{ij} = 0, \quad \forall\, i \in N^{'} \tag{6}$$

$$\sum_{(0,i)\in A} x_{0i} \leq |K| \tag{7}$$

Constraints (2) and (3) enforce the associated delivery methods for all customers. Constraint (4) addresses arc connectivity related to CSs. Constraint (5) ensures that the parcel locker must be visited by vehicles if it keeps parcels for at least one customer. Constraint (6) conserves EV's flows at each node within routes. Constraint (7) limits the number of used electric vehicles.

$$\tau_i + (t_{ij} + s_i)x_{ij} - b_0(1 - x_{ij}) \leq \tau_j, \quad \forall\, i \in C \cup P, (i,j) \in A \tag{8}$$

$$\tau_i + t_{ij}x_{ij} + g(Y_i - y_i) - b_0(1 - x_{ij}) \leq \tau_j, \quad \forall\, i \in F^{'}, (i,j) \in A \tag{9}$$

$$e_i \leq \tau_i \leq l_i, \quad \forall\, i \in N \setminus (C_S \cup C_F) \tag{10}$$

$$e_i(1 - o_i) \leq \tau_i \leq l_i(1 - o_i), \quad \forall\, i \in C_S \cup C_F \tag{11}$$

$$u_j \leq u_i - d_i x_{ij} + Q(1 - x_{ij}), \quad \forall\, (i,j) \in A \tag{12}$$

$$u_j \leq u_i - w_i x_{ij} + Q(1 - x_{ij}), \quad \forall\, i \in P, (i,j) \in A \tag{13}$$

$$u_i \leq Q, \quad \forall\, i \in N \tag{14}$$

$$w_i = \sum_{j\in\mathscr{C}_i} d_j o_j, \quad \forall\, i \in P \tag{15}$$

$$\sum_{j\in\mathscr{C}_i} o_j \leq \mathcal{Q}, \quad \forall\, i \in P \tag{16}$$

$$o_i = 1, \quad \forall\, i \in C_S \tag{17}$$

Constraints (8) and (9) track the starting time for serving at each node. Constraints (10) and (11) impose that vehicles can only visit nodes within their predefined time windows. The current loading of EVs at each node is recorded by constraints (12) and (13). Constraint (14) guarantees that a vehicle's total load does not exceed its capacity. Constraint (15) states that the actual demand of each parcel locker equals the total demand from customers picking up their parcels at that specific parcel locker. Constraint (16) limits that the number of customers assigned to each parcel locker does not exceed the parcel locker's capacity. Constraint (17) imposes that each self-pickup customer must be assigned to the self-pickup method.

$$y_j \leq Y_i - rc_{ij}x_{ij} + B(1 - x_{ij}), \quad i \in F^{+}, \forall (i,j) \in A \tag{18}$$

$$y_j \leq y_i - rc_{ij}x_{ij} + B(1 - x_{ij}), \quad i \in C \cup P, \forall (i,j) \in A \tag{19}$$

$$y_i \leq Y_i \leq B, \quad \forall i \in F^{+} \tag{20}$$

Constraints (18) and (19) track the remaining battery capacity when the EVs visit CSs and the remaining nodes, respectively. Constraint (20) ensures that the battery state of charge does not exceed its capacity at CSs.

4. Solution Method

To solve EVRPTW-PR-PL, we propose an ALNS algorithm that has successfully solved a broad range of routing problems, such as pickup and delivery problems [48], the two-echelon vehicle routing problem (2E-VRP) [49], VRP with handling uncertainty [50], EVRP variants [7,51,52], and VRP with delivery options (VRPDO) [40]. Schiffer and Walther [52] derive a set of functions to efficiently check the feasibility of EV routes in $\mathcal{O}(1)$ time complexity. The efficient operators for handling delivery options described in one study [40] are also utilized in our problem.

A solution of EVRPTW-PR-PL is represented by two arrays. Array 1 represents a set of sequences that include served customers who may visit some CSs in order. Array 2 is associated with array 1, which reports the delivery methods for customers. In particular, if a customer within array 1 is assigned to the self-pickup method, then the designated parcel locker is placed in the corresponding position of the customer in array 1; otherwise, the remaining positions are copied from array 1. Based on this representation, the preferred delivery methods are always ensured for all customers, and the total demand for parcel lockers is also guaranteed. However, it is possible for infeasible solutions to arise that can violate time window, battery, and load violations, resulting in penalties (see Section 1). We note that total cost and penalties are determined based on the second array (see Section 4.4).

Figure 2 presents the solution of the illustrative example (as shown in Figure 1). With array 1, white cells represent home delivery customers and CSs, while blue and red cells represent self-pickup and flexible customers, respectively. Array 2 is then associated with array 1, where all blue cells must be replaced by the designated parcel lockers of self-pickup customers, and red cells can represent either the customers themselves or the designated parcel lockers. For example, route 2 is represented as $0 \rightarrow 11 \rightarrow \text{CS1} \rightarrow \text{P3} \rightarrow 3 \rightarrow 0$, where the flexible customer 11 is assigned to the home delivery method, while another flexible customer 12 is assigned to the self-pickup method at parcel locker 3 (as indicated in array 2). In addition, when two consecutive parcel lockers are the same on a route, we can combine them when updating the routes.

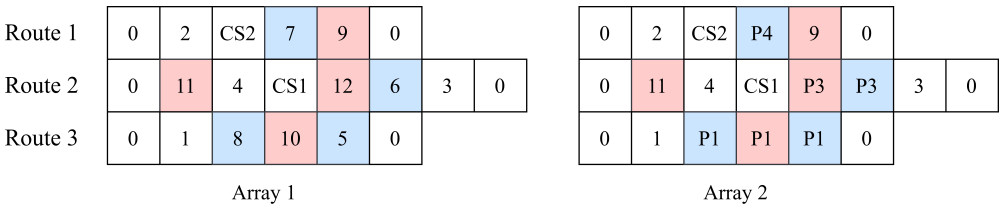


Figure 2. Solution representation for the illustrative example (see Figure 1).

4.1. Construction Algorithm

Based on the structure of a solution representation, an initial solution for ALNS is generated by using a modified nearest neighborhood algorithm that includes two stages. In the first stage, we iteratively select the customer in the set of unassigned customers with the lowest cost increment (i.e., $\Delta f_{gen} = f_{gen}^{after} - f_{gen}^{before}$) and then insert it into the solution. The first stage terminates when all customers are served. In the second stage, an attempt is then made to reduce the battery violation by implementing the station insertion operator (see Section 4.4.3).

To find the best insertion, all customers are tested to be inserted into all positions in the incomplete solution. The customer who results in the lowest cost increment is selected. It is important to note that the three types of customers are considered in different ways: (1) For home delivery customers: the customer locations are taken into account when evaluating cost increments; (2) For self-pickup customers: the designated parcel lockers are considered

when evaluating cost increments; and (3) For flexible customers: both the aforementioned cases (i.e., home delivery and self-pickup) are considered when evaluating cost increments, resulting in a lower cost increment.

4.2. Evaluation of Penalties

To extend the search space, we accept infeasible solutions in terms of time windows, battery consumption, and vehicle loading, along with adding penalty costs [53]. Let $\sigma = \{V_1, \ldots, V_n\}$ denote a solution consisting of a set of n routes. The generalized cost function of a solution σ is denoted by $f_{gen}(\sigma)$ as follows. Next, the total cost function (1) is generalized by taking into account three penalty terms related to time windows, battery, and vehicle load, as shown in the following expression.

$$f_{gen}(\sigma) = \sum_{V \in \sigma} f_{gen}(V) = f(\sigma) + \sum_{V \in \sigma} (\rho_{tw}\lambda_{tw}(V) + \rho_{ba}\lambda_{ba}(V) + \rho_{lo}\lambda_{lo}(V)) \quad (21)$$

Here, $f(\sigma)$ is equivalent to the total cost (1). Values $\lambda_{tw}(V)$, $\lambda_{ba}(V)$, and $\lambda_{vl}(V)$ denote the time window, battery, and load penalties, respectively. These penalties are included in the generalized total cost with associated weights ρ_{tw}, ρ_{ba}, and ρ_{lo}, which are initialized with ρ^0_{tw}, ρ^0_{ba}, and ρ^0_{lo}. These weights are adjusted during the ALNS procedure with the ranges $[\rho^{min}_{tw}, \rho^{max}_{tw}]$, $[\rho^{min}_{ba}, \rho^{max}_{ba}]$, and $[\rho^{min}_{lo}, \rho^{max}_{lo}]$. Based on [53], lower bounds are set as 0.5, and upper bounds are set as the objective value of the initial solution.

It is worth noting that dynamic penalty weights are used to reduce the occurrence of infeasible solutions during the execution of our ALNS procedure. More specifically, the penalty weights are increased to prevent infeasible solutions when the algorithm keeps failing to explore feasible solutions. This concept has been successfully implemented to solve EVRP variants such as [52–54].

Load penalty λ_{lo} is easily determined in $\mathcal{O}(1)$ time complexity [55]. Battery and time window penalties are more challenging, but Schiffer and Walther [52] propose an efficient approach by introducing a set of forward and backward variables. For the sake of calculations, all variables mentioned in this section are used in the same unit as travel time. We also introduce some supported parameters; i.e., $h_{ij} = grd_{ij}$ denotes the recharge time with enough energy to traverse an arc (i, j), and $H = gB$ denotes the time for recharging a full battery capacity. All variables are applied to concatenation operators for evaluating these penalties in $\mathcal{O}(1)$ time complexity. In more detail, forward and backward functions are explained as follows.

4.2.1. Forward Functions

We introduce a set of variables for propagating from node i to node j. In particular, variable a^{min}_j (resp. a^{max}_j) denotes the earliest allowed arrival time adding the minimum required (resp. the maximum possible) charging time at previous CSs. Variables $\tilde{a}^{min}_j$ and $\tilde{a}^{max}_j$ replace a^{min}_j and a^{max}_j, respectively, when penalties occur in order to avoid over-penalization. In addition, slack time due to the waiting time before node j is served results in the available time for charging, denoted by a^{sl}_{ij}. Variable a^{rt}_j tracks the inverse residual battery capacity at node j. Finally, variable a^{add}_{ij} determines the minimum charging time in order to reach from node i to node j. Forward equations derived from [52] are shown as follows. To shorten forward and backward functions, values t_{ij} also involve service time at node i.

$$a^{min}_j = \max\{e_j, \tilde{a}^{min}_i + t_{ij}\} + a^{add}_{ij} \quad (22)$$

$$a^{max}_j = \begin{cases} \max\{e_j, \tilde{a}^{min}_i + \max\{0, a^{rt}_i - s_i\} + t_{ij}\} & \text{if } i \in F \\ \max\{e_j, \tilde{a}^{max}_i + t_{ij}\} & \text{else} \end{cases} \quad (23)$$

$$\tilde{a}_j^{min} = \min\{a_j^{min}, a_j^{max}, l_j\} \tag{24}$$

$$\tilde{a}_j^{max} = \min\{l_j, \tilde{a}_j^{min} + \max\{a_j^{max} - a_j^{min}, 0\}\} \tag{25}$$

$$a_{ij}^{sl} = \max\{0, e_j - \tilde{a}_i^{min} - t_{ij}\} \tag{26}$$

$$a_j^{rt} = \begin{cases} \min\{H, \max\{0, a_i^{rt} - s_i - a_j^{sl}\} + h_{ij}\} & \text{if } i \in F \\ \min\{H, \max\{0, a_i^{rt} - \min\{a_j^{sl}, \tilde{a}_i^{max} - \tilde{a}_i^{min}\}\} + h_{ij}\} & \text{else} \end{cases} \tag{27}$$

$$a_{ij}^{add} = \begin{cases} \max\{0, \max\{0, a_i^{rt} - s_i - a_j^{sl}\} + h_{ij} - H\} & \text{if } i \in F \\ \max\{0, \max\{0, a_i^{rt} - \min\{a_j^{sl}, \tilde{a}_i^{max}, \tilde{a}_i^{min}\}\} + h_{ij} - H\} & \text{else} \end{cases} \tag{28}$$

At the starting depot 0, all values (i.e., a_0^{min}, a_0^{max}, a_0^{sl}, a_0^{rt}, and a_0^{add}) are initialized as 0. Based on the above-mentioned variables, forward battery $\overrightarrow{\lambda}_B$ and time window $\overrightarrow{\lambda}_T$ penalties for each route r are computed as follows.

$$\overrightarrow{\lambda}_B(V) = \sum_{j \in r} \max\{a_j^{min} - a_j^{max}, 0\} \tag{29}$$

$$\overrightarrow{\lambda}_T(V) = \sum_{j \in r} \max\{\min\{a_j^{min}, a_j^{max}\} - l_j, 0\} \tag{30}$$

4.2.2. Backward Functions

The backward variables consisting of b^x, $x \in \{min, max, sl, rt, add\}$, as well as $\tilde{b}^{min}$ and $\tilde{b}^{max}$ are similarly used with the same terminology to update backward penalties. By considering an arc (i, j), the backward variables are updated backwardly from node j to node i as the following equations.

$$b_i^{min} = \min\{l_i, \tilde{b}_j^{min} - t_{ij}\} + b_{ij}^{add} \tag{31}$$

$$b_i^{max} = \begin{cases} \max\{l_i, \tilde{b}_j^{min} - \max\{0, b_j^{rt} - s_j\} - t_{ij}\} & \text{if } j \in F \\ \max\{l_i, \tilde{b}_j^{max} - t_{ij}\} & \text{else} \end{cases} \tag{32}$$

$$\tilde{b}_i^{min} = \max\{b_i^{min}, b_i^{max}, e_i\} \tag{33}$$

$$\tilde{b}_i^{max} = \max\{e_i, \tilde{b}_i^{min} - \max\{b_i^{min} - b_i^{max}, 0\}\} \tag{34}$$

$$b_i^{sl} = \max\{0, \tilde{b}_j^{min} - t_{ij} - l_i\} \tag{35}$$

$$b_i^{rt} = \begin{cases} \min\{H, \max\{0, b_j^{rt} - s_j - b_i^{sl}\} + h_{ij}\} & \text{if } j \in F \\ \min\{H, \max\{0, b_j^{rt} - \min\{b_i^{sl}, \tilde{b}_j^{min} - \tilde{b}_j^{max}\}\} + h_{ij}\} & \text{else} \end{cases} \tag{36}$$

$$b_j^{add} = \begin{cases} \max\{0, \max\{0, b_j^{rt} - s_j - b_i^{sl}\} + h_{ij} - H\} & \text{if } j \in F \\ \max\{0, \max\{0, b_j^{rt} - \min\{b_i^{sl}, \tilde{b}_j^{min}, \tilde{b}_j^{max}\}\} + h_{ij} - H\} & \text{else} \end{cases} \tag{37}$$

$$\overleftarrow{\lambda}_B(V) = \sum_{j \in V} \max\{b_j^{max} - b_j^{min}, 0\} \tag{38}$$

$$\overleftarrow{\lambda}_T(V) = \sum_{j \in V} \max\{e_j - \max\{b_j^{min}, b_j^{max}\}, 0\} \tag{39}$$

4.2.3. Concatenation Operators

By utilizing forward and backward variables, concatenation operators allow us to obtain the penalties in $\mathcal{O}(1)$ time complexity for two cases: (1) when inserting a node into a route, and (2) when removing a node from a route. However, when route segments (i.e., at least two vertices) are inserted, either forward or backward penalties have to be updated for the route segment first before implementing the two above-mentioned cases. (1) Insert a node v into a route V, expressed as $V = \{0, \ldots, x, v, y, \ldots, 0\}$. There are two partial routes partitioned by node v; i.e., $V^1 = \{0, \ldots, x\}$ and $V^2 = \{y, \ldots, 0\}$. Forward and backward variables at node v are calculated by Equations (22)–(28) and Equations (31)–(37), respectively. Time window violation ($\lambda_T(V)$) and battery violation ($\lambda_B(V)$) are then obtained as follows.

$$\begin{aligned}\lambda_T(V) = \overrightarrow{\lambda}_T(V^1) + \overleftarrow{\lambda}_T(V^2) + \max\{0, a_v^{min} - l_v - \max\{0, a_v^{min} - a_v^{max}\}\} + \\ \max\{0, \min\{l_v, \max\{e_v, a_v^{min}\}\} - b_v^{min} - \max\{0, b_v^{max} - b_v^{min}\}\}\end{aligned} \tag{40}$$

$$\lambda_B(V) = \overrightarrow{\lambda}_B(V^1) + \overleftarrow{\lambda}_B(V^2) + \max\{0, a_v^{min} - a_v^{max}\} + \max\{0, b_v^{max} - b_v^{min}\} + \Delta^{(i)} \tag{41}$$

$$\text{where } \Delta^{(i)} = \begin{cases} \max\{0, a_v^{rt} + b_v^{rt} - H - \min\{a_v^{rt}, \max\{0, \max\{b_v^{max}, b_v^{min}\} - \min\{a_v^{min}, a_v^{max}\}\}\}\} & \text{if } v \in F \\ \max\{0, a_v^{rt} + b_v^{rt} - H - \min\{H, \max\{0, b_v^{min} - b_v^{max}\} + \max\{0, a_v^{max} - a_v^{min}\}, \\ \max\{0, \min\{l_v, \max\{b_v^{min}, b_v^{max}\}\} - \min\{a_v^{min}, a_v^{max}\}\}\}\} & \text{else.} \end{cases}$$

(2) Removing a node v from a route V, we obtain two partial routes $V^1 = \{0, \ldots, x\}$ and $V^2 = \{y, \ldots, 0\}$. First, forward variables at node y are extended by Equations (22)–(28). Second, Equations (42) and (43) are derived to calculate the time windows and battery violations, respectively.

$$\begin{aligned}\lambda_T(V) = \overrightarrow{\lambda}_T(V^1) + \overleftarrow{\lambda}_T(V^2) + \max\{0, a_y^{min} - l_y - \max\{0, a_y^{min} - a_y^{max}\}\} \\ + \max\{0, \max\{e_y, \min\{a_y^{min}, a_y^{max}, l_y\}\} - \tilde{b}_y^{min}\}\end{aligned} \tag{42}$$

$$\lambda_B(V) = \overrightarrow{\lambda}_B(V^1) + \overleftarrow{\lambda}_B(V^2) + \max\{0, a_y^{min} - a_y^{max}\} + \Delta^{(r)} \tag{43}$$

$$\text{where } \Delta^{(r)} = \begin{cases} \max\{0, a_y^{rt} + b_y^{rt} - H - \min\{a_y^{rt}, \max\{0, b_y^{min} - \min\{a_y^{min}, a_y^{max}\}\}\}\} & \text{if } y \in F \\ \max\{0, a_y^{rt} + b_y^{rt} - H - \min\{H, \max\{0, b_y^{min} - b_y^{max}\} + \max\{0, a_y^{max} - a_y^{min}\}, \\ \max\{0, \min\{l_y, \max\{b_y^{min}, b_y^{max}\}\} - \min\{a_y^{min}, a_y^{max}\}\}\} & \text{else.} \end{cases}$$

These operators never underestimate the total time window and battery violations, although the equations do not provide exact penalty values for each component.

4.3. Adaptive Large Neighborhood Search

Algorithm 1 presents our proposed ALNS, adopted by ref. [52]. ALNS is an individual-based algorithm, which works on a current solution σ initialized by the construction algorithm (see Section 4.1). We set the current best solution σ^* as σ. While σ^* accepts an infeasible solution, σ_f^* records the current best feasible solutions. During the ALNS procedure, we utilize a temporary solution σ' to execute removal and insertion operators. The algorithm terminates when either of the two following conditions is met: (1) the algorithm reaches η^{max} iterations or (2) no improved solution is found after η^{noimp} iterations.

Algorithm 1 ALNS pseudocode

$\sigma \leftarrow initializeSol()$
$\sigma^{'}, \sigma^{*} \leftarrow \sigma$, $\sigma_f^{*} \leftarrow \emptyset$, $\iota \leftarrow 0$, $\iota^{noi} \leftarrow 0$, $T \leftarrow T_0$
if isFeasible(σ) **then**
    $\sigma_f^{*} \leftarrow \sigma$
while $(\iota < \eta^{max})$ **and** $(\iota - \iota^{noi} < \eta^{noimp})$ **do**
    $\sigma^{'} \leftarrow customerRemovalInsertion()$ ▷ mentioned in Section 4.4.1
    if random(0,1) < δ_L **then**
        $\sigma^{'} \leftarrow lockerRemovalInsertion()$ ▷ mentioned in Section 4.4.2
    $\sigma^{'} \leftarrow stationremovalinsertion()$ ▷ mentioned in Section 4.4.3
    if $f_{gen}(\sigma^{'}) < f_{gen}(\sigma)$ **then**
        $\sigma \leftarrow \sigma^{'}$
        if $f_{gen}(\sigma^{'}) < f_{gen}(\sigma^{*})$ **then**
            $\sigma^{*} \leftarrow \sigma^{'}$
            if isFeasible($\sigma^{'}$) **and** $f_{gen}(\sigma^{'}) < f_{gen}(\sigma_f^{*})$ **then**
                $\sigma_f^{*} \leftarrow \sigma^{'}$
                $\iota^{noi} \leftarrow \iota$
    else
        $\iota^{noi} \leftarrow \iota^{noi} + 1$
        $acceptanceWorse(\sigma, \sigma^{'}, T)$ ▷ mentioned in Section 4.5
    if **modulo**$(\iota, \eta^{penalty}) = 0$ **then**
        $updatePenaltyWeights()$
    if **modulo**$(\iota, \eta^{proba}) = 0$ **then**
        $updateSelectionProb()$
    $\iota \leftarrow \iota + 1$, $T \leftarrow \alpha T$
return σ_f^{*}

Each iteration starts by implementing *customerRemovalInsertion*() into the temporary solution and then applying *lockerRemovalInsertion*() with a probability of δ_L. After that, *stationremovalinsertion*() is executed to complete the removal and insertion phases. The obtained temporary solution $\sigma^{'}$ is then compared with the updated solutions in ALNS: (1) $\sigma^{'}$ replaces σ if $\sigma^{'}$ is better than σ, (2) $\sigma^{'}$ also replaces σ^{*} if σ is better than σ^{*}, and (3) if a better feasible solution is obtained, then σ_f^{*} is updated with σ. These conditions govern the updating process at each iteration. However, we still accept a worse solution $\sigma^{'}$ to be σ with a probability based on the temperature T, denoted by $acceptanceWorse(\sigma, \sigma^{'}, T)$ (see Section 4.5).

Dynamic parameters are then updated based on the performance of operators. The penalty weights are updated (*updatePenaltyWeights*()) based on the following. If no violation occurs within $\eta^{penalty}$ iterations, then the associated penalty weight is divided by a constant $\omega^{penalty}$; otherwise, the penalty weight is multiplied by $\omega^{penalty}$. Similarly, the probability of removal and insertion operators is updated (*updateSelectionProb*()) after η^{proba} consecutive iterations (see Section 4.4). Before proceeding to the next iteration, the temperature of the acceptance criteria is updated with a cooling rate of α. We note that the cooling rate α and the initial SA temperature T_0 are from [53].

4.4. Removal and Insertion Algorithms

This section introduces three classes of removal and insertion algorithms to address the features of our problem, i.e., the solution representation takes into account not only customer nodes, but also parcel lockers and recharging stations (CSs). Removal and insertion algorithms associated with customers are first presented in Section 4.4.1. Section 4.4.2 then introduces removal and insertion mechanisms designed for parcel lockers. Finally,

Section 4.4.3 proposes station removal and insertion algorithms, which exclusively consider recharging stations.

4.4.1. Customer Removal and Insertion

Customers are removed from the solution by three removal operators, while they are re-inserted into the solution by three insertion operators based on array 1 of the solution representation (denoted by *customerRemovalInsertion*()). The customer removal and insertion algorithms are selected in an adaptive manner [53]. The probabilities of selections are re-computed by a smoothing rate ω^{prob} after a learning period (i.e., η^{proba} iterations). During a learning period, when a new solution is generated, the associated removal and insertion algorithms are rewarded based on their performance. The rewards, denoted as $\epsilon_f, \epsilon_b, \epsilon_i$, and ϵ_w, correspond to different scenarios, which are gained when a new best feasible solution, a new best solution (which may be infeasible), a solution improving the current solution, and a worse solution, respectively, are found.

For each customer removal operator, η^{rem} customers are removed. The number of removed customers η^{rem} is randomly selected within a range $\eta^{rem} \in [1, |C|]$. The following removal operators are implemented in our ALNS:

- Random removal randomly removes η^{rem} customers from a given solution.
- Worst removal is adopted by ref. [53], which iteratively removes the customer that increases the generalized total cost the most, defined by $\Delta f_{gen} = f^a_{gen} - f^b_{gen}$, where f^b_{gen} and f^a_{gen} represent the generalized costs before and after removing the customer, respectively. Moreover, to avoid being trapped at local optimum points, we consider randomness by controlling a randomness level δ^{noise}. Instead of choosing the highest cost reduction, the $\lfloor r^{\delta^{noise}} |L^{worst}| \rfloor$-th highest cost reduction is then selected for removal from the solution, where r is a random variable drawn from a uniform distribution $U[0, 1]$, and L^{worst} is a list of customers sorted in descending order by the cost reduction.
- Shaw removal was first introduced in [56] and aims to remove customers that are similar to each other with respect to four criteria (i.e., travel distance, time window, demand, and delivery method). The procedure begins by randomly removing a customer and storing it in a list (denoted by L^{shaw}). The removal process is then repeated until η^{rem} customers are removed. At every iteration, a customer j in the route is selected for removal based on its relatedness to a randomly chosen customer i from the list L^{shaw}. The relatedness between customers i and j is evaluated using the following equation:

$$\mathscr{R}(i, j) = \phi_1 d_{ij} + \phi_2 |q_i - q_j| + \phi_3 |e_i - e_j| + \mathscr{P}(i, j)$$

  Here, the four terms correspond to the relatedness of travel distance, the earliest time, demand, and delivery method, respectively. For the first three terms, each criterion is weighted by ϕ_1, ϕ_2, and ϕ_3, respectively. In addition, $\mathscr{P}(i, j)$ equals 0 if both customers are assigned the same delivery method (either home delivery or self-pickup method) and 1 otherwise. We note that a lower $\mathscr{R}(i, j)$ represents more relatedness between customers i and j.

For each insertion operator, the set of absent customers due to the removals is re-inserted into the solution. Infeasible solutions in terms of time windows, load, and battery are taken into account by the generalized objective function (see Section 4.2). The insertion operators are briefly described as follows.

- Greedy insertion is adopted by ref. [48] and iteratively selects a customer from the list of absent customers with the best insertion for inserting into the solution until all customers are re-inserted. In each iteration, the customer with the best insertion is determined based on the increase in the generalized objective value. Note that for the

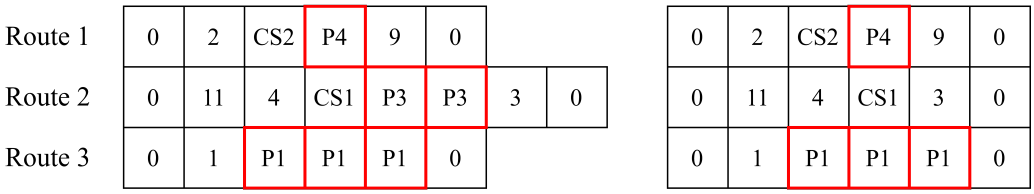
home delivery method, customer locations are considered for insertion, while for the self-pickup method, parcel locker locations are taken into account.

- k-regret insertion was first described by ref. [48] and aims to anticipate the future impact of insertion operations. The process involves iteratively inserting the set of absent customers based on their regret-k cost value. The regret-k value of a customer is determined by calculating the difference between the sum of costs for inserting the customer in the n-th lowest-cost position (denoted by $c^r_{i,n}$, given $n = 2 \dots k$) and the cost of inserting the customer in the lowest-cost position (denoted by $c^r_{i,1}$). Our paper implements this operator with $k = 2$ and $k = 3$.

4.4.2. Locker Removal and Insertion

We adopt a procedure developed in the literature [40] for handling the self-pickup method. This procedure involves locker removal and re-insertion into the solution based on array 2 of the solution representation, denoted by *lockerRemovalInsertion*(). Note that the locker removal and insertion heuristics must be implemented together to effectively handle the self-pickup method. In particular, a parcel locker is first removed from the solution, which consists of one or more associated customers. We then obtain a list of removed customers. Subsequently, all customers with the designated parcel locker are re-inserted by using a locker insertion procedure. Locker removal and insertion mechanisms are presented as follows:

- Locker removal as adopted by ref. [40] is employed to prevent unnecessary repetition of customer removal and insertion operators without yielding any improvement. The procedure begins by recording all utilized parcel lockers in the considered solution that can potentially be removed. Next, a parcel locker is selected based on the highest reduction in distance cost when it is removed from the solution. Note that the parcel locker is removed, resulting in the removal of one or more associated customers. Figure 3 provides an example illustrating the implementation of locker removal, as described in Figure 2. In particular, there are three parcel locker positions (i.e., P4 in route 1, P3 in route 2, and P1 in route 3), as indicated by the red boxes in Figure 3a. Figure 3b shows the incomplete solution after removing parcel locker 3 in route 2.
- Locker insertion is similar to customer greedy insertion, but with the key distinction that it exclusively considers the self-pickup method for inserting customers into the solution. Notably, the set of absent customers is designated to a single parcel locker.



(a) Before removing parcel locker 3 (b) After removing parcel locker 3

Figure 3. An example of implementing the locker removal heuristic.

4.4.3. Station Removal and Insertion

The recharging stations (CSs) are an important component of the problem. Thus, station removal and insertion heuristics are proposed to improve the solution in terms of battery violations, which are discussed as follows.

- Station removal is similar to the worst customer removal. In this case, CSs are targeted for removal instead of customer nodes. This heuristic removes a specified number of stations, denoted as η^{rem}, which is determined by $\eta^{rem} = [1, |CS|]$, where $|CS|$ represents the total number of CSs existing in the solution.
- Station insertion is similar to the greedy customer insertion for inserting CSs into the solution until no further improvement is found.

4.5. Acceptance Criteria

In our ALNS, we employ the simulated annealing (SA) acceptance criteria to escape local optima [53]. In particular, not only are improved solutions accepted, but worse solutions are also accepted with a probability depending on the relative difference between $f_{gen}(\sigma)$, and $f_{gen}(\sigma^{'})$, as well as the SA temperature.

Since infeasible solutions are allowed with dynamic penalty weights $\rho_{tw}, \rho_{ba}, \rho_{lo}$, the generalized cost strongly depends on the values of penalty weights. Let $(\rho^{\sigma}_{tw}, \rho^{\sigma}_{ba}, \rho^{\sigma}_{lo})$ represent the penalty weights used for evaluating $f_{gen}(\sigma)$ and $(\rho^{\sigma^{'}}_{tw}, \rho^{\sigma^{'}}_{ba}, \rho^{\sigma^{'}}_{lo})$ represent the penalty weights used for evaluating $f_{gen}(\sigma^{'})$. To compare the generalized cost between σ and $\sigma^{'}$, we normalize the penalty weights as follows.

$$(\rho^{norm}_{tw}, \rho^{norm}_{ba}, \rho^{norm}_{lo}) = \left(\frac{1}{2}\left(\rho^{\sigma}_{tw} + \rho^{\sigma^{'}}_{tw}\right), \frac{1}{2}\left(\rho^{\sigma}_{ba} + \rho^{\sigma^{'}}_{ba}\right), \frac{1}{2}\left(\rho^{\sigma}_{lo} + \rho^{\sigma^{'}}_{lo}\right)\right)$$

The probability of accepting a worse solution $\sigma^{'}$ is finally determined as follows.

$$p(\sigma, \sigma^{'}, T) = e^{\frac{-\Delta f_{rel}(\sigma^{'}, \sigma)}{T}} \quad \text{where } \Delta f_{rel}(\sigma^{'}, \sigma) = \frac{f^{norm}_{gen}(\sigma^{'}) - f^{norm}_{gen}(\sigma)}{f^{norm}_{gen}(\sigma)}$$

Here, $\Delta f(\sigma^{'}, \sigma)$ denotes the relative difference between the cost values of σ and $\sigma^{'}$.

The initial SA temperature and cooling rate are determined by following [53]. Specifically, the initial SA temperature is configured so that a solution worse than the initial solution by 30% is accepted with a probability of 50%. In addition, the cooling rate is defined to limit the decrease in the temperature to below 0.0001 in the final 20% of iterations.

5. Experimental Results

This section presents the results of numerical experiments to highlight our contributions. Section 5.1 explains the generation of the EVRPTW-PR-PL instances used in our experiments. Section 5.2 discusses the calibration of parameters related to ALNS to improve the solution quality. Section 5.3 discusses the performance of the proposed ALNS versus the state-of-the-art algorithms. Section 5.4 then describes the implementation of the proposed ALNS to solve EVRPTW-PR-PL instances. All experiments are conducted on a computer with an Intel® Core™ i7-9700 CPU at 3.0 GHz. While the GUROBI solver is used to solve the MILP model, the proposed ALNS algorithm is coded in C++ using Microsoft Visual Studio 2022, Version 17.2.3.

5.1. Test Instances

This section generates EVRPTW-PR-PL instances based on EVRPTW instances proposed by ref. [11]. There are 36 small-size instances (i.e., 12 instances with 5, 10, and 15 customers) and 56 large-size instances with 100 customers. These instances are modified to address three types of customers and parcel locker facilities in our problem. The instances are categorized into three classes based on the distribution of customer locations, i.e., clustered (C), randomly distributed (R), and combined clustered and randomly distributed (RC). Furthermore, instances in each class can be divided into two subsets with different widths of time windows, resulting in six subsets, namely C-1, C-2, R-1, R-2, RC-1, and RC-2. The subsets R-1, C-1, and RC-1 have shorter time windows compared with subsets R-2, C-2, and RC-2. In particular, based on [45], we randomly assign each customer to be one of three types of customers (i.e., home delivery, self-pickup, and flexible customers) with an equal probability. There are $\left\lceil \frac{|C|}{20} \right\rceil$ parcel locker locations generated for each instance. Each parcel locker is randomly located within the range between the smallest and largest values of the x- and y-coordinates of customers. Parcel locker's service time is set to half of the customer's service time, and their time window is set the same as the depot. Regarding the self-up method, for self-pickup and flexible customers, the parcel lockers nearest them are assigned to be the designated parcel lockers.

5.2. Parameter Tuning

To tune the parameters of our proposed ALNS, we utilize the one factor at a time (OFAT) approach, successfully employed in tuning parameters for solving VRP variants [50,52]. For this experiment, we randomly selected six large EVRPTW-PR-PL instances (i.e., c103, c202, r110, r205, rc106, and rc207) to evaluate parameter settings.

Table 1 summarizes all the parameters used in the proposed ALNS. In this experiment, eight parameters (i.e., $\omega^{penalty}$, ω^{proba}, η^{noimp}, η^{proba}, $\eta^{penalty}$, ϵ_b, ϵ_f, ϵ_i, ϵ_w, and δ_L) are selected for tuning, while the remaining parameters are referenced from previous works. In particular, parameters related to removal and insertion operators (i.e., ϕ_1, ϕ_2, ϕ_3, and δ^{noise}) are set based on [53]. Parameters associated with the SA acceptance criteria (i.e., T_0 and α) are discussed in Section 4.5. The initial weight penalties and their ranges are adopted from [53] (see Section 4.2). To ensure a good trade-off between computational time and solution quality, the ALNS algorithm is terminated after reaching $\eta^{max} = 30,000$ iterations.

Table 1. ALNS parameters and their domains.

Parameter	Description	Value
η^{max}	The maximum number of iterations implemented in ALNS	30,000 *
η^{noimp}	The maximum number of consecutive iterations without improvement	[1500, 3000, 4500, 6000]
η^{proba}	The number of iterations of a learning period in *updateSelectionProb*()	[50, 75, 100, 125]
$\eta^{penalty}$	The number of iterations for *updatePenaltyWeights*()	[5, 10, 15, 20]
ω^{proba}	Smoothing factor in *updateSelectionProb*()	[0.7, 0.75, 0.8, 0.85]
$\omega^{penalty}$	Reaction factor *updatePenaltyWeights*()	[1.1, 1.3, 1.5, 1.7]
ϵ_f	A reward when obtaining a new best feasible solution	[11, 13, 15, 17]
ϵ_b	A reward when obtaining a new best (infeasible) solution	[5, 7, 9, 11]
ϵ_i	A reward when obtaining a better current solution	[2, 4, 6, 8]
ϵ_w	A reward when obtaining a worse solution	[1, 3, 5, 7]
δ_L	The probability for applying *lockerRemovalInsertion*()	[0.1, 0.3, 0.5, 0.7]
ϕ_1	The weight of travel distance criteria in Shaw removal	6 **
ϕ_2	The weight of earliest time criteria in Shaw removal	4 **
ϕ_3	The weight of demand criteria in Shaw removal	5 **
δ^{noise}	The randomness level used for removal operators	6 **
$\rho^{min}_{tw}, \rho^{max}_{tw}, \rho^{0}_{tw}$	The weights associated with the time window penalty	see Section 4.2
$\rho^{min}_{ba}, \rho^{max}_{ba}, \rho^{0}_{ba}$	The weights associated with the battery penalty	see Section 4.2
$\rho^{min}_{lo}, \rho^{max}_{lo}, \rho^{0}_{lo}$	The weights associated with the load penalty	see Section 4.2
T_0	The initial SA temperature of the acceptance criteria	see Section 4.5
α	The cooling rate of SA temperature over the iteration	see Section 4.5

Notes: * The maximum number of iterations is fixed at 30,000 to ensure reasonable computation time for comparison with the state-of-the-art algorithms. ** These values are taken from [53].

We next vary the values of the eight selected parameters, which are produced by an ad hoc trial-and-error procedure [48] (see Column "Value" of Table 1), and their initial values are randomly selected (see Table 2). All parameters are tuned sequentially, with each parameter being tuned in turn. The tuning order of parameters follows the order reported in Table 2.

For each tuned parameter, we consider a set of four candidate values to test each of them, while the remaining parameters are fixed to their initial values unless their final values have been found. This therefore provides four different parameter settings for the ALNS algorithm. The final value of the tuned parameter is selected based on the best performance observed among the four parameter settings. For each parameter setting, the performance of the ALNS algorithm is determined based on the average gap of average solution values after 5 runs, denoted by $\Delta\bar{f}_{avg}[\%]$ (see Table 2). This process is repeated until all parameters are tuned. Table 2 summarizes the detailed results of the tuning procedure.

Table 2. Parameter setting for the proposed ALNS.

Parameter	Value 1	Value 2	Value 3	Value 4	Parameter	Value 1	Value 2	Value 3	Value 4
$\omega^{penalty}$	1.10	1.30 $^{(i)}$	**1.50**	1.70	ϵ_f	11	13	**15** $^{(i)}$	17
$\Delta\bar{f}_{avg}[\%]$	1.39	0.02	0.00	1.35	$\Delta\bar{f}_{avg}[\%]$	2.27	1.29	0.00	1.64
ω^{proba}	**0.70**	0.75	0.80	0.85 $^{(i)}$	ϵ_b	5	7	**9** $^{(i)}$	11
$\Delta\bar{f}_{avg}[\%]$	0.00	0.55	1.13	0.35	$\Delta\bar{f}_{avg}[\%]$	0.28	0.72	0.00	2.69
η^{noimp}	1500	3000	**4500**	6000 $^{(i)}$	ϵ_i	2	**4** $^{(i)}$	6	8
$\Delta\bar{f}_{avg}[\%]$	2.72	0.51	0.00	0.09	$\Delta\bar{f}_{avg}[\%]$	3.05	0.00	2.52	0.64
η^{proba}	50	75 $^{(i)}$	**100**	125	ϵ_w	**1** $^{(i)}$	3	5	7
$\Delta\bar{f}_{avg}[\%]$	0.21	1.54	0.00	0.15	$\Delta\bar{f}_{avg}[\%]$	0.00	1.24	0.29	3.27
$\eta^{penalty}$	5	**10** $^{(i)}$	15	20	δ_L	0.1	**0.3** $^{(i)}$	0.5	0.7
$\Delta\bar{f}_{avg}[\%]$	1.63	0.00	4.61	2.77	$\Delta\bar{f}_{avg}[\%]$	1.83	0.00	0.40	1.42

Notes: The best values are in bold. Superscipt $^{(i)}$ denotes the initial values.

5.3. Results on EVRPTW-PR

This section reports the results obtained by our proposed algorithm (ALNS) on both EVRPTW-PR small and large instances. These results are then compared with the results achieved by the state-of-the-art algorithms, including those from Keskin and Çatay [7] (KÇ), Schiffer and Walther [52] (SW), and Hiermann et al. [13] (HGA). Note that optimal solutions for EVRPTW-PR small instances are provided by ref. [7].

For the algorithm's performance analysis, several terms are used: (1) f_b represents the best objective value over 10 replications obtained by each algorithm; (2) $\Delta f[\%]$ denotes the gap measured by the results of our ALNS against those of the state-of-the-art algorithms; and (3) $\bar{t}$ denotes the average computational time spent by our ALNS. The details appear in Tables A1–A7.

Regarding the small-sized instances, we compare the results obtained by our ALNS algorithm with the optimal solutions provided in [7]. The details are presented in Table A1. Our ALNS not only achieves optimal solutions for all instances but also operates faster than the GUROBI solver.

Regarding the large-sized instances, the detailed results of subsets C-1, C-2, R-1, R-2, RC-1, and RC-2 are provided in Tables A2–A7, respectively. The tables list the best solution values after 10 runs from all algorithms (i.e., KÇ, SW, HGA, and our ALNS after) as well as the best-known solutions. Based on these tables, we compare the results of our algorithm against those of the state-of-the-art algorithms by providing the overall average objective value ($\bar{f}_b$) and the overall average gap ($\Delta\bar{f}[\%]$) for each subset, presented in Table 3 and Figure 4.

Based on Table 3 and Figure 4, our ALNS outperforms the HGA algorithm by improving the solutions by 0.33% on average. It is worth noting that two new best-known solutions (for c107 and r109 instances) are offered. Our ALNS yields solutions that are worse than those achieved by KÇ, SW, and the BKSs, but the gaps are not significant (ranging from 0.22% to 1.28%). Moreover, our algorithm gives all the best-known solutions for subset C-2, and its solutions for subsets C-1, C-2, and R-1 are better than those obtained by the HGA algorithm by 0.35%, 0.77%, and 1.14%, respectively, on average.

In conclusion, our ALNS algorithm demonstrates effective performance in solving EVRPTW-PR instances within a reasonable computational time. These results highlight its potential applicability in addressing the EVRPTW-PR-PL problem, as implemented in the subsequent section.

Table 3. Comparison of the results of the large-sized EVRPTW-PR instances among algorithms.

Subset	Average of the Best Solution of 10 Runs ($\bar{f}_b$)					Average Gap ($\Delta\bar{f}$[%])					$\bar{t}$(min)
	BKS	KÇ	SW	HGA	ALNS	BKS	KÇ	SW	HGA	ALNS	
C-1	994.57	1004.04	994.99	1002.25	1005.62	1.13	0.20	0.73	−0.34	0.00	12.64
C-2	629.82	629.95	629.82	640.99	629.82	0.00	−0.02	0.00	−1.77	0.00	5.78
R-1	1199.67	1221.70	1213.96	1239.72	1222.96	1.88	0.09	1.57	−1.41	0.00	15.17
R-2	909.69	927.76	923.09	914.76	918.90	0.98	0.32	0.86	0.40	0.00	11.06
RC-1	1365.25	1403.33	1321.07	1389.71	1404.22	2.70	0.52	2.66	1.05	0.00	15.73
RC-2	1130.90	1140.07	1134.66	1142.61	1142.23	0.98	0.19	0.66	0.06	0.00	10.74
Average	1038.32	1054.48	1036.27	1055.01	1053.96	1.28	0.22	1.08	−0.33	0.00	11.86

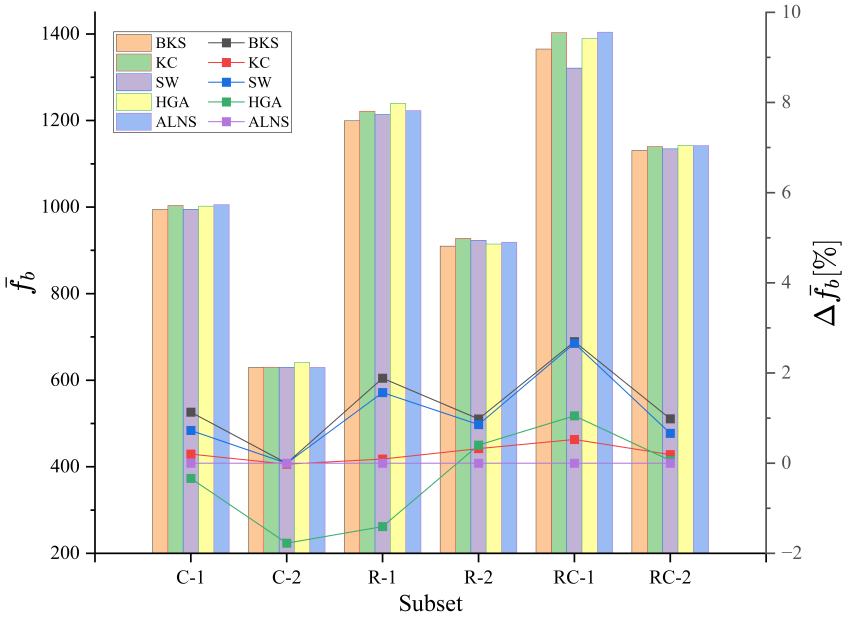


Figure 4. Comparison results on EVRPTW-PR instances among algorithms.

5.4. Results on EVRPTW-PR-PL

This section presents an analysis of the effectiveness of our proposed ALNS for solving EVRPTW-PR-PL instances. We first find solutions to the EVRPTW-PR-PL instances by solving the model presented in Section 3 using a commercial solver (i.e., GUROBI). While optimal solutions are found for small-sized instances, no feasible solution is obtained after 2 h for large-sized instances. Our ALNS is then implemented to solve these instances, especially large ones.

Tables A8–A10 present the results obtained by both GUROBI and ALNS for small-sized instances with 5, 10, and 15 customers, respectively. Here, optimal solutions are provided by GUROBI. Our ALNS algorithm also provides optimal solutions for all instances within an average computational time of 0.51 seconds.

For large-sized instances, the detailed results of subsets C-1, C-2, R-1, R-2, RC-1, and RC-2 are in Tables A11–A16, respectively. Each table presents the best solutions f_b and the average solutions f_a after 10 runs. To assess the algorithm's performance, we calculate the gap between the average cost value (f_a) and the best cost value (f_b), denoted by $\Delta f_{(a-b)}$. This gap indicates the algorithm's stability in finding solutions. Our ALNS performs stability and efficiency analyses when solving large EVRPTW-PR-PL instances with an average gap $\Delta f_{(a-b)}$ of 3.33%.

We also evaluate the impacts of *lockerRemovalInsertion*() operators by varying the value of δ_L within a range of $\{0.1, 0.2, \ldots, 1.0\}$ for solving large-sized instances. Table 4 presents the results from ALNS with different δ_L values. The table includes the overall best objective values obtained for all instances after 10 runs ($\bar{f}_b$) and the average computational time ($\bar{t}$). For each value of δ_L, the gap between $\bar{f}_b$ and the minimum of all $\bar{f}_b$ values is calculated by $\Delta\bar{f}_b = \frac{\bar{f}_b - \min(\bar{f}_b)}{\min(\bar{f}_b)}$. Based on these results, Figure 5 shows that the *lockerRemovalInsertion*() operators significantly improve the performance of our ALNS in solving EVRPTW-PR-PL. In particular, ALNS without using the *lockerRemovalInsertion*() (i.e., $\delta_L = 0$) provides the worst results among all cases, with a gap of 1.02%. It is worth noting that the best performance of ALNS is achieved at $\delta_L = 0.3$.

Table 4. Results of EVRPTW-PR-PL obtained from our ALNS algorithm with different values of δ_L.

δ_L	**0**	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**1**
$\bar{f}_b$	723.31	721.27	720.07	715.98	718.06	718.58	716.85	718.13	719.28	722.99	722.32
$\Delta\bar{f}_b$[%]	1.02	0.74	0.57	0.00	0.29	0.36	0.12	0.30	0.46	0.98	0.89
$\bar{t}(s)$	45.13	43.43	43.56	49.98	44.28	44.20	43.93	56.60	58.64	56.04	56.58

Note: $\min(\bar{f}_b) = 715.98$.

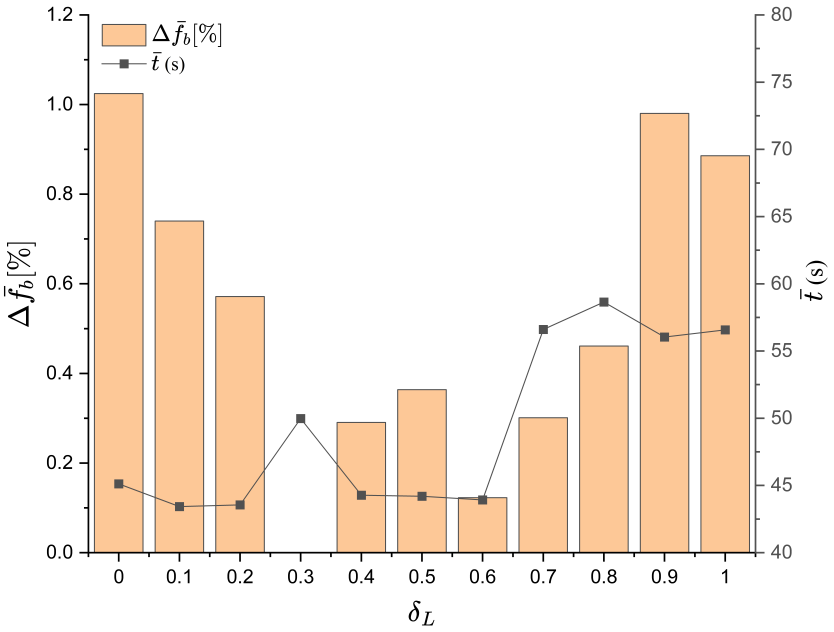


Figure 5. Impact of locker removal and insertion heuristics on the performance of ALNS.

The search process of our ALNS is also analyzed by observing an example of solving instance c101, as shown in Figure 6. The blue line represents the values of the best solution f_b over 30,000 iterations, while the red line shows a linear increase in computational time over iterations. It can be seen that the ALNS algorithm rapidly improves the solution quality at the beginning of the search and achieves convergence after approximately 10,000 iterations.

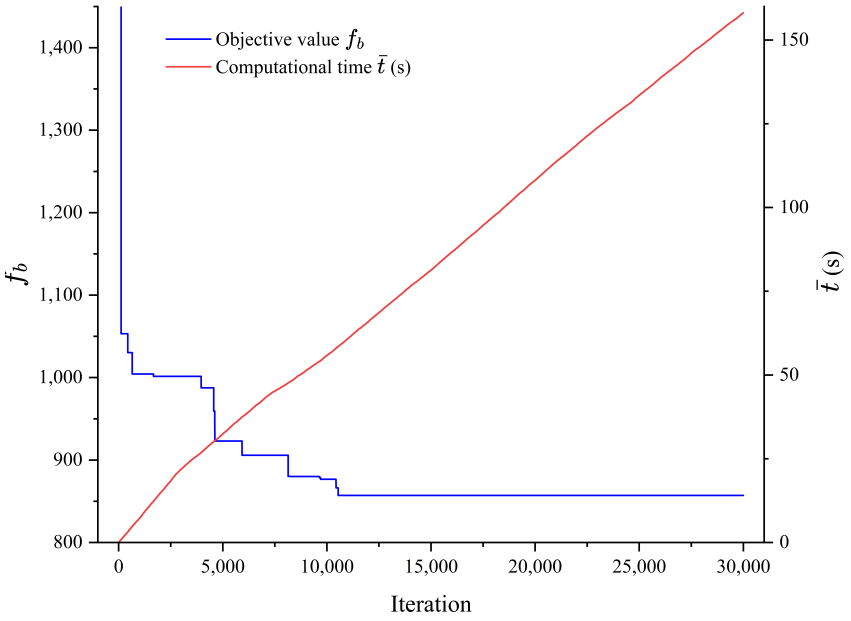


Figure 6. Search process of the proposed ALNS for solving instance c101.

6. Discussion

We further conduct an evaluation to compare the results obtained from EVRPTW-PR and EVRPTW-PR-PL (with $\delta_L = 0.3$) in terms of the cost ratio "CR" and the difference in the number of used vehicles $\Delta|K|$. The cost ratio "CR" is computed as the ratio between the best cost value f_b of EVRPTW-PR and the best cost value f_b of EVRPTW-PR-PL. The difference in used vehicles $\Delta|K|$ is calculated as the number of used vehicles $|K|$ in EVRPTW-PR minus the number of used vehicles $|K|$ in EVRPTW-PR-PL. Therefore, in Tables A8–A16, we recall the results of EVRPTW-PR instances in Section 5.3 and report terms "CR" and $\Delta|K|$ in the right-most columns.

A summary of the results of the average cost ratio "CR" and the average difference in the number of used vehicles for each subset is provided in Table 5. For the results from the large-sized instances, the patterns of "CR" and $\Delta|K|$ are illustrated in Figure 7 by sorting the subsets in ascending order of "CR" values. The figure helps highlight some features, as follows.

Table 5. Comparison of the results of EVRPTW-PR and EVRPTW-PR-PL instances.

Size	Subset	EVRPTW-PR-PL		EVRPTW-PR		CR	$\Delta\|K\|$
		$\|K\|$	$\bar{f}_b$	$\|K\|$	$\bar{f}_b$		
Small-sized	5C	1.42	106.80	1.42	188.90	2.10	0.00
	10C	2.08	156.10	2.08	301.49	2.17	0.00
	15C	2.67	191.27	2.67	357.80	2.11	0.00
Large-sized	C-1	11.00	945.82	10.56	1005.62	1.08	−0.44
	C-2	4.00	526.97	4.00	629.82	1.20	0.00
	R-1	10.42	741.90	12.67	1222.96	1.66	2.25
	R-2	3.27	492.36	2.64	918.90	1.87	−0.64
	RC-1	12.38	1030.82	12.75	1404.22	1.38	0.38
	RC-2	4.00	600.15	3.13	1142.23	1.89	−0.88
	Average	5.69	532.47	5.77	796.88	1.72	0.07

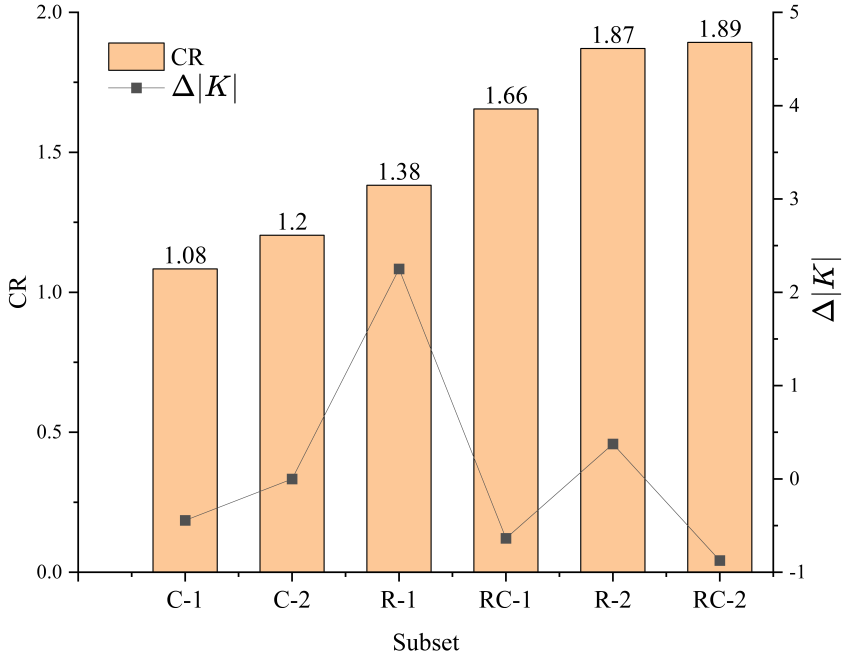
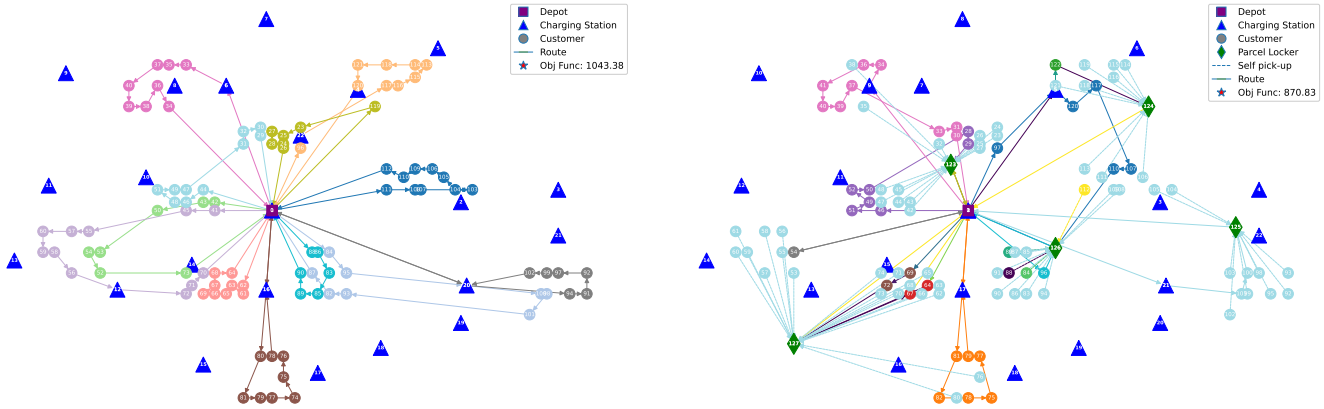


Figure 7. The effects of parcel lockers on the routing plans.

- The implementation of parcel lockers in delivery networks significantly reduces travel costs for routing plans; i.e., the overall cost ratio is 1.72. Figure 8 provides routing plans obtained from instance c101 for EVRPTW-PR and EVRPTW-PR-PL, which illustrates the differences between the two routing plans.
- Considering customer locations, subsets with randomly distributed locations (i.e., subsets (R) and (RC)) show higher "CR" values. For example, the "CR" value for subset C-1 is only 1.08, while those for subsets R-1 and RC-1 are 1.38 and 1.66, respectively. This implies that utilizing parcel lockers becomes more effective when the locations are randomly scattered.
- Considering time windows, subsets with wider customer time windows (i.e., C-1, R-1, and RC-1 subsets) provide higher "CR" values than those with stricter time windows (i.e., C-2, R-2, and RC-2 subsets). This means that instances with wider customer time windows have more potential to find cost-effective solutions when implementing parcel lockers.
- Values of $\Delta|K|$ are not significant for all subsets; i.e., 0.07 vehicles on average. However, for subset R-1, the number of used vehicles is significantly reduced by 2.25 vehicles. This occurs when the customer locations are randomly distributed and the time windows are wider.

The insights noted above can be valuable in reality for delivery companies when making decisions regarding the deployment of parcel lockers in their last-mile delivery networks, aiming to leverage the cost-saving benefits. Specifically, companies can prioritize implementing parcel lockers in areas that reflect customer location patterns of "R" and "RC" subsets, as doing so provides great potential for cutting travel costs.



(**a**) (**b**)

Figure 8. Routing plans of (**a**) EVRPTW-PR and (**b**) EVRPTW-PR-PL for instance c101.

7. Conclusions

This paper addresses an extension of EVRPTW-PR by adding the self-pickup method for serving customers. As a result, there are three types of customers considered: (1) home delivery customers, (2) self-pickup customers, and (3) flexible customers. We formulate a mixed-integer programming (MIP) model and develop an ALNS algorithm adopted by ref. [52] for efficiently solving both EVRPTW-PR and EVRPTW-PR-PL instances. Numerical experiments are conducted to test the performance of our proposed ALNS.

To provide comprehensive managerial insights regarding the implementation of parcel lockers in delivery networks, we have conducted sensitivity analyses based on the results obtained from EVRPTW-PR and EVRPTW-PR-PL. From our observations, the utilization of parcel lockers can lead to additional cost reductions in serving customers. The extent of cost reduction is influenced by various factors, such as the time windows and locations of customers in each instance. Furthermore, in subset R-1, where customer locations are randomly distributed and wider time windows are provided, we observe a decrease in the number of used vehicles for routing plans.

The generality of our algorithm could help tackle other similar problems. For instance, customers' compensation schemes can be investigated by modifying the objective function, which would support policymakers to incentivize customers and enhance delivery network sustainability. Moreover, the cost of allocating parcel locker facilities can be included in the objective function. This provides a more comprehensive model that would help delivery companies make decisions for designing last-mile delivery networks with the use of parcel lockers. However, our research still has some limitations: (1) the scope of the study is limited to analyzing benchmark instances, and no real-world instances have been studied; and (2) the evaluation of the algorithm's performance is challenging due to a lack of results from the existing algorithms or benchmarks for EVRPTW-PR-PL instances.

Some potential future research directions are as follows. First, alternative delivery locations and customers' compensation schemes can be investigated to address a more comprehensive delivery network. More specifically, we could add additional costs (i.e., fixed costs for lockers and compensation costs) to the objective function to align with practical businesses. This model would help policymakers decide compensation strategies for more efficient routing plans and enhance customer satisfaction. Second, in realistic scenarios, the impacts of uncertain factors (e.g., travel time and charging/discharging consumption rates) are also worth studying to capture hidden costs within overall delivery costs. This also poses an interesting challenge through the implementation of stochastic programming or robust optimization. Finally, the applicability of our proposed problem can be enhanced by addressing practical delivery networks and offering more managerial insights.

Author Contributions: Conceptualization, V.F.Y. and P.T.A.; methodology, V.F.Y., P.T.A. and Y.-W.C.; software, P.T.A. and Y.-W.C.; validation, V.F.Y., P.T.A. and Y.-W.C.; formal analysis, Y.-W.C. and P.T.A.; investigation, P.T.A. and Y.-W.C.; resources, V.F.Y.; data curation, V.F.Y. and Y.-W.C.; writing—original draft preparation, P.T.A. and Y.-W.C.; writing—review and editing, V.F.Y.; visualization, P.T.A.; supervision, V.F.Y.; project administration, V.F.Y.; funding acquisition, V.F.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially supported by the National Science and Technology Council of the Republic of China (Taiwan) under grant NSTC 111-2410-H-011-020-MY3 and the Center for Cyber-Physical System Innovation from the Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon reasonable request.

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

Appendix A. Detailed Results

Table A1. Detailed results on EVRPTW-PR small-size instances.

Instance	KÇ (Optimal)		ALNS		Δf[%]	Instance	KÇ (Optimal)		ALNS		Δf[%]
	f	$\bar{t}$ (s)	f_b	$\bar{t}(s)$			f	$\bar{t}(s)$	f_b	$\bar{t}(s)$	
C101-5	257.75	0.31	257.75	0.13	0.00	R201-10	241.52	11.4	241.52	0.48	0.00
C103-5	175.37	2.73	175.37	0.27	0.00	R203-10	218.21	1.62	218.21	0.60	0.00
C206-5	242.56	5.38	242.56	0.17	0.00	RC102-10	423.51	3.07	423.51	0.29	0.00
C208-5	158.48	1.37	158.48	0.17	0.00	RC108-10	345.93	2.9	345.93	0.59	0.00
R104-5	136.69	0.47	136.69	0.06	0.00	RC201-10	412.86 *	-	412.86	0.48	0.00
R105-5	156.08	3.39	156.08	0.05	0.00	RC205-10	325.98	3.26	325.98	0.62	0.00
R202-5	128.78	0.95	128.78	0.09	0.00	C103-15	348.46	1008	348.46	0.66	0.00
R203-5	179.06	1.12	179.06	0.06	0.00	C106-15	275.13	0.47	275.13	0.82	0.00
RC105-5	233.77	3.06	233.77	1.75	0.00	C202-15	383.62	24.07	383.62	0.65	0.00
RC108-5	253.93	3.76	253.93	0.14	0.00	C208-15	300.55	0.92	300.55	0.70	0.00
RC204-5	176.39	2.17	176.39	0.17	0.00	R102-15	412.78 *	-	412.78	0.75	0.00
RC208-5	167.98	1.05	167.98	0.14	0.00	R105-15	336.15	1.39	336.15	0.65	0.00
C101-10	388.25	50.26	388.25	0.39	0.00	R202-15	358	462.89	358.00	0.59	0.00
C104-10	273.93	5.15	273.93	0.57	0.00	R209-15	313.24	610.64	313.24	0.47	0.00
C202-10	304.06	7.52	304.06	0.51	0.00	RC103-15	397.67	20.27	397.67	0.77	0.00
C205-10	228.28	2.01	228.28	0.63	0.00	RC108-15	370.25	101.45	370.25	0.93	0.00
R102-10	249.19	1.83	249.19	0.45	0.00	RC202-15	394.39	113.43	394.39	0.58	0.00
R103-10	206.12	6.76	206.12	0.49	0.00	RC204-15	403.38 *	-	403.38	0.55	0.00
						Average	282.73	74.70	282.73	0.48	0.00

* Solutions obtained are suboptimal even after running for 2 h.

Table A2. Detailed results on EVRPTW-PR large-size instances of subset C-1.

Instance	Best Solution of 10 Runs (f_b)					Gap (Δf[%])				$\bar{t}$(min)
	BKS	KÇ	SW	HGA	ALNS	BKS	KÇ	SW	HGA	
c101	1043.38	1043.38	1043.38	1044.51	1043.38	0.00	0.00	0.00	−0.11	8.05
c102	1017.7	1032.49	1029.44	1033.8	1029.48	1.14	−0.29	0.00	−0.42	16.53
c103	971.19	973.39	971.86	1001.13	989.53	1.85	1.63	1.79	−1.17	9.58
c104	884.38	886.72	884.38	893.04	924.20	4.31	4.06	4.31	3.37	23.41
c105	1015.79	1037.78	1048.06	1052.95	1033.95	1.76	−0.37	−1.36	−1.84	8.85
c106	1009.33	1024.18	1010.56	1043.5	1028.04	1.82	0.38	1.70	−1.50	9.23
c107	1046.5	1058.11	-	-	**1036.17**	−1.00	−2.12	-	-	11.51
c108	1022.48	1033.5	1031.85	-	1025.48	0.29	−0.78	−0.62	-	12.61
c109	940.38	946.84	940.38	946.84	940.38	0.00	−0.69	0.00	−0.69	14.02
Average	994.57	1004.04	994.99	1002.25	1005.62	1.13	0.20	0.73	−0.34	12.64

Bold numbers indicate new best solutions. - We ignore solutions using a different number of vehicles.

Table A3. Detailed results on EVRPTW-PR large-size instances of subset C-2.

Instance	Best Solution of 10 Runs (f_b)					Gap (Δf[%])				$\bar{t}$(min)
	BKS	KÇ	SW	HGA	ALNS	BKS	KÇ	SW	HGA	
c201	629.95	629.95	629.95	658.11	629.95	0.00	0.00	0.00	−4.47	3.18
c202	629.95	629.95	629.95	645.39	629.95	0.00	0.00	0.00	−2.45	6.39
c203	629.95	629.95	629.95	643.45	629.95	0.00	0.00	0.00	−2.14	10.03
c204	628.91	629.95	628.91	636.43	628.91	0.00	−0.17	0.00	−1.20	10.37
c205	629.95	629.95	629.95	638.17	629.95	0.00	0.00	0.00	−1.30	3.14
c206	629.95	629.95	629.95	635.38	629.95	0.00	0.00	0.00	−0.86	4.05
c207	629.95	629.95	629.95	632.8	629.95	0.00	0.00	0.00	−0.45	5.04
c208	629.95	629.95	629.95	638.17	629.95	0.00	0.00	0.00	−1.30	4.05
Average	629.82	629.95	629.82	640.99	629.82	0.00	−0.02	0.00	−1.77	5.78

Table A4. Detailed results on EVRPTW-PR large-size instances of subset R-1.

Instance	Best Solution of 10 Runs (f_b)					Gap (Δf[%])				$\bar{t}$(min)
	BKS	KÇ	SW	HGA	ALNS	BKS	KÇ	SW	HGA	
r101	1606.98	1636.69	1615.5	1630.14	1642.86	2.18	0.38	1.67	0.77	12.63
r102	1461.23	1461.38	1429.8	1521.33	1471.88	0.72	0.71	2.86	−3.36	12.91
r103	1212.37	1262.75	1244.15	1264.81	1261.76	3.91	−0.08	1.40	−0.24	19.07
r104	1051.41	1078.99	1056.87	1089.92	1081.43	2.78	0.23	2.27	−0.79	11.06
r105	1347.8	1373.94	1347.8	1396.8	1375.04	1.98	0.08	1.98	−1.58	16.49
r106	1256.19	1310.46	1268.25	1281.09	1294.95	2.99	−1.20	2.06	1.07	16.92
r107	1108.47	1118.91	1110.95	1127.71	1127.02	1.65	0.72	1.43	−0.06	20.39
r108	1020.52	1031.14	1020.52	1042.8	1039.59	1.83	0.81	1.83	−0.31	11.41
r109	1185.77	1193.76	1186.99	1265.82	**1184.68**	−0.09	−0.77	−0.19	−6.85	10.01
r110	1070.99	1090.92	1070.99	1095	1092.22	1.94	0.12	1.94	−0.25	18.15
r111	1072.46	1084.13	-	1147.23	1102.33	2.71	1.65	-	−4.07	20.43
r112	1001.79	1017.31	1001.79	1013.95	1001.79	0.00	−1.55	0.00	−1.21	12.57
Average	1199.67	1221.70	1213.96	1239.72	1222.96	1.88	0.09	1.57	−1.41	15.17

Bold numbers indicate new best solutions. - We ignore solutions using a different number of vehicles.

Table A5. Detailed results on EVRPTW-PR large-size instances of subset R-2.

Instance	Best Solution of 10 Runs (f_b)					Gap (Δf[%])				$\bar{t}$(min)
	BKS	KÇ	SW	HGA	ALNS	BKS	KÇ	SW	HGA	
r201	1255.81	1262.1	1255.81	1261.64	1280.65	1.94	1.45	1.94	1.48	8.53
r202	1051.46	1052.32	1051.48	1051.46	1058.61	0.68	0.59	0.67	0.68	11.99
r203	895.54	895.54	895.96	900.6	904.09	0.95	0.95	0.90	0.39	15.74
r204	780.91	-	-	783.53	793.88	1.63	-	-	1.30	11.24
r205	987.22	987.36	988.55	987.36	987.18	0.00	−0.02	−0.14	−0.02	9.19
r206	922.7	922.7	922.83	924.48	927.36	0.50	0.50	0.49	0.31	15.56
r207	843.2	846.59	843.2	846.53	853.23	1.18	0.78	1.18	0.79	7.95
r208	736.12	736.12	736.12	736.64	740.12	0.54	0.54	0.54	0.47	9.03
r209	863.36	868.95	863.36	867.8	880.45	1.94	1.31	1.94	1.44	12.35
r210	843.36	843.36	846.33	845.27	849.69	0.74	0.74	0.40	0.52	13.46
r211	826.88	862.56	827.29	857.1	832.60	0.69	−3.60	0.64	−2.94	6.64
Average	909.69	927.76	923.09	914.76	918.90	0.98	0.32	0.86	0.40	11.06

- We ignore solutions using a different number of vehicles.

Table A6. Detailed results on EVRPTW-PR large-size instances of subset RC-1.

Instance	Best Solution of 10 Runs (f_b)					Gap (Δf[%])				$\bar{t}$(min)
	BKS	KÇ	SW	HGA	ALNS	BKS	KÇ	SW	HGA	
rc101	1661.53	1743.9	-	1725.73	1727.07	3.79	−0.97	-	0.08	19.97
rc102	1510.16	1555.5	1510.16	1540.26	1557.11	3.02	0.10	3.02	1.08	10.65
rc103	1346.83	-	-	1388.72	1358.88	0.89	-	-	−2.20	19.63
rc104	1175.06	1202.93	1175.06	1181.26	1175.06	0.00	−2.37	0.00	−0.53	21.10
rc105	1446.3	1458.49	1450.82	1463.49	1494.69	3.24	2.42	2.94	2.09	9.41
rc106	1383.14	1417.4	1385.96	1397.55	1438.59	3.85	1.47	3.66	2.85	11.18
rc107	1244.83	1261.03	1250.3	1255.03	1275.38	2.40	1.13	1.97	1.60	18.02
rc108	1154.14	1184.06	1154.14	1165.6	1206.96	4.38	1.90	4.38	3.43	15.89
Average	1365.25	1403.33	1321.07	1389.71	1404.22	2.70	0.52	2.66	1.05	15.73

- We ignore solutions using a different number of vehicles.

Table A7. Detailed results on EVRPTW-PR large-size instances of subset RC-2.

Instance	Best Solution of 10 Runs (f_b)					Gap (Δf[%])				$\bar{t}$ (min)
	BKS	KÇ	SW	HGA	ALNS	BKS	KÇ	SW	HGA	
rc201	1443.07	1446.84	1445.17	1446.03	1448.78	0.39	0.13	0.25	0.19	9.78
rc202	1403.32	1416.96	1408.08	1434.18	1431.88	1.99	1.04	1.66	−0.16	8.56
rc203	1060.32	1069.27	1060.32	1061.12	1069.12	0.82	−0.01	0.82	0.75	13.43
rc204	884.75	886.23	884.75	887.1	904.66	2.20	2.04	2.20	1.94	14.50
rc205	1249.56	1262.22	1259.69	1289.08	1263.53	1.11	0.10	0.30	−2.02	6.29
rc206	1187.4	1206.09	1189.11	1200.74	1194.28	0.58	−0.99	0.43	−0.54	10.47
rc207	985.67	993.26	997.04	985.67	987.93	0.23	−0.54	-0.92	0.23	11.21
rc208	833.12	839.71	833.12	836.93	837.69	0.55	−0.24	0.55	0.09	11.71
Average	1130.90	1140.07	1134.66	1142.61	1142.23	0.98	0.19	0.66	0.06	10.74

Table A8. Detailed results on EVRPTW-PR-PL small-size instances with 5 customers.

Instance	EVRPTW-PR-PL					Δf[%]	EVRPTW-PR		$\Delta\|K\|$	CR
	GUROBI			ALNS						
	|K|	f	$t(s)$	f_b	$\bar{t}(s)$		|K|	f_b		
C101-5	2	62.72	0.02	62.72	0.06	0.00	2	257.75	0	4.11
C103-5	1	105.69	0.03	105.69	0.06	0.00	1	175.37	0	1.66
C206-5	1	200.1	0.08	200.10	0.14	0.00	1	242.56	0	1.21
C208-5	1	93.13	0.03	93.13	0.12	0.00	1	158.48	0	1.70
R104-5	2	100.00	0.05	100.00	0.06	0.00	2	136.69	0	1.37
R105-5	2	100.22	0.03	100.22	0.35	0.00	2	156.08	0	1.56
R202-5	1	112.65	0.03	112.65	0.11	0.00	1	128.78	0	1.14
R203-5	1	183.23	0.09	183.23	0.20	0.00	1	179.06	0	0.98
RC105-5	2	64	0.03	64.00	1.11	0.00	2	233.77	0	3.65
RC108-5	2	62.8	0.01	62.80	0.05	0.00	2	253.93	0	4.04
RC204-5	1	126.74	0.09	126.74	0.13	0.00	1	176.39	0	1.39
RC208-5	1	70.26	0.03	70.26	0.06	0.00	1	167.98	0	2.39
Average	1.42	106.80	0.04	106.80	0.20	0.00	1.42	188.90	0.00	2.10

Table A9. Detailed results on EVRPTW-PR-PL small-size instances with 10 customers.

Instance	EVRPTW-PR-PL					Δf[%]	EVRPTW-PR		$\Delta\|K\|$	CR
	GUROBI			ALNS						
	|K|	f	$t(s)$	f_b	$\bar{t}(s)$		|K|	f_b		
C101-10	3	224.23	0.16	224.23	0.38	0.00	3	388.25	0	1.73
C104-10	2	227.53	0.22	227.53	1.92	0.00	2	273.93	0	1.20
C202-10	1	147.57	0.09	147.57	0.26	0.00	1	304.06	0	2.06
C205-10	2	182.04	0.2	182.04	0.29	0.00	2	228.28	0	1.25
R102-10	3	88.32	0.05	88.32	0.14	0.00	3	249.19	0	2.82
R103-10	2	99.88	0.19	99.88	0.14	0.00	2	206.12	0	2.06
R201-10	1	59.92	0.06	59.92	0.15	0.00	1	241.52	0	4.03
R203-10	1	196.02	0.22	196.02	0.32	0.00	1	218.21	0	1.11
RC102-10	4	220.12	0.16	220.12	0.20	0.00	4	423.51	0	1.92
RC108-10	3	147.35	0.13	147.35	0.23	0.00	3	345.93	0	2.35
RC201-10	1	115.33	0.09	115.323	0.24	0.00	1	412.86	0	3.58
RC205-10	2	164.94	0.27	164.94	0.38	0.00	2	325.98	0	1.98
Average	2.08	156.10	0.15	156.10	0.39	0.00	2.08	301.49	0.00	2.17

Table A10. Detailed results on EVRPTW-PR-PL small-size instances with 15 customers.

Instance	EVRPTW-PR-PL					Δf[%]	EVRPTW-PR		$\Delta\|K\|$	CR
	GUROBI			ALNS						
	|K|	f	$t(s)$	f_b	$\bar{t}(s)$		|K|	f_b		
C103-15	3	167.05	0.27	167.05	0.39	0.00	3	348.46	0	2.09
C106-15	3	239.85	0.55	239.85	0.93	0.00	3	275.13	0	1.15
C202-15	2	106.02	0.22	106.02	0.42	0.00	2	383.62	0	3.62
C208-15	2	161.14	0.22	161.14	0.96	0.00	2	300.55	0	1.87
R102-15	5	245.52	2.83	245.52	1.80	0.00	5	412.78	0	1.68
R105-15	4	105.2	0.45	105.2	1.54	0.00	4	336.15	0	3.20
R202-15	2	189.4	0.77	189.4	1.13	0.00	2	358	0	1.89
R209-15	1	145.95	1.11	145.95	0.48	0.00	1	313.24	0	2.15
RC103-15	4	267.93	3.78	267.93	1.38	0.00	4	397.67	0	1.48
RC108-15	3	344.55	3.2	344.55	0.52	0.00	3	370.25	0	1.07
RC202-15	2	130.24	0.11	130.24	0.35	0.00	2	394.39	0	3.03
RC204-15	1	192.44	2.63	192.44	1.20	0.00	1	403.38	0	2.10
Average	2.67	191.27	1.35	191.27	0.93	0.00	2.67	357.80	0.00	2.11

Table A11. Detailed results on EVRPTW-PR-PL large-size instances of subset C-1.

Instance	EVRPTW-PR-PL					EVRPTW-PR		$\Delta\|K\|$	CR
	$\|K\|$	f_b	f_a	$\Delta f_{(a-b)}$	$\bar{t}$	$\|K\|$	f_b		
C101	11	870.83	925.722	6.3	49.40	12	1043.38	1	1.20
C102	11	1079.29	1112.66	3.09	61.80	11	1029.48	0	0.95
C103	11	983.09	988.27	0.53	43.40	10	989.53	−1	1.01
C104	11	713.44	724.412	1.54	37.00	10	924.2	−1	1.30
C105	11	1129.97	1162.672	2.89	49.60	11	1033.95	0	0.92
C106	12	965.29	989.214	2.48	47.40	11	1028.04	−1	1.07
C107	11	1110.96	1164.492	4.82	40.80	10	1036.17	−1	0.93
C108	11	799.33	827.026	3.46	47.40	10	1025.48	−1	1.28
C109	10	860.16	885.192	2.91	47.00	10	940.38	0	1.09
Average	11.00	945.82	975.52	3.11	47.09	10.56	1005.62	−0.44	1.08

Table A12. Detailed results on EVRPTW-PR-PL large-size instances of subset C-2.

Instance	EVRPTW-PR-PL					EVRPTW-PR		$\Delta\|K\|$	CR
	$\|K\|$	f_b	f_a	$\Delta f_{(a-b)}$	$\bar{t}$	$\|K\|$	f_b		
C201	4	567.59	579.034	2.02	31.60	4	629.95	0	1.11
C202	5	522.02	522.958	0.18	35.40	4	629.95	−1	1.21
C203	3	487.72	494.3	1.35	53.00	4	629.95	1	1.29
C204	4	523.67	523.918	0.05	46.60	4	628.91	0	1.20
C205	4	478.12	479.392	0.27	43.40	4	629.95	0	1.32
C206	4	620.97	630.752	1.58	57.00	4	629.95	0	1.01
C207	4	473.37	474.394	0.22	34.00	4	629.95	0	1.33
C208	4	542.27	561.37	3.52	71.60	4	629.95	0	1.16
Average	4.00	526.97	533.26	1.15	46.58	4.00	629.82	0.00	1.20

Table A13. Detailed results on EVRPTW-PR-PL large-size instances of subset R-1.

Instance	EVRPTW-PR-PL					EVRPTW-PR		$\Delta\|K\|$	CR
	$\|K\|$	f_b	f_a	$\Delta f_{(a-b)}$	$\bar{t}$	$\|K\|$	f_b		
R101	12	806.21	827.956	2.7	45.20	18	1642.86	6	2.04
R102	10	723.69	730.41	0.93	66.20	15	1471.88	5	2.03
R103	11	776.12	818.944	5.52	47.00	13	1261.76	2	1.63
R104	9	757.79	795.534	4.98	53.80	11	1081.43	2	1.43
R105	12	807.09	863.804	7.03	28.60	14	1375.04	2	1.70
R106	14	742.53	787.044	5.99	29.00	13	1294.95	−1	1.74
R107	10	753.41	803.188	6.61	43.80	12	1127.02	2	1.50
R108	8	557.93	600.61	7.65	38.00	11	1039.59	3	1.86
R109	10	681.17	730.158	7.19	49.60	12	1184.68	2	1.74
R110	9	715.32	756.246	5.72	41.80	11	1092.22	2	1.53
R111	9	781.97	815.734	4.32	39.80	11	1102.33	2	1.41
R112	11	799.56	849.464	6.24	35.60	11	1001.79	0	1.25
Average	10.42	741.90	781.59	5.41	43.20	12.67	1222.96	2.25	1.66

Table A14. Detailed results on EVRPTW-PR-PL large-size instances of subset R-2.

Instance	EVRPTW-PR-PL					EVRPTW-PR		$\Delta\|K\|$	CR
	$\|K\|$	f_b	f_a	$\Delta f_{(a-b)}$	$\bar{t}$	$\|K\|$	f_b		
R201	5	671.93	675.182	0.48	46.00	3	1280.65	−2	1.91
R202	4	591.99	592.742	0.13	51.40	3	1058.61	−1	1.79
R203	3	455.31	456.358	0.23	31.60	3	904.09	0	1.99
R204	3	409.46	412.376	0.71	35.20	2	793.88	−1	1.94
R205	4	534.39	538.784	0.82	44.20	3	987.18	−1	1.85
R206	3	492.25	494.724	0.5	46.00	3	927.36	0	1.88
R207	3	470.36	470.36	0	30.80	2	853.23	−1	1.81
R208	2	369.42	372.222	0.76	39.80	2	740.12	0	2.00
R209	3	465.58	468.302	0.58	56.80	3	880.45	0	1.89
R210	3	474.84	476.172	0.28	44.00	3	849.69	0	1.79
R211	3	480.47	485.254	1	43.00	2	832.6	−1	1.73
Average	3.27	492.36	494.77	0.50	42.62	2.64	918.90	−0.64	1.87

Table A15. Detailed results on EVRPTW-PR-PL large-size instances of subset RC-1.

Instance	EVRPTW-PR-PL					EVRPTW-PR		$\Delta\|K\|$	CR
	$\|K\|$	f_b	f_a	$\Delta f_{(a-b)}$	$\bar{t}$	$\|K\|$	f_b		
RC101	15	1109.8	1210.066	9.03	26.80	15	1727.07	0	1.56
RC102	12	1011.42	1119.514	10.69	33.40	14	1557.11	2	1.54
RC103	12	1262.12	1379.778	9.32	27.00	12	1358.88	0	1.08
RC104	10	905.39	965.9	6.68	35.00	11	1175.06	1	1.30
RC105	17	1156.29	1302.744	12.67	29.80	14	1494.69	−3	1.29
RC106	10	874.47	979.792	12.04	33.20	13	1438.59	3	1.65
RC107	11	829.04	860.446	3.79	40.00	12	1275.38	1	1.54
RC108	12	1098.05	1183.874	7.82	34.00	11	1206.96	−1	1.10
Average	12.38	1030.82	1125.26	9.01	32.40	12.75	1404.22	0.38	1.38

Table A16. Detailed results on EVRPTW-PR-PL large-size instances of subset RC-2

Instance	EVRPTW-PR-PL					EVRPTW-PR		$\Delta\|K\|$	CR
	$\|K\|$	f_b	f_a	$\Delta f_{(a-b)}$	$\bar{t}$	$\|K\|$	f_b		
RC201	5	688.6	689.12	0.08	58.00	4	1448.78	−1	2.10
RC202	4	639.58	641.706	0.33	50.20	3	1431.88	−1	2.24
RC203	4	614.19	616.414	0.36	44.40	3	1069.12	−1	1.74
RC204	3	472.48	483.358	2.3	35.20	3	904.66	0	1.91
RC205	5	698.37	698.514	0.02	49.20	3	1263.53	−2	1.81
RC206	4	607.79	620.66	2.12	41.60	3	1194.28	−1	1.96
RC207	4	580.51	584.89	0.75	53.60	3	987.93	−1	1.70
RC208	3	499.65	505.134	1.1	56.80	3	837.69	0	1.68
Average	4.00	600.15	604.97	0.88	48.63	3.13	1142.23	−0.88	1.89

References

1. Statista. *Retail e-Commerce Sales Worldwide from 2014 to 2026*; Technical Report; Statista: Hamburg, Germany, 2022.
2. Sadati, M.E.H.; Akbari, V.; Çatay, B. Electric vehicle routing problem with flexible deliveries. *Int. J. Prod. Res.* **2022**, *60*, 4268–4294. [CrossRef]
3. Gevaers, R.; Van de Voorde, E.; Vanelslander, T. Characteristics of innovations in last-mile logistics-using best practices, case studies and making the link with green and sustainable logistics. *Assoc. Eur. Transp. Contrib.* **2009**, *1*, 21.
4. World Economic Forum. *The Future of the Last-Mile Ecosystem*; Technical Report; World Economic Forum: Cologny, Switzerland, 2020.
5. Silva, V.; Amaral, A.; Fontes, T. Sustainable urban last-mile logistics: A systematic literature review. *Sustainability* **2023**, *15*, 2285. [CrossRef]
6. Conrad, R.G.; Figliozzi, M.A. The recharging vehicle routing problem. In Proceedings of the 2011 Industrial Engineering Research Conference, IISE Norcross, Singapore, 6–9 December 2011.
7. 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]
8. Deutsch, Y.; Golany, B. A parcel locker network as a solution to the logistics last mile problem. *Int. J. Prod. Res.* **2018**, *56*, 251–261. [CrossRef]
9. Boysen, N.; Fedtke, S.; Schwerdfeger, S. Last-mile delivery concepts: A survey from an operational research perspective. *OR Spectr.* **2021**, *43*, 1–58. [CrossRef]
10. Vukićević, M.; Ratli, M.; Rivenq, A.; Zrikem, M. Covering delivery problem with electric vehicle and parcel lockers: Variable neighborhood search approach. *Comput. Oper. Res.* **2023**, *157*, 106263. [CrossRef]
11. Schneider, M.; Stenger, A.; Goeke, D. The electric vehicle-routing problem with time windows and recharging stations. *Transp. Sci.* **2014**, *48*, 500–520. [CrossRef]
12. Desaulniers, G.; Errico, F.; Irnich, S.; Schneider, M. Exact algorithms for electric vehicle-routing problems with time windows. *Oper. Res.* **2016**, *64*, 1388–1405. [CrossRef]
13. Hiermann, G.; Hartl, R.F.; Puchinger, J.; Vidal, T. Routing a mix of conventional, plug-in hybrid, and electric vehicles. *Eur. J. Oper. Res.* **2019**, *272*, 235–248. [CrossRef]
14. Cortés-Murcia, D.L.; Prodhon, C.; Afsar, H.M. The electric vehicle routing problem with time windows, partial recharges and satellite customers. *Transp. Res. Part E Logist. Transp. Rev.* **2019**, *130*, 184–206. [CrossRef]
15. Keskin, M.; Çatay, B. A matheuristic method for the electric vehicle routing problem with time windows and fast chargers. *Comput. Oper. Res.* **2018**, *100*, 172–188. [CrossRef]
16. Rastani, S.; Çatay, B. A large neighborhood search-based matheuristic for the load-dependent electric vehicle routing problem with time windows. *Ann. Oper. Res.* **2021**, *324*, 761–793. [CrossRef]
17. Montoya, A.; Guéret, C.; Mendoza, J.E.; Villegas, J.G. The electric vehicle routing problem with nonlinear charging function. *Transp. Res. Part Methodol.* **2017**, *103*, 87–110. [CrossRef]

18. Froger, A.; Mendoza, J.E.; Jabali, O.; Laporte, G. Improved formulations and algorithmic components for the electric vehicle routing problem with nonlinear charging functions. *Comput. Oper. Res.* **2019**, *104*, 256–294. [CrossRef]
19. Sassi, O.; Oulamara, A. Electric vehicle scheduling and optimal charging problem: Complexity, exact and heuristic approaches. *Int. J. Prod. Res.* **2017**, *55*, 519–535. [CrossRef]
20. Macrina, G.; Laporte, G.; Guerriero, F.; Pugliese, L.D.P. An energy-efficient green-vehicle routing problem with mixed vehicle fleet, partial battery recharging and time windows. *Eur. J. Oper. Res.* **2019**, *276*, 971–982. [CrossRef]
21. Mancini, S. The hybrid vehicle routing problem. *Transp. Res. Part C Emerg. Technol.* **2017**, *78*, 1–12. [CrossRef]
22. Keskin, M.; Laporte, G.; Çatay, B. Electric vehicle routing problem with time-dependent waiting times at recharging stations. *Comput. Oper. Res.* **2019**, *107*, 77–94. [CrossRef]
23. Froger, A.; Jabali, O.; Mendoza, J.E.; Laporte, G. The electric vehicle routing problem with capacitated charging stations. *Transp. Sci.* **2022**, *56*, 460–482. [CrossRef]
24. Sweda, T.M.; Dolinskaya, I.S.; Klabjan, D. Adaptive routing and recharging policies for electric vehicles. *Transp. Sci.* **2017**, *51*, 1326–1348. [CrossRef]
25. Keskin, M.; Çatay, B.; Laporte, G. A simulation-based heuristic for the electric vehicle routing problem with time windows and stochastic waiting times at recharging stations. *Comput. Oper. Res.* **2021**, *125*, 105060. [CrossRef]
26. Masmoudi, M.A.; Hosny, M.; Demir, E.; Genikomsakis, K.N.; Cheikhrouhou, N. The dial-a-ride problem with electric vehicles and battery swapping stations. *Transp. Res. Part Logist. Transp. Rev.* **2018**, *118*, 392–420. [CrossRef]
27. Jie, W.; Yang, J.; Zhang, M.; Huang, Y. The two-echelon capacitated electric vehicle routing problem with battery swapping stations: Formulation and efficient methodology. *Eur. J. Oper. Res.* **2019**, *272*, 879–904. [CrossRef]
28. Çatay, B.; Sadati, İ. An improved matheuristic for solving the electric vehicle routing problem with time windows and synchronized mobile charging/battery swapping. *Comput. Oper. Res.* **2023**, *159*, 106310. [CrossRef]
29. Lu, J.; Chen, Y.; Hao, J.K.; He, R. The time-dependent electric vehicle routing problem: Model and solution. *Expert Syst. Appl.* **2020**, *161*, 113593. [CrossRef]
30. Erdelić, T.; Carić, T. A survey on the electric vehicle routing problem: Variants and solution approaches. *J. Adv. Transp.* **2019**, *2019*, 5075671. [CrossRef]
31. Behnke, M. Recent Trends in Last Mile Delivery: Impacts of Fast Fulfillment, Parcel Lockers, Electric or Autonomous Vehicles, and More. In Proceedings of the Logistics Management: Strategies and Instruments for Digitalizing and Decarbonizing Supply Chains-Proceedings of the German Academic Association for Business Research, Halle, Germany, 18–20 September 2019; Springer: Berlin/Heidelberg, Germany, 2019; pp. 141–156.
32. de Jong, C.; Kant, G.; Van Vlient, A. *On Finding Minimal Route Duration in the Vehicle Routing Problem with Multiple Time Window*; Department of Computer Science, Utrecht University: Utrecht, The Netherlands, 1996.
33. Doerner, K.F.; Gronalt, M.; Hartl, R.F.; Kiechle, G.; Reimann, M. Exact and heuristic algorithms for the vehicle routing problem with multiple interdependent time windows. *Comput. Oper. Res.* **2008**, *35*, 3034–3048. [CrossRef]
34. Beheshti, A.K.; Hejazi, S.R.; Alinaghian, M. The vehicle routing problem with multiple prioritized time windows: A case study. *Comput. Ind. Eng.* **2015**, *90*, 402–413. [CrossRef]
35. Belhaiza, S.; Hansen, P.; Laporte, G. A hybrid variable neighborhood tabu search heuristic for the vehicle routing problem with multiple time windows. *Comput. Oper. Res.* **2014**, *52*, 269–281. [CrossRef]
36. Reyes, D.; Savelsbergh, M.; Toriello, A. Vehicle routing with roaming delivery locations. *Transp. Res. Part C Emerg. Technol.* **2017**, *80*, 71–91. [CrossRef]
37. Ozbaygin, G.; Karasan, O.E.; Savelsbergh, M.; Yaman, H. A branch-and-price algorithm for the vehicle routing problem with roaming delivery locations. *Transp. Res. Part Methodol.* **2017**, *100*, 115–137. [CrossRef]
38. Ozbaygin, G.; Savelsbergh, M. An iterative re-optimization framework for the dynamic vehicle routing problem with roaming delivery locations. *Transp. Res. Part Methodol.* **2019**, *128*, 207–235. [CrossRef]
39. Tilk, C.; Olkis, K.; Irnich, S. The last-mile vehicle routing problem with delivery options. *OR Spectr.* **2021**, *43*, 877–904. [CrossRef]
40. 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]
41. Zhou, L.; Baldacci, R.; Vigo, D.; Wang, X. A multi-depot two-echelon vehicle routing problem with delivery options arising in the last mile distribution. *Eur. J. Oper. Res.* **2018**, *265*, 765–778. [CrossRef]
42. Sitek, P.; Wikarek, J. Capacitated vehicle routing problem with pick-up and alternative delivery (CVRPPAD): Model and implementation using hybrid approach. *Ann. Oper. Res.* **2019**, *273*, 257–277. [CrossRef]
43. Enthoven, D.L.; Jargalsaikhan, B.; Roodbergen, K.J.; Uit het Broek, M.A.; Schrotenboer, A.H. The two-echelon vehicle routing problem with covering options: City logistics with cargo bikes and parcel lockers. *Comput. Oper. Res.* **2020**, *118*, 104919. [CrossRef]
44. Carotenuto, P.; Ceccato, R.; Gastaldi, M.; Giordani, S.; Rossi, R.; Salvatore, A. Comparing home and parcel lockers' delivery systems: A math-heuristic approach. *Transp. Res. Procedia* **2022**, *62*, 91–98. [CrossRef]
45. Yu, V.F.; Susanto, H.; Jodiawan, P.; Ho, T.W.; Lin, S.W.; Huang, Y.T. A Simulated Annealing Algorithm for the Vehicle Routing Problem with Parcel Lockers. *IEEE Access* **2022**, *10*, 20764–20782. [CrossRef]
46. Orenstein, I.; Raviv, T.; Sadan, E. Flexible parcel delivery to automated parcel lockers: Models, solution methods and analysis. *EURO J. Transp. Logist.* **2019**, *8*, 683–711. [CrossRef]

47. Grabenschweiger, J.; Doerner, K.F.; Hartl, R.F.; Savelsbergh, M.W. The vehicle routing problem with heterogeneous locker boxes. *CEntral Eur. J. Oper. Res.* **2021**, *29*, 113–142. [CrossRef]
48. 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]
49. Hemmelmayr, V.C.; Cordeau, J.F.; Crainic, T.G. An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. *Comput. Oper. Res.* **2012**, *39*, 3215–3228. [CrossRef] [PubMed]
50. Yu, V.F.; Anh, P.T.; Baldacci, R. A robust optimization approach for the vehicle routing problem with cross-docking under demand uncertainty. *Transp. Res. Part E Logist. Transp. Rev.* **2023**, *173*, 103106. [CrossRef]
51. Hiermann, G.; Puchinger, J.; Ropke, S.; Hartl, R.F. The electric fleet size and mix vehicle routing problem with time windows and recharging stations. *Eur. J. Oper. Res.* **2016**, *252*, 995–1018. [CrossRef]
52. Schiffer, M.; Walther, G. An adaptive large neighborhood search for the location-routing problem with intra-route facilities. *Transp. Sci.* **2018**, *52*, 331–352. [CrossRef]
53. Goeke, D.; Schneider, M. Routing a mixed fleet of electric and conventional vehicles. *Eur. J. Oper. Res.* **2015**, *245*, 81–99. [CrossRef]
54. Yu, V.F.; Jodiawan, P.; Gunawan, A. An Adaptive Large Neighborhood Search for the green mixed fleet vehicle routing problem with realistic energy consumption and partial recharges. *Appl. Soft Comput.* **2021**, *105*, 107251. [CrossRef]
55. Kindervater, G.A.; Savelsbergh, M.W. 10. Vehicle Routing: Handling Edge Exchanges. In *Local Search in Combinatorial Optimization*; Princeton University Press: Princeton, NJ, USA, 2018; pp. 337–360.
56. Shaw, P. *A New Local Search Algorithm Providing High Quality Solutions to Vehicle Routing Problems*; University of Strathclyde: Glasgow, UK, 1997.