sustainability

MDPI

Article

An Adaptive Tabu Search Algorithm for Solving the Two-Dimensional Loading Constrained Vehicle Routing Problem with Stochastic Customers

Zheng Zhang [1], Bin Ji [1,*] and Samson S. Yu [2]

[1] School of Traffic & Transportation Engineering, Central South University, Changsha 410075, China
[2] School of Engineering, Deakin University, Waurn Ponds, VIC 3216, Australia
* Correspondence: chcujibin@csu.edu.cn

Abstract: In practical logistic distributions, uncertainties may exist in each distribution process, and sometimes suppliers have to take undesirable measures to deal with the subsequent schedule variances. In light of the uncertainty of customers in logistics distribution and the widely applied two-dimensional loading patterns in transportation, we propose and formulate a two-dimensional loading-constrained vehicle routing problem with stochastic customers (2L-VRPSC), where each customer has a known probability of presence and customers' demands are a set of non-stackable items. A stochastic modeling platform of 2L-VRPSC is established based on a Monte Carlo simulation and scenario analysis to minimize the expected total transportation cost. To achieve this, an enhanced adaptive tabu search (EATS) algorithm incorporating the multi-order bottom-fill-skyline (MOBFS) packing heuristic is proposed, where the EATS algorithm searches for the optimal routing combination and the MOBFS checks the feasibility of each route and guides the EATS to search for feasible solutions. The widely used two-dimensional loading-constrained vehicle routing problem (2L-VRP) benchmarks under different loading configurations considering items' sequential and rotation constraints are applied for experiments, which demonstrates the comparable efficiency of the proposed EATS-MOBFS for solving 2L-VRP. Furthermore, the results and analysis of experiments based on the new 2L-VRPSC instances verify the versatility of the proposed solving approach, which is capable of providing more practical solutions to some real-life scenarios with customers' uncertain information.

Keywords: adaptive tabu search; customer uncertainty; loading constraints; vehicle routing; scenario simulation

Citation: Zhang, Z.; Ji, B.; Yu, S.S. An Adaptive Tabu Search Algorithm for Solving the Two-Dimensional Loading Constrained Vehicle Routing Problem with Stochastic Customers. *Sustainability* **2023**, *15*, 1741. https://doi.org/10.3390/su15021741

Academic Editor: António Abreu

Received: 26 November 2022
Revised: 13 January 2023
Accepted: 13 January 2023
Published: 16 January 2023

1. Introduction

Vehicle routing problem (VRP) and its extensions are classical NP-hard combinatorial optimization problems, which are deemed key components of logistics management and distribution and have been widely studied in operational research [1,2]. In real-world scenarios, such as transporting delicate furniture and household appliances, these rectangular-shaped products cannot be stacked on each other because of the fragility and the height of the freights. These products are normally regarded as rectangular shapes with known length and width, and they cannot be stacked up or overlap in the vehicle's compartment space, which must be reasonably located to ensure a high utilization rate of the carriage space. As such, the VRP needs to simultaneously concern routing and packing problems, engendering the two-dimensional loading-constrained vehicle routing problem (2L-VRP). The aim of solving the problem is to design the optimal routing combination with minimum cost for given vehicles while satisfying the loading constraints, where customers' demands are sets of rectangular weighed items and vehicles are characterized by a two-dimensional loading space with a specified loading capacity [3].

In recent years, 2L-VRP has received increasing attention for operation research due to its practical value and high complexity. 2L-VRP was first presented in [3], where an exact

method was proposed for solving small-scale 2L-VRP instances. Thereafter, a large number of research attempts have been devoted to designing efficient two-dimensional packing heuristics and routing optimization methods in order to improve the solving efficiency for the large-scale problem. In [4], the authors first proposed a heuristic approach that combined Tabu search with a Touching-Perimeter-rule-based packing algorithm, which solved large-scale instances with up to 255 customers. Other heuristic methods, such as genetic algorithm [5,6], extended guided Tabu search [7], promise routing memory packing [8] and variable neighborhood search algorithm [9], have been presented to efficiently solve 2L-VRP. In these pieces of work, only the unrestricted and sequential loading configurations of 2L-VRP were studied. However, allowing items to rotate during the packing process has many practical applications. Lately, the authors in [10] proposed a biased-randomized algorithm to solve the unrestricted 2L-VRP with and without item rotations, while the two-dimensional sequential loading configuration was not studied. In [11], the authors studied four classes of 2L-VRP, including item rotation and sequential loading, and a saving-based ant colony optimization (ACO) algorithm was designed to solve the problem. This is the first published research that deals with the four classes of 2L-VRP, namely, unrestricted oriented loading (2|UO|L), unrestricted rotation-allowed loading (2|UR|L), sequential oriented loading (2|SO|L) and sequential rotation-allowed loading (2|SR|L). Recently, a simulated annealing method [12] was proposed for solving four classes of 2L-VRP, which improved loads of best-known solutions. Thus far, a large number of extended 2L-VRPs have received research attention, e.g., 2L-VRP with backhauls [13], 2L-VRP with heterogeneous fleet [14,15], 2L-VRP with split delivery [16] and three-dimensional constrained vehicle routing problem [17–19]. Nevertheless, the above research generally considers that the geographic location, demand, central depot, vehicle driving status and other information of customers are known in advance and do not change. This, however, may not be consistent with some real-world scenarios involved with uncertain information. In this context, deterministic models for 2L-VRP cannot capture these uncertainties observed in practice.

As stated in [20], the uncertain vehicle routing problem contains two kinds of important uncertain information: information quality and information evolution. Information evolution refers to the fact that some information may not be known in advance and will be revealed over time during the transportation process, which is known as the dynamic VRP. Information quality refers to the uncertainty of some information; for example, the distribution of some information, which is usually modeled as a probability function. This is generally known as the stochastic VRP (SVRP). Taking the uncertainty and the prior knowledge of the uncertain information into account is of significant importance for optimizing the routing schedule that avoids infeasibility and saves transportation costs.

At present, one of the most common solution frameworks for stochastic routing problems is prior optimization. It models the problem in two stages: determining a set of prior routes with uncertain information and making decisions as deterministic information is obtained based on pre-determined policies. Generally, stochastic programming with recourse (SPR) and chance-constrained programming (CCP) are used to determine the "prior solution" [21]. In the CCP, the problem is solved so that the probability of route failure is below a certain level, and the cost of failures is ignored. In the SPR, it determines the first-stage solution that minimizes the expected cost of the second-stage solution. Although the SPR is more difficult to solve than the CCP, the SPR objectives are more meaningful [21]. Therefore, most of the research focused on studies using the SPR approach to address the SVRP. For example, Refs [22–24] studied the VRP with stochastic customers, stochastic travel time and stochastic demand, respectively, which were modeled as stochastic programming and the corresponding solution algorithms were proposed. However, the above studies did not consider the loading problem, which limited the applicability of these studies to certain scenarios, where each customer consists of a certain number of weighed items with a given width and length.

In [25], the authors investigated a SVRP with uncertain demands, where a discrete probability distribution was adopted to characterize the height and width. In this work, the authors presented a two-stage stochastic program to describe this problem and proposed an inter L-shaped method to solve it. Guimarans et al. in [26] studied a 2L-VRP considering stochastic travel time, which was modeled as a two-stage stochastic program. However, only the 2|UO|L and 2|UR|L loading configurations were studied in this work, while the two-dimensional sequential loading configuration was not considered. It is also commonly encountered, for example, in less-than-truckload operations, where a stable set of customers is served on a regular basis, but not every customer requires a visit every day; namely, the presence of customers is probabilistic.

With the aforementioned problems and main purpose of this study, the main contribution of this study is fourfold: (i) By addressing the characteristics of customer-presence uncertainty and two-dimensional patterns in logistics distribution, this study investigates novel extensions of the vehicle routing problem known as the 2L-VRPSC. (ii) A stochastic programming model of 2L-VRPSC is first proposed based on Monte Carlo simulation and scenario analysis. (iii) To solve this problem, an enhanced adaptive tabu search algorithm incorporating a multi-order bottom-fill-skyline packing heuristic, i.e., EATS-MOBFS, is proposed, where multiple neighborhood operators are designed. Monte Carlo simulation and scenario analysis are integrated into the search process to assess the performance of the obtained solutions and provide guidance to the algorithm during the solution-searching process. (iv) Benchmark instances of 2L-VRPSC are generated by extending the classical 2L-VRP benchmark instances. The experimental results verify the applicability of the proposed model and solution approach for dealing with customers' uncertainties in 2L-VRPSC, which can provide more practical and economic solutions to scenarios with customer presence uncertainty.

The remainder of this paper is organized as follows. Section 2 describes 2L-VRPSC and presents its mathematical formulation. Section 3 presents the EATS-MOBFS method for solving the 2L-VRPSC. Extensive computational results are presented in Section 4, followed by the conclusions in Section 5.

2. Mathematical Formulation of the 2L-VRPSC

2.1. Problem Description

In the 2L-VRPSC, there is a central depot, several vehicles and customers, where the presence or absence of a customer is probabilistic. To save time and resources consumed for re-routing, distribution companies usually adopt a pre-optimization strategy; namely, the distribution companies follow the pre-designed routes and directly skip the customer that does not require service and move on to the next customer. Therefore, the purpose of the 2L-VRPSC is to reasonably arrange the routes and customer service order to meet the loading constraints for the purpose of minimizing the total expected cost of the final scheduling plan.

Therein, the assumptions of the 2L-VRPSC are listed as follows: (i) the demand of every customer is known beforehand, but the presence or absence of a customer is uncertain; (ii) the geographical information of the customers and the depot is known, and all nodes in the distribution network are completely connected; and (iii) the attributes of homogeneous vehicles are known.

2.2. Stochastic Programming Model of the 2L-VRPSC

The Monte Carlo method is adopted in this study to simulate customer uncertainties, and a scenario reduction approach is applied to generate a reasonable number of scenarios with good tractability and fidelity, as well as diverse characteristics. Then a stochastic programming model is developed based on the obtained scenarios.

2.2.1. Customer Uncertainty Modeling Based on Monte Carlo Simulation

Due to the fact that the complexity of simulation grows exponentially with the number of customers, we adopt the Monte Carlo method [27,28] to generate scenarios to simulate the uncertainty of customers according to the discrete probability distribution of all customers. Specifically, for each customer, assuming that the probability that customer i requires service is p_i, a random number $u_i \in [0,1]$ is generated by using the random sampling method. If $u_i > p_i$, customer i requires service; otherwise, the customer does not need service. For each scenario, each customer is sampled to generate the probability that requires service; namely, each scenario s corresponds to a set of random variables $\{u_1, u_2, \cdots, u_i, \cdots, u_{n-1}, u_n\}_{i \in V_0}$. After a finite number of samples, a large number of scenarios are generated to simulate the uncertainty of customers.

However, for a given solution τ with n customers, the number of the scenario can be up to 2^n, and such large-scale scenarios will lead to a heavy computation burden. To balance the diversity and fidelity, the size of the scenarios is reduced via the scenario-reduction techniques according to the probability and the correlation among scenarios [29,30]. The simultaneous backward reduction technique [31] is adopted to approximate the original system with high confidence and promote the diverse characteristics of the generated scenarios, which is described in Algorithm 1 below. In Algorithm 1, terms s and $s^{'}$ are used to denote different scenarios. Specifically, the distances between all the scenarios are calculated (line 1). Here the distance between scenario s and $s^{'}$ is defined as $DT_{s,s^{'}} = \sqrt{\sum_{i=1}^{n}(a_i^s - a_i^{s^{'}})^2}, i \in V_0$, where a_i^s is a binary variable that indicates whether customer i is served in scenario s. Afterward, the probability of each pair of the nearest scenarios is calculated, and the pair with the lowest probability is selected (Algorithm 1 line 4). The scenario pair with the lowest probability will be eliminated, and the probability of all other scenarios will be updated (Algorithm 1 line 5). This process is repeated until the desired number of remaining scenarios is obtained.

Algorithm 1: Scenario Reduction.

Input: the initial set of scenarios I
Output: the preserved set of scenarios S
1 Calculate the distances between all of the scenarios.
2 **while** *the predefined size of remaining scenarios is not met* **do**
3 Find other nearest scenario s to each scenario $s^{'}$ (i.e., the distance $DT_{s,s^{'}}$ between scenario s and $s^{'}$ is smallest).
4 Compute the probability for each pair of nearest scenarios and choose the one with the lowest probability.
5 Eliminate the scenarios and update the probability of the remaining scenarios.
6 **end**

2.2.2. Formulation of the 2L-VRPSC

In this study, the 2L-CRPSC is described as a complete undirected graph $G = (V, E)$. Therein, $V = V_0 \cup 0$ consists of n customers ($V_0 = \{1, 2, \cdots, n\}$) and a depot (node 0). Each customer $i (i \in V_0)$ consists of m_i rectangular items with a total weight of q_i. Each item of customer i is denoted by a pair of indices (i, b), and its width and length are denoted by w_{ib} and $l_{ib} (b = 1, 2, \cdots, m_i)$, respectively. A fleet of K homogeneous vehicles is available, and each vehicle has a maximum capacity Q and a rectangular loading surface with width W and length L. Additional notations are described as follows:

Sets and parameters:

V_p, V_a: Set of customers that require service and do not require service ($V_p \cup V_a = V_0$, $V_p \cap V_a = \emptyset$).

S: Set of preserved scenarios, indexed by s and $s^{'}$ ($s, s^{'} \in S$).

R: Set of feasible routes, where R^k represents a feasible route visited by vehicle k.

σ_i: Parameter that indicates the order that customer i is visited along the route ($i \in R^k$).

d_{ij}: Distance (i.e., travel cost) between node i and node j ($(i, j) \in E$).

Variables:

z_{ij}^k: Binary variable, $z_{ij}^k = 1$ if and only if vehicle k travels from node i to node j, 0 otherwise.

x_{ib}, y_{ib}: Integer variables that define the coordinate of the bottom-left corner of item b belonging to customer i in a vehicle.

Ω_{ib}: Integer variable that indicates whether item b of customer i has been rotated.

Let p_i be the probability that customer i requires service (assuming independence between customers), then the probability of scenario $s \in S$ can be expressed as follows:

$$P(s) = \sum_{i \in V_p} p_i \sum_{j \in V_a} (1 - p_j). \tag{1}$$

The objective of the 2L-VRPSC problem is to find a prior solution with minimal expected length. Based on the customer uncertainty modeling method described in Section 2.2.1, we simulate solution τ to construct a desired number of scenarios so as to calculate the expected length $E(L_\tau)$, i.e.,

$$\text{Minimize } E(L_\tau) = \sum_{s \in S} P(s) L_{\tau, V_a}(s), \tag{2}$$

where $L_{\tau, V_a}(s)$ is the length of solution structure corresponding to scenario s, in which the customers V_a that do not require service are skipped. Figure 1 illustrates an example of the prior solution and scenario with nine customers. In the example, customers 2 and 7 do not require service, and thus, vehicles skip them and visit other customers in the same order as they appear in the prior solution.

In the proposed approach, the prior solution is built satisfying the following constraints:

$$\sum_{j \in V_0} z_{0jk} = \sum_{i \in V_0} z_{i0k}, \forall k \in K, \tag{3}$$

$$\sum_{k \in K} \sum_{i \in V} z_{ijk} = 1, \forall j \in V_0, \tag{4}$$

$$\sum_{i \in V} z_{iuk} = \sum_{j \in V} z_{ujk}, \forall u \in V_0, k \in K, \tag{5}$$

$$\sum_{k \in K} \sum_{j \in V_0} z_{0ik} \leq K, \tag{6}$$

$$\sum_{i:(i,j) \in E} q_i z_{ijk} \leq Q, \forall k \in K, \tag{7}$$

Constraint (3) ensures that all vehicles departing the depot have to return to the depot. Constraint (4) enforces that each customer is visited exactly once. Constraint (5) is the flow conservation constraint. Constraint (7) ensures that each vehicle is not overloaded. Constraint (6) guarantees that the maximum number of vehicles is not exceeded.

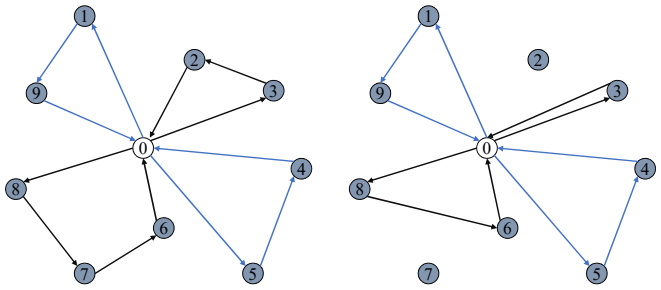

Figure 1. Example of the prior solution and scenario. (**a**) A prior solution visits nine customers. (**b**) The scenario corresponding to the prior solution, where customers 2 and 7 do not require service (disconnected from the routes).

Additionally, the packing feasibility of every route must be guaranteed. In this paper, four different two-dimensional loading configurations with respect to sequential and rotation constraints are considered. The formulation of the loading constraints that adopted from [3], are defined as follows:

$$0 \le x_{ib} \le (W - w_{ib})(1 - \Omega_{ib}) + (W - l_{ib})\Omega_{ib}, \forall i \in V_0, b \in \{1, \cdots, m_i\}, \tag{8}$$

$$0 \le y_{ib} \le (L - l_{ib})(1 - \Omega_{ib}) + (L - w_{ib})\Omega_{ib}, \forall i \in V_0, b \in \{1, \cdots, m_i\}, \tag{9}$$

$$\begin{cases} x_{ib} + w_{ib}(1 - \Omega_{ib}) + l_{ib}\Omega_{ib} \le x_{jb'}, & \text{or} \\ x_{jb'} + w_{jb'}(1 - \Omega_{jb'}) + l_{jb'}\Omega_{ib'} \le x_{ib}, \forall i, j \in V_0, b \in \{1, \cdots, m_i\}, b' \in \{1, \cdots, m_j\}, \end{cases} \tag{10}$$

$$\begin{cases} y_{ib} + l_{ib}(1 - \Omega_{ib}) + w_{ib}\Omega_{ib} \le y_{jb'}, & \text{or} \\ y_{jb'} + l_{jb'}(1 - \Omega_{jb'}) + w_{jb'}\Omega_{ib'} \le y_{ib}, \forall i, j \in V_0, b \in \{1, \cdots, m_i\}, b' \in \{1, \cdots, m_j\}, \end{cases} \tag{11}$$

$$\begin{cases} x_{ib} + w_{ib}(1 - \Omega_{ib}) + l_{ib}\Omega_{ib} \le x_{jb'}, & \text{or} \\ y_{jb'} + l_{jb'}(1 - \Omega_{jb'}) + w_{jb'}\Omega_{jb'} \le y_{ib}, & \text{or} \\ x_{jb'} + w_{jb'}(1 - \Omega_{jb'}) + l_{jb'}\Omega_{jb'} \le x_{ib}, & \\ \forall i, j \in R^k : \sigma(i) < \sigma(j), b \in \{1, \cdots, m_i\}, b' \in \{1, \cdots, m_j\}. & \end{cases} \tag{12}$$

Constraints (8) and (9) enforce that each item must be placed in the loading space. Constraints (10) and (11) enforce that any two items to be transported by the same vehicle do not overlap. The first-in-last-out (FILO) requirements are modeled by constraint set (12).

3. The Proposed EATS-MOBFS Method for the 2L-VRPSC

3.1. Framework of the Solution Mechanism

With the proven superior performance of the tabu search algorithm in a broad range of optimization problems [32,33] and the fact that some packing heuristics have shown high efficiency for solving large-scale two-dimensional packing problems, an EATS-MOBFS is proposed in this study for efficiently solving the 2L-VRPSC, where the EATS searches for the optimal routing combination while the MOBFS checks the feasibility of each route and guides the EATS searches to feasible solutions. To deal with the stochastic nature of the 2L-VRPSC, Monte Carlo simulation and scenario analysis methods are integrated for the solution-searching process of the algorithm to calculate the total expected transportation cost.

The framework of the solution approach is shown in Algorithm 2. Specifically, the solution approach first constructs a feasible initial solution (Algorithm 2 line 1) and iteratively improves the initial solution by performing a non-tabu move or a move that satisfies the aspiration criterion (i.e., better than an incumbent best solution) in each iteration. In each iteration, a neighborhood operator is selected based on the self-adaptive mechanism (Algorithm 2 line 6). Therein, each operator is associated with a weight and is selected

periodically and probabilistically according to its performance in the previous iterations (p_u iterations). After p_u iterations (i.e., inner-layer iterations), a diversification mechanism based on the remove-reinsert strategy inspired by large neighborhood search [34] is called to avoid the algorithm being trapped at a local optimum (Algorithm 2 line 20). Moreover, Monte Carlo simulation is called to estimate the expected transportation cost in lines 2 and 11, and MOBFS is evoked to examine the loading feasibility of the solution in lines 1 and 9. In the following, the key components of the solution approach will be detailed, including initial solution construction in Section 3.1.1, neighborhood operators in Section 3.1.2, adaptive weight adjustment in Section 3.1.3, diversification mechanism in Section 3.1.4 and the proposed MOBFS in Section 3.2.

Algorithm 2: The proposed solution approach for the 2L-VRPSC.

```
   Input: Dataset of the 2L-VRPSC
   Output: Solution of the 2L-VRPSC X*
1  Construct an initial solution X_0, set X = X0, X* = X_0. (Section 3.1.1)
2  Calculate the expected transportation cost by Monte Carlo simulation.
3  while maximum out-layer iterations are not met do
4      while maximum inner-layer iterations are not met do
5          Select a neighborhood operator according to their possibilities.
6          Create a candidate solution list by the selected neighborhood operator.
            (Section 3.1.2)
7          Find the best solution X' from the candidate list.
8          if capacity constraint is satisfied then
9              Check the loading feasibility of the solution. (Section 3.2)
10             if loading constraints is satisfied then
11                 Calculate the expected transportation cost of X' (denoted by
                    Ob(X')) by Monte Carlo simulation.
12                 if Ob(X')<Ob(X) then
13                     Update the current solution, set X = X'.
14                 end
15             end
16         end
17         Update the tabu list.
18     end
19     Update the weights of neighborhood operators. (Section 3.1.3)
20     Call the diversification mechanism to generate new solution. (Section 3.1.4)
21 end
```

3.1.1. Initial Solution Construction

In this study, an insertion-based mechanism is utilized to construct the feasible initial solution. Algorithm 3 provides the pseudocode for the initialization process. Specifically, the customers are first sorted in descending order based on the total area of their items, and K empty routes are generated (Algorithm 3 line 1). Afterward, each customer i is sequentially assigned to the route that has the smallest unoccupied loading surface that does not violate the loading constraints (i.e., can accommodate the items of customer i) (Algorithm 3 line 4–5). If a feasible route is found, the customer is inserted into the position of this route with the minimum incremental cost. Otherwise, if customer i cannot be inserted into any of the K routes, the position of customer i and a randomly selected customer will be exchanged, and the former procedure is restarted under the new customer sequence (Algorithm 3 line 7).

Algorithm 3: Initial Solution Generator.

Input: Set of customers, set of vehicles K
Output: Initial solution

```
Sort the unremoved customers in descending order of the total area of its items.
while customers are left uninserted do
    for customer i to n do
        Sort the vehicles in descending order according to the area of unoccupied
          loading surface.
        Insert the customers into this route in the position that leads to the
          minimum incremental cost.
        if route is not feasible then
            Exchange the position of customer i and another randomly selected
              customer.
        else
            i = i + 1
            Update the unoccupied loading surface of each vehicle.
        end
    end
end
```

3.1.2. Neighborhood Operators

It is widely believed that the performance of the tabu search (TS) algorithm is greatly dependent on the neighborhood operators. To overcome the dependence of the algorithm on a single neighborhood, six neighborhood operators derived from the segment exchange (SE) [9] and move combination (MC) [9] are adopted. It is noteworthy that the loading feasibility of each route must be checked by invoking the proposed MOBFS heuristic in each iteration. During each search for candidate solutions, infeasible operations, which violate the two-dimensional loading constraints or vehicle capacity, are not allowed.

Segment exchange (SE): This operator first selects two segments of customers (i.e., a sequence of customers) from different randomly chosen routes and then exchanges the positions of two segments of customers. Note that if the resulting solution is infeasible, the exchange is canceled. Moreover, considering the weights of items and loading constraints, a longer segment leads to a lower probability with, which this segment can have a successful exchange. To increase the probability of obtaining a feasible solution, the exchange is only executed between segments of the same length. Based on these considerations, the segment length is limited to two and three customers, and the corresponding neighborhood operators are denoted as **SE2** and **SE3**, respectively.

Move combination (MC): This operator randomly selects two types from the six move types presented in Figure 2, which is marked as **MC2**. For each selected type, it repeatedly tries to execute a random move until the new resulting solution obtained by this move is feasible. The detailed move types are described as follows, which correspond to Figure 2a–f, respectively. More details about the move types can be found in [9,12,35].

Intra-swap [35]: This operator tries to swap the positions of two customers selected from the same route.

Inter-swap [9]: This neighborhood operator swaps the positions of two customers selected from different routes.

Intra-shift [35]: For this operator, one customer is selected and removed from the current route and then inserted into another position of this route.

Inter-shift [35]: For this operator, one customer is selected and removed from the current route and then inserted into another route.

Intra-2opt [9]: For a given route, two non-adjacent segments of customers are deleted, the middle segment of the route is reversed, and then two new segments of customers are generated and added to re-connect the route.

Inter-2opt [9]: Two segments of customers from two different routes are eliminated so that each route is divided into two parts; then, the first part of each route is concatenated with the second part of the other route to generate two new routes. If a route is empty, the other route is split into two new routes.

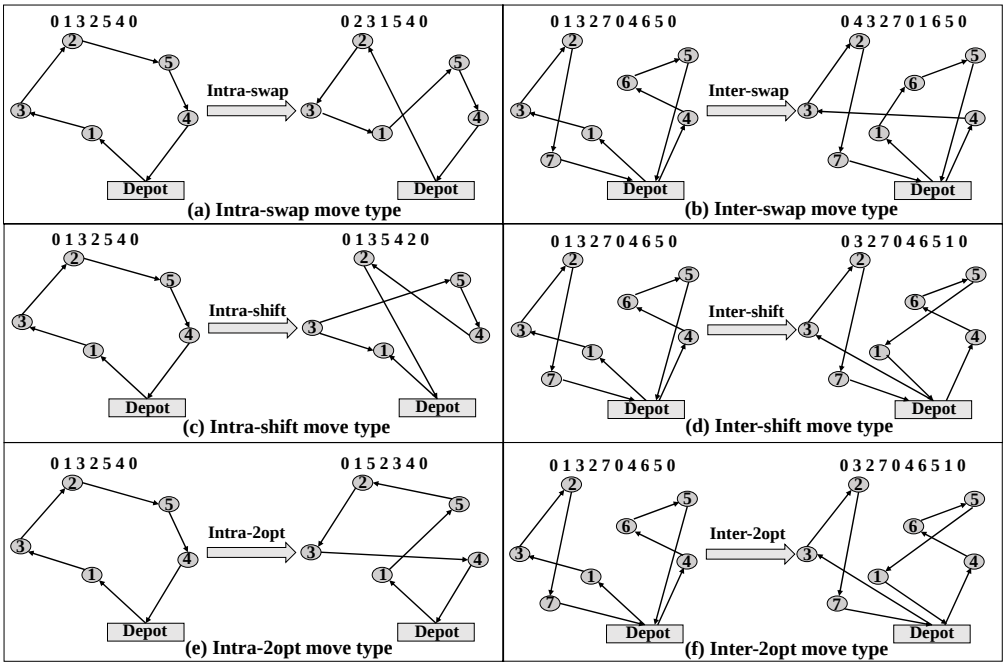

Figure 2. Move types used in the EATS algorithm.

Specifically, under the 2|UO|L and 2|UR|L settings, switching the service order of customers within the same route will not result in loading-infeasible solutions since the FILO constraints are not considered, and thus intra-route moves do not need to examine the loading feasibility. While the sequential constraints are imposed, the feasibility of the resulting solutions constructed by the intra-route and inter-route moves have to be examined, and only the solution satisfying the loading constraints can be accepted during the process.

In order to balance the computation burden and quality of solutions, the EATS algorithm performs each individual operator one or two times. As a result, six neighborhood operators, **SE21** (perform **SE2** once), **SE22** (perform **SE2** twice), **SE31**, **SE32**, **MC21** and **MC22**, are derived.

3.1.3. Adaptive Weight Adjustment

In this study, each neighborhood operator is assigned a weight, and a self-adaptive mechanism is utilized to select neighborhood operators periodically. The weight of each neighborhood operator is based on the score, which measures how well the neighborhood operator has performed in the previous iterations (predefined by p_u iterations). After p_u iterations, the weight of each neighborhood operator is initially identical and calculated as follows:

$$\omega(i) = \begin{cases} (1-r)\omega(i), & \text{if } \theta(i)) = 0 \\ (1-r)\omega(i) + r\frac{\pi(i)}{\theta(i)}, & \text{otherwise} \end{cases} \tag{13}$$

where the reaction factor r controls the influence of the recent success of an operator on its weight. $\theta(i)$ denotes the number of times the neighborhood operator i has been used during the p_u iterations. $\pi(i)$ is the score of neighborhood operator i, which is initially set to zero and updated according to its performance during each iteration as follows: (i) increase δ_1 if the chosen neighborhood operator generates a new solution that is better than the

current best solution; (ii) increase δ_2 if the chosen neighborhood operator generates a better solution compared to the current one.

3.1.4. Diversification Mechanisms

The main advantage of tabu search is that it can avoid revisiting recent solutions via the tabu list, while it emphasizes relatively small and local neighborhoods [36]. Based on this consideration, a diversification mechanism is integrated to enhance the exploration capability of EATS, which is inspired by the remove-reinsert strategy in the large neighborhood search [34]. The basic idea of the remove-reinsert method is to reconstruct the obtained local optimum solution by ruining the incumbent best solution first and then reconstructing a new feasible solution. First, several similar customers are removed from the solution according to the Shaw removal heuristic [37], expecting to create a new, perhaps better, solution by the reinsertion process. To quantify the similarity of two customers, a relatedness indicator $\widetilde{r}(i,j)$ is proposed as follows:

$$\widetilde{r}(i,j) = \alpha_1\left(\frac{d_{ij}}{d_{max}}\right) + \alpha_2\left|q_i - q_j\right| + \alpha_3\left(1 - \frac{\left|K_i \cap K_j\right|}{\min\left\{\left|K_i\right|, \left|K_j\right|\right\}}\right) \tag{14}$$

where d_{max} denotes the maximal distance over all d_{ij} $(i, j \in N)$, which is used for standardizing d_{ij}. Term K_i denotes the set of vehicles able to serve customer i, and $|K_i|$ denotes the number of vehicles in set K_i. Terms $\alpha_1, \alpha_2, \alpha_3$ are weight coefficients.

As shown in (14), the relatedness indicator consists of three aspects: (i) the distance between customers, (ii) the difference in demands between customers, and (iii) the difference in the number of vehicles that can serve customers.

In the reinserting process, the customers that have been removed from the solution are first sorted in descending order according to the total area of their items and reinserted into the partial solution one by one. The route and position of a customer are decided by the same insertion-based rule used in the initial solution construction (Section 3.1.1). When a customer cannot be inserted into any route, other customers in a partial solution are randomly erased one by one until the given customer can be inserted into the partial solution successfully. Then the mechanism restarts the reinserting process with the erased customers. The reinserting process is repeated until a complete solution is found. The pseudocode Algorithm 4 for the removal process is shown as follows. The removal process initially selects a customer i randomly (Algorithm 4 line 1). Then it repeatedly chooses an already selected customer, j, and selects a new customer that is most related to i (Algorithm 4 line 4–7). The process stops when ρ customers have been chosen. Moreover, a determination parameter ϕ ($\phi \geq 1$) (Algorithm 4 line 7) introduces some randomness into the request selection process (a high value of ϕ corresponds to low randomness).

Algorithm 4: Removal Operation.

Input: Destruction degree ρ, integer ϕ and current solution X
Output: The set of customers removed (R_m)

1 Randomly select a customer i to remove from X, $R_m = \{i\}$.
2 **while** $|R_m| < \rho$ **do**
3 Select a request randomly from R_m.
4 R_l be the requests list sorted in descending order of relatedness.
5 Generate a random number a from the iternal $[0, 1)$.
6 Remove $j = R_l\left[|R_l| \cdot a^{\phi}\right]$ from X.
7 $R_m = R_m \cup \{j\}$.
8 **end**

3.2. The Proposed MOBFS Method for Two-Dimensional Loading

In this study, we investigated four classes of the 2L-VRPSC with respect to different loading configurations. A MOBFS packing heuristic is proposed, which is extended from a

bottom-left-fill algorithm and a skyline-based heuristic. The MOBFS method considers not only the area of items but also the length and width of the items as the sorting criterion to generate the loading sequence (Algorithm 5 line 1). When sequential loading constraints are imposed, we will first sort all the items by the reverse visiting order of its customers and then use the above sorting rules to sort the rectangles of each customer (Algorithm 5 line 3). Thereafter, these two heuristics are called in sequence (Algorithm 5 line 4). Specifically, the sophisticated skyline-based heuristic will be called only if the BLF fails to produce a feasible loading solution. If neither heuristic can find a feasible packing solution, a tabu search procedure will be called by swapping two non-tabu items to generate a new sequence (Algorithm 5 line 9). The pseudocode of the presented MOBFS heuristic for two-dimensional loading is shown in Algorithm 5. Detailed improvements in the packing heuristics are described in the following subsections.

Algorithm 5: The proposed MOBFS algorithm for two-dimensional loading.

```
   Input: Route R and corresponding items
   Output: Packing sequence
1  Sort the items to generate three orderings Ord_1, Ord_2, Ord_3.
2  for each ordering Ord_i of the three orderings do
3      Let I_t be the sequence of all rectangular items and |I_t| be the number of items.
4      for j = 1 to 2 do
5          if Heur_j(I_t) places all items then
6              return success
7          else
8              for k = 1 to MaxIter do
9                  Generate |I_t| non-tabu sequences from I_t by swapping two items.
10                 Let I_t^* be the sequence with the highest area utility using Heur_j.
11                 if Heur_j(I_t^*) places all items then
12                     return success
13                 else
14                     I_t = I_t^* and update the tabu list.
15                 end
16             end
17         end
18     end
19 end
```

3.2.1. Improved Bottom-Left-Fill Heuristic

The first heuristic that we use is the modified classical bottom-left-fill (BLF) [38], obtained by generalizing this method to the four classes of the two-dimensional loading configurations. In the BLF heuristic, it maintains a list of locations to indicate where the items may be placed from the bottom left. Each sorted item is then placed in the lowest and leftmost position available that satisfies all constraints.

As shown in Figure 3, the numbers indicate the locations where item 5 may be placed. When item 5 is placed, it may be placed under item 4 according to the BLF method, which violates the sequential loading constraint that may be required in the unloading process. To overcome this drawback, we enhance the BLF by introducing a covering strategy, which enforces the FILO constraint without undermining its high performance in the packing process. Specifically, after each item is placed into the truck, the covering operations, as shown in Figure 4, is implemented first to form a new virtual item. Thereafter, the list of locations is updated, and the process of placing the items in the specified order is repeated, which is described as follows:

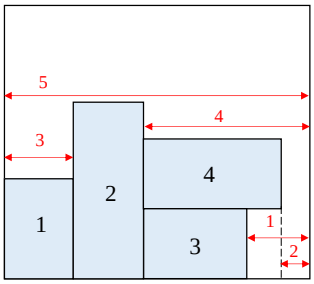

Figure 3. An example of violating the sequential loading constraint when using the BLF.

If the next placed item i completely covers the placed item k, then items i and k are combined into one article, as article i, the information belonging to item k becomes 0. The length of article i is equal to item i, while the height is equal to the sum of items 3 and 4. As shown in Figure 4, item 3 is completely covered by item 4, so items 3 and 4 are combined into article 4.

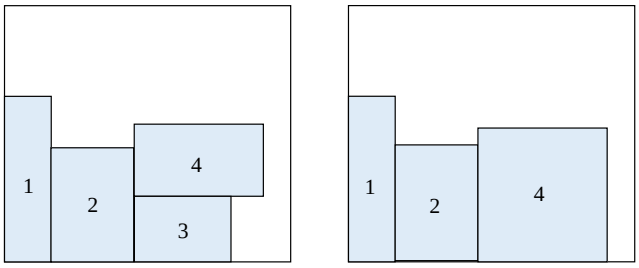

Figure 4. An example of complete covering.

As such, the improved BLF heuristic is capable of handling four classes of loading configurations. Under the 2|UO|L and 2|SO|L settings, we only check the placement of an item in a position without rotation, while the covering strategy is implemented under the 2|SO|L setting. Under the 2|UR|L and 2|SR|L settings, the placement of an item in a position is checked first with the item's original orientation and the placement with a 90° item rotation will be checked only if the attempted placement turns out to be infeasible. Moreover, the covering strategy is implemented only under the 2|SO|L and 2|SR|L settings.

3.2.2. Improved Skyline-Based Heuristic

The skyline-based heuristic was first proposed to solve the two-dimensional rectangle packing problem, and the key is selecting the position of the items on the loading surface. A set of selection criteria were proposed for this purpose, and more details can be found in [9].

One of the selection criteria is selecting the placement with minimum local waste at every stage of the packing process. However, this wasted space measure is only a measure of the actual number of wasted space segments, ignoring a measure of the total area of wasted space segments. Therefore, we further quantify the wasted area and propose the wasted-area-based selection criterion to determine the positions of items, i.e., if there are multiple candidate positions with the same number of wasted space segments, the algorithm will select the position with the minimum total wasted area. It is motivated by the assumption that if the area of wasted space is minimized at every stage of the packing process, the remaining unplaced items will be more likely to be placed. The four types of wasted space are given in Figure 5a–d.

Let w_{min} and h_{min} be the minimum width and height of the remaining rectangular items, respectively. The pattern area in case (a) is always considered wasted space. The pattern areas in cases (b) and (c) are considered wasted if the length of the gap is less than w_{min}. The pattern area in case (d) is wasted if the gap is less than h_{min}.

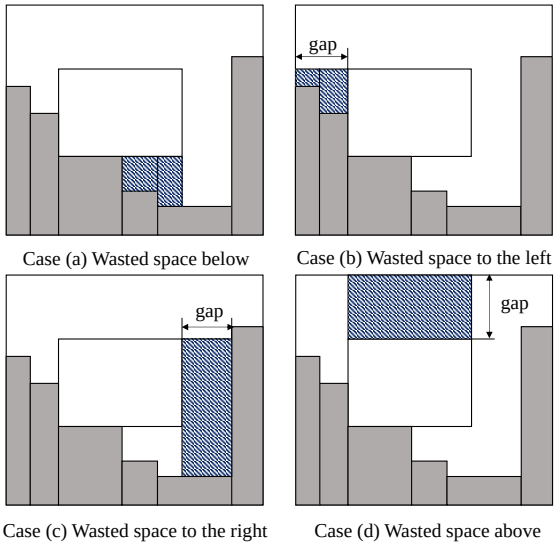

Figure 5. Illustration of four types of wasted space (pattern areas are the wasted space).

Based on the above improvements, we apply the skyline-based heuristic to the four classes of two-dimensional loading configurations. Under the 2|UO|L and 2|SO|L settings in which item rotation is not allowed, it only evaluates the feasibility and area of the wasted space for each pair item-position without rotation. Under the 2|UR|L and 2|SR|L settings, it evaluates the feasibility, wasted area of rotation and non-rotation to decide the placement position and whether to rotate so as to improve the utilization of the wasted space in the packing process.

4. Computational Experiments

The experiments and experimental results based on a number of widely used 2L-VRP benchmarks and 2L-VRPSC instances are presented, which are compared with a few existing methods. The algorithm is coded in MATLAB 2018b, and all experiments are executed on a desktop PC with Intel Core i5-5200U(2.20GHz) and 8 gigabytes of RAM running on a Windows 7 operating system.

4.1. Testing Instances and Parameters

The 2L-VRP benchmarks generated in [4] incorporate a fleet of vehicles with a 40 × 20 squared units for loading and a maximum capacity of weight. According to the characteristics of the demanded items, the instances of 2L-VRP are categorized into 5 classes, as shown in Table 1. Table 2 shows the details of the number of customers (denoted by n), total number of items (denoted by $M = \sum_{i=1}^{n} m_i$), and the number of vehicles (denoted by $|K|$). More details can be found in Ref [3], and all 2L-VRP instances can be downloaded from http://www.or.deis.unibo.it/research.html, accessed on 10 May 2020.

Table 1. Characteristics of the demanded items of Classes 1–5 instances.

Class	$\|m_i\|$	Vertical		Homogeneous		Horizontal	
		Length	Width	Length	Width	Length	Width
1	1	1	1	1	1	1	1
2	[1,2]	[0.4 L,0.9 L]	[0.1 W,0.2 W]	[0.2 L,0.5 L]	[0.2 W,0.5 W]	[0.1 L,0.2 L]	[0.4 W,0.9 W]
3	[1,3]	[0.3 L,0.8 L]	[0.1 W,0.2 W]	[0.2 L,0.4 L]	[0.2 W,0.4 W]	[0.1 L,0.2 L]	[0.3 W,0.8 W]
4	[1,4]	[0.2 L,0.7 L]	[0.1 W,0.2 W]	[0.1 L,0.4 L]	[0.1 W,0.4 W]	[0.1 L,0.2 L]	[0.2 W,0.7 W]
5	[1,5]	[0.1 L,0.6 L]	[0.1 W,0.2 W]	[0.1 L,0.3 L]	[0.1 W,0.3 W]	[0.1 L,0.2 L]	[0.1 W,0.6 W]

Table 2. Details of Classes 1–5 instances.

Instance	n	Class 1		Class 2		Class 3		Class 4		Class 5	
		M	$\|K\|$	M	$\|K\|$	M	$\|K\|$	M	$\|K\|$	M	$\|K\|$
1	15	15	3	24	3	31	3	37	4	45	4
2	15	15	5	25	5	31	5	40	5	48	5
3	20	20	4	29	5	46	5	44	5	49	5
4	20	20	6	32	6	43	6	50	6	62	6
5	21	21	4	31	4	37	4	41	4	57	5
6	21	21	6	33	6	40	6	57	6	56	6
7	22	22	3	32	5	41	5	51	5	55	6
8	22	22	5	29	5	42	5	48	5	52	6
9	25	25	8	40	8	61	8	63	8	91	8
10	29	29	3	43	6	49	6	72	7	86	7
11	29	29	4	43	6	62	7	74	7	91	7
12	30	30	9	50	9	56	9	82	9	101	9
13	32	32	3	44	7	56	7	78	7	102	8
14	32	32	4	47	7	57	7	65	7	87	8
15	32	32	5	48	6	59	6	84	8	114	8
16	35	35	11	56	11	74	11	93	11	114	11
17	40	40	14	60	14	73	14	96	14	127	14
18	44	44	4	66	9	87	10	112	10	122	10
19	50	50	5	82	11	103	11	134	12	157	12
20	71	71	4	104	14	151	15	178	16	226	16

We run the proposed EATS-MOBFS method 10 times for each instance with the same parameters. Therein, the parameters of the proposed EATS-MOBFS algorithm are determined using the Irace package presented in [39]. The irace package is a general-purpose configurator and only requires the definition of a configuration scenario. Given a set of representative instances and a set of possible values for each parameter, the Irace package determines an appropriate combination of values for the parameter configuration. The Irace package iteratively applies a racing procedure in which several configurations are incrementally evaluated on bigger subsets of the training instance set. Specifically, the Irace package uses an iterative racing method or automatic configuration consisting of three steps: (i) sampling new configurations according to a particular distribution, (ii) selecting the best configurations from the newly sampled configurations by means of racing and (iii) updating the sampling distribution to drive the sampling toward the best configuration. These steps are repeated until the termination criterion is met. More details about the Irace package can be found in [39].

For the Irace scenario configuration, instances with 15, 30, 50 and 71 customers are taken as representative instances for parameter tuning. Each parameter of the EATS is provided with a set of possible values in a specified range for tuning. Namely, δ_1 and δ_2 $(\delta_1 > \delta_2)$ range from 0 to 50. Parameters r and μ are set from 0 to 1. Parameters α_1, α_2 and α_3 range from 0 to 10. Afterward, a parameter combination is selected based on the best average result for the testing instances. Specifically, the weight adjustment algorithm is controlled by three parameters, δ_1, δ_2 and r, which are set to 33, 13 and 0.25, respectively. In the diversification procedure, the removal process is controlled by three parameters: α_1, α_2 and α_3, which are 9, 2 and 5, respectively. Parameter μ is set to 0.4 for controlling the maximum number of customers that can be removed in the diversification procedure, and a random number ρ, $1 \leq \rho \leq \min(15, \mu \cdot n)$ determines the exact number of removed customers, where n is the total number of customers. In addition, the inner-layer iterations are set to 50; the outer-layer iterations are set to 5000.

4.2. Computational Results

4.2.1. Results on the 2L-VRP Benchmark Instances

Moreover, four classes of 2L-VRP are tested and compared with existing methods, as shown in Tables 3–8. Tables 4–7 present the results obtained by our method under the 2|UO|L, 2|SO|L, 2|UR|L, 2|SR|L settings, respectively. Therein, "Gap" denotes the percentage gap between the Cost and BKS (best-known solution) (Gap(%) = 100 × (Cost-BKS)/BKS).

Table 3. Comparison results for the 2L-VRP under the 2|UO|L and 2|SO|L settings (average over Classes 2–5). Other state-of-the-art approaches: GRASP [40], PRMP [8], EGTS+LBFH [35], BR-LNS [41] and LS [42].

Inst.	UO				SO			
	GRASP	EGTS+LBFH	LS	EATS-MOBFS	EGTS+LBFH	PRMP	BR-LNS	EATS-MOBFS
1	282.68	295.46	281.23	281.84	303.40	287.09	291.43	288.61
2	343.56	341.89	339.26	339.26	345.23	344.21	344.21	344.21
3	378.07	379.41	374.07	377.35	387.89	381.40	381.40	381.40
4	435.00	440.85	435.01	435.01	443.25	439.97	439.97	441.11
5	380.63	382.23	379.03	379.03	387.61	382.39	382.39	382.85
6	497.62	498.97	496.90	498.83	502.25	499.48	499.48	500.55
7	679.43	699.29	690.68	697.36	715.54	702.27	701.63	702.27
8	694.81	701.77	679.34	678.84	716.36	699.55	702.85	702.87
9	612.01	614.67	612.01	612.98	621.23	614.54	615.94	619.75
10	657.29	705.04	676.73	681.46	731.70	688.48	687.55	688.63
11	709.27	731.41	703.64	721.83	762.83	725.83	724.43	723.47
12	611.12	617.47	611.26	615.21	622.35	614.52	614.52	618.09
13	2457.79	2581.41	2491.18	2527.36	2647.88	2554.93	2556.06	2572.99
14	973.96	1030.50	974.76	985.49	1075.04	1027.38	1022.76	1024.40
15	1180.04	1194.71	1130.36	1128.18	1223.19	1189.97	1185.19	1196.97
16	699.80	702.46	699.79	699.79	703.74	701.02	702.03	702.03
17	861.79	862.63	864.06	864.47	869.93	864.06	865.73	867.13
18	1048.33	1065.94	1031.11	1048.48	1096.57	1057.99	1062.15	1073.18
19	743.50	771.57	740.66	753.67	798.2	766.05	763.75	777.73
20	511.72	545.82	512.84	522.11	559.17	534.87	535.055	541.33

Table 3 compares the best average cost of the classes (Classes 2–5) for each test instance (Instance 1–20) against the previous approaches, which shows our approach is capable of yielding comparable solutions under the 2|UO|L and 2|SO|L settings. Under the 2|UO|L setting, the overall performance of the proposed EATS-MOBFS method slightly outperforms those of GRASP × ELS and EGTS + LBFH. In comparison to GRASP × ELS and EGTS + LBFH, the proposed approach is able to produce better solutions for 6, 19 out of 20 instances, respectively. Under the 2|SO|L setting, our approach is superior to that of EGTS + LBFH and has gained comparable results compared with those obtained by PRMP and BR-LNS. Specifically, our approach can yield better solutions for 20 out of 20 instances compared with EGTS + LBFH.

As shown in Table 4, the best results were obtained by our approach on some instances of Classes 2–5 under the 2|UO|L setting, which shows the proposed EATS-MOBFS method has reached the best-known solutions for 9, 8, 8 and 10 instances in Classes 2–5, respectively. In addition, the average percentage gaps between the solutions obtained by the proposed EATS-MOBFS method and the best-known solutions (BKS) are 0.72%, 1.01%, 0.91% and 0.88% for the instances in Classes 2–5, respectively. It infers that the proposed hybrid heuristic incorporating EATS and MOBFS is efficient for solving the 2L-VRP under the 2|UO|L setting. Moreover, Table 5 presents the best results obtained by our method on some instances of Classes 2–5 under the 2|UR|L setting, which shows the proposed EATS-MOBFS method has reached the best-known solutions for 8, 10, 7 and 10 testing instances in Classes 2–5, respectively. In addition, the average percentage gaps between the solutions obtained by our method and BKS are 0.56%, 0.50%, 0.54% and 0.54% for the instances in Classes 2–5, respectively. It infers that the proposed hybrid heuristic is efficient for solving the 2L-VRP under the 2|UR|L setting.

Table 4. Result for 2|UO|L 2L-VRP (Classes 2–5). "BKS" denotes the best-known solution found among Refs [8,9,12,42].

Inst.	Class 2			Class 3			Class 4			Class 5		
	BKS	Cost	Gap	BKS	Cost	Gap	BKS	Cost	Gap	BKS	Cost	Gap
1	278.73	278.73	**0.00**	284.52	286.95	0.85	282.95	282.95	**0.00**	278.73	278.73	**0.00**
2	334.96	334.96	**0.00**	352.16	352.16	**0.00**	334.96	334.96	**0.00**	334.96	334.96	**0.00**
3	387.70	387.70	**0.00**	394.72	394.72	**0.00**	362.41	368.56	1.70	358.40	358.40	**0.00**
4	430.89	430.89	**0.00**	430.89	430.89	**0.00**	447.37	447.37	**0.00**	430.88	430.88	**0.00**
5	375.28	375.28	**0.00**	381.69	381.69	**0.00**	383.88	383.88	**0.00**	375.28	375.28	**0.00**
6	495.85	499.08	0.65	497.17	499.08	0.38	498.32	498.32	**0.00**	495.75	498.85	0.62
7	725.46	725.46	**0.00**	678.75	701.08	3.29	700.72	702.45	0.25	657.77	660.44	0.41
8	674.55	674.55	**0.00**	738.43	738.43	**0.00**	692.47	692.47	**0.00**	609.90	609.90	**0.00**
9	607.65	611.49	0.63	607.65	607.65	**0.00**	621.23	625.13	0.63	607.65	607.65	**0.00**
10	689.68	689.68	**0.00**	615.68	620.33	0.76	710.87	724.77	1.96	678.66	691.04	1.82
11	684.21	711.08	3.93	706.73	723.00	2.30	784.88	816.45	4.02	624.82	636.77	1.91
12	610.57	614.24	0.60	610.00	622.15	1.99	614.24	614.24	**0.00**	610.00	610.23	0.04
13	2585.72	2588.81	0.12	2436.56	2477.97	1.70	2548.06	2607.66	2.34	2334.78	2434.99	4.29
14	1038.09	1038.20	0.01	996.25	1016.17	2.00	981.00	985.01	0.41	871.22	902.58	3.60
15	1013.29	1017.95	0.46	1149.99	1149.99	**0.00**	1181.30	1184.85	0.30	1159.94	1159.94	**0.00**
16	698.61	698.61	**0.00**	698.61	698.61	**0.00**	703.35	703.35	**0.00**	698.61	698.61	**0.00**
17	863.66	870.86	0.83	861.79	862.62	0.10	861.79	862.62	0.10	861.79	861.79	**0.00**
18	1004.99	1038.77	3.36	1069.45	1081.44	1.12	1116.45	1147.35	2.77	917.94	926.34	0.92
19	754.53	770.96	2.18	771.66	786.43	1.91	775.87	801.25	3.27	644.59	656.04	1.78
20	517.06	525.75	1.68	521.31	541.47	3.87	537.56	540.34	0.52	470.33	480.89	2.24
Ave	738.57	744.34	0.72	740.20	748.64	1.01	756.98	766.20	0.91	701.10	710.72	0.88

Table 5. Result for 2|UR|L 2L-VRP (Classes 2–5). "BKS" denotes the best-known solution found among Refs [10–12,42].

Inst.	Class 2			Class 3			Class 4			Class 5		
	BKS	Cost	Gap	BKS	Cost	Gap	BKS	Cost	Gap	BKS	Cost	Gap
1	278.73	278.73	**0.00**	282.95	282.95	**0.00**	282.95	282.95	**0.00**	278.73	278.73	**0.00**
2	334.96	334.96	**0.00**	352.16	352.16	**0.00**	334.96	334.96	**0.00**	334.96	334.96	**0.00**
3	380.35	380.35	**0.00**	385.32	385.59	0.07	358.40	358.80	0.11	358.40	358.40	**0.00**
4	430.89	430.89	**0.00**	430.89	430.89	**0.00**	447.37	447.37	**0.00**	430.89	430.89	**0.00**
5	375.28	375.28	**0.00**	379.94	381.69	0.46	383.88	383.88	**0.00**	375.28	375.28	**0.00**
6	495.85	495.85	**0.00**	498.16	498.16	**0.00**	498.32	498.32	**0.00**	495.85	495.85	**0.00**
7	715.02	716.82	0.25	664.96	664.96	**0.00**	686.26	686.26	**0.00**	657.77	660.37	0.40
8	665.17	674.20	1.36	738.43	741.12	0.36	688.32	692.47	0.60	609.90	619.18	1.52
9	607.65	611.49	0.63	607.65	615.57	1.30	625.13	625.13	**0.00**	607.65	607.65	**0.00**
10	667.42	675.21	1.17	591.16	615.36	4.09	703.64	703.73	0.01	678.62	699.01	3.00
11	664.48	667.87	0.51	699.35	699.35	**0.00**	771.93	777.14	0.67	624.82	624.82	**0.00**
12	610.00	610.00	**0.00**	610.00	610.00	**0.00**	610.23	614.23	0.66	610.09	610.24	0.02
13	2502.65	2504.53	0.08	2377.39	2379.83	0.10	2500.85	2539.44	1.54	2334.59	2334.99	0.02
14	1029.34	1042.38	1.27	988.79	988.79	**0.00**	955.09	981.00	2.71	871.22	875.07	0.44
15	1001.51	1018.77	1.72	1116.07	1120.75	0.42	1164.63	1168.60	0.34	1159.94	1181.70	1.88
16	698.61	698.61	**0.00**	698.61	698.61	**0.00**	703.35	708.20	0.69	698.61	698.61	**0.00**
17	861.79	874.63	1.49	861.79	861.79	**0.00**	861.79	863.79	0.23	861.79	861.79	**0.00**
18	982.44	988.61	0.63	986.30	999.27	1.32	1100.52	1107.94	0.67	917.94	927.98	1.09
19	711.97	726.51	2.04	749.43	753.66	0.56	747.03	765.51	2.47	644.59	651.97	1.14
20	488.69	489.23	0.11	511.46	518.05	1.29	533.77	534.14	0.07	466.79	472.77	1.28
Ave	725.14	729.75	0.56	726.54	729.93	0.50	747.92	753.69	0.54	700.92	705.01	0.54

Table 6. Results for 2|SO|L 2L-VRP (Classes 2–5). "BKS" denotes the best-known solution found among Refs [8,9,12,41].

Inst.	Class 2			Class 3			Class 4			Class 5		
	BKS	Cost	Gap	BKS	Cost	Gap	BKS	Cost	Gap	BKS	Cost	Gap
1	290.84	290.84	**0.00**	284.52	286.26	0.61	294.25	296.75	0.85	278.73	280.60	0.67
2	347.73	347.73	**0.00**	352.16	352.16	**0.00**	342.00	342.00	**0.00**	334.96	334.96	**0.00**
3	403.93	403.93	**0.00**	394.72	394.72	**0.00**	368.56	368.56	**0.00**	358.40	358.40	**0.00**
4	440.94	440.94	**0.00**	440.68	445.25	1.04	447.37	447.37	**0.00**	430.89	430.89	**0.00**
5	388.72	388.72	**0.00**	381.69	383.52	0.48	383.88	383.88	**0.00**	375.28	375.28	**0.00**
6	499.08	499.08	**0.00**	504.68	508.61	0.78	498.32	498.65	0.07	495.85	495.85	**0.00**
7	734.65	734.65	**0.00**	702.59	709.72	1.01	703.49	703.49	**0.00**	658.64	661.22	0.39
8	725.91	725.91	**0.00**	741.12	741.12	**0.00**	697.92	705.28	1.05	621.85	639.18	2.79
9	611.49	611.49	**0.00**	613.90	631.37	2.85	625.10	628.48	0.54	607.65	607.65	**0.00**
10	700.20	700.20	**0.00**	628.94	637.46	1.35	715.82	717.83	0.28	690.96	699.05	1.17
11	721.54	723.34	0.25	717.37	718.09	0.10	811.56	815.68	0.51	624.82	636.77	1.91
12	619.63	619.63	**0.00**	610.00	610.23	0.04	618.23	628.25	1.62	610.23	614.24	0.66
13	2669.39	2669.39	**0.00**	2486.44	2497.42	0.44	2609.36	2690.15	3.10	2416.04	2434.99	0.78
14	1090.55	1101.61	1.01	1039.06	1069.43	2.92	982.25	1002.28	2.04	922.75	924.27	0.16
15	1041.75	1112.54	6.80	1181.68	1197.56	1.34	1246.49	1247.39	0.07	1229.95	1230.37	0.03
16	698.61	698.61	**0.00**	698.61	698.61	**0.00**	708.20	712.30	0.58	698.61	698.61	**0.00**
17	870.86	875.44	0.53	861.79	867.85	0.70	861.79	862.62	0.10	861.79	862.62	0.10
18	1053.09	1059.44	0.60	1102.17	1123.16	1.90	1134.11	1163.87	2.62	926.34	946.27	2.15
19	792.42	801.90	1.20	801.13	802.65	0.19	801.21	824.15	2.86	652.15	682.20	4.61
20	545.68	553.12	1.36	541.58	553.50	2.20	551.61	567.89	2.95	478.15	490.80	2.65
Ave	762.35	767.93	0.59	754.24	761.43	0.90	770.08	780.34	0.96	713.70	720.21	0.90

Table 7. Result for 2|SR|L 2L-VRP (Classes 2–5). "BKS" denotes the best-known solution found in [10–12].

Inst.	Class 2			Class 3			Class 4			Class 5		
	BKS	Cost	Gap	BKS	Cost	Gap	BKS	Cost	Gap	BKS	Cost	Gap
1	278.73	278.73	**0.00**	284.23	284.52	0.10	282.95	282.95	**0.00**	278.73	278.73	**0.00**
2	334.96	334.96	**0.00**	352.16	352.16	**0.00**	334.96	334.96	**0.00**	334.96	334.96	**0.00**
3	384.93	384.93	**0.00**	385.32	385.32	**0.00**	362.41	364.45	0.56	358.40	358.40	**0.00**
4	430.89	430.89	**0.00**	430.89	430.89	**0.00**	447.37	447.37	**0.00**	430.89	430.89	**0.00**
5	375.28	375.28	**0.00**	379.94	385.69	1.51	383.88	383.88	**0.00**	375.28	375.28	**0.00**
6	495.85	499.08	0.65	498.16	498.16	**0.00**	498.32	502.11	0.76	495.85	507.18	2.28
7	716.82	716.82	**0.00**	668.39	689.01	3.09	686.26	702.45	2.36	657.77	660.37	0.40
8	671.75	674.20	0.36	738.43	741.12	0.36	692.47	705.89	1.94	609.90	609.90	**0.00**
9	607.65	619.21	1.90	607.65	615.57	1.30	625.10	625.10	**0.00**	607.65	618.90	1.85
10	684.37	685.21	0.12	615.68	623.29	1.24	708.68	708.68	**0.00**	680.26	699.01	2.76
11	694.60	694.60	**0.00**	704.77	711.44	0.95	776.69	776.69	**0.00**	624.82	624.82	**0.00**
12	610.00	619.63	1.58	610.00	610.23	0.04	614.24	614.24	**0.00**	601.00	610.24	1.54
13	2526.07	2534.97	0.35	2436.06	2469.98	1.39	2561.65	2623.65	2.42	2334.78	2334.78	**0.00**
14	1032.01	1042.83	1.05	1006.69	1012.46	0.57	981.90	988.75	0.70	921.45	928.91	0.81
15	1005.26	1018.77	1.34	1142.18	1170.82	2.51	1171.41	1172.43	0.09	1160.96	1160.96	**0.00**
16	698.61	698.61	**0.00**	698.61	698.61	**0.00**	703.35	708.20	0.69	698.61	698.61	**0.00**
17	861.79	863.27	0.17	861.79	861.79	**0.00**	861.79	861.79	**0.00**	861.79	861.79	**0.00**
18	988.37	989.21	0.08	1030.69	1031.94	0.12	1104.08	1128.25	2.19	921.29	926.40	0.55
19	731.93	732.64	0.10	757.59	757.59	**0.00**	776.59	776.59	**0.00**	651.97	651.97	**0.00**
20	495.01	501.89	1.39	519.15	536.58	3.36	541.17	549.38	1.52	472.09	488.28	3.43
Ave	731.24	734.79	0.46	736.42	743.36	0.83	755.76	762.89	0.66	703.92	708.02	0.68

Tables 6 and 7 present the best results obtained by our method on some instances of Classes 2–5 for 2L-VRP under the 2|SO|L and 2|SR|L settings, respectively. Table 5 shows that our EATS-MOBFS method can obtain high-quality solutions for all instances, where the average percentage gaps between the solutions obtained by our method and the BKS

are 0.59%, 0.90%, 0.96% and 0.90% for the instances in Classes 2–5, respectively. Under the 2|SR|L setting, the corresponding average percentage gaps are 0.46%, 0.83%, 0.66% and 0.68%, respectively. It is worth noting that the EATS-MOBFS method has reached the best-known solutions for almost 10 instances in each class. It infers that the proposed hybrid heuristic is efficient for solving the 2L-VRP under the 2|SO|L and 2|SR|L settings.

As illustrated in Table 8, the results obtained by the proposed EATS-MOBFS under the 2|UR|L and 2|SR|L settings are compared with the existing heuristic approaches, which shows that the proposed approach is capable of producing comparable results under the 2|UR|L and 2|SR|L settings. Specifically, the overall performance of the proposed EATS-MOBFS method outperforms that of ACO under the 2|UR|L setting. In comparison to ACO, our approach is able to produce better solutions for 11 out of 20 instances. Under the 2|SR|L setting, our approach is superior to that of ACO and has produced comparable results compared with those obtained by BR-LNS. The proposed approach yields better solutions for 11 out of 20 instances compared with ACO.

Table 8. Comparison results for the 2L-VRP under the 2|UR|L and 2|SR|L settings (average over Classes 2–5). Other state-of-the-art approaches: ACO [11], BR-LNS [41] and MS-BR [10].

Inst.	UR			SR		
	ACO	**MS-BR**	**EATS**	**ACO**	**BR-LNS**	**EATS**
1	281.16	280.84	280.84	281.7	281.16	281.23
2	341.02	339.26	339.26	341.021	339.261	339.26
3	372.93	370.62	370.79	376.65	374.07	373.28
4	435.01	435.01	435.01	435.01	435.01	435.01
5	378.60	378.60	379.03	378.59	378.59	380.03
6	497.05	497.05	497.05	497.62	497.045	501.63
7	688.50	681.00	682.10	696.23	685.12	692.16
8	678.75	675.46	681.74	690.10	686.34	682.78
9	612.02	612.01	614.96	612.02	612.01	619.70
10	671.00	667.65	673.33	679.67	674.94	679.05
11	698.25	690.56	692.30	711.67	709.13	701.89
12	611.12	611.06	611.12	613.89	611.12	613.59
13	2468.19	2437.15	2439.70	2513.67	2490.83	2490.85
14	974.81	968.55	971.81	992.53	986.41	993.24
15	1132.49	1112.00	1122.46	1165.38	1131.94	1130.75
16	699.79	699.80	701.01	699.80	699.80	701.01
17	862.37	861.79	865.50	861.79	861.79	862.16
18	1012.20	999.22	1005.95	1018.95	1012.21	1018.95
19	726.96	722.17	724.41	737.73	730.60	729.70
20	508.69	501.90	503.55	517.79	510.35	519.03

To sum up, the proposed approach produces satisfactory performances in solving the 2L-VRP under all four loading configurations, where the average percentage gaps between the solutions obtained by the EATS-MOBFS method and the best-known solutions are 0.97%, 0.48%, 1.14% and 0.64%, respectively. Particularly, our approach has reached the best-known solutions for 34, 35, 25 and 36 out of 80 instances under four settings, respectively. It can be observed that the average percentage gaps over the best-known solutions are smaller under the 2|UR|L and 2|SR|L settings compared with the 2|UO|L and 2|SO|L settings. Namely, the performance of our approach is more satisfactory under the 2|UR|L and 2|SR|L settings, while the performance is inferior to the existing state-of-art methods under the 2|UO|L and 2|SO|L settings. We can also observe that for some large-scale instances, our proposed approach shows a slight disadvantage over the existing methods. This is because the 2L-VRPSC is much more complex than the 2L-VRP with the consideration of the uncertainty of customers. Thus, the proposed solution approach designed for the 2L-VRPSC is more complex, with the specific strategies integrated into the initialization and neighborhood search process, which may undermine the exploration ability of the algorithm when solving the 2L-VRP. Furthermore, the algorithms designed for the simpler 2L-VRP have less computational burden and may find better solutions within the same time window.

4.2.2. Results of the 2L-VRPSC Benchmark Instances

In this section, we extend the Class 2 2L-VRP benchmarks described in the previous section to generate a large stochastic instance pool. Specifically, four classes of stochastic 2L-VRPSC instances are generated by varying the probability that customers require service $p = \{0.2, 0.4, 0.6, 0.8\}$. Here, we only test the performance of the approach under the 2|UO|L setting, with the results presented in Table 9. To verify the necessity of considering customer uncertainty, we use the value of the stochastic solution (VSS) to measure the extra cost due to ignoring the customers' uncertainty. The VSS index is defined as the difference between the expected results of using the expected value problem solution (EEV) and the here-and-now solution (HN), where the value of HN is the objective value obtained by stochastic programming model and the value of EEV represents the expected cost of the scenarios ignoring the customers' uncertainty. See [43–46] for more information about these indices.

Table 9. Comparative results of stochastic instances on 2|UO|L 2L-VRPSC under different probabilities that customers require service.

Instance	0.2			0.4			0.6			0.8		
	EEV	HN	VSS	EEV	HN	VSS	EEV	HN	VSS	EEV	HN	VSS
1	296.5	289.4	7.1	291.9	287.2	4.7	287.6	285.1	2.6	283.4	282.9	0.4
2	366.9	350.6	16.4	357.3	346.0	11.4	353.6	344.1	9.4	349.2	342.0	7.2
3	409.4	398.4	11.0	419.8	403.4	16.4	455.7	420.3	35.4	416.4	401.8	14.6
4	520.1	473.4	46.7	490.9	459.9	31.0	452.2	441.4	10.8	450.3	440.5	9.8
5	451.6	411.7	39.9	434.8	404.0	30.9	421.2	397.6	23.6	391.5	383.3	8.2
6	595.3	545.1	50.2	572.4	534.5	37.9	541.4	519.8	21.6	515.4	507.2	8.2
7	847.1	783.9	63.2	844.3	782.6	61.7	809.6	766.4	43.2	768.2	746.5	21.7
8	809.2	755.8	53.4	802.9	752.8	50.1	736.0	720.8	15.2	734.2	719.9	14.3
9	700.1	654.3	45.8	684.5	647.0	37.6	667.2	638.7	28.5	632.1	621.7	10.4
10	791.6	738.9	52.7	761.5	724.7	36.8	789.8	738.0	51.7	708.5	699.0	9.5
11	843.5	774.5	69.1	812.8	760.2	52.5	812.8	760.2	52.6	740.0	725.4	14.6
12	712.4	661.5	50.9	693.9	652.9	41.0	689.6	650.8	38.8	641.9	627.9	14.0
13	3069.4	2818.9	250.5	3015.2	2793.9	221.3	2829.7	2706.6	123.1	2682.0	2635.0	47.0
14	1278.3	1152.3	126.0	1225.7	1128.3	97.4	1184.9	1109.4	75.5	1089.7	1063.9	25.8
15	1231.0	1134.8	96.2	1207.5	1123.9	83.6	1160.9	1102.0	58.9	1113.0	1079.0	34.0
16	831.6	762.2	69.4	819.5	756.6	62.8	757.1	727.3	29.9	739.1	718.6	20.5
17	1056.7	961.8	94.9	990.5	931.2	59.3	981.1	926.8	54.3	940.4	907.3	33.1
18	1316.0	1169.2	146.8	1218.4	1125.0	93.4	1170.7	1102.8	67.9	1115.6	1076.5	39.1
19	926.9	845.4	81.6	920.2	842.3	77.9	845.4	807.3	38.1	813.8	792.1	21.7
20	685.6	586.2	99.4	664.5	576.8	87.7	598.1	569.1	29.0	576.0	558.5	17.5
Ave	887.0	813.4	73.6	861.4	801.7	59.8	827.2	786.7	40.5	785.0	766.4	18.6

As shown in Table 9, the values of VSS, EEV and HN indices for each instance are given. It can be observed that the value of the VSS index for each instance is greater than zero, which is consistent with our expectation that ignoring the uncertain information raised in the distribution operations incurs extra costs. Meanwhile, the direct application of the distribution schemes ignoring uncertainty may lead to costly ad-hoc arrangements for re-routing, which may substantially increase the actual operating costs and time. On the contrary, the application of distribution schemes considering the customers' uncertainty is conducive to the distribution companies to gaining familiarity with the distribution route and can have a good estimate of the time used to reach each customer so as to improve customers' satisfaction and distribution efficiency.

We can also observe that the average values of the VSS index are 73.6, 59.8, 40.5 and 18.6 under the probabilities of 0.2, 0.4, 0.6 and 0.8, respectively. A lower probability of the presence of customers results in a higher value for VSS, EEV and HN indices. These observations suggest that a lower probability of the presence of customers will aggravate the uncertainty of the distribution scheme, which incurs a significant increase in the

transportation cost. Additionally, the increased number of customers in the distribution scheme causes a significant increase in the transportation cost when customer uncertainty is considered. In terms of computational time, in our experiments, the average computational time for all 2L-VRPSC instances under the deterministic and stochastic settings are 1019 and 1380 s, respectively. Particularly, the computational time is affected by the size of scenarios generated from the Monte Carlo simulation, which contributes to the trade-off between the model fidelity and computational burden.

4.2.3. Validation and Efficiency of the Adaptive and Diversification Mechanisms

To verify the performance of the adaptive weight adjustment and diversification mechanisms of our proposed EATS algorithm, we select one of two 2L-VRPSC instances under the probability of 0.8. Under the same parameters combination, the proposed EATS-MOBFS is compared with ATS-MOBFS (without a diversification mechanism) and ETS-MOBFS (without adaptive weight adjustment), with the comparison results presented in Table 10. For conciseness, these three algorithms are abbreviated as EATS, ATS and ETS, respectively.

Table 10. Comparison of average results between EATS, ATS and ETS algorithms, where "Gap" denotes the percentage gap between the proposed EATS and ATS or ETS.

Instance	EATS		ATS			ETS		
	Cost	Time (s)	Cost	Time (s)	Gap (%)	Cost	Time (s)	Gap (%)
1	282.95	712	282.95	715	0.00	282.95	716	0.00
3	401.81	801	404.82	792	0.75	404.59	797	0.69
5	383.29	741	384.38	734	0.28	383.29	745	0.00
7	746.50	938	741.12	904	−0.72	745.46	943	−0.14
9	621.73	1356	628.65	1309	1.11	627.76	1307	0.97
11	725.40	1745	735.24	1702	1.36	737.29	1737	1.64
13	2634.99	1400	2699.30	1373	2.44	2663.13	1407	1.07
15	1079.00	1901	1082.78	1879	0.35	1074.14	1921	-0.45
17	907.35	1711	915.84	1662	0.94	920.51	1720	1.45
19	792.11	1799	806.79	1779	1.85	799.34	1758	0.91
Ave	857.5	1310.4	868.2	1284.9	0.84	863.85	1305.0	0.61

We can observe that the presented EATS approach shows superiority with respect to solution quality compared with the ATS and ETS algorithms. Compared with ATS and ETS algorithms, the proposed EATS algorithm is capable of producing better solutions for 8 and 6 out of 10 instances, which indicates the adaptive weight adjustment and diversification mechanism have a positive impact on the interplay between intensification and diversification. The better performance of ETS over ATS may be due to the fact that the diversification mechanism can avoid trapping into a local optimum to some extent. Additionally, the average computational time of ATS is slightly shorter than EATS and ETS, which is attributed to the fact that the diversification process is time-consuming to destroy and repair the solution structure. This slight increase in computation time is acceptable given that the diversification mechanism can diversify the search and guide the algorithm search for better solutions.

5. Conclusions

This paper studies a stochastic variant of the 2L-VRP, where each customer has a probability of presence and customers' demands are considered non-stackable rectangular-shape items. A stochastic programming model of the 2L-VRPSC has been established based on Monte Carlo simulation and scenario analysis. An enhanced adaptive tabu search algorithm incorporating a MOBFS packing heuristic has been proposed, which solved four classes of 2L-VRPSC with different loading configurations, namely, unrestricted ori-

ented loading (2|UO|L), unrestricted rotated loading (2|UR|L), sequential oriented loading (2|SO|L) and sequential rotated loading (2|SR|L).

The performance of the proposed approach is verified through experiments on the 2L-VRP and 2L-VRPSC instances, which are compared with existing methods. The experimental results demonstrate the effectiveness and versatility of solving the 2L-VRPSC and 2L-VRP. The comparison between existing methods shows the efficiency of the proposed EATS-MOBFS method for solving the 2L-VRP with different loading configurations, except for some large-scale instances. Moreover, the proposed model is capable of providing more practical solutions for real-life scenarios with uncertain information. Meanwhile, the comparative results indicate that the adaptive weight adjustment and diversification mechanism have a positive impact on the interplay between intensification and diversification. Observations show that a lower presence probability of customers and a larger number of customers will increase the transportation cost in the 2L-VRPSC.

This work has opened up several directions for future research. First, although this research's high-quality solutions have been obtained by the proposed approach, there is still room for improvement. Such improvements can include further enhancing the neighborhood operators in EATS and introducing more exclusive packing strategies. Second, in this paper, we have studied the 2L-VRP with stochastic customers under different loading configurations, while the investigations of the 2L-VRP with other uncertainties, such as stochastic demands, stochastic items and travel time, are yet to be carried out.

Author Contributions: Conceptualization, Z.Z. and B.J.; methodology, Z.Z. and B.J.; software, Z.Z.; validation, Z.Z., B.J. and S.S.Y.; formal analysis, B.J. and S.S.Y.; resources, Z.Z. and B.J.; writing—original draft preparation, Z.Z.; writing—review and editing, Z.Z., B.J. and S.S.Y.; visualization, Z.Z. and B.J.; supervision, B.J.; project administration, B.J. and S.S.Y.; funding acquisition, B.J. and S.S.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Grant from Hubei Provincial Key Laboratory for Operation and Control of Cascaded Hydropower Station under No. 2022KJX02, Central South University under Grant No. 202045007, and National Natural Science Foundation of China under Grant No. 62003091 .

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Acknowledgments: The authors are grateful to the farmers for their support in participating in this study.

Conflicts of Interest: The authors declare that they have no conflict of interest.

References

1. Toth, P.; Vigo, D. *Vehicle Routing: Problems, Methods, and Applications*; SIAM: Sitges, Spain, 2014.
2. Li, J.; Wang, F.; He, Y. Electric vehicle routing problem with battery swapping considering energy consumption and carbon emissions. *Sustainability* **2020**, *12*, 10537. [CrossRef]
3. Iori, M.; Salazar-González, J.J.; Vigo, D. An exact approach for the vehicle routing problem with two-dimensional loading constraints. *Transp. Sci.* **2007**, *41*, 253–264. [CrossRef]
4. Gendreau, M.; Iori, M.; Laporte, G.; Martello, S. A Tabu search heuristic for the vehicle routing problem with two-dimensional loading constraints. *Netw. Int. J.* **2008**, *51*, 4–18. [CrossRef]
5. Sbai, I.; Limam, O.; Krichen, S. An effective genetic algorithm for solving the capacitated vehicle routing problem with two-dimensional loading constraint. *Int. J. Comput. Intell. Stud.* **2020**, *9*, 85–106. [CrossRef]
6. Escobar-Falcón, L.; Alvarez-Martinez, D.; Wilmer-Escobar, J.; Granada-Echeverri, M. A specialized genetic algorithm for the fuel consumption heterogeneous fleet vehicle routing problem with bidimensional packing constraints. *Int. J. Ind. Eng. Comput.* **2021**, *12*, 191–204. [CrossRef]
7. Leung, S.C.; Zhou, X.; Zhang, D.; Zheng, J. Extended guided tabu search and a new packing algorithm for the two-dimensional loading vehicle routing problem. *Comput. Oper. Res.* **2011**, *38*, 205–215. [CrossRef]
8. Zachariadis, E.E.; Tarantilis, C.D.; Kiranoudis, C.T. Integrated distribution and loading planning via a compact metaheuristic algorithm. *Eur. J. Oper. Res.* **2013**, *228*, 56–71. [CrossRef]

9. Wei, L.; Zhang, Z.; Zhang, D.; Lim, A. A variable neighborhood search for the capacitated vehicle routing problem with two-dimensional loading constraints. *Eur. J. Oper. Res.* **2015**, *243*, 798–814. [CrossRef]
10. Dominguez, O.; Juan, A.A.; Faulin, J. A biased-randomized algorithm for the two-dimensional vehicle routing problem with and without item rotations. *Int. Trans. Oper. Res.* **2014**, *21*, 375–398. [CrossRef]
11. Fuellerer, G.; Doerner, K.F.; Hartl, R.F.; Iori, M. Ant colony optimization for the two-dimensional loading vehicle routing problem. *Comput. Oper. Res.* **2009**, *36*, 655–673. [CrossRef]
12. Wei, L.; Zhang, Z.; Zhang, D.; Leung, S.C. A simulated annealing algorithm for the capacitated vehicle routing problem with two-dimensional loading constraints. *Eur. J. Oper. Res.* **2018**, *265*, 843–859. [CrossRef]
13. Pinto, T.; Alves, C.; Valério de Carvalho, J. Variable neighborhood search algorithms for the vehicle routing problem with two-dimensional loading constraints and mixed linehauls and backhauls. *Int. Trans. Oper. Res.* **2020**, *27*, 549–572. [CrossRef]
14. Rivero, O.L.D.; Pérez, A.A.J.; De La Nuez Pestana, I.A.; Ouelhadj, D. An ILS-biased randomization algorithm for the two-dimensional loading HFVRP with sequential loading and items rotation. *J. Oper. Res. Soc.* **2016**, *67*, 37–53. [CrossRef]
15. Sabar, N.R.; Bhaskar, A.; Chung, E.; Turky, A.; Song, A. An adaptive memetic approach for heterogeneous vehicle routing problems with two-dimensional loading constraints. *Swarm Evol. Comput.* **2020**, *58*, 100730. [CrossRef]
16. Ferreira, K.M.; de Queiroz, T.A.; Toledo, F.M.B. An exact approach for the green vehicle routing problem with two-dimensional loading constraints and split delivery. *Comput. Oper. Res.* **2021**, *136*, 105452. [CrossRef]
17. Rajaei, M.; Moslehi, G.; Reisi-Nafchi, M. The split heterogeneous vehicle routing problem with three-dimensional loading constraints on a large scale. *Eur. J. Oper. Res.* **2021**, *299*, 706–721. [CrossRef]
18. Koch, H.; Schlögell, M.; Bortfeldt, A. A hybrid algorithm for the vehicle routing problem with three-dimensional loading constraints and mixed backhauls. *J. Sched.* **2020**, *23*, 71–93. [CrossRef]
19. Chen, Z.; Yang, M.; Guo, Y.; Liang, Y.; Ding, Y.; Wang, L. The split delivery vehicle routing problem with three-dimensional loading and time Windows constraints. *Sustainability* **2020**, *12*, 6987. [CrossRef]
20. Psaraftis, H.N. A dynamic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem. *Transp. Sci.* **1980**, *14*, 130–154. [CrossRef]
21. Gendreau, M.; Jabali, O.; Rei, W. 50th anniversary invited article—Future research directions in stochastic vehicle routing. *Transp. Sci.* **2016**, *50*, 1163–1173. [CrossRef]
22. Rajabi-Bahaabadi, M.; Shariat-Mohaymany, A.; Babaei, M.; Vigo, D. Reliable vehicle routing problem in stochastic networks with correlated travel times. *Oper. Res.* **2021**, *21*, 299–330. [CrossRef]
23. Xia, X.; Liao, W.; Zhang, Y.; Peng, X. A discrete spider monkey optimization for the vehicle routing problem with stochastic demands. *Appl. Soft Comput.* **2021**, *111*, 107676. [CrossRef]
24. Gee, S.B.; Arokiasami, W.A.; Jiang, J.; Tan, K.C. Decomposition-based multi-objective evolutionary algorithm for vehicle routing problem with stochastic demands. *Soft Comput.* **2016**, *20*, 3443–3453. [CrossRef]
25. Côté, J.F.; Gendreau, M.; Potvin, J.Y. The vehicle routing problem with stochastic two-dimensional items. *Transp. Sci.* **2020**, *54*, 453–469. [CrossRef]
26. Guimarans, D.; Dominguez, O.; Panadero, J.; Juan, A.A. A simheuristic approach for the two-dimensional vehicle routing problem with stochastic travel times. *Simul. Model. Pract. Theory* **2018**, *89*, 1–14. [CrossRef]
27. Ahmed, B.S.; Enoiu, E.; Afzal, W.; Zamli, K.Z. An evaluation of Monte Carlo-based hyper-heuristic for interaction testing of industrial embedded software applications. *Soft Comput.* **2020**, *24*, 13929–13954. [CrossRef]
28. Oscar, V.; Aguilasocho-Montoya, D.; Álvarez-García, J.; Simonetti, B. Using Markov-switching models with Markov chain Monte Carlo inference methods in agricultural commodities trading. *Soft Comput.* **2020**, *24*, 13823–13836.
29. Niknam, T.; Azizipanah-Abarghooee, R.; Narimani, M.R. An efficient scenario-based stochastic programming framework for multi-objective optimal micro-grid operation. *Appl. Energy* **2012**, *99*, 455–470. [CrossRef]
30. Bernath, C.; Deac, G.; Sensfuß, F. Impact of sector coupling on the market value of renewable energies—A model-based scenario analysis. *Appl. Energy* **2021**, *281*, 115985. [CrossRef]
31. Niknam, T.; Zare, M.; Aghaei, J. Scenario-based multiobjective volt/var control in distribution networks including renewable energy sources. *IEEE Trans. Power Deliv.* **2012**, *27*, 2004–2019. [CrossRef]
32. Gmira, M.; Gendreau, M.; Lodi, A.; Potvin, J.Y. Tabu search for the time-dependent vehicle routing problem with time windows on a road network. *Eur. J. Oper. Res.* **2021**, *288*, 129–140. [CrossRef]
33. Sadati, M.E.H.; Çatay, B. A hybrid variable neighborhood search approach for the multi-depot green vehicle routing problem. *Transp. Res. Part E Logist. Transp. Rev.* **2021**, *149*, 102293. [CrossRef]
34. Pisinger, D.; Ropke, S. Large neighborhood search. In *Handbook of Metaheuristics*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 399–419.
35. Zachariadis, E.E.; Tarantilis, C.D.; Kiranoudis, C.T. A guided tabu search for the vehicle routing problem with two-dimensional loading constraints. *Eur. J. Oper. Res.* **2009**, *195*, 729–743. [CrossRef]
36. Cai, L.; Wang, X.; Luo, Z.; Liang, Y. A hybrid adaptive large neighborhood search and tabu search algorithm for the electric vehicle relocation problem. *Comput. Ind. Eng.* **2022**, *167*, 108005. [CrossRef]
37. 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]

38. Chazelle, B. The bottomn-left bin-packing heuristic: An efficient implementation. *IEEE Trans. Comput.* **1983**, **32**, 697–707. [CrossRef]
39. López-Ibáñez, M.; Dubois-Lacoste, J.; Cáceres, L.P.; Birattari, M.; Stützle, T. The irace package: Iterated racing for automatic algorithm configuration. *Oper. Res. Perspect.* **2016**, *3*, 43–58. [CrossRef]
40. Duhamel, C.; Lacomme, P.; Quilliot, A.; Toussaint, H. A multi-start evolutionary local search for the two-dimensional loading capacitated vehicle routing problem. *Comput. Oper. Res.* **2011**, *38*, 617–640. [CrossRef]
41. Dominguez, O.; Guimarans, D.; Juan, A.A.; de la Nuez, I. A biased-randomised large neighbourhood search for the two-dimensional vehicle routing problem with backhauls. *Eur. J. Oper. Res.* **2016**, *255*, 442–462. [CrossRef]
42. Zachariadis, E.E.; Tarantilis, C.D.; Kiranoudis, C.T. The vehicle routing problem with simultaneous pick-ups and deliveries and two-dimensional loading constraints. *Eur. J. Oper. Res.* **2016**, *251*, 369–386. [CrossRef]
43. Hu, S.L.; Han, C.F.; Meng, L.P. Stochastic optimization for joint decision making of inventory and procurement in humanitarian relief. *Comput. Ind. Eng.* **2017**, *111*, 39–49. [CrossRef]
44. Nikzad, E.; Bashiri, M.; Oliveira, F. Two-stage stochastic programming approach for the medical drug inventory routing problem under uncertainty. *Comput. Ind. Eng.* **2019**, *128*, 358–370. [CrossRef]
45. Birge, J.R.; Louveaux, F. *Introduction to Stochastic Programming*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2011.
46. Tang, M.; Ji, B.; Fang, X.; Yu, S.S. Discretization-Strategy-Based Solution for Berth Allocation and Quay Crane Assignment Problem. *J. Mar. Sci. Eng.* **2022**, *10*, 495. [CrossRef]