Transportation Research Part E 176 (2023) 103208

Contents lists available at ScienceDirect

Transportation Research Part E

journal homepage: www.elsevier.com/locate/tre

Pickup capacitated vehicle routing problem with three-dimensional loading constraints: Model and algorithms

Jushang Chi, Shiwei He *

Key Laboratory of Transport Industry of Big Data Application Technologies for Comprehensive Transport, Beijing Jiaotong University, Beijing, China

ARTICLE INFO

Keywords:
Three-dimensional loading
Routing
Pickup operation
Loading algorithm
Branch-and-price algorithm

ABSTRACT

This study addresses the pickup capacitated vehicle routing problem with three-dimensional loading constraints (3L-PCVRP), which combined the container loading problem (CLP) and the capacitated vehicle routing problem (CVRP). A mixed-integer linear programming model for 3L-PCVRP is designed with the objective of minimizing the total transportation cost. The basic constraints of CLP and CVRP as well as the practical constraints imposed by the pickup operations are incorporated in 3L-PCVRP. An improved branch-and-price-based (B&P) algorithm is proposed to solve 3L-PCVRP. The 3L-PCVRP model is decomposed into the restricted master problem (RMP) for selecting routes and the subproblem (SP) for generating routes that satisfy the loading and routing constraints. A label-correcting-based algorithm (LCA) is presented to solve the SP, which employs a two-stage method and two enhanced loading algorithms to ensure the loading feasibility of the routes. The two loading algorithms (tree search and greedy heuristic algorithms) are improved by a new evaluation function and an improved space-merging method. The two-stage method is designed to call the two loading algorithms at different frequencies to investigate the route feasibility accurately and efficiently. Numerical experiments are designed to test the performance of the proposed algorithms. The efficiency and effectiveness of the improved loading algorithms are demonstrated in hundreds randomly generated instances. For small-scale 3L-PCVRP instances, the B&P algorithm can generate solutions that are not worse than those of the solver within 0.14% of the solver's running time. The efficiency and effectiveness of the improved B&P algorithm are also validated in large-scale benchmark instances.

1. Introduction

Companies specializing in item pickup or delivery using trucks often encounter vehicle routing problems, which involve computing optimal routes to fulfill supplier/client demands at the lowest possible cost. Effective route planning can significantly reduce the overall transportation cost. The weight attribute of items is widely used to reflect the restriction of the vehicle capacity on loaded items. These problems are classified into the capacitated vehicle routing problem (CVRP) (Tahami et al., 2020).

However, in many practical freight pickup or distribution applications, the loading of items into vehicles can represent a difficult problem, especially for large items such as furniture, mechanical components, and household appliances (Iori et al., 2007). Companies need to define a logistics plan, including loading plan of items in vehicles (which items are loaded and where they are loaded) as well as vehicle routes. These loading issues have a significant impact on vehicle routing and transportation costs (Fuellerer et al., 2010). On

* Corresponding author.
E-mail addresses: 20114029@bjtu.edu.cn (J. Chi), shwhe@bjtu.edu.cn (S. He).

https://doi.org/10.1016/j.tre.2023.103208
Received 14 March 2023; Received in revised form 10 June 2023; Accepted 10 June 2023
Available online 14 July 2023

the basis of studies that integrated the loading problem with the CVRP, the capacitated vehicle routing problem with three-dimensional loading constraints (3L-CVRP) can best reflect the loading and unloading operations of items because of the consideration of their three dimensions (3L). 3L-CVRP seeks to define the routes of a vehicle fleet to deliver items to clients while minimizing the total transportation cost (Gendreau et al., 2006). The problem contains basic loading and routing constraints (flow balance, geometric feasibility, non-overlap, etc.) and practical loading constraints. The latter is derived from the actual working methods of companies, such as the unloading sequence constraint that prohibits repositioning of any item owing to unloading operations (Gendreau et al., 2006). Based on the definition and constraints proposed by Gendreau et al. (2006), many studies have been conducted to explore variants of 3L-CVRP or other efficient algorithms (Bortfeldt and Homberger, 2013; Bortfeldt and Yi, 2020; Rajaei et al., 2022).

On the basis of the previous studies, several research gaps can be identified, as listed below. 1. In the delivery scenario, items are loaded at the depot and unloaded to the clients. In the pickup scenario, items are loaded from different suppliers and unloaded at the depot. Although the working methods differ between the pickup and delivery scenarios, at present, most studies are carried out under the delivery scenario. The current research status reflects the following research gaps to be filled: What are the exact differences between the pickup and delivery scenarios? Which constraints are general under the pickup scenario? 2. Because of the complexity of 3L-CVRP, its description is usually limited to verbal description in most 3L-CVRP studies. At present, a linear 3L-CVRP model that considers all the constraints proposed by Gendreau et al. (2006) and that can be directly solved by commercial solvers is yet to be reported. 3. Efficient algorithms for solving 3L-CVRP and its variants are still being discussed.

To fill these gaps, the objectives of this study were threefold. 1. Problem research. Based on the existing studies on 3L-CVRP and discussion on the constraints caused by the pickup operations, a variant of 3L-CVRP that focuses on the pickup scenario is presented in this study, namely, pickup 3L-CVRP (3L-PCVRP). The details of the pickup operations and the unique constraints imposed by the pickup operations are discussed. This study emphasizes the differences between the pickup and delivery scenarios and fills the gaps in previous research on the pickup scenarios. 2. Model construction. In this study, a model with all the constraints proposed by Gendreau et al. (2006) is presented. The model can be converted to the 3L-CVRP or 3L-PCVRP versions by simply adjusting sets and parameters. 3. Algorithm design. This study proposes an efficient solution method for 3L-PCVRP. The solution method will be introduced in three parts: First, the model is decomposed into the restricted master problem (RMP) for selecting routes and a subproblem (SP) for generating routes that satisfy the loading and routing constraints. The structures of the decomposed models are then discussed. Second, the loading constraints are analyzed, and two loading algorithms (tree search algorithm (TRSA) and greedy heuristic algorithm (GHA)), as well as the corresponding enhancement strategies, are proposed. Finally, a label-correcting-based algorithm (LCA) is presented to solve the SP, in which a two-stage method is developed to employ the two loading algorithms at different frequencies.

The main contributions of this paper are summarized as follows:

1. Differences between the pickup and delivery scenarios are discussed in detail. The practical constraints imposed by the pickup operations are presented. 3L-PCVRP is introduced.

2. A model of 3L-PCVRP, which can be easily converted to a model of 3L-CVRP, is proposed.

3. Improved algorithms, including an improved TRSA and an improved GHA, are developed to investigate the loading feasibility of the routes. The algorithms demonstrate an overall improvement in efficiency and effectiveness.

4. An improved branch-and-price-based algorithm is developed to solve 3L-PCVRP. An LCA is developed for efficient route generation. By employing a two-stage method to call the improved TRSA and the improved GHA at different frequencies, the efficiency and effectiveness of the LCA are enhanced.

This paper is organized as follows: Section 2 presents a literature review of 3L-CVRP. Section 3 discusses the differences between the pickup and delivery operations as well as the constraints imposed by pickup operations. The problem statement and the model of 3L-PCVRP are presented in Section 4. Section 5 introduces details of the solution methods. Numerical experiments are presented in Section 6. Finally, conclusions and suggestions for future work are presented in Section 7.

2. Literature review

As a classical NP-hard problem (Lenstra and Kan, 1981), the CVRP and its variants have been extensively studied over the past few decades (Chen et al., 2020; Gunawan et al., 2021; Pourhejazy et al., 2021; Bombelli & Fazi, 2022). In most CVRP studies, the weight attribute of items is used to judge whether a vehicle can load the corresponding items (Tahami et al., 2020; Yuan et al., 2021; Amine Masmoudi et al., 2022).

For accurately determining the item locations in a container or vehicle, container loading problem (CLP) was proposed (George and Robinson, 1980). The CLP is described as loading a group of rectangular items into a container with fixed length, width, and height to achieve the highest volume utilization (Ren et al., 2011). The CLP has been proven to be NP-hard (Pisinger, 2002), and it has been proven difficult to achieve high-quality solutions within an acceptable time (Zhao et al., 2016; Nascimento et al., 2021). Therefore, heuristic algorithms are typically used to deal with the CLP (Gajda et al., 2022).

In view that neither the CVRP nor the CLP considers the relationship between vehicle routes and item placement, Gendreau et al. (2006) proposed 3L-CVRP and developed a tabu search (TS) algorithm to solve the problem. By adding loading constraints to each route, the CVRP and the CLP were integrated for simultaneously optimizing item placement and vehicle routes. By applying the TS algorithm to search routes and employing a hybrid TS algorithm to investigate the loading feasibility of the routes, the feasibility of each route was ensured. The route is feasible if all items of the suppliers along the route can be loaded into the corresponding vehicle.

In the following sections, related studies are reviewed from two main aspects: variants of 3L-CVRP and existing solution methods of 3L-CVRP. The former offers state-of-the-art 3L-CVRP as well as insights into the characteristics and related constraints of 3L-PCVRP. The latter analyzes the characteristics of the existing solution methods of 3L-CVRP as well as the difficulties in solving 3L-PCVRP.

2.1. Variants of the 3L-CVRP

As described above, 3L-CVRP is composed of the CLP and CVRP. Therefore, the 3L-CVRP constraints can be divided into loading and routing constraints. In the study by Gendreau et al. (2006), the routing constraints conformed to conventional flow-balance and vehicle-capacity constraints. The loading constraints are described below (Gendreau et al., 2006; Rajaei et al., 2022) and are widely used in most 3L-CVRP studies.

It is worth noting that basic constraints are used in almost all studies. They remain unchanged in all cases. On the contrary, practical constraints can be adjusted or removed according to the problem at hand.

(1) Orientation constraint: Each item has a fixed vertical orientation, but a horizontal 90° rotation is allowed (basic constraint).

(2) Geometric feasibility constraint: Each item lies entirely and orthogonally within the loading space, and items within a vehicle do not overlap (basic constraint).

(3) No-split constraint: Items to be delivered to the same clients must be loaded into the same vehicle (practical constraint).

(4) Vertical stability constraint: An item can be placed on other items only if the supporting area is not less than a given threshold percentage (less than100%: partial support; =100%: full support) of the base of the item (practical constraint).

(5) Fragility constraint: Non-fragile items cannot be stacked on the top of fragile items (practical constraint).

(6) Unloading sequence/last-in-first-out (LIFO) constraint: All items must be directly unloaded through a sequence of straight movements parallel to the length of the vehicle without repositioning of other items (practical constraint).

Since Gendreau et al. (2006) proposed 3L-CVRP, many researchers have explored variants of 3L-CVRP by adjusting, adding, or

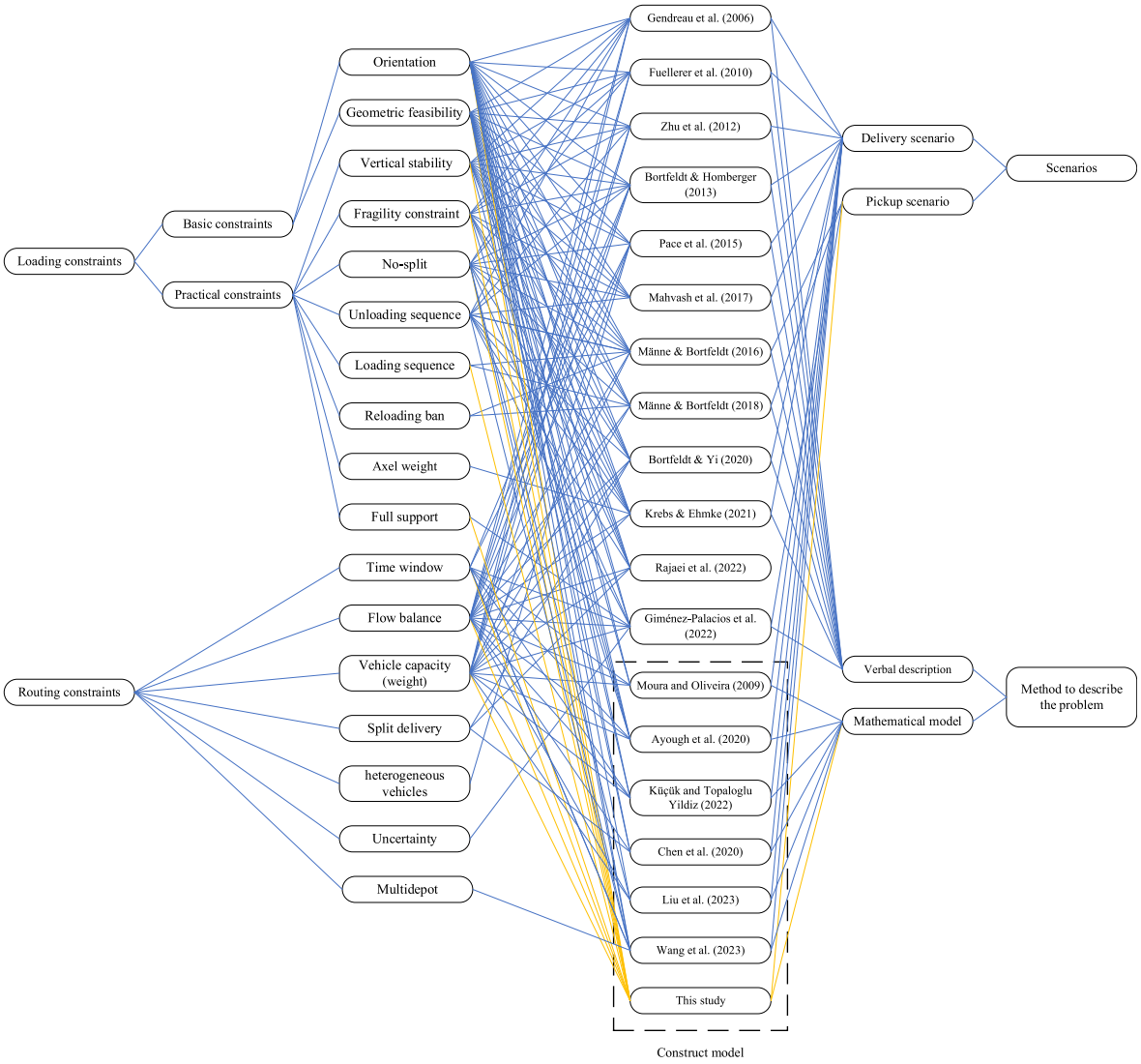

Fig. 1. Typical 3L-CVRP studies.

removing routing or practical loading constraints. Moura and Oliveira (2009), Hu et al. (2015), Moura (2019), and Vega-Mejía et al. (2019) considered the time window constraint. Ceschia et al. (2013), Pace et al. (2015), and Rajaei et al. (2022) considered heterogeneous vehicles. Pollaris et al. (2017) and Krebs and Ehmke (2021) considered the axle weight constraint. Bortfeldt and Yi (2020) and Rajaei et al. (2022) considered split delivery. Giménez-Palacios et al. (2022) considered disruption-caused uncertainties. Wang et al. (2023) studied the multidepot 3L-CVRP.

In addition, the 3L-CVRP variant closest to 3L-PCVRP is the pickup and delivery 3L-CVRP (3L-PDP) proposed by Männel and Bortfeldt (2016). In their study, nodes were defined as pickup and delivery points, and items were picked up at the pickup points and delivered to the delivery points. To avoid item repositioning in the loading and unloading operations, the loading and unloading sequence constraints were considered simultaneously. Furthermore, the reloading ban constraint, which prohibits item repositioning, has been proposed by Männel and Bortfeldt (2018). The descriptions of the two constraints are as follows:

(7) Loading sequence constraint: All items must be directly loaded into a vehicle through a sequence of straight movements parallel to the length of the vehicle without repositioning other items (practical constraint).

(8) Reloading ban constraint: All items cannot be repositioned once they are loaded into the vehicle (practical constraint).

Beyond the development of the variants of 3L-CVRP, the modeling of 3L-CVRP is also worth discussing. Because of the problem's complexity, its description is usually limited to verbal description in most 3L-CVRP studies (Gendreau et al., 2006; Fuellerer et al., 2010; Mahvash et al., 2017; Rajaei et al., 2022). To the best of our knowledge, only a few studies have introduced a mathematical model for 3L-CVRP. Moura and Oliveira (2009) proposed a model with basic loading constraints. Ayough et al. (2020) presented a model with basic loading and fragility constraints. Chen et al. (2020) designed a model with basic loading constraints. Küçük and Topaloglu Yildiz (2022) developed a model with basic loading constraints. Liu et al. (2023) constructed a model with basic loading constraints. Wang et al. (2023) proposed a model with basic and loading sequence constraints, in which items were forced to be placed in pre-divided spaces. Therefore, to date, models that consider all the constraints proposed by Gendreau et al. (2006) have not been reported.

An overview of routing and loading features, scenarios, and methods to describe the problems in papers on 3L-CVRP is presented in Fig. 1. The constraints and methods considered in this study are listed at the bottom for comparison.

2.2. Solution methods of the 3L-CVRP

Since 3L-CVRP was proposed, many researchers attempted to develop better methods to obtain better solutions or reduce the computing time. Tarantilis et al. (2009) have developed a metaheuristic methodology that combines the strategies of TS and guided local search (GLS), in which a packing heuristic bundle algorithm is employed to investigate the loading feasibility of each route. Moura and Oliveira (2009) have designed a greedy randomized adaptive search procedure (GRASP) in which a greedy constructive heuristic is employed to investigate the loading feasibility. Fuellerer et al. (2010) have presented an ant colony optimization algorithm in which a hybrid heuristic algorithm that combines the bottom-left algorithm and the touching perimeter algorithm is employed to investigate the loading feasibility. Zhu et al. (2012) have presented a two-stage TS algorithm in which a deepest-bottom-left-fill (DBLF) heuristic algorithm is applied to investigate the loading feasibility. Bortfeldt and Homberger (2013) have developed a loading-first, routing-second method. By packing the items of each client (TS algorithm), the information on each client's item is represented by the length of the packed items. The conventional CVRP algorithm (hybrid heuristic) is then applied to solve the CVRP with an additional length constraint. Pace et al. (2015) have designed a method that combines an iterated local search (ILS) and simulated annealing (SA). They have developed a heuristic algorithm to investigate the loading feasibility. Mahvash et al. (2017) have designed a

Table 1
Solution methods developed in typical studies.

Publications	Algorithms		
	Loading	Routing (single vehicle)	Overall
Gendreau et al. (2006)	Hybrid TS	TS	
Fuellerer et al. (2010)	Hybrid heuristics	ACO	
Zhu et al. (2012)	DBLF	Two-stage TS	
Bortfeldt & Homberger (2013)	TS	Hybrid heuristic	
Mahvash et al. (2017)	EPHA	Greedy heuristic	C&G
Männel & Bortfeldt (2018)	Tree search (depth first search)	Large neighborhood search	
Bortfeldt & Yi (2020)	GA with construction heuristics	Local search	
Ayough et al. (2020)	SA	SA	
Moura & Oliveira (2009)	Constructive algorithm	GRASP	
Krebs & Ehmke (2021)	DBLF	ALNS	
Rajaei et al. (2022)	EPHA	Label-correcting-based heuristic	C&G
Küçük & Topaloglu Yildiz (2022)	Evolutionary algorithm	Constraint programming	
This paper	**Improved tree search & Improved greedy heuristic**	**Label-correcting-based heuristic**	**B&P**

TS: Tabu search algorithm, ACO: Ant colony optimization, DBLF: deepest-bottom-left-fill heuristic algorithm, EPHA: extreme point-based heuristic algorithm, GA: genetic algorithm, SA: simulated annealing, ALNS: adaptive large neighborhood search, GRASP: greedy randomized adaptive search procedures, C&G: column generation algorithm, B&P: branch-and-price algorithm (C&G and branch process).

column-generation-based algorithm. They have developed an extreme point-based heuristic algorithm (EPHA) to investigate the loading feasibility and an LCA to generate routes. Wu et al. (2019) developed a hybrid heuristic algorithm to solve 3L-PDP. Ayough et al. (2020) have presented an SA algorithm in which a hybrid heuristic algorithm combines an iterative loading heuristic method. The SA method has been employed to investigate the loading feasibility. Krebs and Ehmke (2021) have developed an adaptive large neighborhood search (ALNS) algorithm in which the DBLF heuristic algorithm is employed to investigate the loading feasibility. Rajaei et al. (2022) have designed a column-generation-based algorithm, in which a three-stage heuristic algorithm is applied to generate routes and an EPHA is employed to investigate the loading feasibility.

The solution methods applied in typical studies are summarized in Table 1, and those used in this study are placed at the bottom for comparison.

The following conclusions were drawn from the existing literature: 1. Efficient investigation of the loading feasibility of each route is the main difficulty in solving 3L-CVRP. 2. The loading algorithm is mainly called after obtaining a route (possibly incomplete) to judge its loading feasibility. Therefore, the loading algorithm is called very often. 3. Efficiency is the primary consideration for almost all loading algorithms. 4. To the best of our knowledge, all existing studies have employed only one loading algorithm to investigate the loading feasibility of the routes.

3. Discussion on the pickup operations

In previous 3L-CVRP studies, there were very few studies focusing on pickup operations. Männel and Bortfeldt (2016, 2018) and Wu et al. (2019) considered loading sequence constraint and presented algorithms to solve 3L-PDP. Giménez-Palacios et al. (2022) emphasized unique disruptions in the pickup scenario. In addition to the analysis of loading sequence constraint, research on the differences between the actual working methods under the pickup and delivery scenarios is lacking. To fill these gaps, this study presents the unique constraints encountered during pickup operations. In this section, the constraints to be considered under the pickup scenario are analyzed, and the exact formulation of 3L-PCVRP are presented.

An illustration of 3L-CVRP is given in Fig. 2**(a)**. The 3L-CVRP network is composed of a cluster of clients and a depot. Each client requires a group of items. Workers in the depot load items into a fleet of vehicles and send them to clients. An illustration of 3L-PCVRP is given in Fig. 2**(b)**. The 3L-PCVRP network is composed of a cluster of suppliers and a depot. Each supplier has a group of items to be picked up. Workers at the depot send a fleet of vehicles to pick up the items from suppliers. Comprehensive comparisons of the two scenarios are summarized below.

1. Loading and unloading operations: For 3L-CVRP, items are loaded at the depot and unloaded to the clients. The clients' workers can easily find the items to be unloaded and unload them without repositioning other items. For 3L-PCVRP, items are loaded from the suppliers and unloaded at the depot. Items in the same vehicle are loaded by different workers from different suppliers. If workers of a supplier fail to place the items accurately, workers of the subsequent supplier may have to reposition the misplaced items to place new

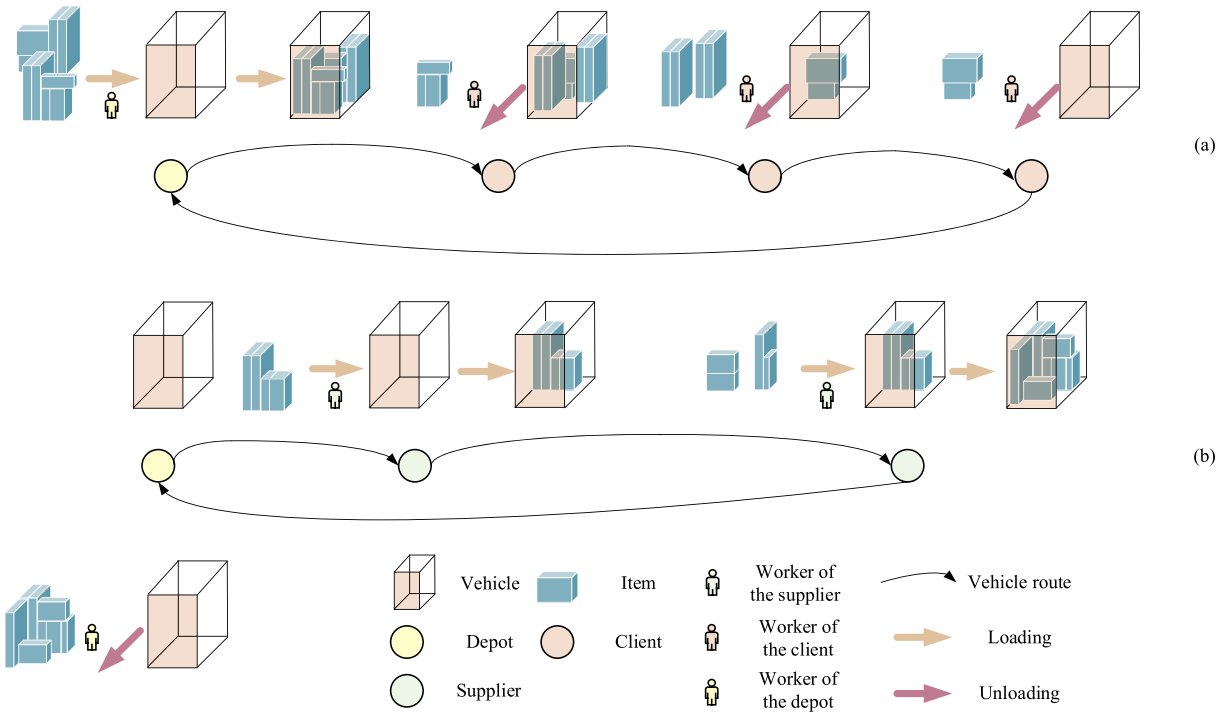

Fig. 2. Schematic diagrams of the delivery scenario (a) and pickup scenario (b).

items. A worse situation may occur if the subsequent supplier does not have proper tools to reposition the misplaced items: the vehicle has to return to the depot or the supplier that has provided the misplaced items for repositioning. In order to reduce the error rate of workers, the base of the items must be fully supported by the bottom items or the vehicle. Fig. 3 shows an overlap due to misplacement under the partial-support constraint. The previously loaded items have to be repositioned to replace the items loaded later. This potential error can be easily avoided by applying the full-support constraint.

2. Loading sequence: For 3L-CVRP, items are unloaded to the clients. The LIFO constraint requires items to be unloaded without repositioning other items. For 3L-PCVRP, items are loaded onto the suppliers. The loading sequence constraint should require the following. (1) Items can be loaded into vehicles without repositioning other items. (2) Previously loaded items cannot be stacked on the items to be loaded later.

3. Transportation time: For 3L-PCVRP, the items to be loaded into vehicles are packed by the suppliers. To avoid occupying the passageway of the suppliers for too long, the suppliers generally provide services within a specific time window. That is, the pickup scenario makes 3L-PCVRP have a stricter requirement for transportation time.

Therefore, 3L-PCVRP has more stringent requirements for the loading positions and transportation time. Based on the studies by Männel and Bortfeldt (2016, 2018), we further summarize the constraints of 3L-PCVRP as follows: 1. The CVRP constraints; 2. The time window constraint; 3. Basic loading constraints; 4. The no-split constraint; 5. The fragility constraint; 6. The loading sequence constraint, which prohibits item repositioning and restricts their loading sequence; and 7. The full-support constraint, which can be viewed as a basic loading constraint of 3L-PCVRP.

The roles of different constraints under different scenarios are clarified in this study by analyzing which constraints should be considered as basic constraints under the pickup scenario and which constraints can be adjusted according to the actual needs of the company. Our findings reflect the particularity of the pickup scenario and its impact on constraints, which fill the research gaps in previous studies on the pickup scenario and are used to construct a mathematical model in this study.

4. Problem statement and mathematical model

The transportation network of 3L-PCVRP can be described as a complete graph $G(V \cup \{o\} \cup \{d\}, A)$, where V is the supplier set and o, d represent the origin and destination of the depot, respectively. Let i, j indicate the index of nodes, $i,j \in V \cup \{o\} \cup \{d\}$. Each ordered pair of nodes $(i,j) \in A$. The transportation cost from node i to j is c_{ij}. The transportation time from node i to j is t_{ij}. The total weight of the items at supplier i is q_i. Let set U_i indicate the items and the corresponding rotated items at supplier i. Let n, n' indicate the index of the items, $n, n' \in U_i$. Let parameter N_i indicate the number of items at supplier i. Items n and $n+N_i$ indicate the original item and rotated item, respectively, $n, n+N_i \in U_i$. The length, width, height, and fragility of item n at supplier i are $l_{in}, w_{in}, h_{in}, \sigma_{in}$ respectively. If item n at supplier i is fragile, then $\sigma_{in} = 0$, and 1 otherwise. Each node i has a time window $[T_i^l, T_i^u]$. The loading time for supplier i is g_i. The depot has a set of homogenous vehicles K to pick up the items. Let k be the index of the vehicles. The length, width, and height of the 3L rectangular loading space of the vehicles are L, W, H, respectively. The capacity of the vehicles is Q. Each vehicle is loaded from the rear door. The coordinate system of the loading space is shown in Fig. 4, where the length, width, and height of the loading space are parallel to the X, Y, Z axes, respectively. Let (x, y, z) and (x', y', z') indicate the system coordinates. If item n at supplier i is placed at (x, y, z) and item n' at supplier j is placed at $(x', y', z - h_{jn'})$, then the length of the projection of the overlap between items n and n' on the X-axis is indicated by $L_{injn'}^{xx'}$, and the width on the Y-axis is indicated by $W_{injn'}^{yy'}$.

3L-PCVRP optimizes the vehicle routes and the location of the items in each vehicle. The variables to be optimized as well as the sets used to define the domains of the variables are summarized as follows.

Vehicles are organized to pick up the items from the suppliers. If vehicle k passes arc (i,j), then variable $\zeta_{ij}^k = 1$, and 0 otherwise. Let

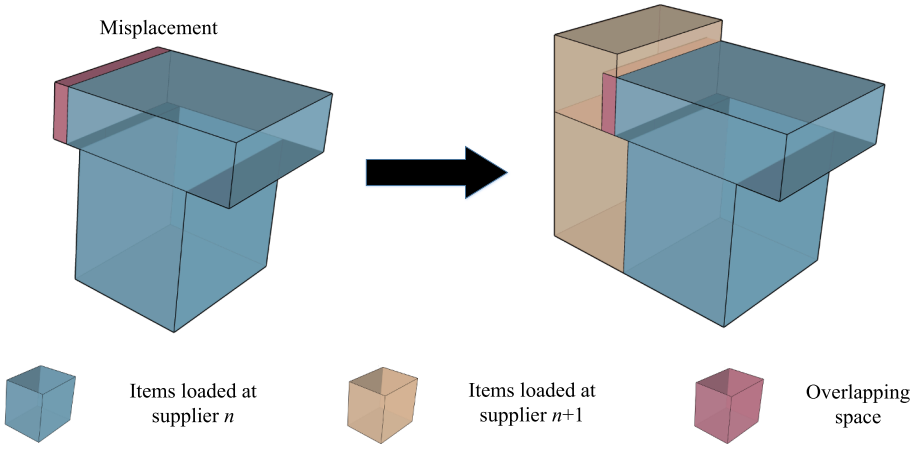

Fig. 3. Schematic diagram of an overlap due to misplacement and the partial-support constraint.

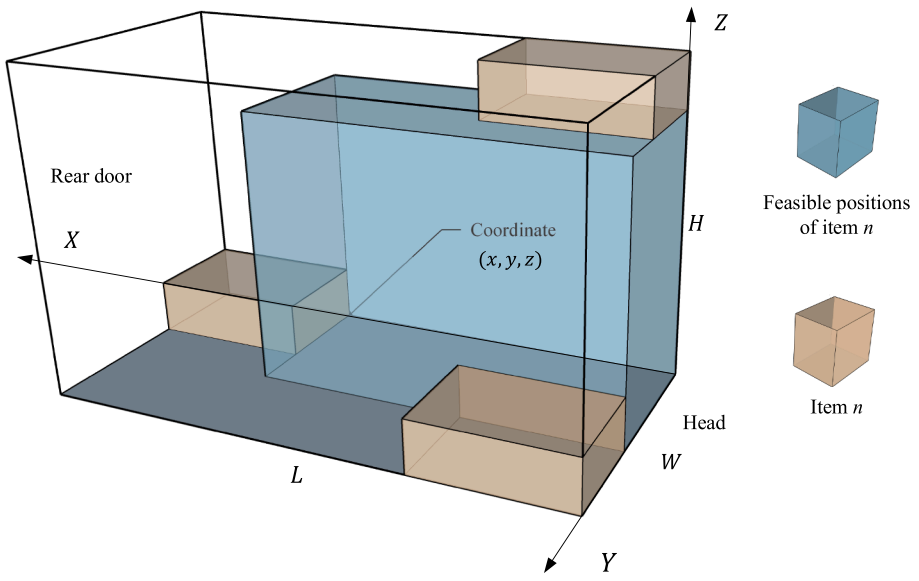

Fig. 4. Schematic diagram of the coordinate system.

continuous variables t_i^k, ω_i^k indicate the arrival and waiting times of vehicle k at node i. Let variable α_{ink}^{xyz} indicate the placement of item n at supplier i. As shown in Fig. 4, if the front-left-bottom corner point of an item is placed at (x, y, z) of vehicle k, then $\alpha_{ink}^{xyz} = 1$, and 0 otherwise.

A schematic diagram of the possible positions is given in Fig. 4. Let sets X_{in}, Y_{in}, Z_{in} be the possible X-, Y-, Z- coordinates/positions of item n at supplier i along the X, Y, Z axes, respectively. Specifically, $X_{in} = \{0, 1, 2, \cdots, L - l_{in}\}, Y_{in} = \{0, 1, 2, \cdots, W - w_{in}\}, Z_{in} = \{0, 1, 2, \cdots, H - h_{in}\}$. Item n at supplier i can only be placed at possible positions to avoid overlapping with the vehicle. Moreover, if the item is placed within vehicle k, then $\sum_{x \in X_{in}} \sum_{y \in Y_{in}} \sum_{z \in Z_{in}} \alpha_{ink}^{xyz} = 1$.

The loading sequence constraint requires (1) all items must be directly loaded into the vehicle through a sequence of straight movements parallel to the length of the vehicle without repositioning of other items; (2) items cannot be stacked on the item that has

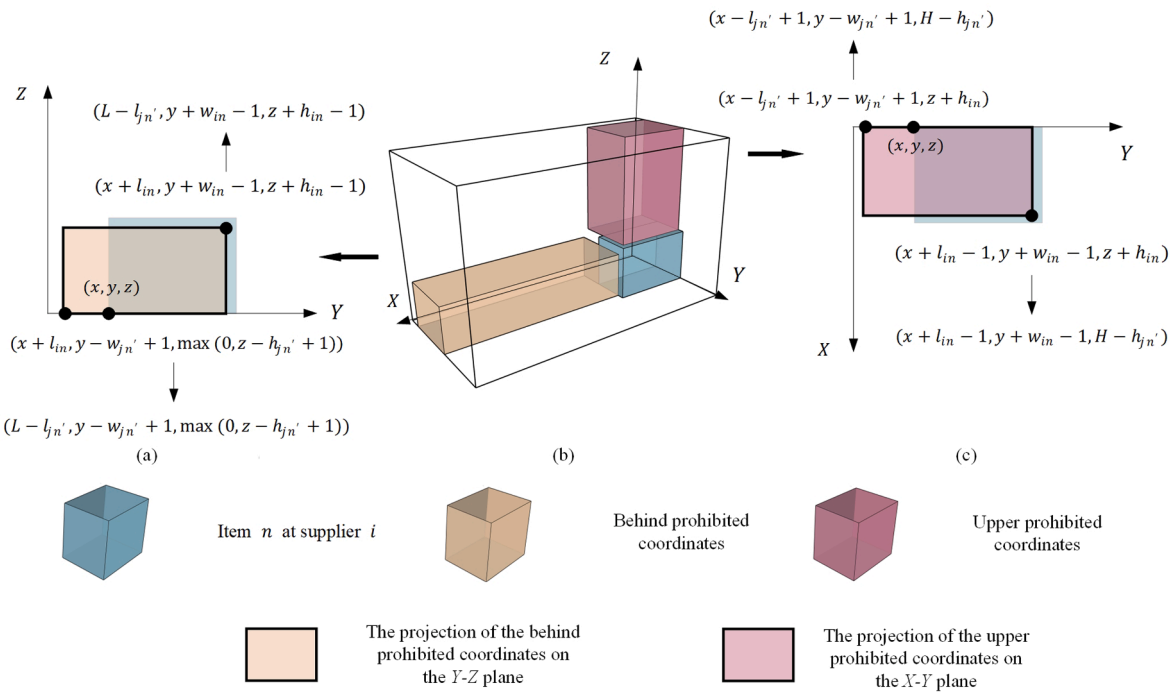

Fig. 5. Schematic diagram of prohibited coordinates.

not been placed (items belonging to the supplier to be visited later.). To implement the loading sequence constraint, a set of coordinates $\theta_{injn'}^{xyz}$ is introduced in this paper, that is, a set of prohibited coordinates. If supplier i is visited later than supplier j (by the same vehicle) and item n at supplier i is placed at (x,y,z), then item n' at supplier j cannot be placed at any coordinates $(x',y',z') \in \theta_{injn'}^{xyz}$. A schematic diagram of $\theta_{injn'}^{xyz}$ is shown in Fig. 5**(b)**. Obviously, if item n' is placed at any coordinates $(x',y',z') \in \theta_{injn'}^{xyz}$, it must occupy the movement space of item n or stack on item n. Specifically, $\theta_{injn'}^{xyz}$ is composed of two parts: the upper prohibited coordinates and the behind prohibited coordinates. The former indicates that items cannot be stacked on items that have not been loaded (Fig. 5**(c)**). The latter indicates that the loading operations of items should not be affected by the already loaded items (Fig. 5**(a)**). The boundaries of the prohibited coordinates are also presented in Fig. 5**(a) and (c)**.

The indices, parameters, sets, and variables of the 3L-PCVRP model are presented in Table 2.

Before introducing the mathematical model of 3L-PCVRP, the following three key assumptions are made to facilitate the modeling process.

1. Vehicles are homogeneous, and each vehicle can only be used once.
2. Items at each supplier can be loaded into each vehicle.
3. Each item has a fixed vertical orientation, but horizontal 90° rotation is allowed.

With the above definitions and assumptions, the 3L-PCVRP model **M1** is presented below.

Table 2
Definitions of indices, parameters, sets, and variables of 3L-PCVRP.

Indices	
x,x',y,y',z,z'	*X*-, *Y*-, *Z*- coordinates
k	Index of vehicles
i,j,o,d	Indices of nodes, origin, and destination
n,n'	Indices of items
Parameters	
N_i	Number of items at supplier i
$l_{in},w_{in},h_{in},\sigma_{in}$	Length, width, height, and fragility of item n at supplier i
L,W,H	Length, width, and height of vehicles
Q	Capacity of vehicles
q_i	Weight of the items at supplier i
c_{ij}	Transportation cost from node i to j
t_{ij}	Transportation time from node i to j
g_i	Loading time at supplier i
$[T_i^l,T_i^u]$	Time window of node i
$L_{injn'}^{xx'}$	Length of the projection of the overlap between item n at supplier i (placed at (x,y,z)) and n' at supplier j (placed at $(x',y',z-h_{jn'})$) on the *X* -axis $L_{injn'}^{xx'} = \max(0,\min(x+l_{in},x'+l_{jn'})-\max(x,x'))$
$W_{injn'}^{yy'}$	Width of the projection of the overlap between item n at supplier i (placed at (x,y,z)) and n' at supplier j (placed at $(x',y',z-h_{jn'})$) on the *Y* -axis $W_{injn'}^{yy'} = \max(0,\min(y+w_{in},y'+w_{jn'})-\max(y,y'))$
M	A large positive value
Sets	
K	Set of vehicles
U_i	Set of original and rotated items at supplier i
V	Set of suppliers
X_{in},Y_{in},Z_{in}	Sets of the possible *X*-, *Y*-, *Z*- coordinates of item n at supplier i, respectively
$\theta_{injn'}^{xyz}$	Set of prohibited coordinates Upper prohibited coordinates: $x' \in \{x' \in X_{jn'} \mid x-l_{jn'}+1 \le x' \le x+l_{in}-1\}$, $y' \in \{y' \in Y_{jn'} \mid y-w_{jn'}+1 \le y' \le y+w_{in}-1\}$, $z' \in \{z' \in Z_{jn} \mid z' \ge z+h_{in}\}$ Behind prohibited coordinates: $x' \in \{x' \in X_{jn'} \mid x' \ge x+l_{in}\}$, $y' \in \{y' \in Y_{jn'} \mid y-w_{jn'}+1 \le y' \le y+w_{in}-1\}$, $z' \in \{z' \in Z_{jn} \mid \max(0,z-h_{jn'}+1) \le z' \le z+h_{in}-1\}$
Variables	
ζ_{ij}^k	0–1 variable, =1 if vehicle k passes arc (i,j); =0 otherwise
α_{ink}^{xyz}	0–1 variable, =1 if item n at supplier i is placed in vehicle k at coordinate (x,y,z); =0 otherwise
t_i^k	Continuous variable, indicating the time when vehicle k arrives at node i
ω_i^k	Continuous variable, indicating the time that vehicle k waiting at node i
$\varphi_{kijnn'}^{xyzx'y'z'}$	Auxiliary 0–1 variable, =1 if $\alpha_{ink}^{xyz} = \alpha_{jn'k}^{x'y'z'} = 1$; = 0 otherwise
ρ_{ij}^k	Auxiliary continuous variable, $\rho_{ij}^k = t_i^k - t_j^k$
$\mu_{kijnn'}^{xyzx'y'z'}$	Auxiliary continuous variable, $\mu_{kijnn'}^{xyzx'y'z'} = \begin{cases} 0 & if\ \varphi_{kijnn'}^{xyzx'y'z'} = 0 \\ \rho_{ij}^k & if\ \varphi_{kijnn'}^{xyzx'y'z'} = 1 \end{cases}$

M1.

Objective function:

$$\min \sum_{k\in K}\sum_{i\in V\cup\{o\}}\sum_{j\in V\cup\{d\}} c_{ij}\zeta_{ij}^{k} \tag{1}$$

Subject to:

$$\sum_{j\in V, j\neq i}\zeta_{ij}^{k}-\sum_{j\in V, j\neq i}\zeta_{ji}^{k}=\begin{cases}1, \forall k\in K, i=o\\ -1, \forall k\in K, i=d\\ 0, otherwise\end{cases} \tag{2}$$

$$t_i^k+g_i+\omega_i^k+t_{ij}-M\left(1-\zeta_{ij}^{k}\right)\le t_j^k, \forall k\in K, i\in V\cup\{o\}, j\in V\cup\{d\} \tag{3}$$

$$t_i^k+g_i+\omega_i^k+t_{ij}+M\left(1-\zeta_{ij}^{k}\right)\ge t_j^k, \forall k\in K, i\in V\cup\{o\}, j\in V\cup\{d\} \tag{4}$$

$$\sum_{i\in V\cup\{o\}}\sum_{j\in V} q_j\zeta_{ij}^{k}\le Q, \forall k\in K \tag{5}$$

$$\sum_{k\in K}\sum_{i\in V\cup\{o\}}\zeta_{ij}^{k}=1, \forall j\in V \tag{6}$$

$$\sum_{i\in V\cup\{o\}}\zeta_{ij}^{k}\le\sum_{x\in X_{jn}}\sum_{y\in Y_{jn}}\sum_{z\in Z_{jn}}\alpha_{jnk}^{xyz}+\sum_{x\in X_{jn+N_j}}\sum_{y\in Y_{jn+N_j}}\sum_{z\in Z_{jn+N_j}}\alpha_{jn+N_jk}^{xyz}, \forall k\in K, j\in V, n\in\{1,2,\cdots,N_j\} \tag{7}$$

$$\sum_{x\in X_{jn}}\sum_{y\in Y_{jn}}\sum_{z\in Z_{jn}}\alpha_{jnk}^{xyz}+\sum_{x\in X_{jn+N_j}}\sum_{y\in Y_{jn+N_j}}\sum_{z\in Z_{jn+N_j}}\alpha_{jn+N_jk}^{xyz}\le 1, \forall k\in K, j\in V, n\in\{1,2,\cdots,N_j\} \tag{8}$$

$$\sum_{j\in V}\sum_{n\in U_j}\left\{\sum_{x\in X_{jn}|\max(0,x'-l_{jn}+1)\le x\le x'}\right\}\left\{\sum_{y\in Y_{jn}|\max(0,y'-w_{jn}+1)\le y\le y'}\right\}$$

$$\left\{\sum_{z\in Z_{jn}|\max\left(0,z'-h_{jn}+1\right)\le z\le z'}\right\}\alpha_{jnk}^{xyz}\le 1$$

$$, \forall k\in K, x'\in\{0,1,2,\cdots,L\}, y'\in\{0,1,2,\cdots,W\}, z'\in\{0,1,2,\cdots,H\} \tag{9}$$

$$\sum_{j\in V}\left\{\sum_{n'\in U_j|z-h_{jn'}\ge 0}\right\}\sum_{x'\in X_{jn'}}\sum_{y'\in Y_{jn'}} L_{injn'}^{xx'}W_{injn'}^{yy'}\alpha_{jn'k}^{x'y'\left(z-h_{jn'}\right)}\le l_{in}w_{in}\alpha_{ink}^{xyz}$$

$$, \forall k\in K, i\in V, n\in U_i, x\in X_{in}, y\in Y_{in}, z\in Z_{in}/\{0\} \tag{10}$$

$$\sigma_{in}\alpha_{ink}^{xy(z'-h_{in})}\ge\sigma_{jn'}\alpha_{jn'k}^{x'y'z'}, \forall k\in K, i,j\in V, n\in U_i, n'\in U_j, x'\in X_{in}, x\in\left[x'-l_{in}+1, x'+l_{jn'}-1\right],$$

$$y'\in Y_{in}, y\in\left[y'-w_{in}+1, y'+w_{jn'}-1\right], z'\in Z_{jn'}/\{0\} \tag{11}$$

$$\left(t_i^k-t_j^k\right)\times\alpha_{ink}^{xyz}\times\alpha_{jn'k}^{x'y'z'}\le 0, \forall k\in K, i,j\in V, i\neq j, n\in U_i, n'\in U_j, x\in X_{in}, y\in Y_{in}, z\in Z_{in}$$

$$, (x', y', z')\in\theta_{injn'}^{xyz} \tag{12}$$

$$\zeta_{ij}^{k}\in\{0,1\}, \forall k\in K, i\in V\cup\{o\}, j\in V\cup\{d\} \tag{13}$$

$$\alpha_{ink}^{xyz}\in\{0,1\}, \forall k\in K, i\in V, n\in U_i, x\in X_{in}, y\in Y_{in}, z\in Z_{in} \tag{14}$$

$$t_i^k\in[T_i^l, T_i^u], \forall k\in K, i\in V\cup\{o\}\cup\{d\} \tag{15}$$

$$\omega_i^k\in[0, T_d^u], \forall k\in K, i\in V \tag{16}$$

Eq. (1) is an objective function that aims to minimize the overall transportation cost. Eq. (2) is the standard flow-balance constraint. Eqs. (3) and (4) are the time window constraints that compute the waiting and arrival times. Eq. (5) is the vehicle capacity constraint. Eq. (6) requires that each supplier has to be visited. Eq. (7) is the no-split constraint, which requires the vehicle to load all items at the

suppliers that it passes. Eq. (8) is the orientation constraint, which allows the rotation of each item: vehicles can load the item or the rotated item. Eq. (9) is the non-overlap constraint derived from the study of Junqueira et al. (2012). Eq. (10) is the full-support constraint derived from the study of Junqueira et al. (2012). Eq. (11) represents the fragility constraint. The bottom of any non-fragile item cannot contact the top of any fragile item. Eq. (12) represents the loading sequence constraint. If vehicle k visits supplier i later than supplier j ($t_i^k - t_j^k \geq 0$) and item n at supplier i is placed at (x,y,z), then item n' at supplier j cannot be placed at $(x',y',z') \in \theta_{injn'}^{xyz}$. That is, if $t_i^k - t_j^k \geq 0$ and $\alpha_{ink}^{xyz} = 1$, then $\alpha_{jn'k}^{x'y'z'} = 0$. Eqs. (13)–(16) are the variable domain constraints.

Eq. (12) is a nonlinear constraint. To linearize the model, the linearization of Eq. (12) is presented in Eqs. (17)–(22). Eqs. (17)–(18) indicate that $\varphi_{kijnn'}^{xyzx'y'z'} = \alpha_{jn'k}^{x'y'z'} \times \alpha_{ink}^{xyz}$: if $\alpha_{ink}^{xyz} = \alpha_{jn'k}^{x'y'z'} = 1$, $\varphi_{kijnn'}^{xyzx'y'z'} = 1$, and 0 otherwise. Eq. (19) computes ρ_{ij}^k between any suppliers. Eqs. (20)–(21) indicate that $\mu_{kijnn'}^{xyzx'y'z'} = \rho_{ij}^k \times \varphi_{kijnn'}^{xyzx'y'z'}$: if $\varphi_{kijnn'}^{xyzx'y'z'} = 1$, $\mu_{kijnn'}^{xyzx'y'z'} = \rho_{ij}^k$, and 0 otherwise. Eq. (22) is the final linear form of Eq. (12). By converting $\alpha_{jn'k}^{x'y'z'} \times \alpha_{ink}^{xyz}$ into variable $\varphi_{kijnn'}^{xyzx'y'z'}$ and $\left(t_i^k - t_j^k\right) \times \varphi_{kijnn'}^{xyzx'y'z'}$ into variable $\mu_{kijnn'}^{xyzx'y'z'}$, $\left(t_i^k - t_j^k\right) \times \alpha_{ink}^{xyz} \times \alpha_{jn'k}^{x'y'z'}$ is converted to $\mu_{kijnn'}^{xyzx'y'z'}$, and Eq. (12) is replaced by Eqs. (17)–(22). Eqs. (23)–(25) are the new variable domain constraints.

$$\begin{cases} \varphi_{kijnn'}^{xyzx'y'z'} \leq \alpha_{ink}^{xyz} \\ \varphi_{kijnn'}^{xyzx'y'z'} \leq \alpha_{jn'k}^{x'y'z'} \end{cases}, \forall k \in K, i,j \in V, i \neq j, n \in U_i, n' \in U_j, x \in X_{in}, y \in Y_{in}, z \in Z_{in}, (x',y',z') \in \theta_{injn'}^{xyz} \tag{17}$$

$$\varphi_{kijnn'}^{xyzx'y'z'} \geq \alpha_{ink}^{xyz} + \alpha_{jn'k}^{x'y'z'} - 1, \forall k \in K, i,j \in V, i \neq j, n \in U_i, n' \in U_j, x \in X_{in}, y \in Y_{in}, z \in Z_{in}$$

$$, (x',y',z') \in \theta_{injn'}^{xyz} \tag{18}$$

$$\rho_{ij}^k = t_i^k - t_j^k, \forall k \in K, i,j \in V, i \neq j \tag{19}$$

$$-T_d^u \varphi_{kijnn'}^{xyzx'y'z'} \leq \mu_{kijnn'}^{xyzx'y'z'} \leq T_d^u \varphi_{kijnn'}^{xyzx'y'z'}$$

$$, \forall k \in K, i,j \in V, i \neq j, n \in U_i, n' \in U_j, x \in X_{in}, y \in Y_{in}, z \in Z_{in}, (x',y',z') \in \theta_{injn'}^{xyz} \tag{20}$$

$$-T_d^u \left(1 - \varphi_{kijnn'}^{xyzx'y'z'}\right) + \rho_{ij}^k \leq \mu_{kijnn'}^{xyzx'y'z'} \leq T_d^u \left(1 - \varphi_{kijnn'}^{xyzx'y'z'}\right) + \rho_{ij}^k$$

$$, \forall k \in K, i,j \in V, i \neq j, n \in U_i, n' \in U_j, x \in X_{in}, y \in Y_{in}, z \in Z_{in}, (x',y',z') \in \theta_{injn'}^{xyz} \tag{21}$$

$$\mu_{kijnn'}^{xyzx'y'z'} \leq 0, \forall k \in K, i,j \in V, i \neq j, n \in U_i, n' \in U_j, x \in X_{in}, y \in Y_{in}, z \in Z_{in}, (x',y',z') \in \theta_{injn'}^{xyz} \tag{22}$$

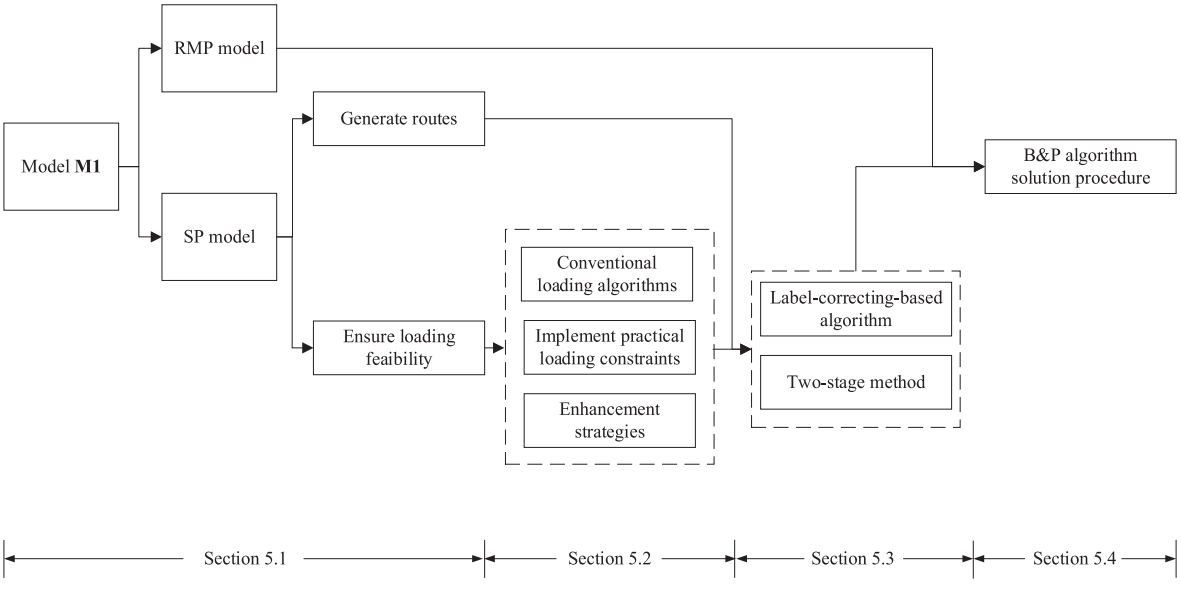

Fig. 6. Arrangement of Section 5.

$$\varphi_{kijnn'}^{xyzx'y'z'} \in \{0,1\}, \forall k \in K, i,j \in V, i \neq j, n \in U_i, n' \in U_j, x \in X_{in}, y \in Y_{in}, z \in Z_{in}, (x', y', z') \in \theta_{injn'}^{xyz} \tag{23}$$

$$\rho_{ij}^{k} \in [-T_d^u, T_d^u], \forall k \in K, i,j \in V, i \neq j \tag{24}$$

$$\mu_{kijnn'}^{xyzx'y'z'} \in [-T_d^u, T_d^u], \forall k \in K, i,j \in V, i \neq j, n \in U_i, n' \in U_j, x \in X_{in}, y \in Y_{in}, z \in Z_{in}, (x', y', z') \in \theta_{injn'}^{xyz} \tag{25}$$

5. Solution method

The branch-and-price-based (B&P) algorithm has been widely used to solve complex and large-scale problems (Jin et al., 2015; Wang et al., 2020; Pan et al., 2021; Chi et al., 2022). For 3L-PCVRP, model **M1** can be decomposed into a set-covering model (restricted master problem, RMP) to select routes and an elementary shortest path problem with resource and loading constraints model (ESPPRLC) (subproblem, SP) to generate independent routes in terms of routing and loading constraints (Mahvash et al., 2017; Rajaei et al., 2022). Clearly, if each route can be optimized independently, the complexity of solving the 3L-PCVRP can be significantly reduced. Therefore, an improved B&P algorithm is to be developed to solve model **M1**.

As presented in Fig. 6, this section is divided into four parts. In Section 5.1, model **M1** is decomposed by using the Dantiz–Wolf (D–W) decomposition method. The structures of the decomposed models are then analyzed. In Section 5.2, two loading algorithms are developed with efficiency and accuracy as the first objectives respectively. First, conventional loading algorithms are described. Second, the methods to implement the fragility and loading sequence constraints (11)–(12) are introduced. Finally, enhancement strategies to improve the algorithm efficiency are developed. In Section 5.3, a label-correcting-based algorithm (LCA) to solve SP is developed, and a two-stage method to call the loading algorithms at different frequencies is presented to investigate the loading feasibility of each route. In Section 5.4, the overall solution procedure of the improved B&P algorithm is given.

5.1. Model decomposition

By using the D–W decomposition method, model **M1** is decomposed into RMP model **M2** (set-covering model) and SP model **M3** (ESPPRLC model). The additional index, parameters, set, and variables in models **M2** and **M3** are listed in Table 3.

RMP model M2.

Objective function:

$$\min \sum_{h \in H} c_h \varepsilon_h \tag{26}$$

Eq. (26) corresponds to Eq. (1).

Subject to:

$$\sum_{h \in H} \tau_j^h \varepsilon_h \geq 1, \forall j \in V \tag{27}$$

$$\sum_{h \in H} \varepsilon_h \leq |K| \tag{28}$$

$$\varepsilon_h \in \{0,1\}, \forall h \in H \tag{29}$$

Eq. (6) of model **M1** is transformed into its set covering formulation (Eq. (27)) to enhance efficiency (Chabrier, 2006; Liu et al.,

Table 3
Definitions of additional index, parameters, set, and variables in **M2** and **M3.**

Index	
h	Index of alternative routes
Parameters	
τ_j^h	If route h passes supplier j, then $\tau_j^h = 1$, and 0 otherwise
ϕ_j	Value of the dual variable of Eq. (27)
μ	Value of the dual variable of Eq. (28)
c_h	Transportation cost of route h
Set	
H	Set of alternative routes
Variables	
ε_h	0–1 route selection variable, = 1 if route h is selected: = 0 otherwise
$t_j^{\|H\|+1}$	Continuous variable indicating the time when the vehicle of route $\|H\|+1$ arrives at node j
$\zeta_{ij}^{\|H\|+1}$	0–1 arc selection variable, = 1 if the vehicle of route $\|H\|+1$ passes arc (i,j); = 0 otherwise
$\alpha_{in\|H\|+1}^{xyz}$	0–1 placement variable, =1 if item n at supplier i is placed into the vehicle of route $\|H\|+1$ at coordinate (x,y,z); = 0 otherwise
$\omega_i^{\|H\|+1}$	Continuous variable, indicating the time that the vehicle of route $\|H\|+1$ waiting at node i

2018). Eq. (28) restricts the number of vehicles to be used. Eq. (29) is the variable domain constraint.

SP model M3.

Objective function:

$$\min \sum_{i\in V\cup\{o\}}\sum_{j\in V\cup\{d\}} c_{ij}\zeta_{ij}^{|H|+1} - \sum_{j\in V}\phi_j \sum_{i\in V\cup\{o\}} \zeta_{ij}^{|H|+1} - \mu \tag{30}$$

Model **M3** is subject to the routing (Eq. (2)–(5)), loading (Eq. (7)–(12)), and related variable domain (Eq. (13)–(16)) constraints. More specifically, the variables ζ_{ij}^{k}, t_i^k, ω_j^k, α_{ink}^{xyz} are replaced by $\zeta_{ij}^{|H|+1}$, $t_j^{|H|+1}$, $\omega_J^{|H|+1}$, $\alpha_{in|H|+1}^{xyz}$. Model **M3** generates feasible routes with a negative reduced cost, which is strongly NP-hard (Rajaei et al., 2022).

The B&P algorithm first creates a set of columns/routes (H) that ensure the feasibility of RMP. To generate feasible routes, the RMP model is linearized. Second, the RMP model is solved to optimality. The dual variables of the RMP model are integrated with the objective function of the SP model as fixed parameters, i.e., the reduced cost (Eq. (30)). Third, the SP model is solved, and new columns (τ_j^h) are updated. The iteration continues until no new column with negative reduced cost can be identified. Finally, the branch-and-bound approach is implemented to obtain the integer solutions.

After decomposing model **M1**, all the complex constraints are retained in SP model **M3**. The constraints in model **M3** can be divided into routing and loading constraints. If the loading constraints are relaxed, then model **M3** generates the shortest routes that satisfy the capacity and time window constraints, and 3L-PCVRP is relaxed to the CVRPTW (CVRP with time window constraints). The formulations of models **M2** and **M3** (without loading constraints) have proven to be efficient in solving the CVRP or CVRPTW (Danna and Le Pape, 2005; Lan et al., 2019; Wang et al., 2021). Therefore, the key to ensuring the efficiency and accuracy of the B&P algorithm is to solve the SP model **M3** efficiently, that is, generate routes in terms of loading and routing constraints. The focus of this study on the solution method is to present an efficient method for solving model **M3**. The following two sections present the algorithms used to solve model **M3**.

5.2. Loading algorithms

The loading constraints in model **M3** are to compute the locations of the items in the vehicle of route $|H|+1$ and ensure the feasibility of each route. In most studies on 3L-CVRP (Gendreau et al., 2006; Fuellerer et al., 2010; Mahvash et al., 2017; Rajaei et al., 2022), a loading algorithm is commonly used to judge the loading feasibility after generating a route, that is, solving a CLP with a given vehicle (container) and items. Because of the CLP complexity, it is impossible to generate optimal solutions within an acceptable time (Zhao et al., 2016; Nascimento et al., 2021). Therefore, heuristic algorithms have been widely developed to solve the CLP efficiently (Gajda et al., 2022). However, a loading algorithm with high computational efficiency may misjudge some feasible routes as infeasible, leading to high transportation costs. A loading algorithm that judges all feasible routes as feasible requires a long computing time (Fuellerer et al., 2010). To overcome these difficulties, two loading algorithms with efficiency and accuracy as the primary objectives

Table 4
Definitions of indices, sets, and parameters to describe the GHA and the TRSA.

Indices	
$s, s^{'}, s_{\lambda}$	Index of spaces
$b, b^{'}, b_{\lambda}$	Index of blocks
ϱ	Index of evaluation functions
$c, c^{'}$	Index of tree nodes (partially filled container/partial solution)
λ	Index of pairs
Sets	
E	Set of spaces
E_c	Set of spaces of tree node c
B_1, B_2	Sets of unplaced blocks (B_1) and placed/loaded blocks (B_2)
B_1^c, B_2^c	Sets of unplaced blocks (B_1^c) and placed/loaded blocks (B_2^c) of tree node c
I_b	Set of the items composing block b
C	Set of tree nodes
G_c	Set of space and block pairs of tree node c
$\theta_{bb^{'}}^{xyz}$	Set of prohibited coordinates
Parameters	
vu	Volume utilization
ϑ	Tree width
u_s	u_s = *true* if space s is usable, and *false* otherwise
v_{ϱ}	Highest value of the evaluation function ϱ
(x_s, y_s, z_s)	Coordinate of space s (front-left-bottom corner point)
d_s	Distance between the coordinate of space s and the origin: $d_s = \sqrt{x_s \times x_s + y_s \times y_s + z_s \times z_s}$
(x_b, y_b, z_b)	Coordinate of the placed block b
L_s, W_s, H_s	Length, width, and height of space s
$l_b, w_b, h_b, \sigma_b, t_b$	Length, width, height, fragility, and sequence of block b
e_{bs}^{ϱ}	Value of evaluation function ϱ of block b and space s

were developed to complement each other in this study. According to the loading constraints (Eqs. (8)–(12)) of model **M3**, the greedy heuristic algorithm (GHA) with high efficiency and the tree search algorithm (TRSA) with high accuracy, proposed by Ren et al. (2011), were selected for further development in this study.

The following sections cover the conventional formulation of the loading algorithms in short, the method to implement the fragility and loading sequence constraints, and the enhancement strategies.

5.2.1. Conventional loading algorithms

The GHA and the TRSA load a set of items into a container/vehicle to maximize the volume utilization. If all the items in the set are loaded into the vehicle, volume utilization is maximized. Therefore, the GHA and the TRSA can be used to investigate the loading feasibility of the routes directly.

Table 4 lists the indices, sets, and parameters to describe the GHA and the TRSA.

Before introducing the procedures of the GHA and TRSA, the following terms are defined: 1. Block: A group of identical items of the same size and orientation, as shown in Fig. 7. 2. Space: Space is part of a container, which has coordinates and dimensions. Each space does not contain any block: part or whole. If any block $b \in B_1$ can be placed into space s, then space s is called a usable space; otherwise, it is called an unusable space. 3. Adjacent: Two spaces s and s' are adjacent if they satisfy the following conditions: $z_s = z_{s'} \wedge y_s + W_s = y_{s'} \wedge (x_s < x_{s'} < x_s + L_s \vee x_{s'} < x_s < x_{s'} + L_{s'})$ or $z_s = z_{s'} \wedge x_s + L_s = x_{s'} \wedge (y_s < y_{s'} < y_s + W_s \vee y_{s'} < y_s < y_{s'} + W_{s'})$. 4. Tree node (TN): A TN indicates a partially filled container or a partial solution that contains a set of placed blocks, spaces, and unplaced blocks. 5. Prohibited coordinates: Because identical items are grouped as blocks in the loading algorithms, the set of prohibited coordinates $\theta^{xyz}_{inj n'}$ is changed into $\theta^{xyz}_{bb'}$ in the loading algorithms. If the sequence of block b is later than that of block b' and b is placed at (x,y,z), then block b' cannot be placed at any coordinate $(x',y',z') \in \theta^{xyz}_{bb'}$.

The GHA procedure can be described as initialization, space selection, block selection, space splitting, space merging, and feasibility testing. Five conventional evaluation functions $e^1_{bs} - e^5_{bs}$ are used to compute the "fitness" between space s and block b, as given in Eqs. (31)–(35). The details of each step are as follows:

$$e^1_{bs} = -\min((L_s - l_b), (W_s - w_b), (H_s - h_b)) \tag{31}$$

$$e^2_{bs} = w_b \times h_b \tag{32}$$

$$e^3_{bs} = l_b \times h_b \tag{33}$$

$$e^4_{bs} = l_b \times w_b \tag{34}$$

$$e^5_{bs} = l_b \times w_b \times h_b \tag{35}$$

Step 1. Initialize. In this step, all items are packed as blocks and added to set B_1. Set $B_2 = \varnothing$. The loading space of the container is set as the first usable ($u_0 = true$) space and added to set E. Turn to Step 2.

Step 2. Select space. In this step, the usable space nearest to the origin (with minimum d_s) is selected from E as the current space s' to place the next block. Turn to Step 3.

Step 3. Select block. In this step, the values of the five evaluations of each possible block $b \in B_1$ (i.e., block b can be placed in space s': $l_b \le L_{s'}, w_b \le W_{s'}, h_b \le H_{s'}$) and space s' $e^1_{bs'} - e^5_{bs'}$ are computed. Subsequently, block b' is selected according to the following criteria in sequence: Block b' is placed at $(x_{s'}, y_{s'}, z_{s'})$. For each $b \in B_1$, if $I_b \cap I_{b'} \neq \varnothing$, remove b from B_1. Add b' into B_2. Remove space s' from E. If $B_1 = \varnothing$, the optimal solution is found; otherwise, turn to Step 4.

Main criterion: The largest value of e^1_{bs}.

Tie-breaker 1: The largest value of e^2_{bs}.

Tie-breaker 2: The largest value of e^3_{bs}.

Tie-breaker 3: The largest value of e^4_{bs}.

Step 4. Split space. In this step, new spaces are generated by splitting space s' according to the newly placed block b'. The method to generate new spaces is shown in Fig. 8 (a). The coordinates and sizes of the three new spaces are upper space: $(x_{s'}, y_{s'}, z_{s'} + h_{b'})$, $l_{b'}, w_{b'}$,

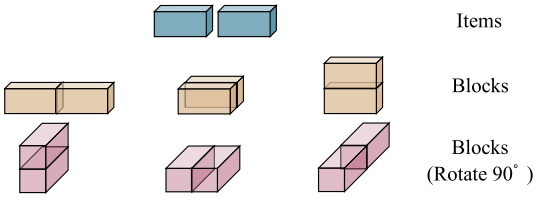

Fig. 7. Schematic diagram of possible blocks.

$H_{s'}-h_{b'}$, right space: $(x_{s'}, y_{s'}+w_{b'}, z_{s'}), l_{b'}, W_{s'}-w_{b'}, H_{s'}$, and behind space: $(x_{s'}+l_{b'}, y_{s'}, z_{s'}), L_{s'}-l_{b'}, W_{s'}, H_{s'}$ (Ren et al., 2011; Li et al., 2022). Turn to Step 5.

Step 5. Merge Spaces. In this step, the adjacent spaces in E are merged into new spaces under the two conditions below. The method to merge the spaces is shown in Fig. 8 (b). Turn to Step 6.

(1) If two spaces s and s' have a common edge along the X- or Y-axis ($z_s = z_{s'}, x_s = x_{s'}, y_s + W_s = y_{s'}, L_s = L_{s'}$, or $z_s = z_{s'}, y_s = y_{s'}, x_s + L_s = x_{s'}, W_s = W_{s'}$), remove spaces s and s' from E. The new space s'' merged from spaces s and s' is added into E (Eley, 2002; Ren et al., 2011; Li et al., 2022).

(2) If two unusable spaces s and s' do not have a common edge along the X- or Y-axis, remove spaces s and s' from E. Three new spaces s'', s''_1, s''_2 merged from spaces s and s' are added into E (Ren et al., 2011; Li et al., 2022).

Step 6. Test feasibility. For each space $s \in E$, if any block $b \in B_1$ can be placed into s ($l_b \le L_s, w_b \le W_s, h_b \le H_s$), then space s is marked as usable $u_s = true$. If $|B_1| \rangle 0$ and there is usable space $s \in E$. Turn to Step 2. If $|B_1| = 0$ or there is no usable space $s \in E$, return the volume utilization (Eq. (36)).

$$vu = \frac{\sum_{b \in B_2} l_b \times w_b \times h_b}{L \times W \times H} \tag{36}$$

The TRSA can be regarded as an extension of the GHA, which allows the selection of different blocks for each space. In the TRSA, a tree node (TN) c concludes the unplaced block set B_1^c, placed block set B_2^c, and empty space set E_c. Obviously, the TN contains all the information required in Step 2 of the GHA.

The solution procedure of the conventional TRSA can be briefly described as initialization, node generation, space splitting, space merging, feasibility testing, and node removal. The details of each step are as follows:

Step 1. Initialize. In addition to the work performed in Step 1 of the GHA, TN c is defined to indicate the initial solution of B_1^c, B_2^c, E_c. Turn to Step 2.

Step 2. Generate nodes. For each TN c in C, multiple pairs of spaces and blocks (each pair concludes a usable space and a block that can be placed in that space) that obtain the highest value (v_ϱ) in five corresponding evaluations are selected to generate new TNs. Let G_c indicate the set of pairs, where each pair $\lambda \in G_c$ represents a space s_λ and a block b_λ. For each λ in G_c, generate a new TN c': $E_{c'} = E_c / \{s_\lambda\}, B_1^{c'} = B_1^c / \{b_\lambda\}, B_2^{c'} = B_2^c \cup \{b_\lambda\}$. For each $b \in B_1^{c'}$, if $I_b \cap I_{b_\lambda} \neq \varnothing$, remove b from $B_1^{c'}$. Add c' into C. Remove c from C. Turn to Step 3.

Step 3. Split spaces. For each λ in G_c, split space $s_\lambda \in E_c$ according to Step 4 of the GHA. New spaces are added into $E_{c'}$. Turn to Step 4.

Step 4. Merge spaces. For each c in C, for each s in E_c, and for each s' in E_c, merge spaces s, s' and add new spaces into $E_{c'}$ according to the methods described in Step 5 of the GHA. Turn to Step 5.

Step 5. Test feasibility. For each c in C and for each s in E_c, mark space s by using the method described in Step 6 of the GHA. Turn to Step 6.

Step 6. Remove nodes. For each c in C, apply the GHA to fill the residual empty spaces of node c (starting from Step 2 of the GHA). If all the blocks in B_1^c can be loaded by the GHA, return its volume utilization. Otherwise, let the volume utilization computed by the GHA be the evaluation value of c. Rank C according to the evaluation value of each TN $c \in C$. If there is no usable space in each c in C, then the highest evaluation value is returned. Otherwise, in C, at most ϑ TNs that rank among the highest evaluation values are retained. Turn to Step 2.

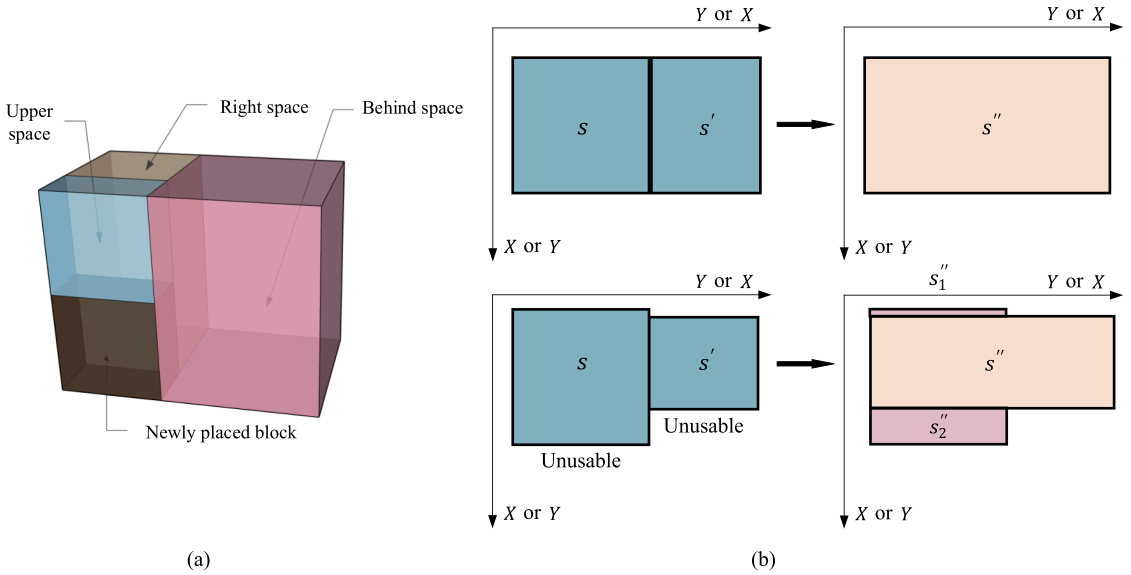

Fig. 8. Schematic diagram of the space-splitting method (a) and space-merging method (b).

5.2.2. Methods to implement the practical constraints

The TRSA has high accuracy, whereas the GHA has high efficiency (Ren et al., 2011). However, the conventional algorithms do not consider the fragility and loading sequence constraints (Eqs. (11)–(12)). In this section, the methods to implement the fragility and loading sequence constraints are presented.

5.2.2.1. Method to implement the fragility constraint. As described in Eq. (11), the bottom of the non-fragile items cannot contact the top of fragile items. Therefore, the fragility constraint can be implemented in the block selection step (Step 3) of the GHA and the node generation step (Step 2) of the TRSA.

When judging whether non-fragile block b ($\sigma_b = 1$) can be placed at (x_s, y_s, z_s), i.e., space s, the top of each placed fragile block $b' \in B_2$ ($\sigma_{b'} = 0$) is tested whether it has contact with the bottom of block b. Specifically, if there is a fragile block $b' \in B_2$ that satisfies $h_{b'} + z_{b'} = z_s \wedge (x_s < x_{b'} < x_s + l_b \vee x_{b'} < x_s < x_{b'} + l_{b'}) \wedge (y_s < y_{b'} < y_s + w_b \vee y_{b'} < y_s < y_{b'} + w_{b'})$, then non-fragile block b cannot be placed in space s.

5.2.2.2. Method to implement the loading sequence constraint. As described above, the loading sequence constraint can be described in two parts: the upper prohibited coordinates and the behind prohibited coordinates. The upper prohibited coordinates can be easily implemented like the fragility constraint (block selection step (Step 3) of the GHA and the node generation step (Step 2) of the TRSA): When judging whether block $b \in B_1$ can be placed in space s, if any block $b' \in B_2$ with sequence $t_{b'} > t_b$ satisfies $h_{b'} + z_{b'} = z_s \wedge (x_s < x_{b'} < x_s + l_b \vee x_{b'} < x_s < x_{b'} + l_{b'}) \wedge (y_s < y_{b'} < y_s + w_b \vee y_{b'} < y_s < y_{b'} + w_{b'})$, then block b cannot be placed in space s.

To implement the behind prohibited coordinates, a new set of blocks is introduced, namely virtual blocks. When block b is placed at (x,y,z), a virtual block vb is created, which is also placed at (x,y,z), but the size of which is $L-x, w_b, h_b$. The virtual block occupies the "movement space" of block b (Shown in Fig. 9 (a)). If virtual block vb overlaps with block b' that is placed at (x', y', z'), block b' must occupy the movement space of block b, and block b' cannot be placed at (x', y', z') unless it is loaded later than block b ($t_{b'} \geq t_b$). Moreover, coordinate (x', y', z') belongs to the prohibited coordinates $\theta^{xyz}_{bb'}$ (Fig. 9 (b)).

According to the discussion above, the method to implement the behind prohibited coordinates is as follows. Let B_3 or B_3^c indicate a set of virtual blocks. When judging whether block b' with sequence $t_{b'}$ can be placed at (x_s, y_s, z_s) (block selection step (Step 3) of the GHA and the node generation step (Step 2) of the TRSA), each virtual block with sequence $t_{vb} > t_{b'}, vb \in B_3$ is tested to determine whether it overlaps with block b'. If vb overlaps with b', then block b' cannot be placed in space s. The three steps to judge the overlap are as follows: It is worth noting that the functions hold because the two blocks/spaces overlap if and only if the projections of the two blocks/spaces on three planes ($X-Y, Y-Z, X-Z$) and the three axes overlap simultaneously.

Step 1: Judge Z-axis. If $z_{vb} < z_{b'} < z_{vb} + h_{vb}$ or $z_{b'} < z_{vb} < z_{b'} + h_{b'}$, turn to Step 2. Otherwise, return false (non-overlap).

Step 2: Judge Y-axis. If $y_{vb} < y_{b'} < y_{vb} + w_{vb}$ or $y_{b'} < y_{vb} < y_{b'} + w_{b'}$, turn to Step 3. Otherwise, return false (non-overlap).

Step 3: Judge X-axis. If $x_{vb} < x_{b'} < x_{vb} + l_{vb}$ or $x_{b'} < x_{vb} < x_{b'} + l_{b'}$, return true (overlap). Otherwise, return false (non-overlap).

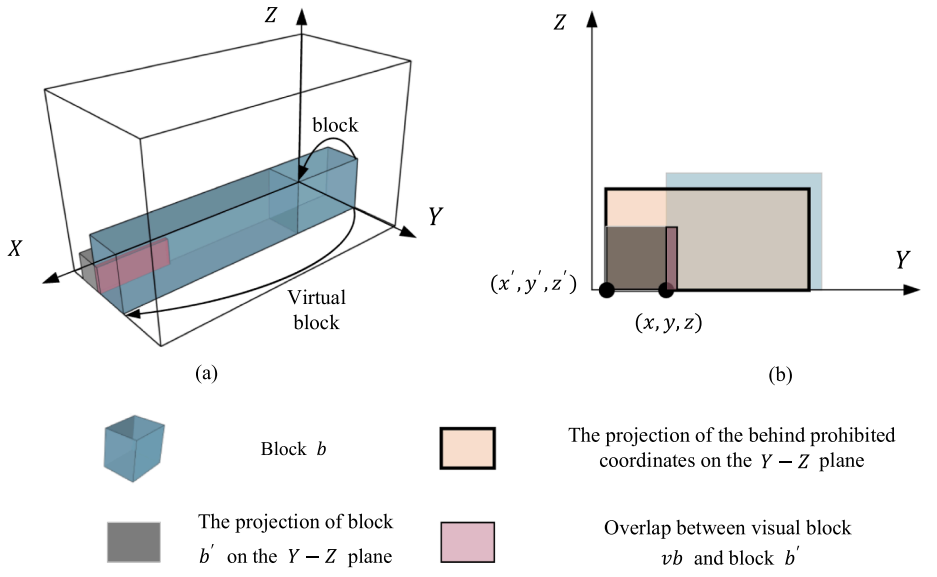

Fig. 9. Schematic diagrams of the virtual block (a) and overlap (b).

5.2.3. Enhancement strategies

Sections 5.2.1–5.2.2 describe conventional algorithms and methods to implement the practical constraints. In this section, strategies to improve the GHA and the TRSA are proposed.

According to the GHA and TRSA procedures, the main factors affecting accuracy and efficiency are the evaluation functions (Eqs. (31)–(35)) and the space-merging method. For the former, an effective method to evaluate and select block(s) can reduce the volume of unusable spaces. For the latter, an effective method to generate usable space(s) from unusable spaces can offer better positions to place blocks and reduce the volume of unusable spaces. Conventional evaluation functions are designed to maximize the utilization of one to three dimensions in each space. The conventional space-merging method is designed to merge unusable spaces into usable spaces without affecting the existing usable spaces. This method ensures that there is no overlap between any placed blocks and the empty spaces.

By observing numerous experiments and analyzing the principles of conventional methods, two enhancement strategies are proposed in this study. The first is to add an evaluation function to evaluate the potential of placing a block in a space to generate usable spaces. The second is to enhance the space-merging method to generate more usable spaces and avoid invalid repeated calculations.

5.2.3.1. New evaluation function. As shown in Fig. 10**(a)**, when blocks are stacked, usable spaces can be generated by merging two adjacent upper spaces. Although the conventional evaluation functions consider the height dimension of the blocks, the merging of the upper spaces is not considered (Fig. 10**(b)**). To generate usable upper spaces to improve the volume utilization, a new evaluation function e^6_{bs} is proposed in Eqs. (37)–(38). The e^6_{bs} evaluates the potential of block b (to be placed in space s) to generate usable upper spaces.

$$e^6_{bs} = \max(e^6_{bs0}, e^6_{bs1}, \cdots, e^6_{bss'}, \cdots, e^6_{bs|E|}) \tag{37}$$

$$e^6_{bss'} = \begin{cases} \begin{cases} (min(y_{s'} + W_{s'}, y_s + w_b) - max(y_{s'}, y_s)) \times (L_{s'} + l_b) \\ (min(x_{s'} + L_{s'}, x_s + l_b) - max(x_{s'}, x_s)) \times (W_{s'} + w_b) \end{cases} \begin{matrix} if x_{s'} + L_{s'} = x_s \\ if y_{s'} + W_{s'} = y_s \end{matrix} if z_{s'} = z_s + h_b \\ 0 \qquad otherwise \\ 0 otherwise \end{cases} \tag{38}$$

5.2.3.2. Improved space-merging method. To avoid affecting usable spaces, the existing space-merging method marks each space as usable or unusable according to whether any block can be placed in that space (Ren et al., 2011; Li et al., 2022). However, the conventional method may also lead to negative situations. Three typical examples are presented in Fig. 11 (all figures in Fig. 11 are projections on the $X-Y$ plane) to illustrate these negative situations. First, as shown in Fig. 11**(a)**, unusable space s_1 and usable space s_2 are not allowed to merge because they do not share a common edge. Consequently, a space that can place block b is not generated. Another complex situation is shown in Fig. 11**(b)**. In Fig. 11**(b)** (1), two unusable spaces s_1 and s_2 are retained in space set E. The two spaces are then merged into a new usable space s'_2, and the original spaces s_1, s_2 are replaced by the newly generated spaces s'_1, s'_2, s'_3. Next, in Fig. 11**(b)** (2), a new space s'_4 is added to set E because of the placement of another block (not shown in Fig. 11**(b)**). Subsequently, spaces s'_4 and s'_3 are merged into space s'' because they share a common edge. Consequently, there is no space to place block b because usable space s'_2 is not allowed to merge with unusable space s''. Third, as shown in Fig. 11**(c)**, spaces s_1 and s_2 are merged at Iteration #1, and spaces s'_1, s'_2, s'_3 are generated. Next, at Iteration #2, spaces s'_2 and s'_3 are merged into space s''_1, and spaces s'_2, s'_3 are replaced by spaces s''_1, s''_2. Consequently, at Iteration #3, spaces s'''_1, s'''_2 (same as spaces s_1, s_2) are obtained again because of the merge of spaces s'_1 and s''_1. That is, after three iterations, the original spaces are obtained again. However, these calculations are ineffective.

This ineffectiveness may lead to long computing time and low accuracy. Therefore, an improved method allows to merge all adjacent spaces and records the previous merge schemes of spaces is developed. In this method, spaces are no longer marked as usable or unusable. The new conditions for merging spaces are as follows:

(1) If two adjacent spaces s and s' have a common edge along the X- or Y-axis ($z_s = z_{s'}, x_s = x_{s'}, y_s + W_s = y_{s'}, L_s = L_{s'}$, or $z_s = z_{s'}, y_s = y_{s'}, x_s + L_s = x_{s'}, W_s = W_{s'}$), remove spaces s and s' from E and add the new space s'' merged from spaces s and s' into E (identical to conventional condition (1).).

(2) If two adjacent spaces s and s' do not have a common edge along the X- or Y-axis and have not merged before, generate a new

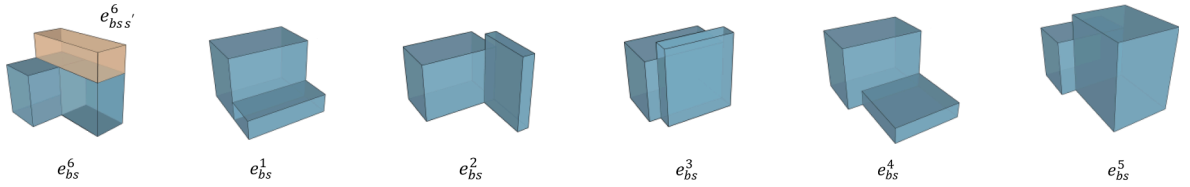

Fig. 10. Schematic diagrams of the effective upper space and e^6_{bs} (a), and other evaluation functions (b).

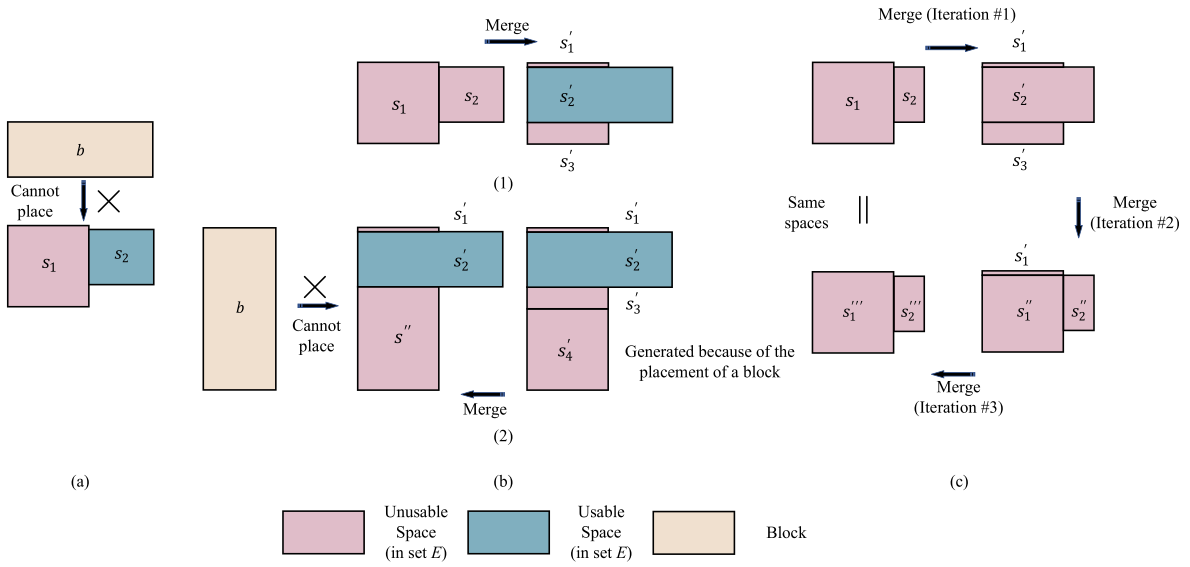

Fig. 11. Schematic diagrams of the shortcomings of the conventional space-merging method.

space s'' (Fig. 12**(a)**) and add it into E. The small spaces (s_1'', s_2'' in Fig. 8**(b)**) are also added into E.

The advantages of the improved method are illustrated in Fig. 12**(a)–(c)**. All figures in Fig. 12 are projections on the $X-Y$ plane. The small spaces are not shown in Fig. 12. First, as shown in Fig. 12**(a)**, because the new method does not mark spaces as usable or unusable, spaces s_1 and s_2 are merged, and the new space s_1' is retained in E with the original spaces s_1, s_2. Block b can be placed in s_1'. Second, as shown in Fig. 12**(b)** (1), spaces s_1 and s_2 are retained in E after merging the two spaces into a new space s_1'. After the new space s_2' is introduced in E, spaces s_1 and s_2' are merged, as shown in Fig. 12**(b)** (2). Space s'' to place block b is then generated. Finally, as shown in Fig. 12**(c)**, the new method records the previously merged spaces. Therefore, spaces are not generated repeatedly, thereby reducing computing time.

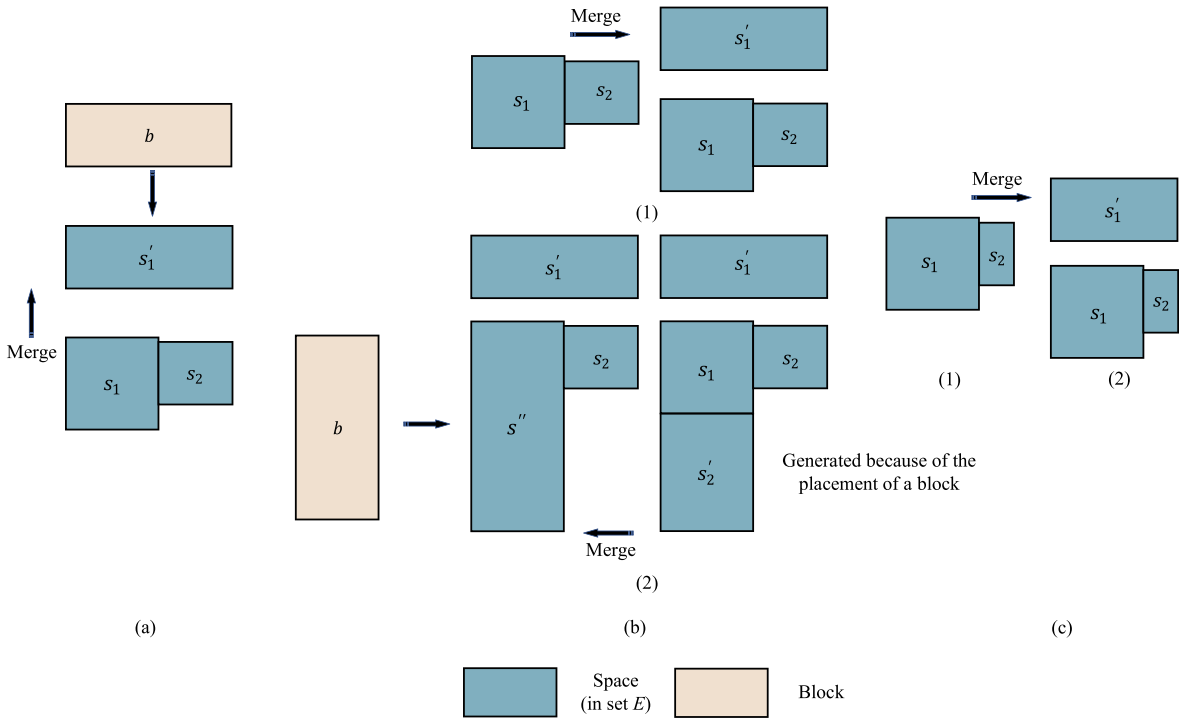

Fig. 12. Schematic diagrams of the improved space-merging method.

For the GHA, the improved space-merging method replaces Steps 5 and 6 as new Step 5 (Merge spaces). For the TRSA, Step 5 is removed and Step 4 is replaced with the improved space-merging method.

Obviously, the spaces in E may overlap because of the retention of the original spaces and addition of new spaces. To ensure that the placed blocks do not overlap, the method of judging overlap (described in Section 5.2.2.2) is also applied in the GHA (Step 3. Select block) and the TRSA (Step 2. Generate nodes). Once a block has been placed, any other spaces that overlap with the block are removed from E.

The GHA and the TRSA with the aforementioned optimized steps are called the improved GHA (IGHA) and the improved TRSA (ITRSA), respectively. The exact solution procedures for the IGHA and ITRSA are given in Appendices A and B, respectively. The following section will introduce the method to solve SP model **M3** as well as the two-stage method to employ the loading algorithms.

5.3. Label-correcting-based algorithm

In Section 5.1, model **M1** is decomposed into RMP model **M2** and SP model **M3**. The challenges of the B&P algorithm are also discussed in Section 5.1, that is, solving SP model **M3** efficiently. In Section 5.2, the two loading algorithms are presented, and their improvement is discussed. In this section, the algorithm for solving model **M3** and the method for calling the two loading algorithms are described.

After decomposition, SP model **M3** is obtained to generate routes in terms of routing and loading constraints. In previous studies, the main method to generate routes/paths (incomplete routes) involved two main steps. First, by applying a conventional routing algorithm, a route/path was generated in terms of transportation cost and routing constraints. Second, an efficient loading algorithm was used to investigate the loading feasibility of the route (Gendreau et al., 2006; Fuellerer et al., 2010; Mahvash et al., 2017; Rajaei et al., 2022).

In this study, the algorithm to solve model **M3** is derived from the label-correcting-based algorithm 2 (LCA2) proposed by Rajaei et al. (2022). In LCA2, the number of labels stored at each node is restricted (at most ϵ labels with lowest costs are retained), thereby ensuring the efficiency of the algorithm. The feasibility of each label (incomplete or complete route) was investigated by using the EPHA. Infeasible labels are then removed. The conventional method of employing a single loading algorithm with high efficiency can reduce the overall computing time but might misjudge some routes. Therefore, in order to improve accuracy under the premise of ensuring efficiency, LCA2 that employs two loading algorithms (LCA2ETLA) is proposed below. The method to call the two loading algorithms (IGHA and ITRSA) is called the two-stage method.

The method of generating labels in LCA2ETLA is the same as that of LCA2. When a label is generated, the IGHA with high efficiency is used to investigate its feasibility first. If the IGHA judges the label as infeasible, then the ITRSA is called with a certain probability. Parameter $\zeta \in [0, M]$ is defined as the given parameter to judge the rate to call the ITRSA according to the problem scale. Let parameter δ indicate the vu obtained by the IGHA. Let parameter φ_1 indicate the number of labels that are identified as infeasible by the IGHA but feasible by the ITRSA. Let parameter φ_2 indicate the number of times that the ITRSA is called. Let $\varphi_1 = \varphi_2 = 1$ if the ITRSA has not been called. Let parameter ξ indicate the rate used to call the ITRSA, as shown in Eq. (39). Once the IGHA judges the label as infeasible, ξ is updated, and a random number $\rho \in [0, 1]$ is generated. If $\rho \leq \xi$, the ITRSA is called to investigate the feasibility of the label. The label is then removed if it is infeasible.

$$\xi = \zeta \times \delta \times \max(\frac{\varphi_1}{\varphi_2}, 0.7) \tag{39}$$

Let $r, r^{'}$ indicate the index of feasible labels. For each label, there are at most three resources to be considered: weight, volume, and arrival time. Therefore, each label r includes five elements: node, cost, weight, volume, and arrival time, which are indicated by (i, c_r, q_r, v_r, t_r) respectively. If two labels $r, r^{'}$ at a node satisfy the following condition, then label $r^{'}$ is dominated and removed.

Domination condition: $r, r^{'}$ are two labels at node i, and the associated attributes are (i, c_r, q_r, v_r, t_r) and $(i, c_{r^{'}}, q_{r^{'}}, v_{r^{'}}, t_{r^{'}})$, respectively. Label r dominates $r^{'}$ if and only if $c_r \leq c_{r^{'}}, q_r \leq q_{r^{'}}, v_r \leq v_{r^{'}}, t_r \leq t_{r^{'}}$.

It is worth noting that multiple routes are generated and updated in LCA2ETLA. The exact solution procedure for LCA2ETLA is given in Appendix C.

5.4. Solution procedure

The solution procedure of the B&P algorithm is detailed in **Algorithm 1**.

Step 1 (Initialize). First, parameters $\epsilon, \zeta, \vartheta, \varphi_1, \varphi_2$ are initialized. Second, the initial routes of the B&P algorithm are generated (Mahvash et al., 2017). Finally, set the iteration number $\gamma = 0$.

In Step 2 (Update dual variables), RMP model **M2** is solved. If the solution is an integer solution, update the value of the objective function as the upper and lower bound values (UB, LB) and turn to Step 6. Otherwise, update the dual variables ϕ_j, μ and the lower bound value LB (the objective function of model **M2**).

Step 3 (Update vehicle routes). LCA2ETLA is applied to generate vehicle routes (solve model **M3**) with negative reduced costs. If no routes with negative reduced costs are generated, update the value of the objective function of model **M2** as the lower bound value (LB) and turn to Step 4, and Step 2 otherwise.

In Step 4 (Update upper bound solution), model **M2** with integer constraints (All variables are set as 0–1 variables.) is solved. Update the value of its objective function as the upper bound value (UB). Turn to Step 5.

In Step 5 (Branch), the integer solution is obtained by applying the most-infeasible-method (Achterberg et al., 2005) and the depth-

first-strategy (Mahvash et al., 2017). Set a variable with the fractional part closest to 0.5 as 1. Turn to Step 6.

Step 6 (Solution quality evaluation and termination condition test). Update LB^{γ}, UB^{γ}, gap (Eqs. (40)–(42)). $\gamma = \gamma + 1$. The termination of the iteration process depends on the condition (i.e., maximum iteration number or termination gap). If the termination condition is not satisfied, turn to Step 2.

$$UB^{\gamma} = \min\{UB^{\gamma-1}, UB\} \tag{40}$$

$$LB^{\gamma} = \max\{LB^{\gamma-1}, LB\} \tag{41}$$

$$\text{gap} = \frac{UB^{\gamma} - LB^{\gamma}}{UB^{\gamma}} \tag{42}$$

Algorithm 1:B&P algorithm

Step 1: Initialize
Initialize ς, $\vartheta \in [15, 50]$, $\epsilon \in [2, 5]$, and $\varphi_1 = \varphi_2 = 1$
Initialize vehicle routes
Set the iteration number $\gamma = 0$
Step 2: Update dual variables
Call the Gurobi 9.5 solver to optimize model **M2**
If the solution is an integer solution
Update upper and lower bound solutions and UB, LB
Turn to **Step 6**
Else
Update ϕ_j, μ, LB
Turn to **Step 3**
End if
Step 3: Update vehicle routes
Call the LCA2ETLA (Appendix C) to generate vehicle routes in terms of routing and loading constraints
If new routes with negative reduced costs are found
Turn to **Step 2**
Else
Update LB
Turn to **Step 4**
End if
Step 4: Update upper bound solution
Call the Gurobi 9.5 solver to optimize model **M2** with integer constraints (without branching constraints)
Update UB
Turn to **Step 5**
Step 5: Branch
For each $h \in H$
If $0 < \varepsilon_h < 1$
Update $\varpi_h = \varepsilon_h - 0.5$
End if
End for
Select the variable with minimum $|\varpi_h|$ to be the variable to be branched
Add $\varepsilon_h = 1$ as the branching constraint
Turn to **Step 6**
Step 6: Solution quality evaluation and termination condition test
Update UB^{γ}, LB^{γ} and gap
If the gap is sufficiently small, or the iteration number γ reaches Ψ (300-500), then terminate the algorithm and output UB^{γ} and LB^{γ}; otherwise, $\gamma = \gamma + 1$, and go back to **Step 2**

6. Numerical experiments

This section describes the performance evaluation of the proposed algorithms on many instances. First, Section 6.1 describes the performance evaluation of the loading algorithms. In Section 6.1.1, the IGHA and the ITRSA are compared with the two existing algorithms. In Section 6.1.2, the ITRSA and the TRSA are compared to assess the effectiveness of the enhancement strategies. Section 6.2 describes the performance evaluation of **Algorithm 1**. In Section 6.2.1, the performance of **Algorithm 1** and the effectiveness of the two-stage method are validated by comparing with the off-the-shelf solver in small-scale instances. Next, in Section 6.2.2, the performance of **Algorithm 1** and the effectiveness of the two-stage method are validated by comparing with the existing algorithm in benchmark instances.

All the algorithms were implemented in C# on the Visual Studio 2019 platform. All the models and algorithms were tested on a desktop computer with an i9-9800H @ 2.30 GHz CPU and 32 GB RAM. To obtain optimal solutions, Gurobi 9.5 was used in the C# environment.

6.1. Experiments of the loading algorithms

The values of the parameters used in the experiments described in Sections 6.1.1 and 6.1.2 are listed in Table 5. Randomly generated items were strongly heterogeneous to verify the efficiency of the algorithms in various situations.

The nouns to describe the experiments presented in Section 6.1 are listed in Table 6.

6.1.1. Test of the loading algorithms

This section demonstrates the efficiency of the loading algorithms proposed in this paper. To ensure the effectiveness of the comparisons, two well-designed loading algorithms are compared with the algorithms proposed in this paper. The ITRSA and the IGHA are compared with EPHA-1 proposed by Rajaei et al. (2022) and EPHA-2 proposed by Mahvash et al. (2017) in 100 randomly generated instances. Owing to the limited space, the results for each instance are not listed. The average *vu* and CT of each algorithm under different conditions and other values for comparison are presented in Table 7.

The calculation results in Table 7 lead to the following conclusions: 1. The IGHA has high efficiency. Compared with EPHA-1, the IGHA can reduce the computing time by up to 88.76%. Compared with EPHA-2, the IGHA can reduce the computing time by up to 96.46%. The IGHA can be called one thousand times within one second (0.84 s). 2. The ITRSA has high accuracy. Compared with EPHA-1, the ITRSA improves the utilization by 8.31% on average and up to 10.61%. Compared with EPHA-2, the ITRSA improves the utilization by 8.12% on average and up to 10.46%. The ITRSA can use space more optimally than the two developed algorithms. 3. The IGHA and the ITRSA can be applied simultaneously to complement each other. By frequently calling the IGHA with high computational efficiency and the ITRSA with high accuracy under a certain probability, the overall accuracy can be improved on the premise of ensuring efficiency (the two-stage method described in Section 5.3).

In conclusion, the results in Table 7 confirm the high computational efficiency of the IGHA and the high accuracy of the ITRSA. Moreover, the accuracy of the IGHA and the computational efficiency of the ITRSA are also acceptable (less than 0.5 s on average). The following section validates the effectiveness of the enhancement strategies.

6.1.2. Test of the enhancement strategies

As an extension of the GHA, the TRSA has higher accuracy and requires a longer computing time. Therefore, the ITRSA and the TRSA were compared. The effectiveness of the enhancement strategies was tested in 100 randomly generated instances. The results for different tree widths (ϑ) are presented to provide more accurate comparisons of the efficiency and effectiveness.

Owing to the limited space, the results for each instance are not listed. The average *vu* and CT of each algorithm under different conditions as well as the IVU and CT gap, which indicate the degree of improvement in the accuracy and efficiency of the enhancement strategies, are presented in Table 8. The exact comparisons (AC) between the TRSA and the ITRSA are shown in Fig. 13. Fig. 13**(a)** shows a comparison between the *vu* values of the TRSA and ITRSA. Fig. 13**(b)** shows the IVU of each instance (with different tree widths). Fig. 13**(c)** shows a comparison between the CTs of the TRSA and ITRSA. Fig. 13**(d)** lists the CT gap of each instance (with different tree widths). It is worth noting that, in Fig. 13**(b) and (d)**, the blue, red, yellow, and purple lines indicate the IVU and the CT gap with $\vartheta = 50,100,150,200$, respectively. The green circles indicate the IVU and CT gap of the improved algorithm with $\vartheta = 50$ and the conventional algorithm with $\vartheta = 200$.

According to the results above, three conclusions are drawn: 1. The enhancement strategies improve the accuracy under the same conditions. Specifically, under the same conditions, the *vu* shows an average increase of 4.29% and a maximum increase of 29.97%. 2. The enhancement strategies improve the computational efficiency under the same conditions. Specifically, under the same conditions, the CT shows an average reduction of 33.31% and a maximum reduction of 81.80%. 3. The enhancement strategies reduce the tree width required by the ITRSA to obtain high-quality solutions. According to the previous description of the tree nodes, the more tree nodes are retained (the wider the tree width, the higher the value of ϑ), the more accurate the result will be, but the longer the computing time will be (Ren et al., 2011). As shown in "50–200" column in Table 8 and Fig. 13**(b) and (d)**, the ITRSA with at most 50 tree nodes can achieve better solutions than the TRSA with at most 200 tree nodes. Moreover, the ITRSA has considerably lower computing time than the TRSA. Specifically, the *vu* shows an average increase of 3.48% and a maximum increase of 28.15%. The CT shows an average reduction of 78.51% and a maximum reduction of 91.91%. In addition, in the improved algorithm, the *vu* increases slowly with the increase in the value of ϑ, which indicates that the high-quality solutions can be achieved with lower ϑ in the ITRSA.

The conclusions above indicate that the enhancement strategies provide a comprehensive enhancement of the efficiency and effectiveness. The objectives of the enhancement strategies are achieved (Mentioned in Section 5.2.3).

Table 5
Parameters setting.

Parameters	Definitions	Values
L, W, H	Length, width, and height of the container	30,30,30
l_n, w_n, h_n	Length, width, and height of item n	$[0.1L, 0.9L], [0.1W, 0.9W], [0.1H, 0.9H]$
σ_n, t_n	Fragility, sequence of item n	$[0, 1], [0, 6]$
	Number of items	35

Table 6
Definitions of symbols in Section 6.1.

Symbols	
vu	Volume utilization (Eq. (36))
IVU	Improved volume utilization of the corresponding algorithms (e.g., $\mathrm{IVU}_{\mathrm{EPHA\_1}} = \frac{(vu_{\mathrm{ITRSA}} - vu_{\mathrm{EPHA\_1}})}{vu_{\mathrm{ITRSA}}} \times 100\%$ or $\mathrm{IVU}_{\mathrm{TRSA}} = \frac{(vu_{\mathrm{ITRSA}} - vu_{\mathrm{TRSA}})}{vu_{\mathrm{ITRSA}}} \times 100\%$)
CT	Computing time
CT gap	Computing time gap between the improved algorithm and the conventional algorithm (e.g., $\mathrm{CT\ gap}_{\mathrm{EPHA\_1}} = \frac{(\mathrm{CT}_{\mathrm{EPHA\_1}} - \mathrm{CT}_{\mathrm{IGHA}})}{\mathrm{CT}_{\mathrm{EPHA\_1}}} \times 100\%$ or $\mathrm{CT\ gap}_{\mathrm{TRSA}} = \frac{(\mathrm{CT}_{\mathrm{TRSA}} - \mathrm{CT}_{\mathrm{ITRSA}})}{\mathrm{CT}_{\mathrm{TRSA}}} \times 100\%$)
BC	Basic constraints: orientation constraint, geometric feasibility constraints, no-split constraints, and full-support constraint
FC	Fragility constraint (practical constraint)
SC	Loading sequence constraint (practical constraint)
AC	All basic and practical constraints
$\mathrm{I}-\vartheta$	Improved loading algorithm - tree width
$\mathrm{O}-\vartheta$	Conventional loading algorithm - tree width

Table 7
Comparisons between the IGHA, ITRSA, EPHA-1, and EPHA-2.

Conditions	EPHA-1	EPHA-2	IGHA	ITRSA	EPHA-1		EPHA-2		IGHA		ITRSA	
					Average	Max	Average	Max	Average	Max	Average	Max
	Average vu (%)								CT (ms)			
BC	86.72	86.92	81.12	91.53	5.07	30.98	16.10	39.50	0.57	16.31	352.22	783.27
BC&FC	83.03	83.23	78.72	90.27	4.09	17.16	13.29	29.76	0.59	5.11	391.34	1747.09
BC&SC	79.67	79.84	74.61	87.91	2.71	9.86	9.21	22.79	1.13	6.55	423.55	1663.83
BC&SC&FC	77.59	77.72	72.95	86.80	2.42	7.47	8.16	20.32	1.05	6.85	442.05	1782.18
Average	81.75	81.93	76.85	89.13	3.57	16.37	11.69	28.09	0.84	8.71	402.29	1494.09
Maximum		–			5.07	30.98	16.10	39.50	1.13	16.31	442.05	1782.18
	Average IVU (%) (Compared with the ITRSA)				Average CT gap (%) (Compared with the IGHA)							
BC	5.26	5.04	11.37	–	88.76		96.46		–		–	
BC&FC	8.02	7.80	12.79	–	85.57		95.56		–		–	
BC&SC	9.37	9.18	15.13	–	58.30		87.73		–		–	
BC&SC&FC	10.61	10.46	15.96	–	56.61		87.13		–		–	
Average	8.31	8.12	13.81	–	72.31		91.72		–		–	
Maximum	10.61	10.46	15.96	–	88.76		96.46		–		–	

6.2. *Experiments of 3L-PCVRP*

In Section 6.1, the efficiency of the loading algorithms proposed in this study is validated through a comparison with existing algorithms. The effectiveness of the enhancement strategies is also validated. This section describes the validation of the efficiency of **Algorithm 1** and effectiveness of the two-stage method by comparing with those of 1. the off-the-shelf solver and 2. the existing algorithm.

Section 6.2.1 presents the solutions of randomly generated small-scale instances by the solver, **Algorithm 1**, and the B&P algorithm without the two-stage method (B&P-IGHA). The impact of each constraint on the complexity of model **M1** and the efficiency and effectiveness of the proposed algorithms are analyzed in Section 6.2.1.

Section 6.2.2 describes the solution of the benchmark instances solved using **Algorithm 1**, the B&P-IGHA, and the algorithm proposed by Mahvash et al. (2017) (C&G-HP). By comparing **Algorithm 1** proposed in this paper with the developed algorithm that has been proven to be effective, the efficiency and effectiveness of the proposed algorithms are validated. Moreover, by comparing **Algorithm 1** with the B&P-IGHA, the effectiveness of the two-stage method is validated.

The additional symbols used in Section 6.2 are defined in Table 9.

Table 8
Comparisons between the TRSA and the ITRSA.

	Conditions	Average *vu* (%)								IVU_{TRSA} (%) ($\vartheta_{ITRSA} - \vartheta_{TRSA}$)				
		ITRSA				TRSA								
	Tree width (ϑ)	50	100	150	200	50	100	150	200	50–50	100–100	150–150	200–200	50–200
vu	BC	89.69	89.83	89.86	89.89	87.09	87.75	88.07	88.29	2.59	2.08	1.79	1.60	1.40
	BC&FC	88.67	88.79	88.82	88.89	85.74	86.47	86.78	86.98	2.92	2.32	2.04	1.90	1.68
	BC&SC	85.43	85.71	86.05	86.20	80.74	81.35	81.77	82.03	4.69	4.36	4.28	4.17	3.40
	BC&SC&FC	84.34	84.90	85.14	85.22	75.92	76.33	76.57	76.91	8.42	8.57	8.57	8.31	7.43
	Average	87.03	87.31	87.47	87.55	82.37	82.97	83.30	83.55	4.66	4.33	4.17	3.99	**3.48**
	Maximum									29.97	29.21	**4.29** 28.15	28.15	**28.15**

	Conditions	Average CT (s)								CT gap_{TRSA} (%) ($\vartheta_{ITRSA} - \vartheta_{TRSA}$)				
		ITRSA				TRSA								
	Tree width (ϑ)	50	100	150	200	50	100	150	200	50–50	100–100	150–150	200–200	50–200
CT	BC	0.38	0.66	0.96	1.28	0.57	0.87	1.10	1.37	33.33	24.14	12.73	6.57	72.26
	BC&FC	0.49	0.94	1.37	1.83	1.06	1.67	2.09	2.57	53.77	43.71	34.45	28.79	80.93
	BC&SC	0.45	0.80	1.20	1.60	0.87	1.32	1.70	2.14	48.28	39.39	29.41	25.23	78.97
	BC&SC&FC	0.49	0.94	1.39	1.85	1.13	1.66	2.16	2.70	56.64	43.37	35.65	31.48	81.85
	Average	**0.45**	0.84	1.23	1.64	0.91	1.38	1.76	2.19	48.00	37.65	**33.31** 28.06	23.02	**78.51**
	Maximum	**1.77**	4.55	6.81	8.74	3.45	5.88	8.19	11.20	81.80	79.50	74.27	68.69	**91.91**

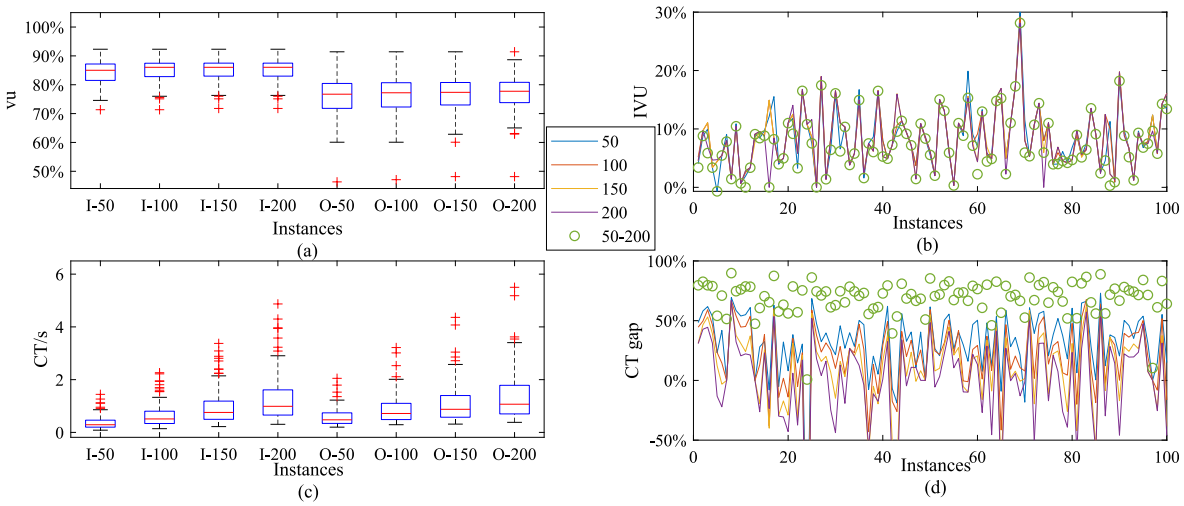

Fig. 13. Exact comparisons (BC&SC&FC) between the ITRSA and the TRSA.

Table 9
Definitions of symbols in Section 6.2.

Symbols	
TW	3L-PCVRP with time window constraint
NO TW	3L-PCVRP without time window constraint
CTG	Computing time between the improved algorithm and the conventional algorithm (s) (e.g., $CTG_{C\&G-HP} = CT_{Algorithm\,1} - CT_{C\&G-HP}$)
IG	Improved gap,IG $= \frac{UB_{Conventional_{Algorithm}} - UB_{Algorithm1}}{UB_{Conventional_{Algorithm}}} \times 100\%$

6.2.1. *Test of small-scale instances*

In Table 10, small-scale randomly generated instances with 10–15 nodes under different conditions are solved by Gurobi, **Algorithm 1**, and B&P-IGHA (using LCA2 to generate routes and the IGHA to investigate the loading feasibility of each route, i.e., $\varsigma = 0$). The locations, weights, time windows of nodes, and capacity of vehicles are derived from the Solomon benchmark instance C101. $L = W = H = 10$. The "0.1——0.9" and "0.2——0.6" in the second column of Table 10 indicate the size range of the items. At most, three items were loaded at each supplier. The bold values in Table 10 indicate the best solutions. The "-" indicates that the solution cannot be obtained by applying the corresponding method (out of memory).

The five main conclusions of the small-scale instances are: 1. **Algorithm 1** has high efficiency and can generate solutions in 21.15 s. 2. The computing time of all the instances with the time window constraints is significantly lower than that without the time window constraints. The time window constraints significantly reduce the searching space of the problem. Specifically, it reduces the number of labels created in LCA2ETLA and LCA2, thereby reducing the number of calls of the loading algorithms. The results also verify the existing research discussion on the complexity of 3L-CVRP/3L-PCVRP: The loading constraints are the most important factor that influences the complexity of the problem. 3. The loading sequence constraint has the greatest influence on the model scale. To linearize the loading sequence constraint, numerous variables are introduced. 4. **Algorithm 1** has high accuracy and can generate high-quality solutions under various conditions (optimal solutions are obtained in some instances). 5. The two-stage method can fully exploit the respective advantages of the ITRSA and IGHA. The B&P-IGHA can efficiently solve 3L-PCVRP (within 2.87 s). However, the low accuracy of the loading algorithm leads to high transportation costs. **Algorithm 1** can generate solutions with lower transportation costs within an acceptable timeframe.

6.2.2. *Test of the benchmark instances*

In this section, the benchmark instances derived from the Solomon benchmark instances and the 27 3L-CVRP instances provided by Gendreau et al. (2006) were tested. The locations, weights, time windows of nodes, and capacity of vehicles were derived from Solomon benchmark instance C101. The number, size, and fragility of the items at each node were derived from the 27 instances proposed by Gendreau et al. (2006).

The C&G-HP proposed by Mahvash et al. (2017) was proven to be accurate and efficient in their study by comparing it with state-of-the-art algorithms. Therefore, **Algorithm 1** is compared with the C&G-HP in this section. Moreover, to prove the advantages of applying the two-stage method, the B&P-IGHA is also used to solve the benchmark instances for comparison.

For space reasons, the results of the experiments are presented in Appendix D. The average values of the results are listed in Table 11. The intuitive comparisons of the IG and CTG under different conditions are presented in Table 12. The transportation costs (UB) of the different algorithms under different conditions are given in Fig. 14. The IGs under different conditions are presented in

Table 10
Comparisons between the solver, **Algorithm 1**, and the B&P-IGHA.

10 Nodes	0.1–0.9	Gurobi				**Algorithm 1**				B&P-IGHA ($\varsigma = 0$)			
		UB	LB	gap (%)	CT (s)	UB	LB	gap (%)	CT (s)	UB	LB	gap (%)	CT (s)
NO TW	BC	87.39	35.59	59.27	7200.00	**67.17**	64.56	3.89	2.56	70.74	70.74	0.00	0.58
	BC&FC	92.18	89.27	3.15	7200.00	**91.83**	88.95	3.14	10.34	110.85	105.74	4.61	1.71
	BC&SC			–		**70.74**	70.74	0.00	14.67	137.12	136.40	0.52	1.79
	BC&SC&FC			–		**99.15**	94.85	4.34	18.46	133.71	129.43	3.20	1.17
TW	BC	**67.17**	67.17	0.00	2207.00	**67.17**	67.17	0.00	1.53	70.74	70.74	0.00	0.48
	BC&FC	**97.78**	97.78	0.00	1553.00	**97.78**	95.43	2.41	2.50	131.15	124.82	4.83	0.48
	BC&SC			–		**99.15**	96.73	2.45	2.99	176.55	176.55	0.00	0.66
	BC&SC&FC			–		**99.15**	99.15	0.00	3.98	185.10	185.10	0.00	0.63
10 Nodes	0.2–0.6	UB	LB	gap (%)	CT (s)	UB	LB	gap (%)	CT (s)	UB	LB	gap (%)	CT (s)
NO TW	BC	153.29	77.05	49.74	7200.00	**122.31**	122.31	0.00	3.61	138.79	134.81	2.87	0.37
	BC&FC	154.53	62.32	59.67	7200.00	**129.03**	126.15	2.23	10.34	**129.03**	126.79	1.73	1.42
	BC&SC			–		**123.84**	123.54	0.24	13.41	135.93	135.93	0.00	1.66
	BC&SC&FC			–		**132.84**	132.84	0.00	16.71	135.93	135.93	0.00	0.68
TW	BC	**122.31**	122.31	0.00	2742.00	**122.31**	122.31	0.00	3.53	130.89	126.24	3.55	0.80
	BC&FC	**129.03**	129.03	0.00	1593.00	**129.03**	129.03	0.00	6.61	135.94	135.94	0.00	0.83
	BC&SC			–		**133.95**	133.95	0.00	9.79	136.12	134.91	0.88	0.91
	BC&SC&FC			–		**133.95**	133.95	0.00	6.03	136.12	136.12	0.00	0.72
15 Nodes	0.1–0.9	UB	LB	gap (%)	CT (s)	UB	LB	gap (%)	CT (s)	UB	LB	gap (%)	CT (s)
NO TW	BC	197.40	78.54	60.21	7200.00	**89.92**	89.92	0.00	3.59	101.87	101.87	0.00	0.98
	BC&FC	199.27	51.57	74.12	7200.00	**106.50**	106.50	0.00	17.52	131.00	129.08	1.46	2.87
	BC&SC			–		**106.50**	106.50	0.00	19.72	139.20	135.20	2.87	1.85
	BC&SC&FC			–		**106.50**	106.50	0.00	21.15	144.02	144.02	0.00	0.88
TW	BC	**125.48**	116.33	7.29	7200.00	**125.48**	125.48	0.00	4.42	127.71	127.71	0.00	0.59
	BC&FC	**127.71**	126.62	0.85	7200.00	**127.71**	127.71	0.00	13.15	135.11	135.11	0.00	0.43
	BC&SC			–		**131.32**	129.76	1.19	17.50	166.23	166.23	0.00	0.34
	BC&SC&FC			–		**140.80**	140.21	0.42	20.49	215.73	215.73	0.00	0.62

Table 11
Results of the benchmark instances (Average values).

Algorithms	Conditions		UB	gap (%)	CT (s)	Conditions		UB	gap (%)	CT (s)
Algorithm 1	NO TW	BC	707.58	2.62	743.44	TW	BC	719.00	3.76	152.65
		BC&FC	726.51	2.65	417.90		BC&FC	737.06	3.18	186.91
		BC&SC	765.82	2.52	342.00		BC&SC	807.40	2.49	128.10
		BS&FC&SC	784.20	2.10	376.70		BS&FC&SC	828.73	2.34	187.14
C&G-HP	NO TW	BC	731.44	3.32	842.55	TW	BC	770.29	3.28	88.34
		BC&FC	788.11	2.29	271.50		BC&FC	794.23	2.82	88.19
		BC&SC	867.08	0.97	216.73		BC&SC	947.14	1.65	78.49
		BS&FC&SC	898.54	1.30	215.29		BS&FC&SC	974.64	1.40	77.61
B&P-IGHA ($\varsigma = 0$)	NO TW	BC	783.35	2.34	134.11	TW	BC	822.02	2.22	0.21
		BC&FC	836.21	2.23	108.78		BC&FC	896.01	1.83	16.95
		BC&SC	1084.65	0.72	15.34		BC&SC	1329.81	0.42	3.63
		BS&FC&SC	1105.36	0.76	17.98		BS&FC&SC	1359.30	0.28	2.61

Table 12
Result comparisons.

Conditions			IG (%)			CTG (s)		
			Minimum	Average	Maximum	Minimum	Average	Maximum
Algorithm 1 & C&G-HP	NO TW	BC	0.06	3.54	12.48	−2033.60	−99.11	198.07
		BC&FC	1.10	7.92	14.83	35.14	146.40	452.01
		BC&SC	5.70	11.00	20.33	22.58	125.27	418.51
		BS&FC&SC	2.68	11.41	19.74	18.78	161.40	382.16
		Overall	–	**8.47**	–	–	**83.49**	–
	TW	BC	1.52	6.44	15.04	−3.68	56.90	442.18
		BC&FC	0.16	7.19	16.45	−4.39	57.97	314.72
		BC&SC	4.95	14.01	22.34	−157.30	49.61	440.29
		BS&FC&SC	4.16	13.45	**26.38**	−8.96	78.57	421.26
		Overall	–	**9.27**	–	–	**73.39**	–
	Overall		–	**9.37**	–	–	**73.05**	–

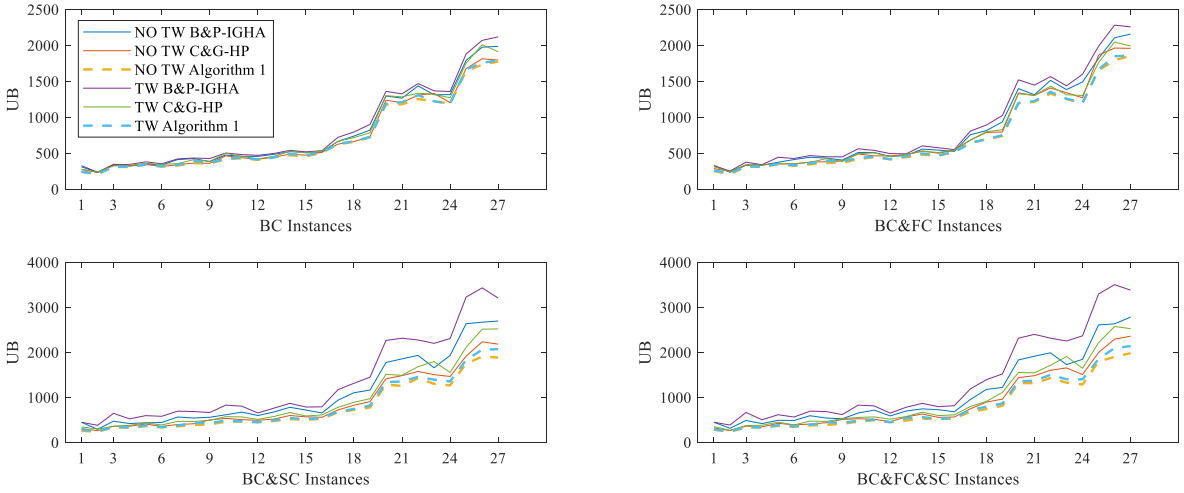

Fig. 14. Transportation costs of different algorithms under different conditions.

Fig. 15. The CTs of the algorithms under different conditions are given in Fig. 16.

The efficiency and effectiveness of **Algorithm 1** were verified in the benchmark instances. **Algorithm 1** yields best results in all instances. The computing time of **Algorithm 1** is also acceptable. In addition, three exact conclusions were drawn: 1. The two-stage method effectively combines the advantages of the two loading algorithms. Specifically, the IGHA with high efficiency can test the feasibility of most routes efficiently, whereas the ITRSA compensates for the relatively low accuracy of the IGHA. Compared with the C&G-HP, **Algorithm 1** brings a significant cost saving: 9.37% on average and 26.38% at maximum. Moreover, the additional time for obtaining high-quality solutions is also acceptable: 73.05 s on average. 2. **Algorithm 1** proposed in this paper can adapt to the actual needs of the company by simply adjusting the parameters. According to the results of the B&P-IGHA, the computational efficiency of the algorithm can be greatly improved by reducing the call probability (ζ) of the ITRSA. 3. With the time window constraint, the average transportation cost reduction (9.27%) is higher than that without the time window constraint (8.47%). As described above, the

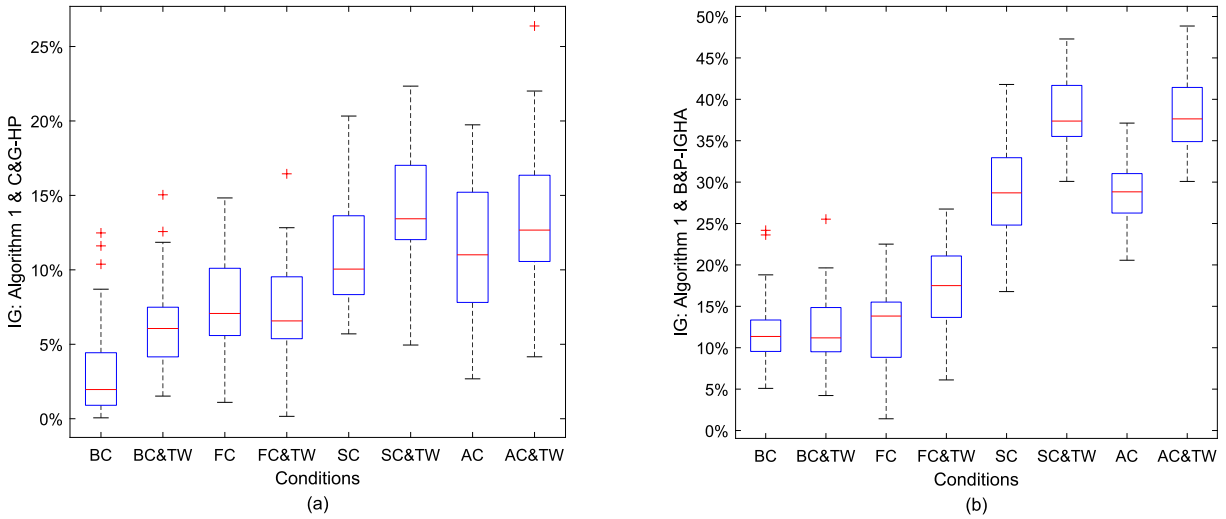

Fig. 15. IGs under different conditions.

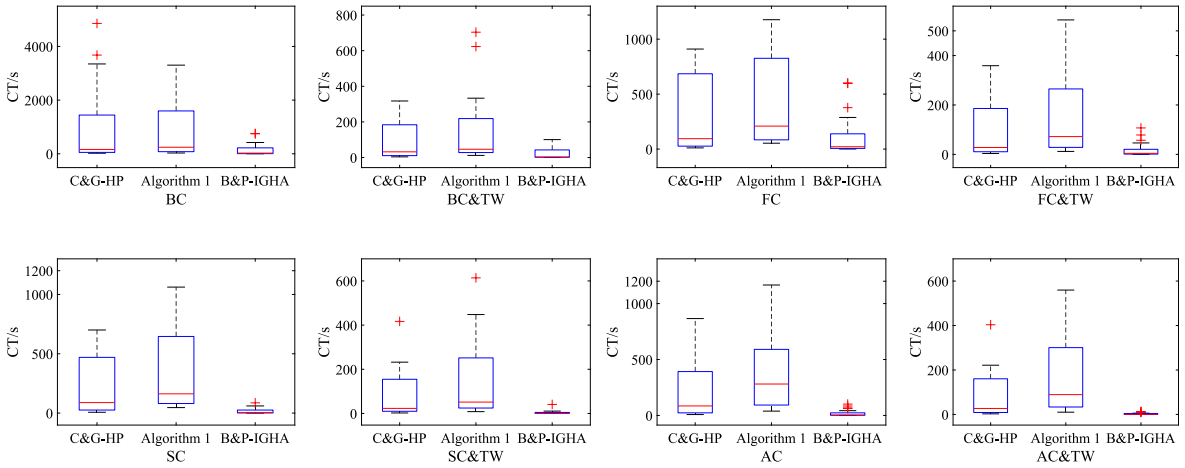

Fig. 16. CTs of different algorithms under different conditions.

time window constraint greatly reduces the number of labels. Therefore, the importance of each label increases significantly. By calling the ITRSA, the feasibility of each label can be investigated more accurately, and a more accurate loading algorithm has greater significance. Moreover, the computing time is reduced for the same reason.

In summary, the performance of the algorithms proposed in this paper was verified through numerical experiments. The enhancement strategies of the loading algorithms improve their efficiency and effectiveness. The model was proven to be efficient, and the complexity of each constraint was discussed. The two-stage method of calling the two loading algorithms at different frequencies is proven to be efficient, which makes LCA2ETLA generate routes accurately and efficiently. **Algorithm 1** proposed in this paper can solve large-scale instances and generate high-quality solutions within an acceptable computational time.

7. Conclusion

In this study, the pickup capacitated vehicle routing problem with three-dimensional loading constraints (3L-PCVRP) was addressed. By discussing the working methods of the pickup operations, the loading sequence, full-support, and time-window constraints are proposed. The 3L-PCVRP model presented herein can be easily converted into the 3L-CVRP model under the delivery scenario.

To solve 3L-PCVRP efficiently, three main studies were performed in this work. First, 3L-PCVRP was decomposed by using the Dantiz–Wolf decomposition method. The restricted master problem (RMP) model was used to select routes, and the subproblem (SP) model was used to generate routes. The structures of the decomposed models were analyzed in detail, and challenges in their solutions were discussed. Second, two loading algorithms: greedy heuristic algorithm (GHA) and tree search algorithm (TRSA), were introduced. Two strategies were developed to enhance the efficiency and effectiveness of the loading algorithms. Finally, a label-correcting-based algorithm to solve the SP was presented. To improve accuracy while ensuring efficiency, a two-stage method employing two loading algorithms at different frequencies was presented.

The numerical experiment was divided into two parts to explore the efficiency and effectiveness of the proposed algorithms. In the first part, the efficiency and effectiveness of the loading algorithms were verified by comparing them with those of the developed algorithms. The results confirm that the two loading algorithms complement each other. Moreover, the enhancement strategies result in an average time reduction of 78.51% and an average increase in volume utilization of 3.48%. In the second part, the efficiency and effectiveness of the 3L-PCVRP model and branch-and-price-based (B&P) algorithm proposed herein were validated. By solving small-scale 3L-PCVRP instances, the efficiency of the 3L-PCVRP model and the effectiveness of the two-stage method were validated. By solving the benchmark instances, the efficiency of the proposed B&P algorithm was verified by comparing it with that of the developed algorithm. The proposed algorithm reduces the cost by up to 26.38%. The results indicate that the two-stage method can be used as a conventional method to handle 3L-PCVRP.

The model and algorithms proposed in this study can be directly applied to companies specializing in item pickup using trucks. Future studies on 3L-PCVRP focusing on the pickup scenario can be carried out considering the findings of this study to discuss the formulation of exact problems further. Relevant constraints involved in the actual loading operations can be discussed, such as the working space of the forklift and different positions of the vehicle doors. In addition, uncertainties caused by road congestion or dynamic orders are a future direction of 3L-PCVRP research. A simpler 3L-PCVRP model is also worth exploring. Finally, the probability function of calling an accurate loading algorithm deserves further discussion.

CRediT authorship contribution statement

Jushang Chi: Conceptualization, Methodology, Software, Visualization, Investigation, Writing – original draft. **Shiwei He:** Conceptualization, Methodology, Formal analysis, Writing – original draft, Project administration.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

Funding: This work was supported by the Fundamental Research Funds for the Central Universities (Science and technology leading talent team project) (No.2022JBQY006); National Natural Science Foundation of China (No.62076023); Research Project of China Railway Corporation (No.K2022X015).

Appendix A. Solution procedure of the IGHA

IGHA

Step 1: Initialize
Initialize $B_1 = B_2 = E = \varnothing$
Build all the possible blocks and add the blocks into B_1. Each block b corresponds to a set of items I_b
Set the loading space of the container as the initial space s. The space set $E = \{s\}$
Step 2: Select space
Set the shortest distance $d = M$
Update d_s for each $s \in E$
Sort S according to the value of d_s from low to high
Set index $q = 0$
Set the space $s_q \in E$ with the lowest d_s as $s^{'}$
Turn to **Step 3**
Step 3: Select block
Set $B^{'} = B_1$
Set $f = false$
For each $\varrho \in \{1,2,3,4,5,6\}$, do
Set $v_\varrho = -M$
For each $b \in B^{'}$, do
$e^{\varrho}_{bs^{'}} = -M$
If $l_b \le L_{s^{'}}, w_b \le W_{s^{'}}, h_b \le H_{s^{'}}$
Update $e^{\varrho}_{bs^{'}}$
$f = true$
End if
End for
If $f = false$
$q = q + 1$
If $q + 1 > |E|$
Return *vu* (CLP) and false (3L-PCVRP) as the final solution
Else if
$s^{'} = s_{q+1}$
Turn to **Step 3**
End if
End if
Sort $e^{\varrho}_{bs^{'}}$ from low to high
Set v_ϱ as the max $e^{\varrho}_{bs^{'}}$
For each $b \in B^{'}$, do
If $e^{\varrho}_{bs^{'}} < v_\varrho$
Remove b from $B^{'}$
End if
End for
End for
Set the first element in $B^{'}$ as b_0
For each $b \in B_1$, do
If $I_b \cap I_{b_0} \neq \varnothing$
Set $B_1 = B_1/\{b\}$
End if
End for
For each $s \in E$, do
If s and b_0 overlap
Set $E = E/\{s\}$
End if
End for
Turn to **Step 4**
Step 4: Split space
Remove b_0 from B_1. Add b_0 to B_2. Remove $s^{'}$ from S
Generate new spaces s_1, s_2, s_3 and add them into S
Turn to **Step 5**
Step 5: Merge spaces
If $B_1 = \varnothing$
Return *vu* (CLP) and true (3L-PCVRP) as the final solution
Else if $E = \varnothing$
Return *vu* (CLP) and false (3L-PCVRP) as the final solution
End if
For each $s \in E$, do
For each $s^{'} \in E$, do
If spaces s and $s^{'}$ have a common edge along the *X*- or *Y*- axis same side
Generate new space $s^{''}$ following the new method 1
Add $s^{''}$ into E, and remove s and $s^{'}$ from E

(*continued on next page*)

(*continued*)

Else if spaces s and $s^{'}$ are adjacent and do not have a common edge along the X- or Y- axis
Generate new space $s^{''}, s_1^{'}, s_2^{'}$ following the new method 2
Add $s^{''}, s_1^{'}, s_2^{'}$ into E
End if
End for
End for
Turn to **Step 2**

Appendix B. Solution procedure of the ITRSA

ITRSA

Step 1: Initialize
Initialize the initial tree node c with $B_1^c = B_2^c = E_c = \varnothing$
Build all possible blocks and add the blocks into B_1^c. Each block b corresponds to a set of items I_b
Set the loading space of the container as the initial space s. The space set $E_c = \{s\}$
Step 2: Generate nodes
For each $c \in C$, do
For each $\varrho \in \{1,2,3,4,5,6\}$, do
Initialize $v_\varrho = -M$
For each $s \in E_c$, do
For each $b \in B_1^c$, do
Set $e_{bs}^\varrho = -M$
If $l_b \le L_s, w_b \le W_s, h_b \le H_s$
Update e_{bs}^ϱ
End if
End for
End for
Sort e_{bs}^ϱ from low to high
Set v_ϱ as the max e_{bs}^ϱ
For each $s \in E_c$, do
For each $b \in B_1^c$, do
If $e_{bs}^\varrho = v_\varrho$
Set $b_\lambda = b, s_\lambda = s$
Set $G_c = G_c \cup \{\lambda\}$
End if
End for
End for
End for
For each $\lambda \in G_c$, do
Generate $c^{'}$, $E_{c^{'}} = E_c/\{s_\lambda\}, B_1^{c^{'}} = B_1^c/\{b_\lambda\}, B_2^{c^{'}} = B_2^c \cup \{b_\lambda\}$
Record the block b_λ as $b_{c^{'}}$, and space s_λ as $s_{c^{'}}$
For each $b \in B_1^{c^{'}}$, do
If $I_b \cap I_{b_\lambda} \neq \varnothing$
Set $B_1^{c^{'}} = B_1^{c^{'}}/\{b\}$
End if
End for
For each $s \in E_{c^{'}}$, do
If s and b_λ overlap
Set $E_{c^{'}} = E_{c^{'}}/\{s\}$
End if
End for
$C = C \cup \{c^{'}\}/\{c\}$
End for
End for
Turn to **Step 3**
Step 3: Split space
For each $c \in C$, do
Add b_c to B_2^c
Generate new spaces s_1, s_2, s_3 and add them into S
End for
Turn to **Step 4**
Step 4: Merge spaces
For each $c \in C$, do
For each $s \in E$, do
For each $s^{'} \in E$, do
If spaces s and $s^{'}$ have a common edge along the X- or Y- axis same side

(*continued on next page*)

(*continued*)

```
                                        Generate new space s'' following the new method 1
                                    Else if spaces s and s' are adjacent, and do not have a common edge along the X- or Y- axis
                                        Generate new space s'', s'_1, s'_2 following the new method 2
                                        Set E = E ∪ {s'', s'_1, s'_2}
                                    End if
                                End for
                            End for
                        End for
                        Turn to Step 5
Step 5: Remove nodes
                        For each c ∈ C, do
                            If E_c = ∅
                                Remove c from C
                            Else
                                Call the IGHA to evaluate the value of c
                                Set the volume utilization obtained by the IGHA as vu_c
                                Set B_1 as the B_1 obtained in the IGHA
                                If B_1 = ∅
                                    Return vu_c (CLP) or true (3L-PCVRP) as the final solution
                                End if
                            End if
                        End for
                        Sort vu_c from low to high
                        If C = ∅
                            Return vu_c (CLP) or false (3L-PCVRP) as the final solution
                        Else
                            Retain at most ϑ tree nodes in C with higher vu_c
                            Turn to Step 2
                        End if
```

Appendix C. Solution procedure of the LCA2ETLA

LCA2ETLA

```
Step 1: Initialize
        Generate initial label in the depot o, the attributes of the label are (o, 0, 0, 0, 0)
        For each i ∈ V ∪ {o}
            For each j ∈ V
                Update c̄_ij = c_ij − φ_j
            End for
        End for
Step 2: Develop labels
        For each i ∈ V ∪ {o}
            For each r in R_i
                For each j ∈ V ∪ {d}
                    If c_r + c̄_ij < 0 and t_r + t_ij ≤ T_j^u
                        Generate label r'. The attributes of label r' are (j, c_r + c̄_ij, q_r + q_j, v_r + v_j, t_r + t_ij)
                        Update the items of label r', I_r' = I_r ∪ U_j
                        Call the IGHA to load I_r'
                        If I_r' can be loaded (IGHA return true)
                            Add r' into R_j
                        Else
                            Update δ
                            Generate ρ ∈ [0, 1]
                            If ρ < ξ
                                Call the ITRSA to load I_r'
                                φ_2 = φ_2 + 1
                                If I_r' can be loaded (ITRSA return true)
                                    Add r' into R_j
                                    φ_1 = φ_1 + 1
                                End if
                            End if
                        End if
                    End if
                End for
            End for
        End for
        Turn to Step 3
```

(*continued on next page*)

(*continued*)

Step 3: Domination rule
  For each $i \in V$
    For each r in R_i
      For each $r^{'}$ in R_i
        If $c_r < c_{r^{'}}, q_r < q_{r^{'}}, v_r < v_{r^{'}}, t_r < t_{r^{'}}$
          Remove $r^{'}$ from R_i
        End if
      End for
    End for
  End for
  Turn to **Step 4**
Step 4: Remove labels
  For each $i \in V$
    Sort R_i according to c_r
    Retain at most ϵ labels with lower c_r
  End for
  If there are no labels to be developed
    Return labels in R_d as the new routes
  Else
    Turn to **Step 2**
  End if

Appendix D. Results of the benchmark instances

Instances	Constraints		**Algorithm 1**				B&P algorithm with the IGHA				
			UB	LB	gap	CT (s)	UB	LB	gap	CT (s)	IG
E016-03m	NO TW	BC	240.86	236.47	1.82%	40.12	311.90	311.68	0.07%	1.43	22.78%
		BC&FC	252.50	250.22	0.90%	73.42	325.85	323.96	0.58%	0.78	22.51%
		BC&SC	260.74	260.74	0.00%	55.25	447.96	447.96	0.00%	0.21	41.79%
		BC&FC&SC	282.80	272.50	3.64%	77.41	449.50	449.50	0.00%	0.19	35.45%
	TW	BC	243.49	231.80	4.80%	21.34	326.95	326.95	0.00%	0.26	25.53%
		BC&FC	261.68	261.68	0.00%	12.01	328.64	325.13	1.07%	0.37	20.37%
		BC&SC	285.84	273.08	4.47%	8.22	449.52	449.52	0.00%	0.20	36.41%
		BC&FC&SC	288.13	281.22	2.40%	10.16	449.54	449.54	0.00%	0.05	34.52%
E016-05m	NO TW	BC	212.14	203.62	4.02%	81.74	231.90	225.37	2.82%	3.48	7.62%
		BC&FC	212.14	203.62	4.02%	78.80	235.56	223.81	4.99%	2.39	7.31%
		BC&SC	236.50	227.82	3.67%	73.31	294.20	288.32	2.00%	0.44	19.61%
		BC&FC&SC	234.42	223.51	4.66%	84.11	315.76	315.76	0.00%	0.29	23.05%
	TW	BC	212.14	204.27	3.71%	20.54	237.35	235.52	0.77%	0.42	9.74%
		BC&FC	218.34	218.34	0.00%	21.59	252.00	241.59	4.13%	0.48	13.36%
		BC&SC	249.79	245.41	1.75%	18.31	381.35	369.67	3.06%	40.83	34.50%
		BC&FC&SC	252.36	247.90	1.77%	13.46	387.64	387.64	0.00%	0.08	34.90%
E021-04m	NO TW	BC	307.67	294.25	4.36%	29.61	333.89	333.89	0.00%	5.80	6.48%
		BC&FC	312.26	298.15	4.52%	53.64	333.51	333.25	0.08%	5.91	6.37%
		BC&SC	341.86	327.12	4.31%	51.13	473.45	473.45	0.00%	0.52	28.39%
		BC&FC&SC	339.02	321.81	5.08%	94.39	488.60	488.60	0.00%	0.79	30.61%
	TW	BC	312.83	301.06	3.76%	11.95	339.87	335.14	1.39%	1.14	6.58%
		BC&FC	320.88	306.00	4.63%	21.11	376.16	376.16	0.00%	1.09	14.70%
		BC&SC	341.13	331.71	2.76%	15.24	647.16	631.34	2.44%	0.31	46.98%
		BC&FC&SC	341.13	329.19	3.50%	21.33	666.95	651.13	2.37%	0.33	48.04%
E021-06m	NO TW	BC	310.70	305.93	1.54%	50.79	322.33	322.33	0.00%	6.55	3.61%
		BC&FC	310.70	298.39	3.96%	83.83	333.68	318.55	4.54%	8.07	6.89%
		BC&SC	325.03	313.01	3.70%	58.45	420.43	420.10	0.08%	1.42	20.73%
		BC&FC&SC	333.29	317.22	4.82%	82.01	419.57	399.58	4.76%	2.40	20.56%
	TW	BC	310.70	302.31	2.70%	27.80	344.85	329.49	4.45%	1.40	8.09%
		BC&FC	310.70	302.00	2.80%	44.11	340.58	324.22	4.80%	0.90	8.77%
		BC&SC	335.27	319.71	4.64%	33.94	524.41	524.41	0.00%	0.19	35.17%
		BC&FC&SC	335.27	322.15	3.91%	61.30	504.68	504.68	0.00%	0.14	33.57%
E022-04g	NO TW	BC	343.87	312.71	9.06%	78.17	349.74	336.17	3.88%	4.14	1.85%
		BC&FC	347.44	341.27	1.78%	85.31	375.23	365.52	2.59%	4.13	7.41%
		BC&SC	364.19	355.12	2.49%	88.78	437.62	437.62	0.00%	0.95	16.78%
		BC&FC&SC	367.54	362.97	1.24%	90.74	493.26	478.07	3.08%	0.81	25.49%
	TW	BC	343.29	329.68	3.96%	42.48	380.51	361.49	5.00%	1.18	9.78%
		BC&FC	352.27	335.05	4.89%	54.08	443.67	436.16	1.69%	0.82	20.60%
		BC&SC	374.90	357.07	4.76%	43.30	596.41	593.72	0.45%	0.37	37.14%
		BC&FC&SC	383.55	369.88	3.56%	44.73	616.61	616.61	0.00%	0.20	37.80%
E022-06m	NO TW	BC	312.82	303.82	2.88%	66.55	336.93	336.93	0.00%	7.89	6.35%

(*continued on next page*)

(*continued*)

Instances	Constraints		Algorithm 1				B&P algorithm with the IGHA				
			UB	LB	gap	CT (s)	UB	LB	gap	CT (s)	IG
E023-03g	TW	BC&FC	325.55	315.35	3.13%	76.67	413.01	396.11	4.09%	2.39	21.18%
		BC&SC	335.94	327.71	2.45%	72.37	443.83	422.48	4.81%	1.71	24.31%
		BC&FC&SC	348.41	336.43	3.44%	84.68	489.09	482.78	1.29%	1.20	28.77%
		BC	319.09	308.22	3.41%	36.68	354.37	354.37	0.00%	1.48	9.96%
	NO TW	BC&FC	332.17	319.19	3.91%	28.87	427.72	427.72	0.00%	0.25	21.69%
		BC&SC	342.91	334.73	2.39%	28.60	581.65	581.65	0.00%	0.27	41.05%
		BC&FC&SC	360.49	346.14	3.98%	33.25	565.95	565.95	0.00%	0.19	36.30%
	TW	BC	330.71	315.26	4.67%	41.22	412.75	398.92	3.35%	5.58	17.91%
		BC&FC	346.42	332.76	3.94%	56.89	444.47	444.47	0.00%	4.52	20.16%
		BC&SC	359.85	345.15	4.08%	46.28	566.27	566.27	0.00%	0.59	36.45%
		BC&FC&SC	378.23	373.71	1.20%	38.63	594.67	594.52	0.03%	0.48	36.40%
	NO TW	BC	338.83	325.96	3.80%	18.20	421.65	414.30	1.75%	1.60	19.64%
		BC&FC	359.82	346.09	3.81%	15.71	468.16	464.67	0.74%	1.35	23.14%
		BC&SC	373.95	357.46	4.41%	10.94	695.89	693.58	0.33%	0.55	46.26%
		BC&FC&SC	393.98	376.42	4.46%	11.57	695.89	695.89	0.00%	0.32	43.18%
E023-05s	TW	BC	360.89	346.12	4.09%	74.43	426.24	410.81	3.62%	4.77	13.54%
		BC&FC	365.73	349.67	4.39%	76.18	433.32	421.12	2.82%	4.54	14.29%
		BC&SC	391.15	382.66	2.17%	82.64	543.48	543.48	0.00%	0.49	28.03%
		BC&FC&SC	391.15	391.15	0.00%	93.57	543.48	543.48	0.00%	0.70	28.03%
	NO TW	BC	394.15	379.02	3.84%	25.87	433.98	425.35	1.99%	1.03	9.18%
		BC&FC	394.15	390.18	1.01%	25.15	451.63	448.22	0.76%	0.68	12.73%
		BC&SC	438.23	427.28	2.50%	22.19	687.90	687.90	0.00%	0.27	34.89%
		BC&FC&SC	447.88	431.29	3.71%	34.09	687.90	687.90	0.00%	0.18	34.89%
E026-08m	TW	BC	358.08	358.08	0.00%	107.58	384.67	384.67	0.00%	5.25	6.91%
		BC&FC	367.42	365.55	0.51%	124.60	405.04	404.00	0.26%	8.28	9.29%
		BC&SC	400.31	395.64	1.17%	79.86	560.27	549.48	1.92%	2.66	28.41%
		BC&FC&SC	413.56	413.56	0.00%	92.75	525.99	525.99	0.00%	1.71	21.37%
	NO TW	BC	366.30	366.30	0.00%	20.75	426.61	418.02	2.01%	1.44	14.14%
		BC&FC	386.49	375.54	2.83%	18.17	449.64	433.98	3.48%	1.55	14.04%
		BC&SC	431.25	418.26	3.01%	17.28	664.95	664.95	0.00%	0.35	35.01%
		BC&FC&SC	433.17	411.92	4.91%	22.42	619.55	619.55	0.00%	0.24	30.25%
E030-03g	TW	BC	415.85	403.42	2.99%	184.20	474.81	459.34	3.26%	9.77	12.18%
		BC&FC	417.31	408.88	2.02%	248.06	512.02	503.69	1.63%	15.22	17.64%
		BC&SC	462.32	441.58	4.49%	161.29	615.69	608.78	1.12%	3.46	24.91%
		BC&FC&SC	462.32	449.72	2.73%	172.42	654.37	641.09	2.03%	3.37	29.35%
	NO TW	BC	424.08	409.34	3.48%	28.27	503.46	503.46	0.00%	2.65	15.77%
		BC&FC	432.97	418.41	3.36%	28.21	562.55	552.89	1.72%	2.45	23.03%
		BC&SC	486.19	486.06	0.03%	23.21	830.41	830.41	0.00%	0.20	41.45%
		BC&FC&SC	486.19	485.50	0.14%	31.48	830.41	830.41	0.00%	0.18	41.45%
E030-04s	TW	BC	430.95	428.59	0.55%	369.49	458.04	457.84	0.04%	14.11	4.79%
		BC&FC	446.86	438.86	1.79%	178.98	510.21	500.33	1.94%	13.52	11.85%
		BC&SC	464.91	457.36	1.63%	130.68	674.87	671.76	0.46%	3.39	30.72%
		BC&FC&SC	494.50	487.49	1.42%	146.72	719.57	716.06	0.49%	2.10	31.28%
	NO TW	BC	442.33	439.06	0.74%	94.55	482.20	481.71	0.10%	2.63	8.27%
		BC&FC	453.09	443.40	2.14%	99.00	540.20	537.55	0.49%	3.01	15.76%
		BC&SC	470.32	465.94	0.93%	64.61	807.31	807.31	0.00%	0.53	41.74%
		BC&FC&SC	505.87	505.87	0.00%	90.93	812.16	812.16	0.00%	0.35	37.59%
E031-09h	TW	BC	408.08	396.10	2.93%	424.95	456.58	441.60	3.28%	16.57	10.62%
		BC&FC	416.21	400.67	3.73%	170.05	456.18	439.67	3.62%	21.32	7.60%
		BC&SC	450.69	433.26	3.87%	153.49	599.16	593.08	1.01%	3.12	23.46%
		BC&FC&SC	450.69	444.30	1.42%	127.48	589.79	589.79	0.00%	3.17	20.75%
	NO TW	BC	415.12	401.44	3.29%	46.04	471.40	453.29	3.84%	2.69	11.49%
		BC&FC	417.45	405.73	2.81%	83.72	495.72	475.50	4.08%	5.35	14.97%
		BC&SC	450.69	433.46	3.82%	51.49	655.27	655.27	0.00%	0.59	27.34%
		BC&FC&SC	450.69	447.02	0.81%	104.03	652.12	652.12	0.00%	0.51	26.40%
E033-03n	TW	BC	445.67	430.02	3.51%	219.89	485.15	468.93	3.34%	30.57	8.97%
		BC&FC	445.67	434.14	2.59%	215.50	468.30	460.08	1.75%	40.10	1.11%
		BC&SC	478.40	458.84	4.09%	205.34	676.60	665.58	1.63%	4.57	29.29%
		BC&FC&SC	488.51	486.56	0.40%	281.89	699.44	697.72	0.25%	3.80	29.66%
	NO TW	BC	441.62	419.75	4.95%	46.56	497.26	483.38	2.79%	2.97	11.19%
		BC&FC	461.94	440.41	4.66%	66.09	492.00	484.69	1.49%	4.20	5.88%
		BC&SC	493.13	487.98	1.04%	45.93	762.00	754.77	0.95%	0.70	35.28%
		BC&FC&SC	539.77	528.16	2.15%	65.05	784.20	778.99	0.66%	0.50	30.54%
E033-04g	TW	BC	472.80	469.07	0.79%	179.50	521.62	513.54	1.55%	18.66	6.09%
		BC&FC	481.42	464.61	3.49%	209.05	558.65	557.17	0.26%	16.48	13.05%
		BC&SC	518.28	499.11	3.70%	143.91	781.37	772.26	1.17%	1.75	33.19%
		BC&FC&SC	531.18	512.32	3.55%	185.69	746.48	742.36	0.55%	3.17	28.84%
	NO TW	BC	489.85	467.35	4.59%	40.08	540.81	527.97	2.37%	3.08	9.42%

(*continued on next page*)

(*continued*)

Instances	Constraints		Algorithm 1				B&P algorithm with the IGHA				
			UB	LB	gap	CT (s)	UB	LB	gap	CT (s)	IG
E033-05s	NO TW	BC&FC	497.45	479.42	3.62%	55.68	602.98	597.49	0.91%	2.90	17.50%
		BC&SC	544.05	536.30	1.42%	31.45	868.71	868.71	0.00%	0.40	37.37%
		BC&FC&SC	562.61	541.86	3.69%	63.32	868.71	868.71	0.00%	0.25	35.24%
	TW	BC	454.83	449.95	1.07%	296.37	514.89	497.12	3.45%	25.71	11.66%
		BC&FC	466.13	455.94	2.19%	223.46	544.11	527.18	3.11%	18.67	14.33%
		BC&SC	503.66	481.98	4.31%	269.05	721.87	720.00	0.26%	2.81	30.23%
		BC&FC&SC	516.91	509.68	1.40%	298.34	728.15	717.87	1.41%	4.33	27.62%
E036-11h	NO TW	BC	474.75	453.94	4.38%	84.86	518.98	507.33	2.24%	4.68	8.47%
		BC&FC	474.75	451.13	4.98%	113.51	576.67	565.81	1.88%	3.61	15.88%
		BC&SC	516.20	495.35	4.04%	55.57	787.86	786.65	0.15%	0.87	34.29%
		BC&FC&SC	526.13	500.13	4.94%	89.17	796.07	787.65	1.06%	1.05	33.79%
	TW	BC	509.08	488.57	4.03%	186.18	515.16	505.45	1.88%	40.63	0.30%
		BC&FC	518.85	499.20	3.79%	191.54	526.32	516.49	1.87%	38.30	1.42%
		BC&SC	526.97	508.50	3.50%	197.49	655.57	643.99	1.77%	4.78	19.62%
		BC&FC&SC	536.53	526.96	1.78%	282.79	682.97	682.97	0.00%	4.66	21.42%
E041-14h	NO TW	BC	513.61	476.00	7.32%	29.67	536.27	523.16	2.45%	7.83	4.23%
		BC&FC	515.67	507.79	1.53%	35.50	551.35	524.15	4.93%	7.77	5.81%
		BC&SC	553.29	525.93	4.95%	47.53	791.40	773.09	2.31%	1.38	30.07%
		BC&FC&SC	548.40	527.31	3.85%	92.99	817.67	817.67	0.00%	0.64	32.77%
	TW	BC	624.12	611.30	2.05%	244.08	665.10	632.56	4.89%	48.70	4.40%
		BC&FC	650.39	630.02	3.13%	207.38	758.17	733.73	3.22%	34.46	15.08%
		BC&SC	669.51	659.28	1.53%	162.18	939.04	939.04	0.00%	2.61	28.70%
		BC&FC&SC	681.77	677.89	0.57%	280.83	954.46	944.56	1.04%	4.04	27.67%
E045-04f	NO TW	BC	631.40	607.46	3.79%	73.96	721.80	699.08	3.15%	8.22	11.91%
		BC&FC	641.33	614.61	4.17%	72.00	808.10	805.32	0.34%	4.96	20.32%
		BC&SC	679.75	660.12	2.89%	77.11	1171.95	1171.95	0.00%	1.10	41.53%
		BC&FC&SC	728.26	711.65	2.28%	113.42	1190.38	1189.96	0.04%	1.35	38.82%
	TW	BC	659.30	643.54	2.39%	279.25	739.43	732.35	0.96%	40.04	8.40%
		BC&FC	691.08	672.06	2.75%	367.52	815.44	784.06	3.85%	57.41	14.53%
		BC&SC	723.32	695.98	3.78%	278.47	1105.33	1099.00	0.57%	7.61	35.58%
		BC&FC&SC	741.36	722.01	2.61%	397.56	1179.16	1178.52	0.05%	4.34	36.63%
E051-05e	NO TW	BC	667.45	643.24	3.63%	59.76	796.13	775.13	2.64%	13.21	14.93%
		BC&FC	696.98	664.17	4.71%	93.98	892.94	871.61	2.39%	11.19	21.95%
		BC&SC	745.64	728.52	2.30%	51.57	1315.17	1314.79	0.03%	1.85	43.30%
		BC&FC&SC	793.77	791.84	0.24%	79.61	1394.44	1394.44	0.00%	0.87	42.92%
	TW	BC	722.01	691.75	4.19%	562.11	821.78	791.80	3.65%	69.29	12.14%
		BC&FC	740.12	723.64	2.23%	431.45	936.94	910.11	2.86%	49.15	21.01%
		BC&SC	777.54	761.06	2.12%	399.99	1163.44	1163.44	0.00%	8.96	31.85%
		BC&FC&SC	814.79	800.09	1.80%	428.11	1222.63	1217.35	0.43%	10.48	33.36%
E072-04f	NO TW	BC	727.66	692.35	4.85%	120.09	901.64	887.63	1.55%	10.97	17.85%
		BC&FC	751.39	730.93	2.72%	134.79	1025.77	1016.77	0.88%	9.08	26.74%
		BC&SC	818.89	804.83	1.72%	100.21	1449.26	1443.72	0.38%	2.72	43.50%
		BC&FC&SC	867.25	848.49	2.16%	131.89	1521.94	1511.88	0.66%	1.43	43.02%
	TW	BC	1177.97	1161.83	1.37%	1695.88	1296.82	1276.91	1.54%	279.17	9.11%
		BC&FC	1198.74	1180.66	1.51%	1162.59	1399.24	1386.07	0.94%	121.57	14.33%
		BC&SC	1283.81	1274.07	0.76%	848.29	1776.40	1763.86	0.71%	30.81	27.73%
		BC&FC&SC	1321.13	1298.77	1.69%	930.51	1833.22	1824.55	0.47%	26.54	27.93%
E076-07s	NO TW	BC	1191.96	1138.57	4.48%	226.54	1360.64	1341.67	1.39%	46.22	11.10%
		BC&FC	1198.74	1151.82	3.91%	288.98	1521.77	1509.25	0.82%	20.72	21.23%
		BC&SC	1341.27	1314.99	1.96%	312.14	2266.82	2259.97	0.30%	5.13	40.83%
		BC&FC&SC	1357.38	1351.04	0.47%	287.43	2315.71	2312.20	0.15%	4.18	41.38%
	TW	BC	1180.60	1172.17	0.71%	1906.68	1266.85	1214.68	4.12%	421.01	6.08%
		BC&FC	1214.20	1182.05	2.65%	1176.72	1313.85	1280.95	2.50%	287.76	7.58%
		BC&SC	1257.39	1251.08	0.50%	909.07	1856.81	1850.92	0.32%	48.96	30.59%
		BC&FC&SC	1316.69	1293.76	1.74%	914.55	1915.65	1897.17	0.96%	42.72	30.81%
E076-08s	NO TW	BC	1210.97	1166.76	3.65%	328.21	1326.90	1313.01	1.05%	51.58	8.74%
		BC&FC	1225.76	1172.34	4.36%	260.25	1448.06	1430.40	1.22%	31.62	15.35%
		BC&SC	1354.67	1342.02	0.93%	306.67	2315.15	2315.15	0.00%	2.61	41.13%
		BC&FC&SC	1373.57	1338.23	2.57%	355.84	2400.81	2373.25	1.15%	9.02	42.62%
	TW	BC	1254.89	1245.48	0.75%	1300.23	1437.84	1376.83	4.24%	207.83	8.93%
		BC&FC	1330.91	1266.34	4.85%	802.70	1514.58	1504.09	0.69%	188.19	12.13%
		BC&SC	1427.49	1366.54	4.27%	662.47	1934.29	1922.02	0.63%	31.71	26.20%
		BC&FC&SC	1431.20	1405.44	1.80%	782.03	1990.92	1975.71	0.76%	35.20	28.11%
E076-10e	NO TW	BC	1309.26	1247.08	4.75%	330.50	1470.24	1409.01	4.16%	55.31	9.11%
		BC&FC	1355.07	1302.27	3.90%	333.59	1567.16	1559.36	0.50%	34.26	13.16%
		BC&SC	1458.03	1420.89	2.55%	267.45	2275.71	2272.05	0.16%	5.72	34.60%
		BC&FC&SC	1501.88	1460.90	2.73%	347.89	2316.70	2312.41	0.19%	5.05	35.17%
	TW	BC	1224.76	1183.72	3.35%	1713.12	1314.11	1280.49	2.56%	264.50	7.08%

(*continued on next page*)

(*continued*)

Instances	Constraints		Algorithm 1				B&P algorithm with the IGHA				
			UB	LB	gap	CT (s)	UB	LB	gap	CT (s)	IG
E076-14s	TW	BC&FC	1254.72	1214.96	3.17%	855.65	1385.82	1350.07	2.58%	273.16	9.82%
		BC&SC	1301.43	1274.35	2.08%	595.81	1658.47	1643.38	0.91%	56.83	21.44%
		BC&FC&SC	1329.07	1313.72	1.15%	550.70	1730.85	1709.63	1.23%	62.54	23.54%
		BC	1221.55	1192.43	2.38%	333.22	1369.14	1342.62	1.94%	44.57	10.55%
	NO TW	BC&FC	1258.55	1213.17	3.61%	266.45	1438.78	1409.56	2.03%	46.29	13.02%
		BC&SC	1395.13	1366.61	2.04%	277.23	2201.42	2200.42	0.05%	6.58	37.20%
		BC&FC&SC	1405.32	1392.66	0.90%	332.88	2253.42	2252.23	0.05%	10.40	38.01%
	TW	BC	1192.00	1171.45	1.72%	1898.44	1316.35	1281.07	2.68%	225.65	9.34%
		BC&FC	1197.24	1185.20	1.01%	833.43	1492.36	1448.27	2.95%	144.69	19.31%
		BC&SC	1268.38	1257.05	0.89%	798.38	1933.05	1933.05	0.00%	19.76	33.60%
		BC&FC&SC	1286.02	1270.84	1.18%	603.93	1846.41	1846.41	0.00%	13.38	30.35%
	NO TW	BC	1192.00	1147.30	3.75%	195.00	1358.81	1324.90	2.50%	37.38	11.26%
		BC&FC	1204.41	1179.12	2.10%	276.44	1599.66	1548.10	3.22%	19.92	23.56%
		BC&SC	1355.02	1335.18	1.46%	227.34	2308.23	2298.22	0.43%	5.08	41.30%
		BC&FC&SC	1402.34	1387.45	1.06%	304.99	2368.50	2357.01	0.49%	2.22	40.28%
E101-08e	TW	BC	1645.14	1622.41	1.38%	1917.16	1791.93	1729.40	3.49%	757.49	8.19%
		BC&FC	1655.07	1627.29	1.68%	1053.93	1820.18	1780.78	2.16%	597.52	9.07%
		BC&SC	1758.10	1735.51	1.28%	785.98	2637.73	2637.73	0.00%	60.29	33.35%
		BC&FC&SC	1796.76	1763.49	1.85%	1166.09	2610.35	2578.95	1.20%	101.51	31.17%
	NO TW	BC	1667.88	1584.95	4.97%	623.25	1882.31	1796.98	4.53%	100.65	11.28%
		BC&FC	1667.88	1607.46	3.62%	517.99	1989.55	1908.91	4.05%	107.06	15.57%
		BC&SC	1832.66	1788.60	2.40%	447.77	3228.76	3221.79	0.22%	5.16	43.19%
		BC&FC&SC	1853.05	1835.89	0.93%	463.03	3297.37	3297.37	0.00%	11.59	43.20%
E101-10c	TW	BC	1732.63	1714.79	1.03%	3302.36	1975.89	1923.59	2.65%	745.14	11.88%
		BC&FC	1795.07	1773.01	1.23%	1135.17	2107.30	2058.14	2.33%	601.49	14.50%
		BC&SC	1903.08	1893.64	0.50%	1061.87	2670.73	2669.50	0.05%	27.45	28.74%
		BC&FC&SC	1903.08	1817.81	4.48%	980.89	2635.06	2635.06	0.00%	84.17	27.78%
	NO TW	BC	1758.40	1724.18	1.95%	903.79	2070.93	2006.91	3.09%	86.34	15.09%
		BC&FC	1849.32	1787.62	3.34%	544.28	2282.94	2255.11	1.22%	78.54	18.59%
		BC&SC	2059.63	2032.02	1.34%	613.82	3433.65	3433.65	0.00%	3.47	40.02%
		BC&FC&SC	2097.23	2074.24	1.10%	559.52	3504.39	3504.39	0.00%	5.86	40.15%
E101-14s	TW	BC	1776.27	1752.16	1.36%	2822.81	1987.85	1953.40	1.73%	361.25	10.64%
		BC&FC	1855.71	1845.27	0.56%	1110.71	2158.30	2116.43	1.94%	376.95	14.02%
		BC&SC	1886.28	1871.15	0.80%	862.08	2697.60	2697.60	0.00%	86.34	29.60%
		BC&FC&SC	1982.46	1960.51	1.11%	901.96	2785.37	2770.36	0.54%	67.36	27.61%
	NO TW	BC	1792.26	1709.41	4.62%	331.54	2119.36	2057.92	2.90%	77.65	15.43%
		BC&FC	1861.36	1815.24	2.48%	435.17	2257.98	2242.71	0.68%	57.15	17.57%
		BC&SC	2072.05	2055.76	0.79%	259.54	3206.52	3206.01	0.02%	10.68	34.79%
		BC&FC&SC	2139.90	2117.15	1.06%	450.98	3381.44	3355.32	0.77%	13.30	36.72%

References

Achterberg, T., Koch, T., Martin, A., 2005. Branching rules revisited. Oper. Res. Lett. 33, 42–54. https://doi.org/10.1016/j.orl.2004.04.002.

Amine Masmoudi, M., Coelho, L.C., Demir, E., 2022. Plug-in hybrid electric refuse vehicle routing problem for waste collection. Transport. Res. Part E: Logist. Transport. Rev. 166, 102875 https://doi.org/10.1016/j.tre.2022.102875.

Ayough, A., Khorshidvand, B., Massomnedjad, N., Motameni, A., 2020. An integrated approach for three-dimensional capacitated vehicle routing problem considering time windows. J. Model. Manag. 15, 995–1015. https://doi.org/10.1108/JM2-11-2018-0183.

Bombelli, A., Fazi, S., 2022. The ground handler dock capacitated pickup and delivery problem with time windows: A collaborative framework for air cargo operations. Transport. Res. Part E: Logist. Transport. Rev. 159, 102603 https://doi.org/10.1016/j.tre.2022.102603.

Bortfeldt, A., Homberger, J., 2013. Packing first, routing second—a heuristic for the vehicle routing and loading problem. Comput. Oper. Res. 40, 873–885. https://doi.org/10.1016/j.cor.2012.09.005.

Bortfeldt, A., Yi, J., 2020. The Split Delivery Vehicle Routing Problem with three-dimensional loading constraints. Eur. J. Oper. Res. 282, 545–558. https://doi.org/10.1016/j.ejor.2019.09.024.

Ceschia, S., Schaerf, A., Stützle, T., 2013. Local search techniques for a routing-packing problem. Comput. Ind. Eng. 66, 1138–1149. https://doi.org/10.1016/j.cie.2013.07.025.

Chabrier, A., 2006. Vehicle Routing Problem with elementary shortest path based column generation. Comput. Oper. Res. 33, 2972–2990. https://doi.org/10.1016/j.cor.2005.02.029.

Chen, X., He, S., Zhang, Y., Tong, L. (Carol), Shang, P., & Zhou, X. (2020). Yard crane and AGV scheduling in automated container terminal: A multi-robot task allocation framework. Transportation Research Part C: Emerging Technologies, 114, 241–271. https://doi.org/10.1016/j.trc.2020.02.012.

Chen, Z., Yang, M., Guo, Y., Liang, Y., Ding, Y., Wang, L., 2020. The split delivery vehicle routing problem with three-dimensional loading and time Windows constraints. Sustainability 12 (17), 6987. https://doi.org/10.3390/su12176987.

Chi, J., He, S., Song, Z., Xue, S., Feng, X., 2022. Optimization of double stack car carriers transportation problem based on branch-and-price algorithm. In Chinese Control and Decision 37 (01), 185–195. https://doi.org/10.13195/j.kzyjc.2020.0917.

Danna, E., Le Pape, C., 2005. Branch-and-Price Heuristics: A Case Study on the Vehicle Routing Problem with Time Windows. In: Desaulniers, G., Desrosiers, J., Solomon, M.M. (Eds.), Column Generation. Springer, Boston, MA. https://doi.org/10.1007/0-387-25486-2_4.

Eley, M., 2002. Solving container loading problems by block arrangement. Eur. J. Oper. Res. 141, 393–409. https://doi.org/10.1016/S0377-2217(02)00133-9.

Fuellerer, G., Doerner, K.F., Hartl, R.F., Iori, M., 2010. Metaheuristics for vehicle routing problems with three-dimensional loading constraints. Eur. J. Oper. Res. 201, 751–759. https://doi.org/10.1016/j.ejor.2009.03.046.
Gajda, M., Trivella, A., Mansini, R., Pisinger, D., 2022. An optimization approach for a complex real-life container loading problem. Omega 107, 102559. https://doi.org/10.1016/j.omega.2021.102559.
Gendreau, M., Iori, M., Laporte, G., Martello, S., 2006. A Tabu Search Algorithm for a Routing and Container Loading Problem. Transp. Sci. 40, 342–350. https://doi.org/10.1287/trsc.1050.0145.
George, J.A., Robinson, D.F., 1980. A heuristic for packing boxes into a container. Comput. Oper. Res. 7, 147–156. https://doi.org/10.1016/0305-0548(80)90001-5.
Giménez-Palacios, I., Parreño, F., Álvarez-Valdés, R., Paquay, C., Oliveira, B.B., Carravilla, M.A., Oliveira, J.F., 2022. First-mile logistics parcel pickup: Vehicle routing with packing constraints under disruption. Transportation Research Part E: Logistics and Transportation Review 164, 102812. https://doi.org/10.1016/j.tre.2022.102812.
Gunawan, A., Kendall, G., McCollum, B., Seow, H.-V., Lee, L.S., 2021. Vehicle routing: Review of benchmark datasets. J. Oper. Res. Soc. 72, 1794–1807. https://doi.org/10.1080/01605682.2021.1884505.
Hu, Z.-H., Zhao, Y., Tao, S., Sheng, Z.-H., 2015. Finished-vehicle transporter routing problem solved by loading pattern discovery. Ann. Oper. Res. 234, 37–56. https://doi.org/10.1007/s10479-014-1777-1.
Iori, M., Salazar-González, J.-J., Vigo, D., 2007. An Exact Approach for the Vehicle Routing Problem with Two-Dimensional Loading Constraints. Transp. Sci. 41 (2), 253–264. https://doi.org/10.1287/trsc.1060.0165.
Jin, J.G., Lee, D.-H., Hu, H., 2015. Tactical berth and yard template design at container transshipment terminals: A column generation based approach. Transportation Research Part E: Logistics and Transportation Review 73, 168–184. https://doi.org/10.1016/j.tre.2014.11.009.
Junqueira, L., Morabito, R., Sato Yamashita, D., 2012. Three-dimensional container loading models with cargo stability and load bearing constraints. Comput. Oper. Res. 39, 74–85. https://doi.org/10.1016/j.cor.2010.07.017.
Krebs, C., Ehmke, J.F., 2021. Axle Weights in combined Vehicle Routing and Container Loading Problems. EURO Journal on Transportation and Logistics 10, 100043. https://doi.org/10.1016/j.ejtl.2021.100043.
Küçük, M., Topaloglu Yildiz, S., 2022. Constraint programming-based solution approaches for three-dimensional loading capacitated vehicle routing problems. Comput. Ind. Eng. 171, 108505 https://doi.org/10.1016/j.cie.2022.108505.
Lan, Z., He, S., Song, R., Hao, S., 2019. Optimizing Vehicle Scheduling Based on Variable Timetable by Benders-and-Price Approach. J. Adv. Transp. 2019, 2781590. https://doi.org/10.1155/2019/2781590.
Lenstra, J.K., Kan, A.H.G.R., 1981. Complexity of vehicle routing and scheduling problems. Networks 11, 221–227. https://doi.org/10.1002/net.3230110211.
Li, Y., Chen, M., Huo, J., 2022. A hybrid adaptive large neighborhood search algorithm for the large-scale heterogeneous container loading problem. Expert Syst. Appl. 189, 115909 https://doi.org/10.1016/j.eswa.2021.115909.
Liu, S., Qin, S., Zhang, R., 2018. A branch-and-price algorithm for the multi-trip multi-repairman problem with time windows. Transportation Research Part E: Logistics and Transportation Review 116, 25–41. https://doi.org/10.1016/j.tre.2018.05.009.
Liu, Y., Yue, Z., Wang, Y., Wang, H., 2023. Logistics Distribution Vehicle Routing Problem with Time Window under Pallet 3D Loading Constraint. SUSTAINABILITY 15. https://doi.org/10.3390/su15043594.
Mahvash, B., Awasthi, A., Chauhan, S., 2017. A column generation based heuristic for the capacitated vehicle routing problem with three-dimensional loading constraints. Int. J. Prod. Res. 55, 1730–1747. https://doi.org/10.1080/00207543.2016.1231940.
Männel, D., Bortfeldt, A., 2016. A hybrid algorithm for the vehicle routing problem with pickup and delivery and three-dimensional loading constraints. Eur. J. Oper. Res. 254, 840–858. https://doi.org/10.1016/j.ejor.2016.04.016.
Männel, D., Bortfeldt, A., 2018. Solving the pickup and delivery problem with three-dimensional loading constraints and reloading ban. Eur. J. Oper. Res. 264, 119–137. https://doi.org/10.1016/j.ejor.2017.05.034.
Moura, A., 2019. A model-based heuristic to the vehicle routing and loading problem. Int. Trans. Oper. Res. 26, 888–907. https://doi.org/10.1111/itor.12586.
Moura, A., Oliveira, J.F., 2009. An integrated approach to the vehicle routing and container loading problems. OR Spectr. 31, 775–800. https://doi.org/10.1007/s00291-008-0129-4.
Nascimento, O.X. do, Alves de Queiroz, T., Junqueira, L., 2021. Practical constraints in the container loading problem: Comprehensive formulations and exact algorithm. Computers & Operations Research 128, 105186. https://doi.org/10.1016/j.cor.2020.105186.
Pace, S., Turky, A., Moser, I., Aleti, A., 2015. Distributing Fibre Boards: A Practical Application of the Heterogeneous Fleet Vehicle Routing Problem with Time Windows and Three-dimensional Loading Constraints. Procedia Comput. Sci. 51, 2257–2266. https://doi.org/10.1016/j.procs.2015.05.382.
Pan, H., Liu, Z., Yang, L., Liang, Z., Wu, Q., Li, S., 2021. A column generation-based approach for integrated vehicle and crew scheduling on a single metro line with the fully automatic operation system by partial supervision. TransportationResearch Part E: Logistics and Transportation Review 152, 102406. https://doi.org/10.1016/j.tre.2021.102406.
Pisinger, D., 2002. Heuristics for the container loading problem. Eur. J. Oper. Res. 141, 382–392. https://doi.org/10.1016/S0377-2217(02)00132-7.
Pollaris, H., Braekers, K., Caris, A., Janssens, G., Limbourg, S., 2017. Iterated Local Search for the Capacitated Vehicle Routing Problem with Sequence-Based Pallet Loading and Axle Weight Constraints. Networks 69, 304–316. https://doi.org/10.1002/net.21738.
Pourhejazy, P., Zhang, D., Zhu, Q., Wei, F., Song, S., 2021. Integrated E-waste transportation using capacitated general routing problem with time-window. Transportation Research Part E: Logistics and Transportation Review 145, 102169. https://doi.org/10.1016/j.tre.2020.102169.
Rajaei, M., Moslehi, G., Reisi-Nafchi, M., 2022. The split heterogeneous vehicle routing problem with three-dimensional loading constraints on a large scale. Eur. J. Oper. Res. 299, 706–721. https://doi.org/10.1016/j.ejor.2021.08.025.
Ren, J., Tian, Y., Sawaragi, T., 2011. A tree search method for the container loading problem with shipment priority. Eur. J. Oper. Res. 214, 526–535. https://doi.org/10.1016/j.ejor.2011.04.025.
Tahami, H., Rabadi, G., Haouari, M., 2020. Exact approaches for routing capacitated electric vehicles. Transportation Research Part E: Logistics and Transportation Review 144, 102126. https://doi.org/10.1016/j.tre.2020.102126.
Tarantilis, C.D., Zachariadis, E.E., Kiranoudis, C.T., 2009. A Hybrid Metaheuristic Algorithm for the Integrated Vehicle Routing and Three-Dimensional Container-Loading Problem. IEEE Trans. Intell. Transp. Syst. 10, 255–271. https://doi.org/10.1109/TITS.2009.2020187.
Vega-Mejía, C.A., Montoya-Torres, J.R., Islam, S.M.N., 2019. A nonlinear optimization model for the balanced vehicle routing problem with loading constraints. Int. Trans. Oper. Res. 26, 794–835. https://doi.org/10.1111/itor.12570.
Wang, D., Liao, F., Gao, Z., Rasouli, S., Huang, H.-J., 2020. Tolerance-based column generation for boundedly rational dynamic activity-travel assignment in large-scale networks. Transportation Research Part E: Logistics and Transportation Review 141, 102034. https://doi.org/10.1016/j.tre.2020.102034.
Wang, M., Miao, L., Zhang, C., 2021. A branch-and-price algorithm for a green location routing problem with multi-type charging infrastructure. Transportation Research Part E: Logistics and Transportation Review 156, 102529. https://doi.org/10.1016/j.tre.2021.102529.
Wang, Y., Wei, Y., Wang, X., Wang, Z., Wang, H., 2023. A clustering-based extended genetic algorithm for the multidepot vehicle routing problem with time windows and three-dimensional loading constraints. Appl. Soft Comput. 133 https://doi.org/10.1016/j.asoc.2022.109922.
Wu, J., Zheng, L., Huang, C., Cai, S., Feng, S., Zhang, D., IEEE, 2019. An Improved Hybrid Heuristic Algorithm for Pickup and Delivery Problem with Three-Dimensional Loading Constraints, in: Xiamen University. Presented at the 2019 IEEE 31ST INTERNATIONAL CONFERENCE ON TOOLS WITH ARTIFICIAL INTELLIGENCE (ICTAI 2019), pp. 1607–1612. https://doi.org/10.1109/ICTAI.2019.00233.
Yuan, Y., Cattaruzza, D., Ogier, M., Semet, F., Vigo, D., 2021. A column generation based heuristic for the generalized vehicle routing problem with time windows. Transportation Research Part E: Logistics and Transportation Review 152, 102391. https://doi.org/10.1016/j.tre.2021.102391.
Zhao, X., Bennell, J.A., Bektaş, T., Dowsland, K., 2016. A comparative review of 3D container loading algorithms. Int. Trans. Oper. Res. 23, 287–320. https://doi.org/10.1111/itor.12094.
Zhu, W., Qin, H., Lim, A., Wang, L., 2012. A two-stage tabu search algorithm with enhanced packing heuristics for the 3L-CVRP and M3L-CVRP. Comput. Oper. Res. 39, 2178–2195. https://doi.org/10.1016/j.cor.2011.11.001.