

Received December 14, 2020, accepted December 28, 2020, date of publication January 13, 2021, date of current version January 25, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3051248

A Double Traveling Salesman Problem With Three-Dimensional Loading Constraints for Bulky Item Delivery

MINZHI RUAN¹, CHUNYUE SHEN², JIANQIANG TANG³, CHAO QI³, AND SHUANG QIU⁴

¹School of Naval Architecture & Ocean Engineering, Naval University of Engineering, Wuhan 430072, China

²School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan 430074, China

³School of Management, Huazhong University of Science and Technology, Wuhan 430074, China

⁴School of Business Administration, Hong Kong University of Science and Technology, Hong Kong, China

Corresponding author: Jianqiang Tang (jqtang@hust.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFC0807500, in part by the Key Project of the 13th Five-Year Pre-Research Foundation under Grant 6140004050, and in part by the National Natural Science Foundation of China under Grant 71821001.

ABSTRACT This article considers bulky item delivery problems in which multiple items are retrieved and loaded onto a vehicle from different warehouses and then delivered. This problem is described as a double traveling salesman problem with three-dimensional container loading constraints with multiple stacks. The double TSP with multiple stacks is used to determining the shortest route performing pickups and deliveries in two separated networks (one for pickups and one for deliveries) using only one container. Repacking is not allowed after loading the items into the container. An integer linear programming model is proposed to solve this problem, a standard genetic algorithm and an improved genetic algorithm is designed. In the improved genetic algorithm, a Lin-Kernighan algorithm is used to improve the delivery route, a k-means clustering algorithm, and a heuristic packing scheme improvement rules work together to improve the loading route. The results show that the improved genetic algorithm is superior to the standard genetic algorithm in large scale problems.

INDEX TERMS Double traveling salesman problem, three-dimensional container loading, bulky item delivery, genetic algorithm.


I. INTRODUCTION

The rapid development of e-commerce has significantly propelled the growth and improvement of the modern logistics industry. According to the National Retail Federation, in 2015, almost 60% of Amazon's online transactions included free shipping, which meant the company paid billions in shipping expenses. With such a trend and the fact that shipping and delivery are very costly, companies make every effort to reduce the transportation expenses on a per package and per order basis. Of course, the features of the shipping and delivery process largely depend upon the nature of goods.

It is not surprising for a consumer to spend a few minutes online and easily place an order for a refrigerator, washing machine, or treadmill, with delivery and installation in two days. For e-commerce players and logistics companies, the most challenging part is delivery. Among their biggest

challenges is working out the logistics for heavy and bulky items. According to our survey conducted at a distribution center of RRS Logistics, a leading bulky item logistics company in China that provides logistics services for heavy and large household appliances orders, the following features have to be considered when shipping and delivering bulky items.

First, bulky items with large volumes and heavy mass commonly require large storage spaces, specific storage facilities, and material handling equipment. In bulky item logistics companies, warehouses are often designed for particular types of merchandise. Thus, different types of items are stored in and retrieved from different warehouses. When the orders are delivered, the orders assigned to a truck consist of multiple items with various delivery destinations. As illustrated in Fig.1, five items a, b, c, d, and e are assigned to a truck. These items are stored in three warehouses (D1, D2, and D3). The truck driver usually is the one who decides how to pick up, to load the truck, and to deliver the items in logistics practice. Fig.1 presents one possible loading route

The associate editor coordinating the review of this manuscript and approving it for publication was Hao Luo .

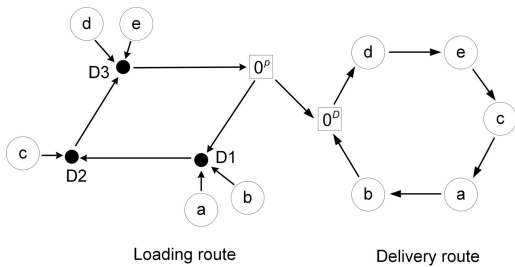


FIGURE 1. Illustration of bulky-item distribution process.

$0^L \rightarrow D1 \rightarrow D2 \rightarrow D3 \rightarrow 0^L$, and one possible delivery route $0^D \rightarrow d \rightarrow e \rightarrow c \rightarrow a \rightarrow b \rightarrow 0^D$, where 0^L and 0^D are pseudo-depots for loading and delivery operations. Therefore, two routing problems are necessary when considering loading and delivery.

Second, since handling bulky items is laborious and time-consuming, the loading and positioning sequences significantly influence the loading and delivery efficiency and are inherently related to routing problems. There are blocking relationships among bulky items in a limited space. Thus, reloading is inevitable if an item is delivered before the ones spatially blocking it, significantly diminishing delivery efficiency.

The bulky item delivery problem with the aforementioned features can be abstracted as a double traveling salesman problem (DTSP) with loading constraints. Two routes are identified in two separate networks, i.e., a loading route and a delivery route, with a corresponding feasible vehicle loading plan. Reference [1] has investigated a similar DTSP problem considering two-dimensional container loading constraints with multiple stacks. In their model, the vehicle space is considered a rectangular shape and divided into several horizontal stacks. No vertical stacking is allowed. However, from a practical perspective, both the packages and the vehicle space are cuboid-shaped. Thus, vertical stacking is usually inevitable, which means three-dimensional (3D) container loading constraints should be considered. In a related area, vehicle routing problems with loading constraints were researched by [2] and [3], who indicated that most related works decoupled the problems into two stages. The routing problems are handled at the first stage, and the container loading problems are solved at the second stage with the previously generated routes as the constraints. This two-stage approach does not lead to joint optimization since the routing and loading problems are inherently coupled. However, integrating 3D loading constraints into a routing problem complicates the model significantly as it affects the formulation at the bottom in terms of the definitions of the variables and descriptions of the constraints.

To our knowledge, [4] first proposed a mathematical model that integrated capacity-constrained vehicle routing problems with 3D loading constraints (3L-CVRP). In their work, the orientation and stability constraints in a 3D space were formulated. Reference [5] and [6] further extended the loading feasibility constraints by considering support surface and

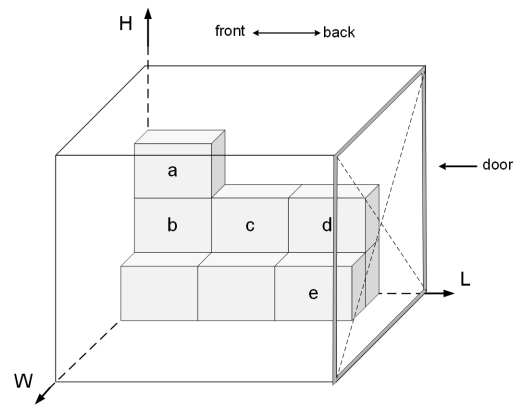


FIGURE 2. Blocking relationship in a 3D container.

vulnerability constraints. In these existing 3L-CVRP models, sequence-based loading is usually assumed, which means that the loading sequence is the exact reverse of the delivery sequence. As shown in the example in Fig.2, if the possible loading sequence is $b \rightarrow c \rightarrow d \rightarrow e \rightarrow a$, then the only delivery sequence is $a \rightarrow e \rightarrow d \rightarrow c \rightarrow b$. This assumption greatly restrains the possible loading and delivery sequences for cargo loaded in 3D spaces. Multiple possible delivery sequences correspond to the loading sequence $b \rightarrow a \rightarrow c \rightarrow e \rightarrow d$, such as $e \rightarrow d \rightarrow a \rightarrow c \rightarrow b$ or $d \rightarrow e \rightarrow c \rightarrow a \rightarrow b$. Thus, without the assumption of sequence-based loading, the spatial constraints interacting with the loading and delivery sequences become much more complicated. Besides, for the bulky item delivery situation considered in this article, where different item types are retrieved from different warehouses, sequence-based loading inevitably leads to repeat visits to the same pickup nodes.

This article aims to study bulky item delivery problems by pursuing the joint optimization of the DTSP and 3D container loading problems. An integer linear programming model, abbreviated as 3L-DTSPMS, is established, and the coupling relationship between pickup/delivery routing and the item-loading sequence is specifically formulated. It is assumed that different types of items with identical physical sizes are loaded into a single container with loading stability constraints and blocking relationship constraints. The assumption of sequence-based loading commonly considered in related works is relaxed in this study. An improved genetic algorithm is designed to solve the problem. A Lin-Kernighan algorithm is used to improve the delivery route, a k-means clustering algorithm, and a heuristic packing scheme improvement rules work together to improve the loading route. A numerical study is conducted to verify the efficiency of the proposed algorithm. The remainder of the article is organized as follows. In section II, the related works are reviewed. The proposed 3L-DTSPMS model is formulated in section III, followed by the presentation of the algorithms in section IV. In section V, a numerical study is carried out. Conclusions and future works are discussed in section VI.

TABLE 1. Models and Algorithms for the DTSPMS.

Reference	RC	IDIS	2DLI	3DLI	SV	MV	2DLC	3DLC	HS	VS	TS	LNS	SA	ILS	VND	VNS	BC	BB	BP	HY
Petersen and Madsen (2009)	x	x	x	-	x	-	x	-	x	-	x	x	x	x	-	-	-	-	-	-
Felipe et al. (2009a)	-	x	-	-	x	-	x	-	-	-	-	-	-	-	x	x	-	-	-	-
Felipe et al. (2009b)	-	x	-	-	x	-	x	-	-	-	-	x	-	-	-	x	-	-	-	-
Petersen et al.(2010)	x	x	x	-	x	-	x	-	x	-	-	-	-	-	-	-	x	-	-	-
Lusby and Larsen(2010)	x	x	-	-	x	-	x	-	-	-	-	-	-	-	-	-	-	-	-	x
Lusby and Larsen (2011)	x	x	-	-	-	x	x	-	-	-	-	-	-	-	-	-	-	-	-	x
Felipe et al. (2011)	x	x	x	-	-	x	x	-	-	-	-	-	-	-	-	x	-	-	-	-
Casazza et al.(2012)	x	x	-	-	x	-	x	-	-	-	-	-	-	-	-	-	-	-	-	x
Martinez et al.(2013)	x	x	x	-	x	-	x	-	x	-	-	-	-	-	-	-	x	-	-	-
Carrabs et al.(2013)	x	x	x	-	x	-	x	-	x	-	-	-	-	-	-	-	x	-	-	-
Batista-Galvan et al.(2013)	x	x	x	-	x	-	x	-	x	-	-	-	-	-	-	-	x	-	-	-
Urrutia et al.(2015)	x	x	-	-	x	-	x	-	-	-	-	-	-	-	-	-	-	-	-	x
Iori and Riera-Ledesma (2015)	x	x	x	-	-	x	x	-	x	-	-	-	-	-	-	-	x	-	x	-
Silveira et al.(2016)	-	x	-	-	-	x	x	-	-	-	-	-	x	x	x	-	-	-	-	-
Barbato,et al. (2016a)	x	x	-	-	x	-	x	-	-	-	-	-	-	-	-	-	x	-	-	-
Barbato,et al. (2016b)	x	x	-	-	x	-	x	-	-	-	-	-	-	-	-	-	x	-	-	-
Chagas et al.(2016)	-	x	-	-	-	x	x	-	-	-	-	-	x	x	-	-	-	-	-	-
Chagas et al.(2017)	-	x	-	-	-	x	x	-	-	-	-	-	-	-	-	-	-	-	x	-
Chagas et al.(2018)	-	x	-	-	-	x	x	-	-	-	-	-	-	-	-	-	-	-	-	x
Chagas et al.(2020)	-	x	-	-	-	x	x	-	-	-	-	-	-	-	-	x	-	-	-	-
Hvattum,et al. (2020)	x	x	-	-	x	-	x	-	-	-	-	-	-	-	-	x	-	-	-	-

Model features : RC = route constraints, IDIS = identical dimensions items, i.e. all items have identical shape and size, 2DLI = two-dimensional load-infeasible constraints in rectangular shape container, while 3DLI means in cuboid shape container, SV = single-vehicle, MV = multiple-vehicle, 2DLC = two-dimensional loading constraints, means container and boxes are considered as rectangular shape, while 3DLC means container and boxes as cuboid shape, HS = horizontal stability, VS = vertical stability.

Solution methods : TS = tabu search, LNS = large neighbourhood search, SA = simulated annealing, ILS = iterated local search, VND = variable neighborhood descent, VNS = variable neighborhood search, BC = branch-and-cut, BB = branch-and-bound, BP = branch-and-price, HY = hybrid algorithm, x = considered in the reference, - = not applicable in the reference.

II. RELATED WORKS

Reference [1] initially established a DTSPMS model with two-dimensional loading constraints. Since then, the problem has received increasing attention. In the past decades, formulations and algorithms have been developed for this problem, which is summarized in Table 1. From the model features listed in Table 1, it is obvious that all the literature on DTSPMS considered two-dimensional container loading constraints. In other words, only horizontal stacking and infeasible load constraints in the horizontal direction were considered. However, in the bulky item delivery problem, the cuboid features of the boxes and vehicles are critical for feasible loading and delivery plans. Therefore, vertical stacking and 3D load-infeasible constraints have to be considered.

The 3D container loading constraints were considered in some research on capacity-constrained vehicle routing problems (3L-CVRP) and pickup and delivery problems (3L-PDP). In the first study to address 3L-CVRP, [27] proposed a tabu search algorithm to solve the 3L-CVRP and considered sequence-based loading, stacking, and vertical stability constraints, as well as the fixed vertical orientation of the items in the vehicles. Reference [4] established the first mathematical model for a 3L-CVRP and proposed a GRASP algorithm to solve it. Reference [6] proposed a 3L-CVRP model, which was solved using a mathematical programming solver Gurobi, with the assumptions of a homogeneous vehicle fleet, sequence-based loading, stacking constraints, orientation constraints, and stability constraints. Reference [28] proposed a heterogeneous fleet vehicle routing problem with 3D loading constraints (3L-HFVRP) and developed an

adaptive variable neighborhood search utilizing an extreme point-based first-fit heuristic to find a feasible loading pattern for each route.

Reference [29] modeled a capacitated vehicle routing problem minimizing fuel consumption under 3D loading constraints (3L-FCVRP) and adopted an evolutionary local search framework that incorporated a recombination method to explore the solution space. Reference [30] proposed a column generation technique-based heuristic for 3L-CVRP and applied an efficient heuristic pricing method to speed up the column generation. Reference [31] and [32] investigated VRPs with backhauls, time windows, and 3D loading constraints. Reference [33], [34] extended the classic pickup and delivery problem to integrate routing and 3D loading problems (3L-PDP). They did not give the mathematical problem formulation but proposed a hybrid algorithm to solve the 3L-PDP.

Reference [35] focused on the split delivery vehicle routing problem with three-dimensional loading constraints combining vehicle routing (3L-SDVRP) with a proposed hybrid algorithm consisting of a local search algorithm for routing and a genetic algorithm and several construction heuristics for packing. Reference [36] investigated a vehicle routing problem with three-dimensional loading constraints and mixed backhauls (3L-VRPMB) and proposed a hybrid metaheuristic consisting of a reactive tabu search for the routing problem and different packing heuristics for the loading problem. Reference [37] proposed a new integrated model to incorporate both the three-dimensional and time window aspects of the routing problem (3L-CVRPTW), developing a two-stage algorithm solve it. In the first stage, a simulated annealing

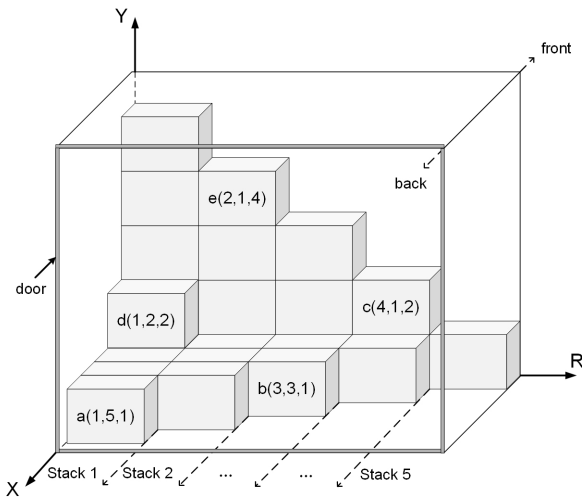


FIGURE 3. Cartesian coordinate system for the 3D vehicle loading space.

algorithm was used to determine the number of vehicles required, the number of routes, the order of the service, and each node’s service time; then, these decisions were entered three-dimensional loading problem.

To our best knowledge, 3D loading constraints have not been addressed in the related works on DTSPMS. Thus, this paper’s major contribution is to mathematically formulate a 3D loading constrained DTSPMS by considering the coupling relationship of the pickup/delivery routes and the item loading sequence. An improved genetic algorithm is designed correspondingly.

III. 3L-DTSPMS FORMULATION

The objective of the 3L-DTSPMS is to identify the shortest routes performing loading and delivery in two relatively separate regions and identify an associated, feasible 3D container loading plan. In this article, the Cartesian coordinate system is employed for the 3D vehicle loading space, as illustrated in Fig.3. It is assumed that the vehicle is rear-loaded (i.e., X dimension in Fig.3). Lifting the boxes in the height direction (i.e., Y dimension in Fig.3) or moving them in the width direction (i.e., R dimension in Fig.3) is not permitted in the loading or unloading operations.

A. NOTATIONS

The notations for the 3L-DTSPMS formulation are shown in Table 2.

B. MATHEMATICAL FORMULATION

The model is formulated based on the following three aspects. First, the loading position is described using the first octant of the Cartesian coordinate system. Fig.3 illustrates the coordinate system with the back-bottom-left corner of the vehicle’s container at the origin of the coordinate system. The boxes’ front-upper-right corner is mapped to Cartesian coordinate point (r, x, y) , where r , x , and y denote the width position, length position, and height position, respectively. We use \mathcal{R} , \mathcal{X} , and \mathcal{Y} to represent the index set of the

TABLE 2. Notations of the 3L-DTSPMS Formulation.

Notation	Explanation
$G^T = (V^T, A^T)$	a directed graph
$T \in \mathcal{T}$	$\mathcal{T} = \{P, D\}$
$I = \{1, 2, \dots, n\}$	the set of items that must be loaded and delivered
$V_0^P = \{1^P, 2^P, \dots, n^P\}$	the set of loading nodes
$V_0^D = \{1^D, 2^D, \dots, n^D\}$	the set of delivery nodes
$0^T = \{0^P, 0^D\}$	the nodes 0^T represent the pseudo depot
$V^T = V_0^T \cup \{0^T\}$	a set of nodes in the directed graph
$A^T = \{(i^T, j^T) \in V^T \times V^T \mid i^T \neq j^T\}$	a set of arcs in the directed graph
$\mathcal{R} = \{1, 2, \dots, W\}$	cardinality of the width
$\mathcal{X} = \{1, 2, \dots, L\}$	cardinality of the length
$\mathcal{Y} = \{1, 2, \dots, H\}$	cardinality of the height
$c_{ij}^T \geq 0, \forall (i, j) \in V^T$	symmetric arc transport costs

corresponding elements. The container is divided into multiple stacks. For instance, the container in Fig.3 is divided into five stacks. Since all the items are identical, we do not consider the items’ true dimensions, but the dimension is used as a unit (i.e., stack 1, stack 2, and so on). Since the item of customer i and the location of customer i correspond one-to-one, without loss of generality, the item of customer i and the location of customer i are represented with the same notation, i . It is assumed that the container is cuboid shaped and that the width, the length, and the height of the container are integers. All items are assumed to have identical shapes and sizes.

Second, the loading position feasibility constraints are considered in terms of vertical stability and blocking relationships. To ensure the boxes’ vertical stability, certain restrictions are introduced to avoid suspending the boxes in the vertical direction. When a box is loaded into the container, it must be placed at the bottom of the container (i.e., boxes a(1,5,1) and b(3,3,1) in Fig.3) or directly above other boxes (i.e., box c(4,1,2), d(1,2,2), and box e(2,1,4) in Fig.3). To describe the blocking relationship constraints, the container is divided into several independently accessible loading stacks. When a box is moved out of the container, all the boxes immediately above and to the right in front of it must first be moved. The boxes are allowed to be piled in the vertical direction since fragility is not considered in this work.

Third, the DTSP route constraints with multiple loading warehouses are considered. It is assumed that each order consists of exactly one item with an associated loading location and delivery destination. Several orders can be picked up from the same warehouse in a practical situation, storing the same item type, which means a warehouse with shared pickup items might be revisited in the loading process. To manage this situation, it is assumed that different orders correspond to different loading nodes, while different loading nodes may have the same coordinates. The distance between any two loading nodes with the same coordinates is assumed to be zero.

An integer linear programming formulation for the 3L-DTSPMS is given as follows. The objective is to minimize the total loading and delivery route costs with DTSP

TABLE 3. Decision Variables of the 3L-DTSPMS Formulation.

Decision variables	Explanation
x_{ij}^T	equals 1 if the truck goes directly from loading node i to loading node j , and 0 otherwise $i, j \in V^T, T \in \mathcal{T}$
y_{ij}^T	equals 1 if loading node i is visited before loading node j , and 0 otherwise $i, j \in V^T, T \in \mathcal{T}$
z_i^{rxy}	equals 1 if item $i \in I$ is placed in the cartesian coordinate point (r, x, y) , and 0 otherwise $r \in \mathcal{R}, x \in \mathcal{X}, y \in \mathcal{Y}$
ξ^{rxy}	equals 1 if there is an item located in the cartesian coordinate point (r, x, y) , and 0 otherwise $r \in \mathcal{R}, x \in \mathcal{X}, y \in \mathcal{Y}$

route constraints, loading stability constraints, and blocking relationship constraints in a 3D vehicle container.

Objective function:

$$\min \sum_{\substack{T \in \mathcal{T} \\ i, j \in V^T}} c_{ij}^T x_{ij}^T \tag{1}$$

$$s.t. \sum_{i \in V^T} x_{ij}^T = 1, \quad T \in \mathcal{T}, j \in V^T \tag{2}$$

$$\sum_{j \in V^T} x_{ij}^T = 1, \quad T \in \mathcal{T}, i \in V^T \tag{3}$$

$$y_{ij}^T + y_{ji}^T = 1, \quad T \in \mathcal{T}, i, j \in I \tag{4}$$

$$y_{ik}^T + y_{kj}^T \leq y_{ij}^T + 1, \quad T \in \mathcal{T}, i, j, k \in I \tag{5}$$

$$x_{ij}^T \leq y_{ij}^T, \quad T \in \mathcal{T}, i, j \in I \tag{6}$$

$$\sum_{r \in \mathcal{R}, x \in \mathcal{X}, y \in \mathcal{Y}} z_i^{rxy} = 1, \quad i \in I \tag{7}$$

$$\sum_{i \in I} z_i^{rxy} = 1, \quad r \in \mathcal{R}, x \in \mathcal{X}, y \in \mathcal{Y} \tag{8}$$

$$z_i^{rx(y+1)} \leq \xi^{rxy}, \quad r \in \mathcal{R}, x \in \mathcal{X}, y \in \mathcal{Y}, i \in I \tag{9}$$

$$\xi^{rxy} = \sum_{i \in I} z_i^{rxy}, \quad r \in \mathcal{R}, x \in \mathcal{X}, y \in \mathcal{Y} \tag{10}$$

$$z_i^{rxy} + z_j^{r(x+s)y} \leq 1 + y_{ij}^D, \quad r \in \mathcal{R}, x \in \mathcal{X}, s \in \mathcal{S}, \tag{11}$$

$$\mathcal{S} = \{1, 2, \dots, L - x\}, y \in \mathcal{Y}, i, j \in I, i \neq j$$

$$z_i^{rxy} + z_j^{r(x+s)y} \leq 1 + y_{ji}^D, \quad r \in \mathcal{R}, x \in \mathcal{X}, s \in \mathcal{S}, \tag{12}$$

$$\mathcal{S} = \{1, 2, \dots, L - x\}, y \in \mathcal{Y}, i, j \in I, i \neq j$$

$$z_i^{rxy} + z_j^{rx(y+1)} \leq 1 + y_{ij}^D, \tag{13}$$

$$r \in \mathcal{R}, x \in \mathcal{X}, y \in \mathcal{Y}, i, j \in I, i \neq j$$

$$z_i^{rxy} + z_j^{rx(y+1)} \leq 1 + y_{ji}^D, \tag{14}$$

$$r \in \mathcal{R}, x \in \mathcal{X}, y \in \mathcal{Y}, i, j \in I, i \neq j$$

$$x_{ij}^T, y_{ij}^T \in \mathbb{B} \quad T \in \mathcal{T}, i, j \in V^T, i \neq j \tag{15}$$

$$z_i^{rxy}, \xi^{rxy} \in \mathbb{B}, \quad r \in \mathcal{R}, x \in \mathcal{X}, y \in \mathcal{Y}, i \in I \tag{16}$$

1) DTSP ROUTE CONSTRAINTS

Equation (2) and (4) restrict each loading or delivery operation to only one loading or delivery node. Equation (4) guarantees the only precedence relationship between any two nodes. Equation (5) ensures that if node i is visited before node k , and node k is visited before node j , then node i is always visited before node j . In addition, (5) also plays a role

as a subtour eliminator. Equation (6) restricts the relationship between variable x_{ij}^T and variable y_{ij}^T . Equation (15) indicates that the decision variable is binary.

2) LOADING STABILITY CONSTRAINTS

Equation (7) ensures that each box can only be loaded in one position in the container. Equation (8) indicates that one position can only be assigned to one box. Equation (9) indicates that if ξ^{rxy} takes the value 1, then there is a box at coordinate point (r, x, y) , and box i may or may not be loaded immediately above the coordinate point (r, x, y) . Otherwise, if ξ^{rxy} takes the value 0, then box i cannot be loaded immediately above the coordinate point (r, x, y) . Additionally, (10) ensures that there is only one box at coordinate point (r, x, y) , and restricts the relationship between variable ξ^{rxy} and variable z_i^{rxy} . Equation (16) indicates that the decision variable is binary.

3) BLOCKING RELATIONSHIP CONSTRAINTS

Equations (11) - (14) bind the container loading variables to the routing problem variables and ensure that box reloading will not happen in the delivery process. To describe the blocking relationship in the horizontal direction, (11) indicates that if box i is loaded in position (r, x, y) , and box j is to be immediately loaded next, then box j might be loaded in position $(r, x + 1, y)$ or $(r, x + 2, y), \dots$, or (r, L, y) , and thus $y_{ij}^D = 1$. In this case, (12) guarantees that box j must be delivered before box i in the delivery route, i.e. $y_{ji}^D = 1$, since box i is blocked by box j in the horizontal direction. To describe the blocking relationship in the vertical direction, (13) indicates that if box i is loaded in position (r, x, y) , the box j is to be immediately loaded next, and then box j might be loaded in position $(r, x, y + 1)$, and thus $y_{ij}^D = 1$. In this case, (14) guarantees that box j must be delivered before box i in the delivery route, i.e. $y_{ji}^D = 1$, since box i is blocked by box j in the vertical direction.

IV. GENETIC ALGORITHM FOR 3L-DTSPMS

The proposed model can be solved using a standard MIP solver, Gurobi, to obtain optimal solutions when the problem size is smaller than 12 items (i.e., there are 24 nodes in the loading and delivery network). However, when the problem size reaches 13 items (i.e., there are 26 nodes in the loading and delivery network), it will take hours for Gurobi to find the optimal solution. According to the data from the RRS Logistics Company in China, a typical number of orders handled by a vehicle is commonly in the range of 30 to 50 items. Therefore, the genetic algorithm is employed in this article.

A. STANDARD GENETIC ALGORITHM

In the standard genetic algorithm, the delivery route is represented as a chromosome, and chromosomes make up the initial population G_0 . A loading route and loading plan based on the current population generation are then constructed. The complete solution consists of a loading route, a 3D container loading plan, and a delivery route. After a delivery

route (a chromosome in this population) has been generated, a loading route is generated by inverting the delivery route. For example, when the delivery route is $v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_n$, the loading route is $v_n \rightarrow v_{s-1} \rightarrow \dots \rightarrow v_1$. A feasible 3D container loading plan is then generated as follows. The loading route identifies the loading sequence. The first box is loaded in the back-bottom-left corner of the container, i.e., the initial position (1, 1, 1). For subsequently loaded boxes, the set of available positions are generated in turn. These positions must be in front, to the right, or immediately above the previously loaded box. The loading position is randomly selected from the set of available positions when an item is loaded until all the items are loaded into the container. The total route cost C of a complete solution is used to indicate the fitness of the population Gen_k ($k = 0, 1, \dots, k_{max}$), the fitness function is expressed as $s_k = 1/C$. After the current population's fitness is calculated, the genetic algorithm selection, crossover, and mutation are carried out below. The crossover and mutation operators work only on the delivery routes, adopting the genetic algorithm's standard operation, namely, the roulette wheel selection strategy, position-based crossover, and reciprocal exchange mutation. Multiple iterations are then carried out. The procedure for the standard genetic algorithm is given in Algorithm 1.

Reference [38] showed that the roulette wheel selection complexity was $O(n^2)$, where n is the number of customer points in the delivery route. Therefore, the genetic algorithm's complexity can approximately express as $O(MaxGen \times \lambda n^2)$. Where $MaxGen$ represents the maximum number of iterations, and λ represents the population size. In the standard genetic algorithm for solving 3L-DTSPMS, the loading route is reversely generated according to the delivery route. The complexity analysis of generating packing scheme according to packing route is as follows: load the first customer's box on the loading route to the (1,1,1) position of the container, the second customer's box can be placed in a maximum of three positions in the container, i.e., in front, to the right, or immediately above the previously loaded box, the third customer's goods can be placed in a maximum of 5 positions in the container, i.e., in front, to the right, or immediately above the previously loaded box, and so on. The formula for calculating the worst time complexity of the packing part is shown in the formula: $O(1) + O(3) + O(5) + \dots + O(2n - 1) = O(n^2)$. Therefore, the standard genetic algorithm's complexity is $O(MaxGen \times \lambda(n^2 + n^2))$.

B. IMPROVED GENETIC ALGORITHM

An improved genetic algorithm is now introduced to tackle the 3L-DTSPMS and improve the delivery and loading route. In generating and improving the delivery route, the genetic algorithm is used to optimize and improve the delivery route. The Lin-Kernighan algorithm ([39]–[41]) improves the genetic algorithm in the initial population generation part. As the pickup warehouses are independent of each other, and because customers' items may be located in the same warehouse, a k-means clustering algorithm ([42]) is used

Algorithm 1 Standard Genetic Algorithm

Require: 3L-DTSPMS problem data, item number N , crossover probability p_c , mutation probability p_m , maximum iteration number k_{max}

Ensure: Best solution to 3L-DTSPMS

- 1: Construct initial population G_0 , each chromosome is encoded by a delivery route
 - 2: **for** ($k := 1$ to k_{max}) **do**
 - 3: Reverse all chromosomes to form the population G_{k-1} loading route
 - 4: Generate the loading plan according to the loading route G_{k-1}
 - 5: Evaluate the fitness of initial population G_{k-1} , $s^* \in G_{k-1}$ is the current best solution
 - 6: Select the delivery route in G_{k-1}
 - 7: Parental chromosomes are selected in G_{k-1} by roulette wheel selection
 - 8: Crossover using position based crossover operator with probability p_c
 - 9: Mutate using reciprocal exchange mutation operator with probability p_m
 - 10: Generate complete solution
 - 11: Generate G_k
 - 12: Calculate the fitness of population G_k , and set the current best solution as s_c
 - 13: Set $s_k = s_c$
 - 14: **if** the fitness of $s_c \leq$ the fitness of s^* **then**
 - 15: $s^* = s_c$
 - 16: **end if**
 - 17: **end for**
 - 18: **Return** the best solution.
-

to optimize the loading route when the route is processed. The blocking relation of the loading plan is also considered at this point, and a heuristic method is adopted to regenerate the loading path.

First, an approximate optimal delivery route is generated according to a genetic algorithm. In this stage, the Lin-Kernighan algorithm is used to optimize the generation of the delivery route's initial solution. The pseudo-code of the algorithm is shown in Algorithm 2. In contrast to Algorithm 1, the delivery route's cost is used as the evaluation of fitness function. According to the Lin-Kernighan algorithm ([41]), suppose $g_i = D_{start(X_i)end(X_i)} - D_{start(Y_i)end(Y_i)}$ is the gain of deleting an edge X_i and adding Y_i . The D value of an edge X_i is expressed as the distance from $start(X_i)$ to $end(X_i)$, G_i be the sum of $g_1 + \dots + g_i$. See Algorithm 3 for the pseudo code of the Lin-Kernighan algorithm used. Refer to the [39] for the specific content of the Lin-Kernighan algorithm. The complexity of Lin-Kernighan algorithm was $O(n^{2.2})$ (see reference [39] for details). Therefore, the complexity of improved delivery route GA algorithm is $O(MaxGen \times \lambda(n^2 + n^{2.2}))$.

Since the loading route in GA (Algorithm 1) is generated by the direct reversal of the delivery route, many routes

Algorithm 2 Improved Delivery Route Genetic Algorithm

Require: Items delivery location data, item number N , crossover probability p_c , mutation probability p_m , maximum iteration number k_{max}

Ensure: Best solution to delivery route

- 1: Construct initial population Gen_0 , each chromosome is encoded by a delivery route
- 2: **for** ($k := 1$ to k_{max}) **do**
- 3: **for** (each chromosome of population Gen_{k-1}) **do**
- 4: The Lin-Kernighan algorithm(Algorithm 3) is used to improve the initial chromosome which consisting of a distribution route
- 5: Generate new each chromosome of population Gen_{k-1}
- 6: **end for**
- 7: Selection, crossover, mutation and evolution of Gen_{k-1} chromosome are perform. The specific steps are shown in step 5-16 of Algorithm 1
- 8: **end for**
- 9: **Return** the best solution.

that repeatedly visit the same warehouse. Therefore, so the next step is to improve the loading route. Due to the coupling relationship between the loading route and the delivery route, when the approximate optimal delivery route and loading route are generated, respectively, a blocking relationship is established between items. The blocking relationship between the loading route and the delivery route should be considered first when optimizing the loading route. Therefore, steps 3 and 4 in Algorithm 1 need to be improved. When improving the loading route, the loading plan also needs to be changed to align the current loading route and packing scheme with the current delivery route.

The blocking relationship between items can be defined as follows: if two items i and j in two route networks (loading route and delivery route) are not accessed in the same order (i.e., i precedes j or j precedes i), then the two items have a blocking relationship. The constraint of the load blocking relationship is that for an item with coordinates (r, x, y) , only items with coordinates $(r, x, y+1)$ above and $(r, x+1, y)$ behind may blocked that item. The route's adjustment mode is illustrated in Fig.4, which is shown from the same perspective as that in Fig.2. If the items i and j are located in the same stack, there are three possible situations (i.e. in B1, B2, and U1 Fig.4). For the situations B1 and B2 situations in Fig.4, since items i and j have a blocking relationship, their positions in the loading route and the loading plan will be exchanged to preserve the feasibility. For situation U1 in Fig.4, there is no need to exchange the positions of items i and j in the loading route since they do not have a blocking relationship. If items i and j are located in different stacks as illustrated by U2 in Fig.4, they do not have a blocking relationship. Therefore, it is not necessary to exchange their positions in the loading route.

Algorithm 3 Lin-Kernighan Algorithm

Require: An initial chromosome consisting of a delivery route T_s

Ensure: An improved route T^*

- 1: **while** No stopping criterion is triggered **do**
- 2: Set $i = 1$. Choose a node from route T_s as t_1 . t_1 is the start node of the entire search procedure
- 3: **if** every node has already been tested as t_1 **then**
- 4: Stop and return the improved route T^*
- 5: **end if**
- 6: Choose an edge $X_1 = (t_1, t_2)$ that belongs to T
- 7: **if** all edges have been tried as X_1 **then**
- 8: go to step 2
- 9: **end if**
- 10: Choose an edge $Y_1 = (t_2, t_3)$ that does not belong to T . so that $G_1 = g_1 > 0$
- 11: **if** this is impossible and all choices for Y_1 have been tested **then**
- 12: go to step 6
- 13: **end if**
- 14: Set $i = i + 1$;
- 15: Choose $X_i = (t_{2i-1}, t_{2i})$, such that:
- 16: a. $X_i \neq Y_p$, for all $p < i$, and
- 17: b. Add an edge between t_{2i} and t_1 so that $T \cup Y \setminus X$ can form a route T'
- 18: **if** T' is a shorter route than T **then**
- 19: set $T = T'$ and $T^* = T'$
- 20: **end if**
- 21: **if** $i = 2$ and all choices for X_i have been tested **then**
- 22: set $i = 1$, go to step 10
- 23: **end if**
- 24: **if** $i > 2$ and all choices for X_i have been tested **then**
- 25: set $i = 2$, go to step 27
- 26: **end if**
- 27: Choose $Y_i = (t_{2i}, t_{2i+1})$, such that:
- 28: a. $G_i = g_1 + \dots + g_i > 0$,
- 29: b. $Y_i \neq X_p$ for all $p \leq i$, and
- 30: c. X_{i+1} exists
- 31: **if** Y_i can be chosen **then**
- 32: go to step 14
- 33: **end if**
- 34: **if** $i = 2$ and all choices for Y_i have been tested **then**
- 35: go to step 15
- 36: **end if**
- 37: **if** $i > 2$ and all choices for Y_i have been tested **then**
- 38: go to step 27
- 39: **end if**
- 40: **end while**

In the improved loading route stage, the loading route is optimized by the k-means clustering algorithm ([42]). After the k-means clustering algorithm is completed, each category's nodes are randomly connected according to the clustering category to form a complete loading route. A feasible

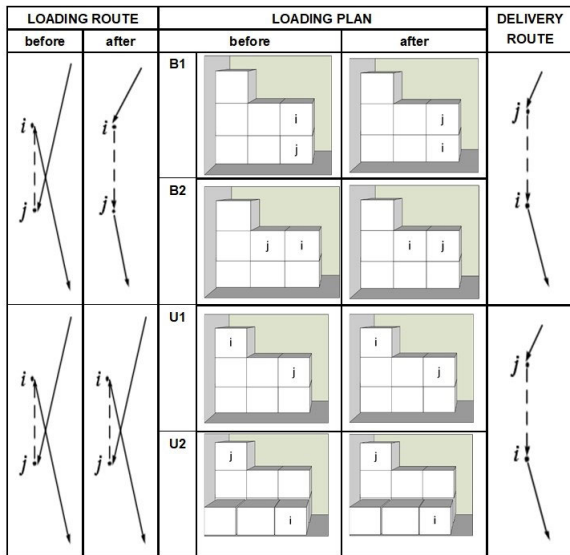


FIGURE 4. The relationship between DTSP and loading plan.

3D container loading plan is then generated according to Section IV-A. Then the blocking relationship between the loading route and the delivery route is then determined. If there is a blocking relationship between items, the loading route of the item corresponding to the blocked location is exchanged until all the loading plans are not blocking with the delivery route. The complexity of k-means clustering is $O(n \log^k n)$ (see reference [43] for details), where k is the number of clustering. The improved loading route clustering algorithm also includes generating the final packing plan, where the complexity of the packing scheme is also express as $O(1) + O(3) + O(5) + \dots + O(2n - 1) = O(n^2)$. Therefore, the complexity of the improved loading route clustering algorithm is $O(n^2 + n \log^k n)$.

The complete solution in this part consists of a loading route and a 3D container loading plan. Thereafter, the improved delivery route genetic algorithm (Algorithm 2) combined with the improved loading route clustering algorithm (Algorithm 4) is consequently referred to as the improved genetic algorithm (IGA).

V. NUMERICAL STUDY

The proposed 3L-DTSPMS model is solved using Gurobi 8.1.0 when the number of orders is less than 12, which means there are less than 24 nodes in the loading and delivery network. The standard genetic algorithm and the improved genetic algorithm are implemented using Java 4.6.3. Three groups of numerical studies are conducted with 80 instances (R01-R80) in total and are run on a Lenovo 20H1001NCD laptop with 12 GB RAM and a 2.70 GHz processor. Unless noted otherwise, the maximum allowable runtime is set as one hour.

A. TEST INSTANCES

The test instances used are designed following the idea of generating the DTSPMS test instances in [1]. With the warehouse located at (0,0), the items' loading locations are

Algorithm 4 Improved Loading Route Clustering Algorithm

Require: Items loading location data, item number n , delivery route.

Ensure: Best solution to loading route.

- 1: Randomly assign all data items to a k initial cluster
- 2: **repeat**
- 3: Compute centroids for each cluster
- 4: Reassign each data item to cluster of closest centroid
- 5: **until** no change in cluster assignments
- 6: Randomly connect the nodes in each category according to the clustering category to form a complete loading route
- 7: Generate the loading scheme according to the loading route
- 8: Update new loading routes and loading schemes until there is no blocking relationship between loading scheme and delivery route
- 9: **Return** the best solution.

randomly positioned in the area of -10×-10 . The associated delivery destinations are then randomly generated in the area of 100×100 . Euclidean distances are rounded to the smallest integer.

In this numerical study, different container configurations are tested. These are defined by (W, L, H) , with W, L , and H representing the width, the length, and the height of the container, respectively. The optimal solutions are obtained by Gurobi for test instances from R01 to R50 with the number of nodes ranging from 16 to 24. The number of nodes for the test instances from R51 to R80 ranges from 36 to 100, in which Gurobi cannot obtain optimal solutions within the prescribed maximum allowable runtime of one hour. In the instances from R01 to R50, the genetic algorithm's performances and improved genetic algorithm are compared with the optimal solutions obtained by Gurobi. In the instances from R51 to R80, the appropriate lower bounds are calculated for use as the performance benchmark. For the GA and IGA, the parameters of the crossover rate, mutation rate, and population size are set as 0.70, 0.10, and 70, respectively. A sensitivity analysis is conducted to evaluate the impact of parameter setting.

B. COMPUTATIONAL RESULTS

The numerical study was divided into three groups for testing: the instances from R01 to R20 considering 16 nodes in the same container configuration, from R21 to R50 considering 16 to 24 nodes in different container configurations, and from R51 to R80 considering 36 to 100 nodes in different container configurations.

1) RESULTS OF THE INSTANCES WITH 16 NODES

In the instances from R01 to R20, the number of nodes is set as $N = 16$ (i.e., containing 8 orders), excluding two pseudo-depots in the loading and delivery regions. A lower bound for the proposed 3L-DTSPMS problem is calculated

TABLE 4. Results of the Instances With 16 Nodes.

Instance	N	CC	OPT	LB	GA/OPT	IGA/OPT
R01	16	(2,3,2)	371	370	1.00	1.00
R02	16	(2,3,2)	345	345	1.00	1.00
R03	16	(2,3,2)	367	366	1.00	1.00
R04	16	(2,3,2)	306	305	1.00	1.00
R06	16	(2,3,2)	279	278	1.00	1.00
R07	16	(2,3,2)	347	346	1.00	1.00
R08	16	(2,3,2)	373	372	1.00	1.00
R09	16	(2,3,2)	319	317	1.00	1.00
R10	16	(2,3,2)	373	372	1.00	1.00
R11	16	(2,3,2)	342	340	1.00	1.00
R12	16	(2,3,2)	342	342	1.00	1.00
R13	16	(2,3,2)	319	318	1.00	1.00
R14	16	(2,3,2)	333	332	1.00	1.00
R15	16	(2,3,2)	315	314	1.00	1.00
R16	16	(2,3,2)	360	359	1.00	1.00
R17	16	(2,3,2)	373	373	1.00	1.00
R18	16	(2,3,2)	331	331	1.00	1.00
R19	16	(2,3,2)	355	354	1.00	1.00
R20	16	(2,3,2)	343	342	1.00	1.00
Avg	-	-	-	-	1.00	1.00

$p_c = 0.70, p_m = 0.10, \text{population size} = 70$

using Gurobi, referred to as *LB* in Table 4. The loading route and delivery route are considered two separate TSPs and are solved using Gurobi. The sum of the costs of these two TSPs is set as the lower bound. The results are given in Table 4, where *OPT* is the optimal solution obtained by Gurobi, *CC* is the container configuration (W, L, H), and *GA/OPT* and *IGA/OPT* are $1/\text{gaps}$ between the genetic algorithm and the improved genetic algorithm to the optimal solution, respectively.

The results showed that the GA and IGA could converge rapidly (within 10 seconds), and the optimal solution was found for all instances from R01 to R20. The optimal solutions of the proposed 3L-DTSPMS are quite close to the lower bound. This result is because when the problem size is small, there are many feasible loading plans in a container with a fixed volume. Thus, finding an optimal loading plan is possible to be found associated with the optimal loading and delivery routes.

2) RESULTS OF THE INSTANCES WITH 16 TO 24 NODES

In the instances from R21 to R50, the impacts of the container configurations and problem size are evaluated. In the instances from R21 to R35, the number of nodes is fixed at $N = 20$, while the container configuration is adjusted. In the instances from R36 to R50, both the number of nodes and the container configuration are adjusted. The parameter setting here is consistent with Section V-B1. The maximum problem size is limited to $N = 24$ since it is the maximum size of the problem that can be solved by Gurobi within one hour. The results are given in Table 5.

The results show that the GA and IGA converged rapidly (less than 20 seconds); again, the optimal solutions of the proposed 3L-DTSPMS are quite close to the lower bound (*LB*). In these instances, the performance of the IGA is worse than that of the GA, which may be because the optimal loading route of a single commodity was destroyed during the loading route clustering process, increasing the

TABLE 5. Results of the Instances With 16 to 24 Nodes.

Instance	N	CC	OPT	LB	GA/OPT	IGA/OPT
R21	20	(2,3,2)	404	404	1.00	1.00
R22	20	(2,3,2)	359	359	1.00	1.04
R23	20	(2,3,2)	371	371	1.00	1.00
R24	20	(2,3,2)	331	331	1.00	1.06
R25	20	(2,3,2)	367	365	1.00	1.00
R26	20	(3,2,2)	404	404	1.00	1.00
R27	20	(3,2,2)	359	359	1.00	1.04
R28	20	(3,2,2)	371	371	1.00	1.00
R29	20	(3,2,2)	331	331	1.00	1.07
R30	20	(3,2,2)	366	365	1.00	1.00
R31	20	(2,2,3)	404	404	1.00	1.00
R32	20	(2,2,3)	359	359	1.00	1.04
R33	20	(2,2,3)	371	371	1.00	1.00
R34	20	(2,2,3)	331	331	1.00	1.03
R35	20	(2,2,3)	367	365	1.00	1.00
R36	16	(2,4,2)	371	371	1.00	1.00
R37	18	(2,4,2)	358	358	1.00	1.03
R38	20	(2,4,2)	371	371	1.00	1.00
R39	22	(2,4,2)	348	348	1.00	1.02
R40	24	(2,4,2)	389	388	1.01	1.00
R41	16	(4,2,2)	371	371	1.00	1.00
R42	18	(4,2,2)	358	358	1.00	1.00
R43	20	(4,2,2)	371	371	1.00	1.00
R44	22	(4,2,2)	348	348	1.00	1.00
R45	24	(4,2,2)	388	388	1.01	1.01
R46	16	(2,2,4)	371	371	1.00	1.00
R47	18	(2,2,4)	358	358	1.00	1.00
R48	20	(2,2,4)	371	371	1.00	1.00
R49	22	(2,2,4)	348	348	1.00	1.00
R50	24	(2,2,4)	389	388	1.01	1.00
Avg	-	-	-	-	1.001	1.011

$p_c = 0.70, p_m = 0.10, \text{population size} = 70$

total route cost. By comparing the results under the different container configurations (2,3,2), (3,2,2), and (2,2,3) in the instances from R21 to R25, R26 to R30, and R31 to R35, respectively, it is observed that for a certain problem size and a certain container volume, changing the dimension of the container does not significantly influence the optimal solution. This outcome can also be observed when the container configuration is set as (2,4,2), (4,2,2), and (2,2,4). Notably, in the instances R40, R45, and R50 with the problem size set as $N = 24$ and the container configuration set as (2,4,2), (4,2,2), and (2,2,4), respectively, a change in the container's dimensions significantly impacts the computational time of Gurobi. It took Gurobi less than 120 seconds to find the optimal solution for the instances R45 and R50 associated with configurations (4,2,2) and (2,2,4). In comparison, it took Gurobi approximately 660 seconds for instance R40 associated with the configuration (2,4,2). This indicates that the container's length impacts the difficulty of solving the problem more significantly than the container's width or height. A similar conclusion was drawn by [7] for a DTSPMS in a 2D container situation.

3) RESULTS OF THE INSTANCES WITH 36 TO 100 NODES

When the problem size is larger than 24 nodes, the optimal solution cannot be obtained by Gurobi within one hour. Thus, in the instances from R51 to R80, where $36 \leq N \leq 100$, the lower bound (*LB*) calculated by Gurobi is used as a benchmark to evaluate the performance of the algorithms. The parameter setting here is consistent with Section V-B1.

TABLE 6. Results of the Instances With 36 to 100 Nodes.

Instance	N	CC	LB	GA/LB	IGA/LB
R51	36	(3,3,3)	449	1.08	1.07
R52	36	(3,3,3)	452	1.09	1.08
R53	36	(3,3,3)	453	1.06	1.06
R54	36	(3,3,3)	412	1.06	1.06
R55	36	(3,3,3)	422	1.05	1.05
R56	40	(3,3,3)	455	1.10	1.07
R57	40	(3,3,3)	455	1.11	1.08
R58	40	(3,3,3)	452	1.08	1.06
R59	40	(3,3,3)	457	1.05	1.05
R60	40	(3,3,3)	431	1.05	1.04
Avg 1	-	-	-	1.073	1.062
R61	60	(3,4,3)	573	1.17	1.05
R62	60	(3,4,3)	564	1.18	1.10
R63	60	(3,4,3)	551	1.19	1.10
R64	60	(3,4,3)	543	1.16	1.11
R65	60	(3,4,3)	548	1.16	1.12
R66	76	(3,4,3)	639	1.19	1.12
R67	76	(3,4,3)	645	1.40	1.13
R68	76	(3,4,3)	582	1.36	1.14
R69	76	(3,4,3)	619	1.25	1.15
R70	76	(3,4,3)	614	1.34	1.17
Avg 2	-	-	-	1.250	1.116
R71	80	(3,5,3)	680	1.26	1.08
R72	80	(3,5,3)	686	1.34	1.08
R73	80	(3,5,3)	603	1.42	1.16
R74	80	(3,5,3)	656	1.37	1.07
R75	80	(3,5,3)	653	1.35	1.11
R76	100	(4,5,3)	782	1.31	1.08
R77	100	(4,5,3)	723	1.33	1.11
R78	100	(4,5,3)	686	1.49	1.14
R79	100	(4,5,3)	734	1.27	1.15
R80	100	(4,5,3)	763	1.37	1.11
Avg 3	-	-	-	1.351	1.109

$p_c = 0.70, p_m = 0.10, \text{population size} = 70$

It can be seen that the performance of the IGA is significantly better than that of the GA, no matter in terms of time consumption or numerical results. When the problem scale is small ($36 \leq N \leq 40$), the effect of the GA and IGA is very similar. As the size of the problem increases from 60 to 100, the quality of the solution obtained by the GA becomes increasingly worse, and it takes approximately 5 minutes to converge.

When the problem scale is less than 60 nodes, the algorithm converges within 60 seconds; when the problem scale is 80 nodes, the algorithm converges within 200 seconds; and when the problem scale is 100 nodes, the algorithm converges within 600 seconds. The quality of the solution obtained by the IGA also diminishes, but it is a great improvement over the GA, and the solution can be obtained with this quality in a few seconds. When the data size increased, the IGA performed better than the GA, possibly because the optimal loading routes for many items were not broken during the loading route clustering. Moreover, the bigger the problem size was, the better the IGA performed over GA. This result is consistent with objective reasoning: When the problem scale is small, clustering is not significant, and when the problem scale is large, clustering is more accurate and targeted.

C. SENSITIVITY ANALYSIS

The population size, crossover rate, and mutation rate are the GA parameters that might impact the algorithm’s performance. In this study, the scale of test cases is between

16 and 100 nodes. The population size is set between 50 and 100 based on experience, and 50, 70, 90, and 100 are selected as the test points for observing the influence of population size on algorithm performance. The crossover rate is the probability of two chromosomes crossing, which is generally high. In this study, the crossover rate ranges from 0.3 to 0.9, and the interval is divided into three segments on average. The endpoint values of 0.3, 0.5, 0.7, and 0.9 are selected to test the crossover rate’s impact. The mutation rate is the probability of gene mutation in the chromosome, which is relatively low. In this study, the mutation rate ranges from 0.05 to 0.20, and the interval is divided into three segments on average. The endpoint values of 0.05, 0.10, 0.15, and 0.20 are selected to test the mutation rate’s impact.

For all the instances of R01 to R80, a sensitivity analysis is conducted to evaluate the impact of population size, crossover rate, and mutation rate on the performance of the proposed IGA. The results show that the impacts of the parameter settings are not statistically significant.

VI. CONCLUSION

In bulky item logistics practice, loading and delivery routes are inherently coupled with vehicle loading plans in a 3D circumstance. Thus, this article modeled a bulky item delivery problem as a double traveling salesman problem with three-dimensional container loading constraints with multiple stacks (3L-DTSPMS). The major contribution of this work is to take 2D container loading constraints from existing DTSPMS literature and extend it into 3D container loading constraints using the proposed integer linear programming model, in which loading stability constraints, blocking relationship constraints, and DTSP routing constraints are formulated to obtain the joint optimization of DTSP and 3D container loading problems. To solve the model, Gurobi can obtain an optimal solution for relatively small problem instances (i.e., less than 24 nodes in the loading and delivery network in our numerical study). For larger problems, the standard genetic algorithm and the improved genetic algorithm are proposed. Three different implementations have been tested, and the results are summarized as follows: when considering the homogeneous container, 16 nodes, the GA and the IGA can converge to the optimal solution in 10 seconds; when considering heterogeneous container, nodes from 16 to 24, the performance of IGA was slightly worse than that of GA. By comparing the test results of the length, width, and height of the container under the heterogeneous container, it is found that the length of the container has a greater influence on the difficulty of solving the problem than the width or height of the container; when considering nodes from 36 to 100, the proposed IGA is superior to the standard GA. The results of the sensitivity analysis showed that the influence of parameter setting was not statistically significant.

However, in the real-world, loading and delivering bulky items are much more complicated processes. Thus, further studies are necessary, possibly in the following directions. First, the assumption of identical item sizes should be relaxed

TABLE 7. Abbreviations of This Paper.

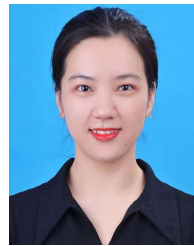
Abbreviation	The meaning of an abbreviation
DTSP	Double traveling salesman problem
DTPMS	Double traveling salesman problem with multiple stacks
3D	Three-dimensional
3L-CVRP	Capacity constrained vehicle routing problem with three-dimensional loading constraints
3L-DTSPMS	DTSP with the 3D container loading constraints with multiple stacks
3L-PDP	Pickup and delivery problem with 3D loading constraints
3L-HFVRP	Heterogeneous fleet vehicle routing problem with 3D loading constraints
3L-FCVRP	Capacitated vehicle routing problem minimizing fuel consumption under 3D loading constraints
3L-VRPMB	Vehicle routing problem with three-dimensional loading constraints and mixed backhauls
3L-CVRPTW	Three-dimensional capacitated vehicle routing problem considering time windows
3L-SDVRP	Split delivery vehicle routing problem with three-dimensional loading constraints combines vehicle routing

in future work. Second, multi vehicles with different capacity limitations is another direction worthy of further study, especially when a delivery time window needs to be met. Third, more complicated loading constraints should be considered, including the vulnerability of items, the loading direction, item weights, and center of gravity.

APPENDIX. A REFERENCES

- [1] H. L. Petersen and O. B. G. Madsen, "The double travelling salesman problem with multiple stacks—Formulation and heuristic solution approaches," *Eur. J. Oper. Res.*, vol. 198, no. 1, pp. 139–147, Oct. 2009, doi: [10.1016/j.ejor.2008.08.009](https://doi.org/10.1016/j.ejor.2008.08.009).
- [2] M. Iori and S. Martello, "Routing problems with loading constraints," *Top*, vol. 18, no. 1, pp. 4–27, Jul. 2010, doi: [10.1007/s11750-010-0144-x](https://doi.org/10.1007/s11750-010-0144-x).
- [3] H. Pollaris, K. Braekers, A. Caris, G. K. Janssens, and S. Limbourg, "Vehicle routing problems with loading constraints: State-of-the-art and future directions," *OR Spectr.*, vol. 37, no. 2, pp. 297–330, Dec. 2015, doi: [10.1007/s00291-014-0386-3](https://doi.org/10.1007/s00291-014-0386-3).
- [4] A. Moura and J. F. Oliveira, "An integrated approach to the vehicle routing and container loading problems," *OR Spectr.*, vol. 31, no. 4, pp. 775–800, Mar. 2009, doi: [10.1007/s00291-008-0129-4](https://doi.org/10.1007/s00291-008-0129-4).
- [5] S. Ceschia, A. Schaefer, and T. Stutzle, "Local search techniques for a routing-packing problem," *Comput. Ind. Eng.*, vol. 66, no. 4, pp. 1138–1149, Dec. 2013, doi: [10.1016/j.cie.2013.07.025](https://doi.org/10.1016/j.cie.2013.07.025).
- [6] L. Junqueira, J. F. Oliveira, M. A. Carravilla, and R. Morabito, "An optimization model for the vehicle routing problem with practical three-dimensional loading constraints," *Int. Trans. Oper. Res.*, vol. 20, no. 5, pp. 645–666, Sep. 2013, doi: [10.1111/j.1475-3995.2012.00872.x](https://doi.org/10.1111/j.1475-3995.2012.00872.x).
- [7] H. L. Petersen, C. Archetti, and M. G. Speranza, "Exact solutions to the double travelling salesman problem with multiple stacks," *Networks*, vol. 56, no. 4, pp. 229–243, Dec. 2010, doi: [10.1002/net.20375](https://doi.org/10.1002/net.20375).
- [8] M. A. A. Martinez, J. F. Cordeau, A. M. Dell, and M. Iori, "A branch-and-cut algorithm for the double traveling salesman problem with multiple stacks," *Inform. J. Comput.*, vol. 25, no. 1, pp. 41–55, Feb. 2013, doi: [10.1287/ijoc.1110.0489](https://doi.org/10.1287/ijoc.1110.0489).
- [9] M. Batista-Galván, J. Riera-Ledesma, and J. J. Salazar-González, "The traveling purchaser problem, with multiple stacks and deliveries: A branch-and-cut approach," *Comput. Oper. Res.*, vol. 40, no. 8, pp. 2103–2115, Aug. 2013, doi: [10.1016/j.cor.2013.02.007](https://doi.org/10.1016/j.cor.2013.02.007).
- [10] M. Iori and J. Riera-Ledesma, "Exact algorithms for the double vehicle routing problem with multiple stacks," *Comput. Oper. Res.*, vol. 63, pp. 83–101, Nov. 2015, doi: [10.1016/j.cor.2015.04.016](https://doi.org/10.1016/j.cor.2015.04.016).
- [11] M. Barbato, R. Grappe, M. Lacroix, and R. W. Calvo, "Polyhedral results and a branch-and-cut algorithm for the double traveling salesman problem with multiple stacks," *Discrete Optim.*, vol. 21, pp. 25–41, Aug. 2016, doi: [10.1016/j.disopt.2016.04.005](https://doi.org/10.1016/j.disopt.2016.04.005).
- [12] M. Barbato, R. Grappe, M. Lacroix, and W. C. Roberto, "A set covering approach for the double traveling salesman problem with multiple stacks," *Int. Symp. Combinat. Optim.*, Vietri sul Mare, Italy, May 2016, pp. 260–272, doi: [10.1007/978-3-319-45587-7_23](https://doi.org/10.1007/978-3-319-45587-7_23).
- [13] A. Felipe, M. T. Ortuno, and G. Tirado, "The double traveling salesman problem with multiple stacks: A variable neighborhood search approach," *Comput. Oper. Res.*, vol. 36, no. 11, pp. 2983–2993, Nov. 2009, doi: [10.1016/j.cor.2009.01.015](https://doi.org/10.1016/j.cor.2009.01.015).
- [14] A. Felipe, M. T. Ortuno, and G. Tirado, "New neighborhood structures for the double traveling salesman problem with multiple stacks," *Top*, vol. 17, no. 1, pp. 190–213, Mar. 2009, doi: [10.1007/s11750-009-0080-9](https://doi.org/10.1007/s11750-009-0080-9).
- [15] A. Felipe, M. T. Ortuno, and G. Tirado, "Using intermediate infeasible solutions to approach vehicle routing problems with precedence and loading constraints," *Eur. J. Oper. Res.*, vol. 211, no. 1, pp. 66–75, May 2011, doi: [10.1016/j.ejor.2010.11.011](https://doi.org/10.1016/j.ejor.2010.11.011).
- [16] U. E. F. D. Silveira, M. P. L. Benedito, and A. G. Santos, "Heuristic approaches to double vehicle routing problem with multiple stacks," in *Proc. Int. Conf. Intell. Syst. Design Appl.*, Marrakech, Morocco, Dec. 2016, pp. 231–236, doi: [10.1109/ISDA.2015.7489230](https://doi.org/10.1109/ISDA.2015.7489230).
- [17] J. B. C. Chagas, U. E. F. Silveira, M. P. L. Benedito, and A. G. Santos, "Simulated annealing metaheuristic for the double vehicle routing problem with multiple stacks," in *Proc. Int. Conf. Intell. Transp. Syst.*, Rio de Janeiro, Brazil, Nov. 2016, doi: [10.1109/ITSC.2016.7795726](https://doi.org/10.1109/ITSC.2016.7795726).
- [18] R. M. Lusby and J. Larsen, "Improved exact method for the double TSP with multiple stacks," *Networks*, vol. 58, no. 4, pp. 290–300, Dec. 2011, doi: [10.1002/net.20473](https://doi.org/10.1002/net.20473).
- [19] R. M. Lusby and J. Larsen, "An exact method for the double TSP with multiple stacks," *Int. Trans. Oper. Res.*, vol. 58, no. 4, pp. 290–300, Sep. 2011, doi: [10.1111/j.1475-3995.2009.00748.x](https://doi.org/10.1111/j.1475-3995.2009.00748.x).
- [20] M. Casazza, A. Ceselli, and M. Nunkesser, "Efficient algorithms for the double traveling salesman problem with multiple stacks," *Comput. Oper. Res.*, vol. 39, no. 5, pp. 1044–1053, May 2012, doi: [10.1016/j.cor.2011.06.008](https://doi.org/10.1016/j.cor.2011.06.008).
- [21] S. Urrutia, A. Milanés, and A. Løkketangen, "A dynamic programming based local search approach for the double traveling salesman problem with multiple stacks," *Int. Trans. Oper. Res.*, vol. 22, no. 1, pp. 61–75, Oct. 2015, doi: [10.1111/itor.12053](https://doi.org/10.1111/itor.12053).
- [22] F. Carrabs, R. Cerulli, and M. G. Speranza, "A branch-and-bound algorithm for the double travelling salesman problem with two stacks," *Networks*, vol. 61, no. 1, pp. 58–75, Jan. 2013, doi: [10.1002/net.21468](https://doi.org/10.1002/net.21468).
- [23] L. Hvattum, G. Tirado, and A. Felipe, "The double traveling salesman problem with multiple stacks and a choice of container types," *Mathematics*, vol. 6, no. 8, pp. 979–990, Jun. 2020, doi: [10.3390/math8060979](https://doi.org/10.3390/math8060979).
- [24] J. B. C. Chagas, U. E. F. Silveira, A. G. Santos, and M. J. F. Souza, "A variable neighborhood search heuristic algorithm for the double vehicle routing problem with multiple stacks," *Int. Trans. Oper. Res.*, vol. 27, no. 1, pp. 112–137, Jan. 2020, doi: [10.1111/itor.12623](https://doi.org/10.1111/itor.12623).
- [25] J. Chagas and A. G. Santos, "A branch-and-price algorithm for the double vehicle routing problem with multiple stacks and heterogeneous demand," in *Proc. 16th Int. Conf. Intell. Syst. Design Appl.*, Porto, Portugal, Feb. 2017, pp. 921–934, doi: [10.1007/978-3-319-53480-0_91](https://doi.org/10.1007/978-3-319-53480-0_91).
- [26] J. Chagas and A. G. Santos, "An effective heuristic algorithm for the double vehicle routing problem with multiple stack and heterogeneous demand," in *Proc. 17th Int. Conf. Intell. Syst. Design Appl.*, Delhi, India, Mar. 2018, pp. 785–796, doi: [10.1007/978-3-319-76348-4_75](https://doi.org/10.1007/978-3-319-76348-4_75).
- [27] M. Gendreau, M. Iori, G. Laporte, and S. Martello, "A tabu search algorithm for a routing and container loading problem," *Transp. Sci.*, vol. 40, no. 3, pp. 342–350, Aug. 2006, doi: [10.1287/trsc.1050.0145](https://doi.org/10.1287/trsc.1050.0145).
- [28] L. Wei, Z. Zhang, and A. Lim, "An adaptive variable neighborhood search for a heterogeneous fleet vehicle routing problem with three-dimensional loading constraints," *IEEE Comput. Intell. Mag.*, vol. 9, no. 4, pp. 18–30, Oct. 2014, doi: [10.1109/MCI.2014.2350933](https://doi.org/10.1109/MCI.2014.2350933).
- [29] L. Wei, Z. Zhang, and A. Lim, "An evolutionary local search for the capacitated vehicle routing problem minimizing fuel consumption under three-dimensional loading constraints," *Transp. Res. B, Methodol.*, vol. 82, pp. 20–35, Dec. 2015, doi: [10.1016/j.trb.2015.10.001](https://doi.org/10.1016/j.trb.2015.10.001).
- [30] B. Mahvash, A. Awasthi, and S. Chauhan, "A column generation based heuristic for the capacitated vehicle routing problem with three-dimensional loading constraints," *Int. J. Prod. Res.*, vol. 55, no. 6, pp. 1730–1747, Sep. 2017, doi: [10.1080/00207543.2016.1231940](https://doi.org/10.1080/00207543.2016.1231940).
- [31] H. Koch, A. Bortfeldt, and G. Wascher, "A hybrid algorithm for the vehicle routing problem with backhauls, time windows and three-dimensional loading constraints," *OR Spectr.*, vol. 40, no. 4, pp. 1029–1075, Feb. 2018, doi: [10.1007/s00291-018-0506-6](https://doi.org/10.1007/s00291-018-0506-6).

- [32] S. Reil, A. Bortfeldt, and L. Monch, "Heuristics for vehicle routing problems with backhauls, time windows, and 3D loading constraints," *Eur. J. Oper. Res.*, vol. 266, no. 3, pp. 877–894, Oct. 2017, doi: [10.1016/j.ejor.2017.10.029](https://doi.org/10.1016/j.ejor.2017.10.029).
- [33] D. Mannel and A. Bortfeldt, "A hybrid algorithm for the vehicle routing problem with pickup and delivery and three-dimensional loading constraints," *Eur. J. Oper. Res.*, vol. 254, no. 3, pp. 840–858, Apr. 2016, doi: [10.1016/j.ejor.2016.04.016](https://doi.org/10.1016/j.ejor.2016.04.016).
- [34] D. Mannel and A. Bortfeldt, "Solving the pickup and delivery problem with three-dimensional loading constraints and reloading ban," *Eur. J. Oper. Res.*, vol. 264, no. 1, pp. 119–137, May 2017, doi: [10.1016/j.ejor.2017.05.034](https://doi.org/10.1016/j.ejor.2017.05.034).
- [35] A. Bortfeldt and Y. Junmin, "The split delivery vehicle routing problem with three-dimensional loading constraints," *Eur. J. Oper. Res.*, vol. 282, no. 2, pp. 545–558, Apr. 2020, doi: [10.1016/j.ejor.2019.09.024](https://doi.org/10.1016/j.ejor.2019.09.024).
- [36] H. Koch, M. Schlgell, and A. Bortfeldt, "A hybrid algorithm for the vehicle routing problem with three-dimensional loading constraints and mixed backhauls," *J. Scheduling*, vol. 23, no. 4, pp. 71–93, Feb. 2020, doi: [10.1007/s10951-019-00625-7](https://doi.org/10.1007/s10951-019-00625-7).
- [37] A. Ayough, B. Khorshidvand, N. Massomnedjad, and A. Motameni, "An integrated approach for three-dimensional capacitated vehicle routing problem considering time windows," *J. Model. Manag.*, vol. 15, no. 3, pp. 1746–5672, Feb. 2020, doi: [10.1108/JM2-11-2018-0183](https://doi.org/10.1108/JM2-11-2018-0183).
- [38] D. E. Goldberg and K. A. Deb "A comparative analysis of selection schemes used in genetic algorithms," *Found. Genetic Algorithms*, vol. 1, pp. 69–93, Dec. 1991, doi: [10.1016/B978-0-08-050684-5.50008-2](https://doi.org/10.1016/B978-0-08-050684-5.50008-2).
- [39] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling-salesman problem," *Oper. Res.*, vol. 21, no. 2, pp. 498–516, Apr. 1973, doi: [10.1287/opre.21.2.498](https://doi.org/10.1287/opre.21.2.498).
- [40] Y. Wu, T. Weise, and R. Chiong, "Local search for the traveling salesman problem: A comparative study," in *Proc. IEEE 14th Int. Conf. Cognit. Informat. Cognit. Comput.*, Beijing, China, Jul. 2015, pp. 213–220, doi: [10.1109/ICCI-CC.2015.7259388](https://doi.org/10.1109/ICCI-CC.2015.7259388).
- [41] B. W. Kernighan and S. Lin, "An efficient heuristic procedure for partitioning graphs," *Bell Labs Tech. J.*, vol. 49, no. 2, pp. 291–307, Feb. 1970, doi: [10.1002/j.1538-7305.1970.tb01770.x](https://doi.org/10.1002/j.1538-7305.1970.tb01770.x).
- [42] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 66–651, Jun. 2010, doi: [10.1016/j.patrec.2009.09.011](https://doi.org/10.1016/j.patrec.2009.09.011).
- [43] J. Matousek, "On approximate geometric k-clustering," *Discrete Comput. Geometry*, vol. 24, no. 1, pp. 61–84, Dec. 2000, doi: [10.1007/s004540010019](https://doi.org/10.1007/s004540010019).



CHUNYUE SHEN received the B.E. degree in automation from the Huazhong University of Science and Technology, Wuhan, China, in 2017, where she is currently pursuing the M.E. degree in control science and engineering. Her research interests include intelligent task planning and scheduling, emergency task planning and scheduling, and the application and optimization of meta-heuristic algorithm in logistics field.



artificial intelligence, and emergency management.

JIANQIANG TANG received the B.E. degree in mathematics from Hefei University, Hefei, China, in 2017. He is currently pursuing the Ph.D. degree with the School of Management, Huazhong University of Science and Technology. From 2017 to 2019, he began to study in the School of Mathematics and Statistics, Huazhong University of Science and Technology, engaged in operations research and cybernetics. His research interests include logistics and supply chain optimization, artificial intelligence, and emergency management.



CHAO QI received the Ph.D. degree from Nanyang Technological University, Singapore, in 2006. She is currently a Professor with the School of Management, Huazhong University of Science and Technology. Her research interests include intelligent decision-making for management systems including production and logistics, social security, and emergency management.



interests include naval warship equipment logistic support, support resource optimization, and equipment support modeling and simulation.

MINZHI RUAN received the B.S. degree in warship equipment command from Naval Academy, Dalian, China, in 2006, the M.S. degree in electronic warfare from Naval academy, Dalian, in 2009, and the Ph.D. degree in weapon system and application engineering from the Naval University of Engineering, Wuhan, in 2012. He is currently a Postdoctoral Research with the College of Naval Architecture & Ocean Engineering, Naval University of Engineering. His current research



SHUANG QIU was born in Zoucheng, Shandong, China. She received the B.S. degree in logistics management from Huazhong University of Science and Technology, Wuhan, Hubei, in 2020. She is currently pursuing the M.S. and Ph.D. degrees in operations management with The Hong Kong University of Science and Technology, Hong Kong.

...