



# Dynamic feedback algorithm based on spatial corner fitness for solving the three-dimensional multiple bin-size bin packing problem

Yi Liu<sup>1</sup> · Xiaoyun Jiang<sup>1</sup>

Received: 25 July 2023 / Accepted: 30 January 2024  
© The Author(s) 2024

## Abstract

To improve cargo loading efficiency and achieve diverse needs of companies for the loading process, this paper innovatively establishes a multiple objective mixed integer programming model for the three-dimensional multiple bin-size bin packing problem (3D-MBSBPP). The model is designed to maximize container space utilization rate and cargo load balance, minimize container usage costs, and incorporates some practical constraints. On this basis, we propose a novel dynamic feedback algorithm based on spatial corner fitness (SCF\_DFA) to solve this model, which consists of three stages. Specifically, Stage 1 employs a heuristic algorithm based on spatial corner fitness to optimize the search of the remaining spaces. Stage 2 employs a container type selection algorithm to dynamically adjust and optimize container types. Stage 3 uses an improved genetic algorithm to improve the quality of the solutions of the first two stages. We demonstrate the effectiveness of the proposed algorithm through comparative experiments on benchmark instances, and apply it to solve the real-life instances for the 3D-MBSBPP. The results show that the proposed algorithm can make the average container space utilization rate reach 85.38%, which is 1.48% higher than that of baseline method, while the loading results obtained are more balanced, indicating the advantages of the SCF\_DFA in solving 3D-MBSBPP. Furthermore, we conduct ablation experiments to confirm the effectiveness of each component within the algorithm.

**Keywords** Three-dimensional multiple bin-size bin packing problem · Multiple objective mixed integer programming model · Spatial corner fitness assessment strategy · Dynamic feedback algorithm · Improved genetic algorithm

## Introduction

With the global development of digitization, technologization, and urbanization, online shopping has propelled the rapid growth of the logistics and transportation systems. For example, the global logistics market was valued at 103 billion USD in 2017, with a projected growth of 129 billion USD by 2023 at an annual rate of 3.5% [1]. However, amidst this opportunity and rapid progress, logistics and transportation systems face significant difficulties and challenges. Especially for the core field of logistics and transportation systems, namely the three-dimensional bin packing problem. The rising quantity of transported cargoes has raised the complexity and difficulty of cargo loading. According

to the Pitney Bowes Parcel Shipping Index, across 13 major markets with a total population of 3.7 billion, an average of 23 parcels were shipped per person, equivalent to 2760 parcels shipped every second. Additionally, projections indicate that the global volume of transported parcels is expected to hit 200 billion by 2025 [2]. On the other hand, with the increasing variety of vehicle types for cargo transport, the company faces difficulties in efficiently selecting the optimal combination of vehicle types for delivery. For example, according to a survey by Eurostat, one-fifth of freight vehicles in the EU operate while empty [3]. These circumstances not only exacerbate traffic congestion and reduce the efficiency of cargo delivery, but also lead to increase transportation costs for the company. Therefore, it is imperative to improve the methods and technologies used in the field of three-dimensional bin packing problem to adapt to the rapid development of current logistics and transportation systems. This will improve the efficiency and cost-effectiveness of the cargo loading process.

✉ Xiaoyun Jiang  
jxycom@163.com

<sup>1</sup> Present Address: School of Economics and Management, Xiamen University of Technology, Xiamen 361024, Fujian, China

Scholars typically employ operations research techniques to optimize the three-dimensional bin packing problem. This involves formulating a mathematical model for the three-dimensional bin packing problem and constructing the corresponding algorithms to solve the model. At present, research on the three-dimensional bin packing problem can be categorized into two subproblems: the Three-Dimensional Single Bin-Size Bin Packing Problem (3D-SBSBPP) and the Three-Dimensional Multiple Bin-Size Bin Packing Problem (3D-MBSBPP). Significant progress has been made in the research of the former [4]. Scholars are primarily focused on continuously refining the models and solution algorithms for this subproblem. For example, Martínez et al. [5] introduced two stability indicators to enhance the precise assessment of cargo stability during transportation: the number of fallen boxes and the number of boxes likely to be damaged in case of acceleration. These two indicators effectively ensure the security of cargoes in transit. Yang et al. [6] and Que et al. [7] introduced machine learning algorithms into the solution of 3D-SBSBPP.

However, with the rising quantity of transported cargoes and types of containers, the company prefers to choose the combination of container types that can reduce container usage costs and increase the rate of space utilization. Consequently, scholars have begun to focus their studies on the solution and optimization of the 3D-MBSBPP. The definition of the 3D-MBSBPP can be described as follows: given the information of a batch of cargoes, there is a group of limited container types with different sizes and costs, and therefore it is necessary to select several different types of containers, load all cargoes and meet the set goals, such as maximizing the container space utilization rate and minimizing the container usage cost. The difference between 3DSBSBPP and 3DMBSBPP is that the former uses only one type of container to load cargo. However, the latter chooses to use different types of containers to load cargo, ensuring that while making full use of the container space utilization rate, the container usage cost is reduced. The difference between 3D-MBSBPP and 3D-SBSBPP is shown in Fig. 1.

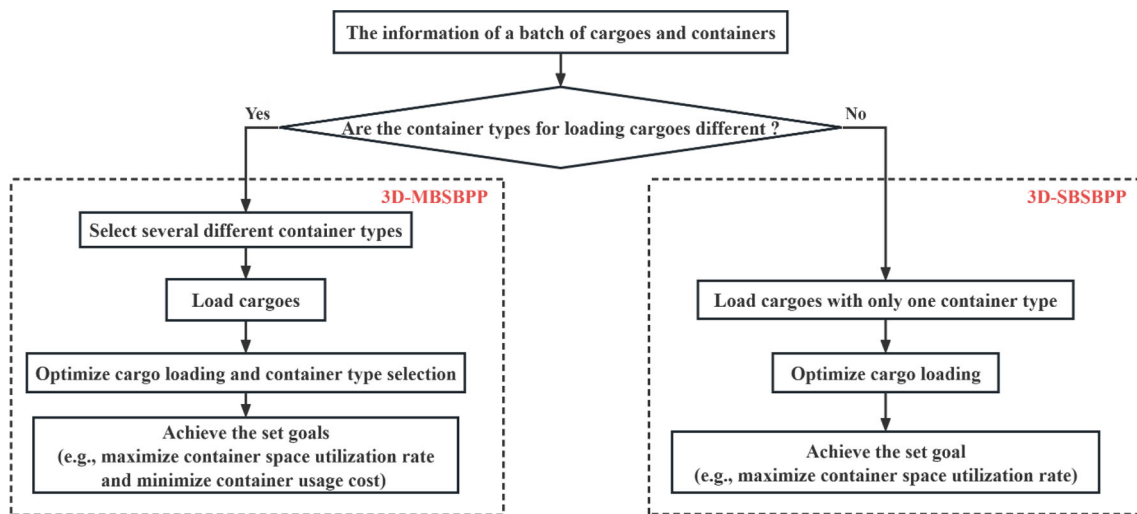
Currently, the study of mathematical models for the 3D-MBSBPP is primarily focused on minimizing the number or usage cost of containers used, incorporating practical constraints such as stability [8]. Hence the scholars lack extensive research on the construction of the multiple objective functions of 3D-MBSBPP. Besides, the selection of appropriate container types is crucial to ensure the efficient loading of cargo when solving the 3D-MBSBPP. However, there are fewer studies involving the selection of different container types. Existing studies have determined the type of container before the cargo is loaded [4], which may lead deficiencies in the initially designed container types as the loading of the cargo is completed. In conclusion, there are still the following

difficulties in the current research on the three-dimensional bin packing problem:

- (1) The three-dimensional bin packing problem itself is a non-deterministic polynomial hard (NP-hard) problem with high complexity.
- (2) The mathematical model of the 3D-MBSBPP has difficulties in meeting the diversified needs of the company. Since the existing mathematical model of the 3D-MBSBPP is mainly based on single objective function, it has difficulties in realizing the multiple objective requirements of the company.
- (3) The timely optimization of container types presents difficulties. The adjustments of container types in the existing study are determined before cargo loading, thus making it difficult to dynamically optimize container types based on feedback from the results.

These difficulties make the study of the 3D-MBSBPP complex and challenging. Generally heuristic algorithms are mostly used to solve the 3D-MBSBPP and its design is mainly divided into two stages: construction methods stage and improvement methods stage (we will introduce them in detail in “Literature reviews”). We notice that the current methods for these two stages present certain challenges: first, for the design of methods in the construction methods stage, the primary purpose is to find suitable placements for the cargoes, thereby effectively utilizing the remaining space within the containers. However, existing construction methods still lack sufficient strategies for reducing gaps between layers and enhancing the balance of upper-layer cargo placements. Second, for the design of methods in the improvement methods stage, the current methods are deficient in local search capability and are prone to premature convergence, leading to less pronounced optimization effects on the solutions.

Therefore, in this paper, we overcome all the above difficulties and challenges. For the 3D-MBSBPP, this paper establishes a mixed-integer programming mathematical model to maximize the container space utilization rate and cargo load balance and minimize the container usage cost, and incorporates some practical constraints in the model. Meanwhile, given the complexity of the 3D-MBSBPP under study, a dynamic feedback algorithm based on spatial corner fitness (SCF\_DFA) is proposed to solve the model. Our proposed algorithm consists of three components: the heuristic algorithm based on spatial corner fitness (SCF\_HA), the container type selection algorithm (CTSA), and the improved genetic algorithm (IGA). Specifically, during the cargo loading stage, the SCF\_HA is constructed to load the cargo. Its goals are to optimize cargo placement, reduce the gaps between the cargoes, and improve the space utilization rate. Based on the feedback of the results after loading the cargoes, we design a CTSA to dynamically adjust and optimize container types,



**Fig. 1** Difference between 3D-MBSBPP and 3D-SBSBPP

helping the company to reduce costs, improving the space utilization rate, and enhancing the balance of container loading. Finally, we employ an IGA to optimize the container loading results generated in the first two stages, aiming to improve the quality of solutions.

In the experimental analysis, we test our algorithm on the standard benchmark instances proposed by Bischoff and Ratcliff. Compared to several state-of-the-art algorithms, the experimental results demonstrate that our algorithm can increase the average container space utilization rate by 1.79%, which confirms the effectiveness of our algorithm. Based on this foundation, the algorithm proposed in this paper is applied to solve the real-life instance for the 3D-MBSBPP. We compare the proposed algorithm with the most widely used container type selection algorithm (as the baseline method). The results show that our proposed algorithm improves the container space utilization rate by 1.48% and the loading results obtained are more balanced, indicating the superiority of our algorithm in solving the 3D-MBSBPP. Besides, to further analyze the effectiveness of our proposed algorithm, an ablation experiment is conducted. The main purpose of this experiment is to comprehensively evaluate the role of each component in the algorithm and exhibit their contribution to the performance of the algorithm.

The remainder of this paper is structured as follows. In “[Literature reviews](#)”, related works about the three-dimensional bin packing problem are analyzed. In “[Construction of the 3D-MBSBPP model](#)”, the problem statement and modeling are explained. Then, in “[Dynamic feedback algorithm based on spatial corner fitness](#)”, a detailed description of the proposed dynamic feedback algorithm based on spatial corner fitness is presented. “[Computational experiments](#)” reports the computational experiments. Finally, “[Conclusion and future works](#)” concludes the paper and provides some future work.

## Literature reviews

The three-dimensional bin packing problem is characterized by high complexity and strong practical application value, which has been widely studied by scholars and numerous works in this field have been published [1, 9, 10]. The study of the three-dimensional bin packing problem can be broadly categorized into two main aspects: the construction of mathematical models and the design of solution methods. We conduct a review on the construction of mathematical models for the three-dimensional bin packing problem in “[Mathematical model for the three-dimensional bin packing problem](#)”. We review and categorize the methods for solving the three-dimensional bin packing problem in “[Solving methods](#)”, and finally we summarize the literature reviews on the three-dimensional bin packing problem in “[Summary of literature reviews](#)”.

## Mathematical model for the three-dimensional bin packing problem

The construction of mathematical models for the three-dimensional bin packing problem primarily includes formulating objective functions and constraints. Specifically, when formulating the mathematical model for the three-dimensional bin packing problem with a single objective function, most scholars adopt the maximization of space utilization rate as their objective function. For example, Eley [11] established an objective function dominated by the maximum volume utilization rate of the container and proposed an algorithm to form homogeneous blocks to improve the volume utilization rate of the container. Araya et al. [12] innovatively proposed a new evaluation function to rank loaded boxes to ensure that the packed boxes could maximize the

use of the remaining space. The results showed that the proposed method could solve the three-dimensional bin packing problem efficiently. Oliveira et al. [13] established the maximization of the space utilization rate as the objective function, and to increase the stability of cargo loading, integrated cutting planes into the traditional branch-and-bound method to better locate the optimal positions for cargo stability. These objective functions are set for single-container type, but as for the 3D-MBSBPP, most scholars primarily focus on minimizing the number of containers used or minimizing container usage costs as the single objective function. For example, Wei et al. [4] used the minimization of total transportation costs as the objective function of the 3D-MBSBPP and designed a new parameter to estimate the gross space utilization rate of containers. Che et al. [14] based on the previous study by Eley [11], constructed an objective function for the 3D-MBSBPP that focused on minimizing the container usage costs. However, with the increase in the number of cargoes and container types, the formulation of a single objective function cannot meet the needs of the company for multiple objectives. Scholars have begun to study the construction of a multiple objective mathematical model for the three-dimensional bin packing problem. For the formulation of mathematical models with multiple objective functions, extensive studies have been conducted for single-container types [15, 16]. However, for the 3D-MBSBPP, there have been relatively fewer studies involving multiple objective functions in the mathematical model. The main reasons are that the 3D-MBSBPP is an NP-hard problem and also requires solving multiple objective functions, which makes the complexity of the model grow exponentially. The research closest to our study was conducted by Kurpel et al. [17], where they established an upper-level model to minimize container usage volume and a lower-level model to maximize container space utilization rate. They employed exact algorithms to solve the models. However, our study differs from theirs in two aspects: (i) we introduce the maximization of cargo load balance as the third objective function in the model to enhance cargo loading safety; (ii) we use a heuristic algorithm to solve the constructed model.

Regarding the constraints on the three-dimensional bin packing problem are mainly included: weight limit constraints [4, 18, 19], volume constraints [20–22], orientation constraints [23–25], and overlap constraints [26–29]. Among them, the weight limit constraints refer mainly to the fact that the total weight of the cargo cannot exceed the maximum load of the container. Volume constraints mainly ensure that the volume of the cargo placed does not exceed the capacity of the entire container. Orientation constraints have strict limitations for some cargoes, such as wine bottles and other fragile cargoes. Overlap constraints indicate the placement of cargoes without overlapping. The above are the general constraints of the three-dimensional bin packing problem. To

simulate real-life packing processes more accurately, some scholars have introduced practical constraints into the mathematical models. For example, stability constraints refer to the requirement that upper-layer cargo receive sufficient support from lower-layer cargo during placement [13, 26]. Allocation constraints define those different types of cargoes cannot be placed in the same container, such as food and gasoline, they will affect each other's quality [30, 31]. Center of gravity constraints refer that the center of gravity of the container is within the safety interval [32, 33]. Complexity constraints refer to the complexity of cargoes placement [34, 35]. However, fewer studies have been conducted on the setting of practical constraints for 3D-MBSBPP, especially for the setting of some practical constraints jointly. This circumstance makes it more challenging to meet the various constraints set by the company for the loading process.

### Solving methods

The solution of the three-dimensional bin packing problem is mainly categorized into the exact algorithm and the heuristic algorithm. The latter can be widely used mainly for the following reasons. First of all, since the three-dimensional bin packing problem belongs to an NP-hard problem, the exact algorithm can only solve the small-scale instances. Secondly, when the mathematical model of the three-dimensional bin packing problem becomes complicated, the exact algorithm may fail to find a solution. Thirdly, logistics and supply chain managers require a short time to perform the cargo loading stage, while the exact algorithm needs a longer time to solve the problem. Therefore, most scholars use heuristic algorithms to solve the three-dimensional bin packing problem. By summarizing the current heuristic algorithms used to solve the three-dimensional bin packing problem, they can be primarily categorized into two stages: construction methods and improvement methods [36]. The primary purpose of construction methods is to load the cargoes into the containers, whereas improvement methods aim to enhance the quality of the solution generated in the previous stage.

For the design of construction methods, scholars mainly developed heuristic algorithms based on specific placement criteria. Among them, a classical construction method is the wall-building approach, first proposed by George et al. [37]. The core idea of this algorithm is that the available space within the container is formed by the walls created by the surfaces of the cargoes inside the container. Influenced by this concept, Pisinger et al. [38] introduced layer-building strategies. This strategy mainly involves filling the bottom layer of the container first before placing the cargoes in the second layer. However, when faced with an increasing number of cargoes, these methods could increase gaps within the container. Therefore, some scholars incorporated a pre-processing stage before employing layer-building strategies,

which involved grouping identical cargoes into blocks before loading [12, 39]. The second classic construction method is the maximal-space strategy, first proposed by Lai et al. [40]. Its primary purpose is to guide current cargo placement based on the cargo arrangement within the container. On this basis, Parreño et al. [41] extended it by adding a block heuristic algorithm to better obtain cargo placement information. The last classic construction method is the corner points, first proposed by Martello et al. [42]. These corner points represent the ideal positions for placing cargoes in the space. As the cargoes are loaded into the container, the positions of these corner points are updated. Expanding on the concept of corner points, Crainic et al. [43] introduced the concept of extremal points, which are generated by projecting cargo onto the inner walls of the container or adjacent cargo surfaces. In conclusion, the primary purpose of these heuristic algorithms based on specific placement criteria is to best use the remaining space inside the container. However, as the number and variety of cargoes increase, it may lead to more significant gaps between layers, consequently reducing the performance of these algorithms.

The design of improvement methods consists mainly of some intelligent algorithms. For example, local search [44], tabu search [45, 46], ant colony optimization [47, 48], and so forth. Iori et al. [49] compared the previously proposed tabu search, guided tabu search, and ant colony optimization, concluding that the ant colony optimization exhibited the most effective in improving the quality of the solution. Based on the foundation of interval graphs introduced by Fekete et al. [50], Trivella et al. [44] incorporated multiple local search stages to optimize solutions. Ramos and Silva et al. [51] used an improved genetic algorithm to optimize the solution in the improvement stage. Additionally, to enhance the stability of cargoes during transportation, they integrated a stability evaluation strategy into the fitness function. The same authors, Ramos and Silva et al. [3] improved the previously designed fitness function by multiplying the center of gravity of the container with the density of the box as the fitness function. Eventually, they drew an important conclusion: for stable transportation, it is advisable to position the heaviest cargoes closer to the rear axle of the vehicle. The greedy randomized adaptive search procedure (GRASP) was first proposed by Moura et al. [52]. This algorithm also generates the corresponding solution from the construction methods stage and then conducts a local search algorithm to explore the neighborhood of the current solution. If a superior solution is found, it replaces the previous one. Correcher et al. [21] incorporated a ruin-and-recreate strategy into the GRASP algorithm. This strategy involves removing previous containers and generating new ones to optimize the solution. In conclusion, the primary purpose of these algorithms in the improvement methods stage is to improve the quality of the solutions. However, the high complexity of the problem

and the poor quality of the initial solutions make these algorithms face the drawbacks such as poor local search ability and insignificant improvement effect on the quality of the solutions.

Because of various types of containers in 3D-MBSBPP, selecting the better combination of container types can improve the cost-effectiveness of the company. Scholars usually design a container type selection algorithm as a new stage to adjust and optimize the combination of container types. Currently, there are few studies on container type selection algorithms for the 3D-MBSBPP. Scholars primarily focus on loading cargoes into different containers with less emphasis on selecting different container types [53]. The studies closest to the container type selection algorithm we proposed were Li et al. [54], Piyachayawat et al. [55], and Han [56]. However, the container type selection algorithms they proposed had determined the type of container before loading the cargo. They were unable to dynamically adjust and optimize container types based on the results after loading the cargoes.

## Summary of literature reviews

In summary, some research results have been achieved with the three-dimensional bin packing problem, but they are limited to the single-container type. There are fewer studies on the 3D-MBSBPP and the main aspects for improvement in the study are as follows.

- (1) Mathematical model construction of the 3D-MBSBPP. The mathematical model of the 3D-MBSBPP cannot satisfy the requirements of the company with multiple objectives in the loading process, such as the space utilization rate, the usage cost, and the cargo load balance. Also, the model contains fewer practical constraints.
- (2) The solution methods of the 3D-MBSBPP. For the design of algorithms in the construction methods stage, they cannot effectively reduce the gaps in the remaining space. For the design of algorithms in the improvement methods stage, these algorithms are weak in local search. For the design of container type selection algorithms, these methods lack the ability to dynamically adjust and optimize container types based on the feedback from the loading results.

Therefore, considering practical constraints in the real-life loading process, this paper establishes a mixed integer programming mathematical model to maximize container space utilization rate and cargo load balance and minimize container usage cost. A SCF\_DFA is proposed to solve the model. In this algorithm, the SCF\_HA optimizes the search for the remaining space. At the same time, based on the characteristics of 3D-MBSBPP, a CTSA is proposed to

dynamically adjust the selection of container types. Finally, the IGA improves the quality of the solutions generated in the first two stages.

## Construction of the 3D-MBSBPP model

We describe the parts included in the study of 3D-MBSBPP in “Description of the problem”, and then make some assumptions and explain the variables that will appear in the model in “Variables and symbols description”. We introduce the rules for placing the cargoes and the pretreatment before packing in “Placement rules and preprocessing of cargo”, and finally formulate a multiple objective mixed-integer programming model for 3D-MBSBPP in “Model building”.

### Description of the problem

The 3D-MBSBPP studied in this paper is divided into four parts: establishing the mathematical model, determining the cargo placement state, selecting the container type, and improving the solutions. Establishing a mathematical model mainly involves the establishment of assumptions, constraints, and objective functions. The determination of the cargo placement state mainly involves finding the optimal orientation and position of cargo in the container to ensure complete use of the container’s space. For the selecting of container types, the single-container type only needed to calculate the number of containers that could be fully loaded, while 3D-MBSBPP not only needed to ensure that the cargoes were fully loaded, but also needed to optimize the container types to ensure the low cost of the combination of container types. Improving the quality of the solutions is mainly the optimization of the previously generated solutions.

### Variables and symbols description

Considering the complexity of 3D-MBSBPP, it is necessary to make some assumptions before establishing the mathematical model:

- (1) The cargoes to be loaded are cuboids with a uniform texture and the barycenter coincides with its geometric center;
- (2) The extrusion of external force between cargoes is negligible;
- (3) Cargo is not allowed to be placed in suspension or an oblique position;
- (4) Cargo is not prone to deformation due to squeezing;
- (5) The length of any single cargo should not exceed the length of the container.

Table 1 shows the symbols used in the mathematical model of the 3D-MBSBPP established in this paper.

### Placement rules and preprocessing of cargo

- (1) The orientation of cargo. Under normal circumstances, based on the relevant relationship between the cargo side and the container side, there are six placement methods, as shown in Fig. 2. If some cargoes have special requirements (such as not being allowed to be placed backward), determine the placement orientation according to the relevant regulations.
- (2) The placement sequence of the cargo. At present, the settings of the sequence of cargo placement mainly include descending sequence of volume size, sequence of arrival of cargo, random arrangement, and so forth. Considering that large-volume cargoes occupy more surplus space and to avoid excessive gaps caused by large volume differences between cargoes placed, this paper adopts the descending sequence of volume size for the cargoes placed.
- (3) Pretreatment of cargo. To reduce the generation of gaps in the remaining space, this paper first forms homogeneous blocks of the same type of cargo before loading. The combination of homogeneous blocks has mainly three forms, as shown in Fig. 3. Cargoes of the same type are closely arranged in a matrix. At the same time, the volume of homogeneous blocks increases with increasing construction dimension. To avoid the large volume and verbosity of the homogeneous block, which causes inconvenience of loading, method (a) in Fig. 3 is used to construct the homogeneous block.

### Model building

With the objective functions of maximizing the container space utilization rate (Eqs. 1 and 2), minimizing the container usage cost (Eq. 3), and maximizing the container load balance (Eq. 4), and combined with the constraints in the actual loading process, the corresponding mathematical model of the 3D-MBSBPP is established. The specific mathematical expression is as follows:

$$\max \sum_{k=1}^c \sum_{j=1}^m \frac{\sum_{i=1}^n v_i \alpha_i^{kj}}{V_{kj}}, \quad (1)$$

$$\max \sum_{k=1}^c \sum_{j=1}^m \frac{\sum_{i=1}^n g_i \alpha_i^{kj}}{G_{kj}}, \quad (2)$$

$$\min \sum_{k=1}^c C_k \times N_k, \quad (3)$$

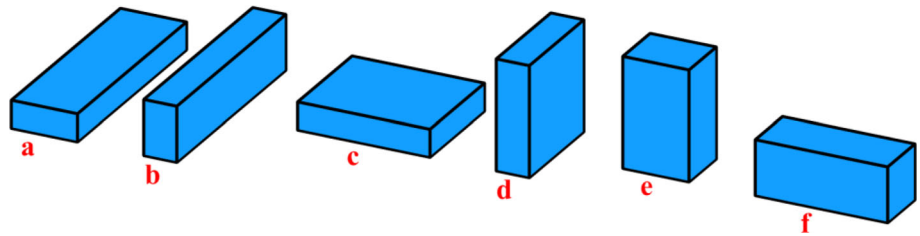
**Table 1** Symbols and names

Type	Symbols	Description
Collection	$U_u$	Collection of cargoes at the second layer and above, $U_u = \{1, 2, 3, \dots r\}$
	$U_p$	Collection of cargoes at the first layer, $U_p = \{1, 2, 3, \dots f\}$
	$U_i$	Collection of all cargoes, $U_i = \{1, 2, 3, \dots n\}$
	$N_s$	Number of cargoes in the $U_u$ collection
	$N_k$	Number of containers used for the $k$ th type
	$(x_p, y_p)$	Length and width of the cargo $p$ in contact with cargo $u$
	$(x_u, y_u)$	Length and width of the $u$ th cargo in the $U_u$ set
	$(l_i, w_i, h_i)$	Length, width, and height of the cargo $i$
	$(L_k, W_k, H_k)$	Length, width, and height of the $k$ th container type
	$(x_i, y_i, z_i)$	Coordinates of the point where cargo $i$ is placed (bottom left rear point, spatial corner)
Parameters	$(x'_i, y'_i, z'_i)$	Coordinates of cargo $i$ after placement (bottom right front point)
	$(X_i, Y_i, Z_i)$	Coordinates of the center of gravity of cargo $i$
	$[ValX_1^k, ValX_2^k]$	Safe offset interval of the lateral center of gravity for the $k$ th container type
	$[ValY_1^k, ValY_2^k]$	Safe offset interval of the longitudinal center of gravity for the $k$ th container type
	$[0, ValZ_2^k]$	Safe offset interval of the vertical center of gravity for the $k$ th container type
	$g_i$	Weight of cargo $i$
	$v_i$	Volume of cargo $i$
	$G_{kj}$	The load of the $j$ th container of the $k$ th type
	$V_{kj}$	The volume of the $j$ th container of the $k$ th type
	$C_k$	The cost of using the $k$ th type of container
Decision making variables	$c$	The total number of container types
	$m$	The total number of containers of the $k$ th type
	$\alpha_i^{kj}$	$\begin{cases} 1, & \text{Indicate that cargo } i \text{ is loaded by the } j \text{th container of the } k \text{th type} \\ 0, & \text{Others} \end{cases}$
	$S_u$	$\begin{cases} 1, & S_u \geq \gamma, \\ 0, & S_u < \gamma, \end{cases}$ Contact area of two adjacent cargoes
	$\beta_1$	$\begin{cases} 1, & x'_1 - x_2 < 0 \text{ Indicate that the two cargoes are not in contact} \\ 0, & x'_1 - x_2 \geq 0 \text{ Indicate that the two cargoes are possibly in contact} \end{cases}$
	$\beta_2$	when $x'_1 - x_2 \geq 0$ , $\begin{cases} 1, & y'_1 - y_2 < 0 \text{ Indicate that the two cargoes are not in contact} \\ 0, & y'_1 - y_2 \geq 0 \text{ Indicate that the two cargoes are possibly in contact} \end{cases}$
	$\beta_3$	when $x'_1 - x_2 \geq 0, y'_1 - y_2 \geq 0$ , $\begin{cases} 1, & z'_1 - z_2 < 0 \text{ Indicate that the two cargoes are not in contact} \\ 0, & z'_1 - z_2 \geq 0 \text{ Indicate that the two cargoes are possibly in contact} \end{cases}$
	$(l_i^x, l_i^y, l_i^z)$	Indicate whether the length edge of cargo $i$ is parallel to X-axis, Y-axis, or Z-axis. For example, $l_i^x$ $\begin{cases} 1, & \text{Indicate that the length edge of cargo } i \text{ is parallel to X -axis} \\ 0, & \text{Otherwise} \end{cases}$

**Table 1** (continued)

Type	Symbols	Description
	$(w_i^x, w_i^y, w_i^z)$	Indicate whether the width edge of cargo $i$ is parallel to X-axis, Y-axis, or Z-axis. For example, $w_i^x \begin{cases} 1, & \text{Indicate that the width edge of cargo } i \text{ is parallel to X-axis} \\ 0, & \text{Otherwise} \end{cases}$
	$(h_i^x, h_i^y, h_i^z)$	Indicate whether the height edge of cargo $i$ is parallel to X-axis, Y-axis, or Z-axis. For example, $h_i^x \begin{cases} 1, & \text{Indicate that the height edge of cargo } i \text{ is parallel to X-axis} \\ 0, & \text{Otherwise} \end{cases}$

**Fig. 2** Orientation of cargo



$$\max \frac{\sum_{u=1}^r S_u}{N_s}, \tag{4}$$

$$l_i^x + l_i^y + l_i^z = 1, \tag{11}$$

$$\text{s.t. } \sum_{k=1}^c \sum_{j=1}^m \alpha_i^{kj} = 1, \tag{5}$$

$$w_i^x + w_i^y + w_i^z = 1, \tag{12}$$

$$h_i^x + h_i^y + h_i^z = 1, \tag{13}$$

$$\sum_{k=1}^c \sum_{j=1}^m \sum_{i=1}^n g_i \alpha_i^{kj} \leq G_{kj}, \tag{6}$$

$$l_i^x + w_i^x + h_i^x = 1, \tag{14}$$

$$\sum_{k=1}^c \sum_{j=1}^m \sum_{i=1}^n v_i \alpha_i^{kj} \leq V_{kj}, \tag{7}$$

$$l_i^y + w_i^y + h_i^y = 1, \tag{15}$$

$$S_u = \frac{x_u y_u}{\sum_{p=1}^f x_p y_p}, \tag{8}$$

$$l_i^z + w_i^z + h_i^z = 1, \tag{16}$$

$$x'_i \times \alpha_i^{kj} \leq L_k \quad y'_i \times \alpha_i^{kj} \leq W_k \quad z'_i \times \alpha_i^{kj} \leq H_k, \tag{9}$$

$$\beta_1 + \beta_2 + \beta_3 = 1. \tag{17}$$

$$\left\{ \begin{aligned} ValX_1^k &\leq \frac{\sum_{k=1}^c \sum_{j=1}^m \sum_{i=1}^n X_i g_i \alpha_i^{kj}}{\sum_{k=1}^c \sum_{j=1}^m \sum_{i=1}^n g_i \alpha_i^{kj}} \leq ValX_2^k \\ ValY_1^k &\leq \frac{\sum_{k=1}^c \sum_{j=1}^m \sum_{i=1}^n Y_i g_i \alpha_i^{kj}}{\sum_{k=1}^c \sum_{j=1}^m \sum_{i=1}^n g_i \alpha_i^{kj}} \leq ValY_2^k, \\ 0 &\leq \frac{\sum_{k=1}^c \sum_{j=1}^m \sum_{i=1}^n Z_i g_i \alpha_i^{kj}}{\sum_{k=1}^c \sum_{j=1}^m \sum_{i=1}^n g_i \alpha_i^{kj}} \leq ValZ_2^k \end{aligned} \right. \tag{10}$$

Among them: Equation 5 indicates that cargo  $i$  can only be loaded into one of the containers. Equations 6 and 7 indicate that the total weight and volume of cargoes loaded in each container should not exceed the limit weight and volume of the container. Equation 8 indicates whether the total area of contact between the upper cargo and lower cargoes is greater than the set threshold  $\gamma$ . Equation 9 indicates that all the cargoes loaded in the container should not exceed the length, width, and height limit of the container. Equation 10 indicates that the center of gravity of the container after it is fully loaded should be within the specified safety range; Eqs. 11–16 indicate that the cargoes can be placed in six orientations. Equation 17 indicates that no overlap is allowed between cargoes.

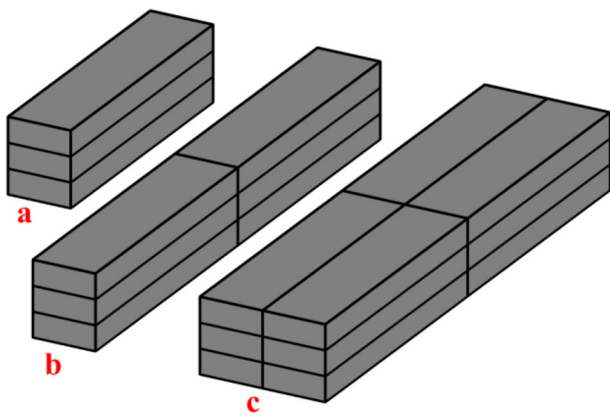


Fig. 3 Three combinations of homogeneous blocks

### Dynamic feedback algorithm based on spatial corner fitness

As for 3D-MBSBPP, the selection of different cargo placement positions will significantly influence the subsequent loading of cargoes and the space utilization rate within the container. Moreover, when facing multiple types of containers, the company prefers to choose the cost-effective container type combinations for transporting cargoes. For these reasons, we propose a dynamic feedback algorithm based on spatial corner fitness (SCF\_DFA) to solve the 3D-MBSBPP. The proposed algorithm consists of three components: the heuristic algorithm based on spatial corner fitness (SCF\_HA), the container type selection algorithm (CTSA), and the improved genetic algorithm (IGA). Among them, the SCF\_HA aims to find a better placement for the cargoes, the CTSA aims to optimize the container types and reduce their usage costs, and the IGA aims to improve the quality of the solutions. The flowchart of the proposed algorithm in this paper is shown in Fig. 4 and the symbolic definitions of the relevant variables are shown in Table 2.

As shown in Fig. 4, the SCF\_HA is employed to load the cargoes into containers in Stage 1 and output the solutions  $V$  with fully loaded cargoes. Subsequently, based on the feedback of solutions  $V$ , a decision is made regarding whether to proceed to the second stage. Stage 2 uses the CTSA to adjust the container types within solutions  $V$  and output the adjusted solutions  $V'$ . Stage 3 adopts the IGA to optimize the solutions  $V'$  generated through the first two stages or the solutions  $V$  generated in the first stage, thereby enhancing the quality of the solutions. The optimized solutions  $V''$  is judged to determine whether further adjustment and optimization of the container types are needed. If not needed, the final solutions  $V_{best}$  is output directly. If needed, then  $V''$  is backtracked to the second stage to adjust the container types and output the final solutions  $V_{best}$ . We introduce the

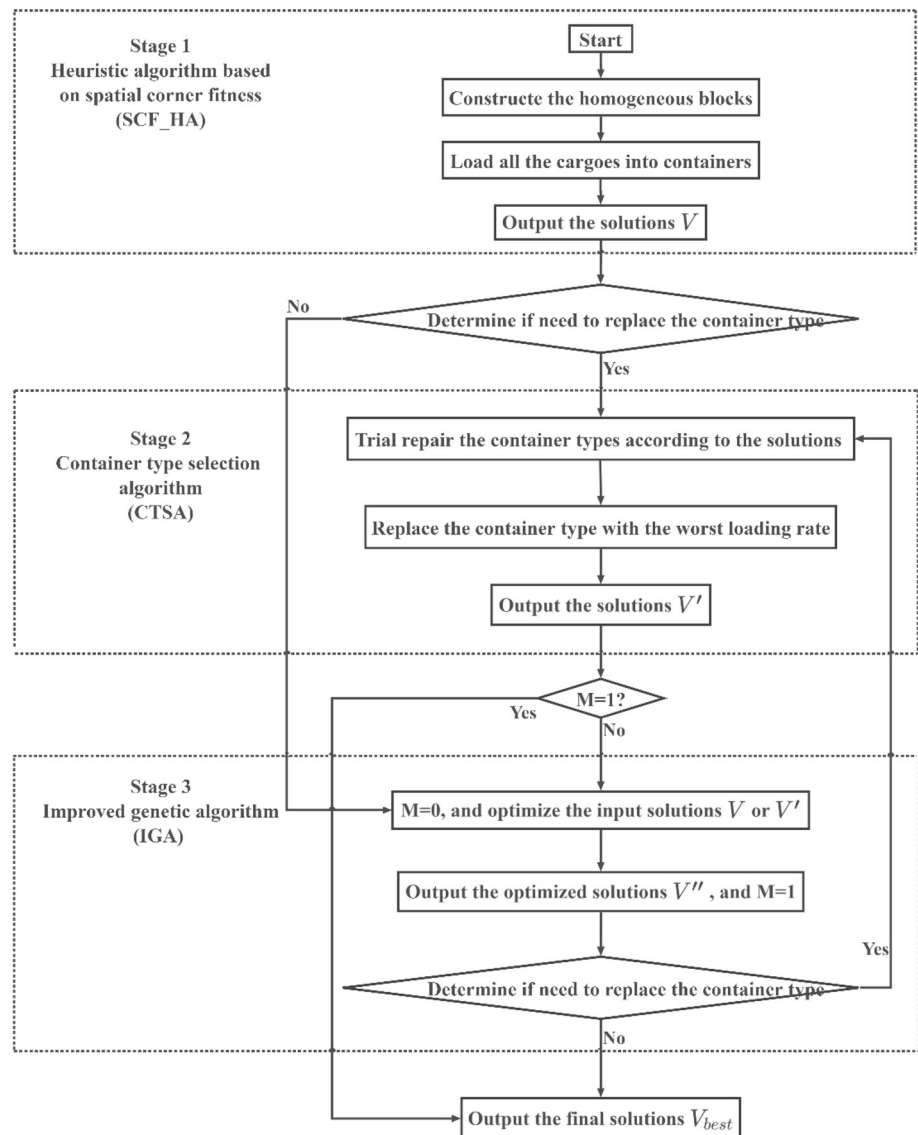
SCF\_HA and explain how it finds the optimal placement of cargoes in “[Heuristic algorithm based on spatial corner fitness](#)”, describe the CTSA in “[Container type selection algorithm](#)”, and specifically explain how the IGA optimizes the solutions in “[Improved genetic algorithm](#)”.

### Heuristic algorithm based on spatial corner fitness

From the introduction of construction methods in “[Solving methods](#)”, it is evident that these heuristic algorithms based on specific placement criteria primarily conduct a search for placement positions within the residual space defined by the shapes of the cargoes. However, as the types and shapes of cargoes increase, the height of the remaining space is diversified. In such cases, these algorithms may struggle to effectively utilize the remaining space, consequently increasing the gaps between layers and wasting the remaining space. Besides, these construction methods lack consideration of the reasonableness of cargo placement positions. For example, when placing cargo along the Z-axis, these algorithms may fail to select positions that provide sufficient support for the center of gravity of the cargo.

To overcome the weaknesses of the existing algorithms, we propose a heuristic algorithm based on spatial corner fitness (SCF\_HA) to reduce the gaps in the remaining space and ensure the balance of the upper cargo. Specifically, this algorithm divides the cargo load process into four different directions. The spatial corners in each direction mainly refer to the left bottom point in the remaining space. In the first two directions, the cargo is placed close to the inner wall of the container. In the latter two directions, the current cargo is placed in advance at each spatial corner of the remaining space, while the fitness values of these spatial corners are calculated based on different assessment operators designed for each direction. For example, when loading cargoes in the X–Y-axis direction, the assessment operators mainly calculate the fitness value of the spatial corners from two aspects: the height difference between the adjacent cargoes and the contact area between the adjacent cargoes. These two assessment operators help the cargo find the most suitable remaining space, and reduce the gaps between layers. When loading in the Z-axis direction, the assessment operators mainly calculate whether the center of gravity of the cargo is sufficiently supported and the contact area with the lower-layer cargoes. These two assessment operators ensure the balance of the upper-layer cargoes. The higher the fitness value of the spatial corner, the stronger the fit degree between cargoes with the remaining space and finally the spatial corner with the highest fitness is the best position to place the current cargo. The specific steps of the algorithm are as follows.

- (1) X-axis direction loading

**Fig. 4** Flowchart of the proposed SCF\_DFA

Loading in the first direction is carried out in the positive direction of the X-axis according to the sequence of cargoes placed in  $U_d$ . The initial spatial corner is the bottom left point of the container, the origin  $S_0 = (0, 0, 0)$ . The loading constraints Eqs. (5) to (17) should be satisfied by the cargoes in the loading process; if the cargo  $i$  do not satisfy the container's loading constraints, it will be added to  $U_n$  and traverse the next cargo  $i + 1$ . The position selected for loading each time is the corner where the remaining space falls on the X-axis, as shown in Fig. 5, that is,  $S_i^X = (l_{i-1}, 0, 0) + (l_i, 0, 0)$ . Until  $U_d = \emptyset$  is satisfied, discontinue loading in the X-axis direction, update  $U_a$ , and assign the value  $U_d = U_n$ . Finally, initialize  $U_n = \emptyset$  and begin loading in the next direction.

## (2) Y-axis direction loading

The loading process in the Y-axis direction is carried out in the positive direction of the Y-axis according to the sequence of the cargoes in  $U_d$ . The initial spatial corner is  $S_1^Y = (0, w_i, 0)$ , as shown in Fig. 6. Each time load the cargo at the location of the spatial corner where the remaining space falls on the Y-axis. The loading process is performed in the same manner as in (1). Until  $U_d = \emptyset$  is satisfied, stop loading the Y-axis direction, update  $U_a$ , and assign the value  $U_d = U_n$ . Finally, initialize  $U_n = \emptyset$  and begin loading in the next direction.

## (3) X–Y-axis direction loading

The goal of loading in the X–Y axis direction is to enable the cargo to cover the entire bottom surface while reducing the excessive gap generated when loading in the Z-axis direction and avoiding the waste of remaining space. Therefore, based on the above objectives, we design two spatial corner fitness assessment operators

**Table 2** Symbolic definition of related variables

Types	Variables	Definitions
Collection	$U_a$	Collection of loaded cargoes
	$U_d$	Collection of cargoes to be loaded
	$U_n$	Collection of unloadable cargoes
Remaining space	Among them $U_a \cup U_d \cup U_n = U_i$	
	$P$	Remaining placeable space
	$S_i$	Cargo $i$ selects a spatial corner in $P$
	$S_{best}$	The highest fitness value spatial corner in $P$
Container loading solution	$I$	Loading solutions for all containers
	$I_{best}(I_{worst})$	Loading scheme for containers with a maximum (minimum) space utilization rate in $I$
	$I_l$	Loading solutions for container types except the $I_{best}$
	$I_{new}$	New loading scheme after changing the container type
	$L_{d_x}^X, H_{d_x}^X$	Length and height of contact between the current cargo and the adjacent cargo $d_x$ in the orientation parallel to the X-axis
Adjacent cargo Information	$L_{d_y}^Y, H_{d_y}^Y$	Length and height of contact between the current cargo and the adjacent cargo $d_y$ in the orientation parallel to the Y-axis
	$S_v^Z$	The contact area of the upper cargo and their adjacent lower cargoes
Others	$b$	Current loaded container type
	$i$	Current loaded cargo

in the direction of the X–Y axis, which are  $\delta_1^{XY}$  adjacent cargoes contact area assessment operator and  $\delta_2^{XY}$  adjacent cargoes height difference assessment operator. The specific calculation formula is shown in Eqs. 18–20.

$$\delta_1^{XY} = \frac{1}{n} \times \left( \frac{\sum_{d_x=1}^n L_{d_x}^X \times H_{d_x}^X}{l_i \times h_i} \right) + \frac{1}{m} \times \left( \frac{\sum_{d_y=1}^m L_{d_y}^Y \times H_{d_y}^Y}{w_i \times h_i} \right), \tag{18}$$

$$\delta_2^{XY} = \left[ \frac{1}{n} \times \sum_{d_x=1}^n \frac{\min(H_{d_x}^X, h_i)}{\max(H_{d_x}^X, h_i)} \right] + \left[ \frac{1}{m} \times \sum_{d_y=1}^m \frac{\min(H_{d_y}^Y, h_i)}{\max(H_{d_y}^Y, h_i)} \right], \tag{19}$$

$$\text{Score}_p^{XY} = \sigma_1^{XY} \delta_1^{XY} + \sigma_2^{XY} \delta_2^{XY}. \tag{20}$$

Put the current cargo  $i$  into each spatial corner, calculate the fitness value of cargo  $i$  in each spatial corner (the calculation formula is shown in Eq. 20, where  $\sigma_1^{XY}$ ,  $\sigma_2^{XY}$  represent the weight of each assessment operator), select the corner with the highest fitness value as  $S_{best}$  and put the current cargo into it, as shown in Fig. 7. If the cargo cannot be loaded without satisfying the constraints, it will be added to the  $U_n$  set. Until  $U_d = \emptyset$  is satisfied, discontinue loading in the X–Y axis direction, update  $U_a$ , and assign the value

$U_d = U_n$ . Finally, initialize  $U_n = \emptyset$  and begin loading in the next direction.

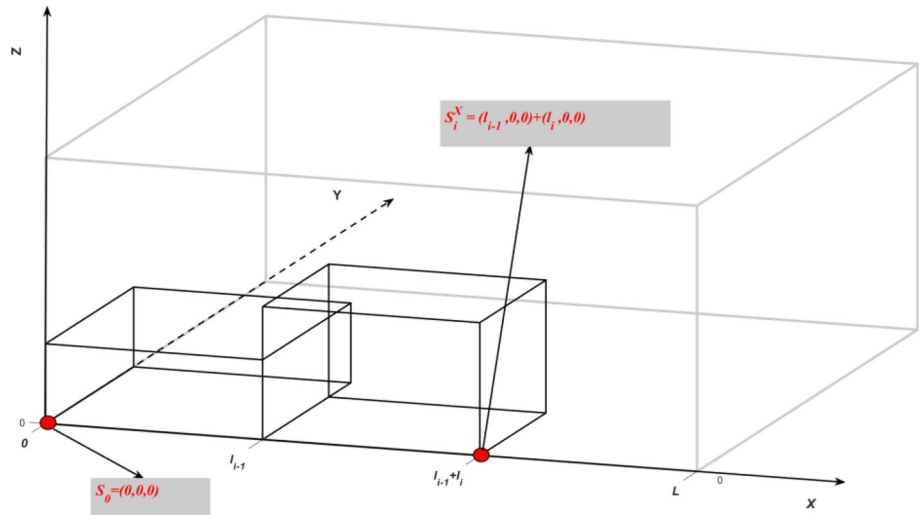
(4) Z-axis direction loading

When loading in the Z-axis direction, it is necessary to ensure that the center of gravity of the cargo is fully supported and that the cargoes between the layers have sufficient contact area to avoid the suspension of the cargo and ensure the stability of the transportation process. On this basis, two spatial corner fitness assessment operators about the Z-axis direction are designed, which are  $\delta_1^Z$  cargoes contact area assessment operator and  $\delta_2^Z$  current cargo center of gravity assessment operator. The calculation formula of  $\delta_1^Z$  is shown in Eq. 21.

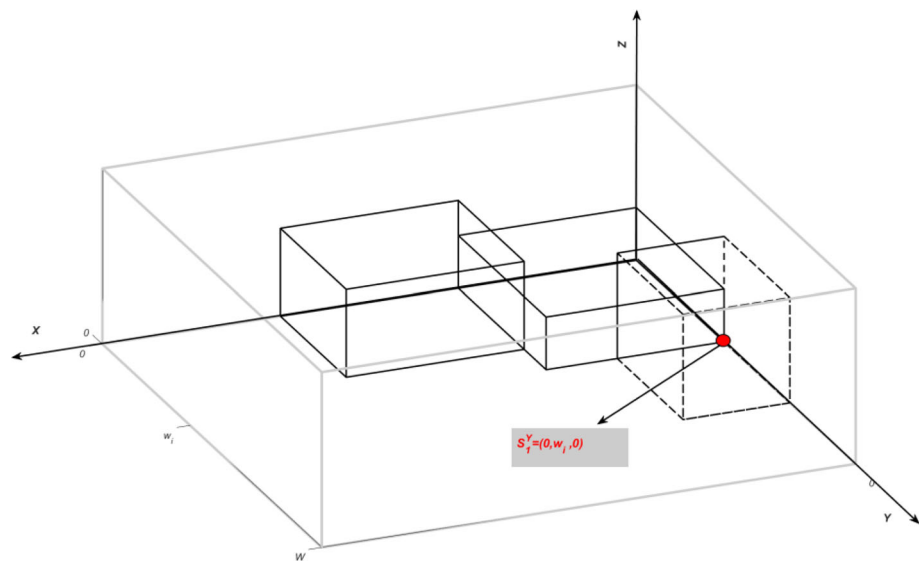
$$\delta_1^Z = \frac{\sum_{v=1}^l S_v^Z}{l_i \times w_i}. \tag{21}$$

The position of the center of gravity of the upper cargo  $n$  may have the following three cases, as shown in Fig. 8. The first case indicates that the center of gravity of cargo  $n$  can be fully supported by the lower cargo  $m$ . The second case indicates that the center of gravity of cargo  $n$  falls on the edge of cargo  $m$  and can be partially supported. The third case indicates that the center of gravity of cargo  $n$  is not supported at all by cargo  $m$ . Therefore,  $\delta_2^Z$  and the fitness value of the spatial corners in the direction of the Z-axis is calculated using Eqs. 22–23.

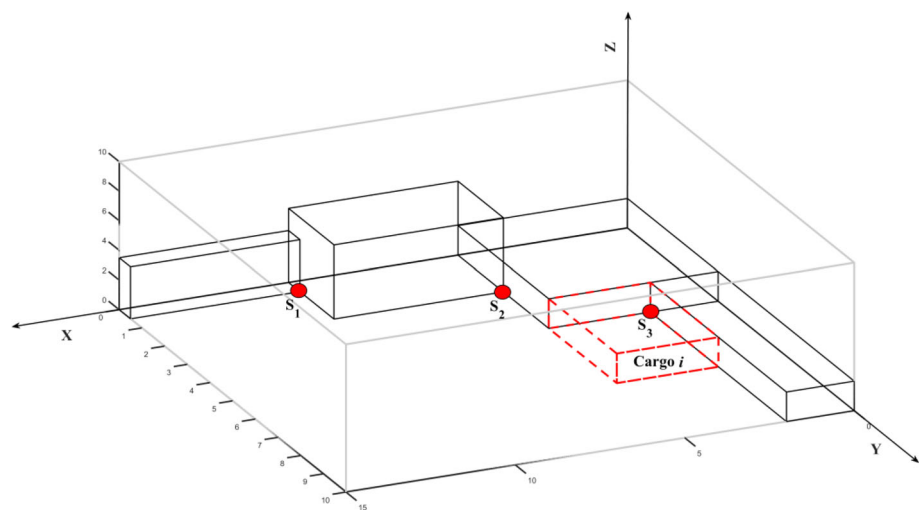
**Fig. 5** Spatial corner in the X-axis direction



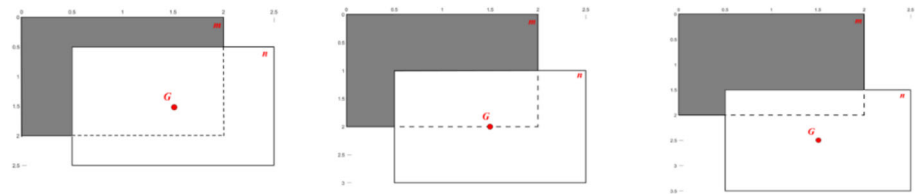
**Fig. 6** Spatial corner in the Y-axis direction



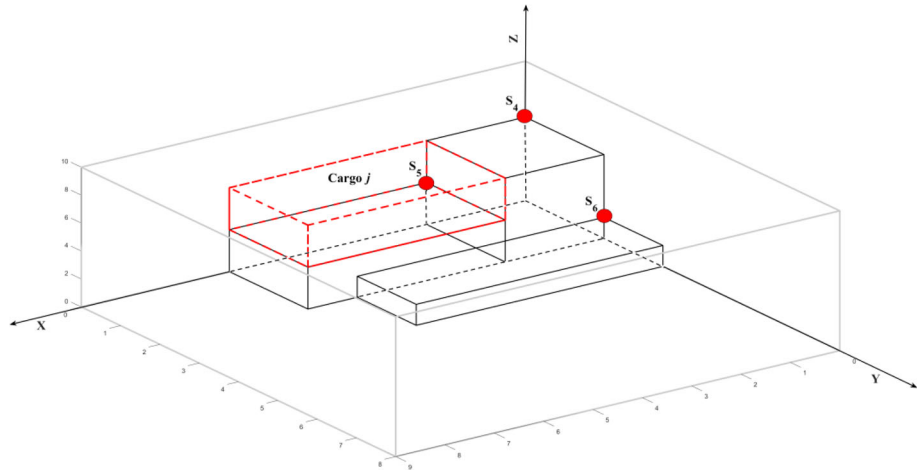
**Fig. 7** Cargo *i* placed at the spatial corner  $S_3$  in the X–Y-axis direction



**Fig. 8** Distribution of the center of gravity of the upper cargo



**Fig. 9** Cargo  $j$  placed at the spatial corner  $S_5$  in the Z-axis direction



$$\delta_2^Z = \begin{cases} \theta_1, & \text{The center of gravity of the upper cargo is fully supported} \\ \theta_2, & \text{The center of gravity of the upper cargo is partially supported,} \\ \theta_3, & \text{The center of gravity of the upper cargo is not supported} \end{cases} \quad (22)$$

$$\text{Score}_p^Z = \sigma_1^Z \delta_1^Z + \sigma_2^Z \delta_2^Z. \quad (23)$$

Put the current cargo  $j$  in each spatial corner in turn, calculate the fitness value of each spatial corner (the calculation formula is shown in Eq. 23, where  $\sigma_1^Z$  and  $\sigma_2^Z$  represent the weight of each assessment operator), select the corner with the highest fitness value as  $S_{best}$  and put the current cargo into it, as shown in Fig. 9. If the cargo cannot be loaded without satisfying the constraints, it will be added to the  $U_n$  set. Until  $U_d = \emptyset$  is satisfied, discontinue loading the Z-axis direction, update  $U_a$ , and assign the value  $U_d = U_n$ . Finally, initialize  $U_n = \emptyset$  and end loading in the current container. If  $U_a = U_i$ , the whole loading process is terminated. If  $U_a \neq U_i$ , open the next container to continue loading.

In summary, the flowchart of the SCF\_HA is shown in Fig. 10.

### Container type selection algorithm

Through summarizing and analyzing the research on the container type selection algorithm in “Solving methods”, we find that some scholars optimize the container types before loading cargoes. However, as the cargo loading is completed,

the container type selection algorithms they proposed cannot dynamically adjust the container type based on the loading results, leading to a lack of timely optimization for container types. Meanwhile, in Stage 1, the SCF\_HA primarily focuses on placing the cargoes into the container, without exploring and optimizing the combination of different types of containers.

Therefore, we propose a container type selection algorithm (CTSA) to optimize container types. Importantly, this algorithm can dynamically adjust and optimize the container types based on the feedback of the results after loading the cargoes, overcoming the weaknesses of the previous methods. Within this algorithm, based on the feedback from the solutions  $V$  generated in Stage 1, it is determined whether need to adjust the container types that already have been loaded. If no adjustment is required, the algorithm directly proceeds to Stage 3, which uses the IGA to optimize solutions  $V$ . If adjustments and optimizations are needed, our proposed CTSA is used to trial repair the container types and decide whether to replace the type of containers with the worst loading rate. The specific algorithm flowchart is shown in Fig. 11, and the detailed algorithm steps are as follows,

Fig. 10 Flowchart of SCF\_HA

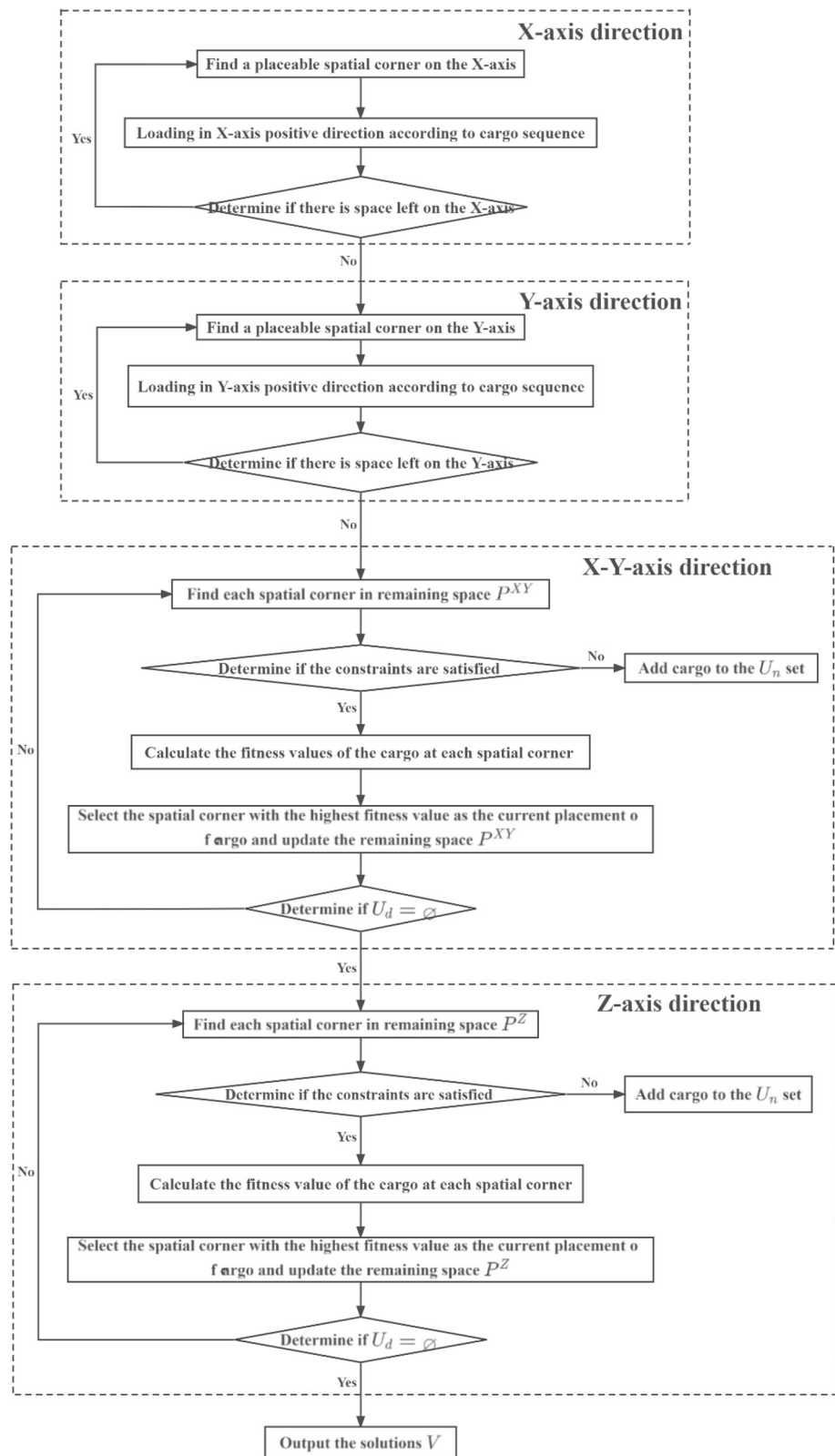
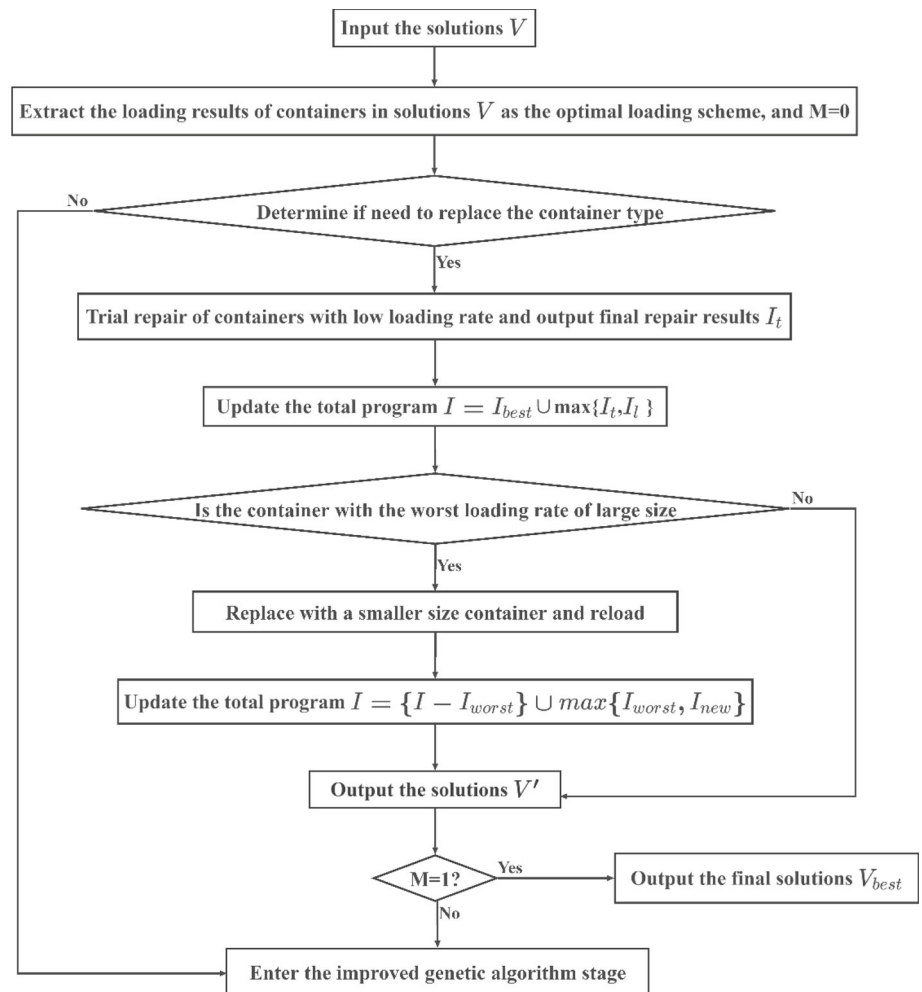


Fig. 11 Flowchart of CTSA



where Step 1 to Step 6 denote the loading of cargoes in Stage 1.

**Step 1** Enter the information of the cargo. Based on the cargo information, construct the homogeneous blocks, and estimate the number of containers ( $B_N$ ). Initialize sets  $U_n$ ,  $P$ , and  $U_a$ , and assign the value of  $U_i$  to  $U_d$ .

**Step 2** Generate the container type sequence for loading cargo. In this paper, the container sequence is generated in two ways. The first way is alternately generated according to the volume of the descending sequence of the container space. In the second way, the container sequence is randomly generated to improve the diversity of solutions and avoid falling into local optimal solutions.

**Step 3** Load the cargo according to the sequence of cargoes and the type of container. Sequentially iterate over the cargoes in set  $U_d$  and determine whether cargo  $i$  satisfies the constraints in  $b$ . If satisfied, search for remaining space in the container. If not, add it to the set that cannot load, update the set  $U_n = U_n + \{i\}$ , and start loading the next cargo  $i = i + 1$ .

**Step 4** Search for available space. If  $P = \emptyset$  or  $U_d = \emptyset \cap U_n \neq \emptyset$ , proceed to Step 5. If  $P \neq \emptyset$  and  $U_d \neq \emptyset$ , proceed to Step 6 to search for  $P$ .

**Step 5** Open the next container for loading according to the generated container type sequence in Step 2, initialize  $U_d$  assign the value of  $U_d = U_n$ , and proceed to Step 3.

**Step 6** Calculate the fitness values of various spatial corners by the SCF\_HA, and choose  $S_{best}$  to load the cargo. The current cargo  $i$  is traversed sequentially through the spatial corners  $S_i$  of the remaining placeable space in  $P$ , and calculate the fitness value of  $S_i$ . Select the spatial corner with the highest fitness as the  $S_{best}$  and load the cargo. Update the sets  $U_a = U_a + \{i\}$ ,  $U_d = U_d - \{i\}$ , and  $i = \{i\} + 1$ .

**Step 7** Determine whether there is currently unloaded cargo. If the set  $U_n = \emptyset$  or  $U_a = U_i$ , it means that all the cargoes have been loaded into the container and output the solutions  $V$ ; otherwise, proceed to Step 3 to continue loading.

**Step 8** Extract the loading results of containers in solutions  $V$  or  $V''$  as the optimal loading scheme  $I$ , and initialize  $M = 0$ .

**Step 9** Try to repair the container types with a low-loading-rate. Destroy the  $I_l$ , and its loaded cargoes are added to  $U_d$  for

reloading. Meanwhile, the type sequence of the containers is generated in the second way in Step 2.

**Step 10** Reload the cargo. Reload the cargoes in  $U_d$  and calculate the final loading scheme  $I_t$ .

**Step 11** Determine the quality of the repair container. If  $I_t \geq I_l$ , choose the loading scheme  $I_t$  of Step 10; otherwise, select the loading scheme  $I_l$  of Step 9. Finally, update the total loading scheme  $I = I_{\text{best}} \cup \max\{I_t, I_l\}$ .

**Step 12** Determine the container type with the worst loading rate. Determine whether the container used in  $I_{\text{worst}}$  is of a large type, if so, proceed to Step 13; otherwise, proceed to Step 14.

**Step 13** Change the container type. Change the current container size to a smaller container and reload the cargoes into the container. If all cargoes are loaded into the new type of container, it means that the container type has been successfully repaired and output  $I_{\text{new}}$ . If there are still unloaded cargoes, it means that the container type cannot be changed, and the output result is still  $I_{\text{worst}}$ . Finally, update the total loading scheme  $I = \{I - I_{\text{worst}}\} \cup \max\{I_{\text{worst}}, I_{\text{new}}\}$  and proceed to Step 14.

**Step 14** Output the solutions  $V$  and determine whether  $M$  is equal to 1. If  $M = 1$ , directly output the final solutions  $V_{\text{best}}$ , otherwise enter Stage 3.

The pseudocode of the SCF\_HA and the CTSA are summarized as Algorithms 1 and 2.

---

**Algorithm 1** SCF\_HA (Stage 1)

---

**Input:**

the information of cargoes and containers

**Output:**

$V$ : the loading solutions;  $M$ : determine if the solutions are optimized in Stage 3

1:  $U_N, P, U_a \leftarrow \emptyset$ , and  $U_d \leftarrow U_i, t \leftarrow 0, M \leftarrow 0$  // initialization

2: **for** each container  $b$  in  $T_b$  **do**

3:   **if**  $t=0$  **then**

4:      $T_b \leftarrow T_b - b$

5:     **for** each cargo  $i$  in  $U_d$  **do**

6:       determine the direction axis for loading the cargo  $i$

7:       determine whether satisfies the constraints; if not, proceed to next iteration

8:       calculate the fitness values for each spatial corner and select the spatial corner with the highest fitness as the  $S_{\text{best}}$  and load the cargo  $i$

9:     **endfor**

10:  **else**

11:    $V \leftarrow$  the optimal solutions at this time

12:   **break**

13:  **endif**

14:  **if**  $U_d = \emptyset$  or  $T_b = \emptyset$  **then**

15:    $t \leftarrow t + 1$

16:  **else**

17:    $t \leftarrow 0$

18:  **endif**

19: **endfor**

20: **return**  $V$  and  $M$

---

**Algorithm 2** CTSA (Stage 2)**Input:** $V$ : the cargo loading solutions from Stage 1; and  $M$  $V''$ : the optimized solutions from Stage 3**Output:** $V'$  or  $V_{best}$  $V'$ : the solutions used to optimize in Stage 3;  $V_{best}$ : the final solutions1: extract the loading results of containers in solutions  $V$  as the optimal loading scheme  $I$ 2:  $U_d \leftarrow$  destroy the  $I_l$  and its loaded cargoes are added to  $U_d$  for reloading3:  $I \leftarrow I - I_l$ 4: use the Algorithm 1 to reload the cargoes in  $U_d$  and calculate the final loading scheme  $I_t$ .5:  $I \leftarrow I_{best} + \max \{I_t, I_l\}$ 6: **if** the container used in  $I_{worst}$  is of a large type **then**7: change the current container type to a smaller container, reload the cargoes into the container, and  $I \leftarrow I - I_{worst}$ 8: **endif**9: update the solutions  $V'$ 10: whether to proceed to the next stage of optimization is based on the value of  $M$ 11: **return** the loading solutions  $V'$  or  $V_{best}$ , and  $M$ **Improved genetic algorithm**

Based on the literature review on solving methods in “[Solving methods](#)”, the design of algorithm for improving the solutions quality mainly consists of some intelligent algorithms. Among them, some scholars have used genetic algorithms to achieve optimization of the solutions [3, 51]. However, for the 3D-MBSBPP studied in this paper, the existing genetic algorithm (GA) is unable to effectively improve the quality of solutions. This is primarily due to the following two reasons: firstly, this paper incorporates minimizing the container usage cost as one of the constructed objective functions, which leads to the need to optimize container types during the evolutionary process to improve the solution quality. However, the existing GA primarily focuses on optimizing cargo placement state (cargo loading sequence and cargo placement orientation) and lacks the capability to optimize container types. This limitation renders the existing GA ineffective in optimizing solutions generated during the first two stages of this paper. Secondly, the three-dimensional bin packing problem itself is an NP-Hard problem with high complexity and it also involves optimizing container types. All these factors not only exponentially increase the difficulty of problem-solving

but also expand the search space for solutions, ultimately causing the existing GA to be prone to premature convergence during the iteration process, thereby impacting the algorithm’s local search capability.

Therefore, to overcome the weaknesses of the existing GA and enhance its performance, this paper proposes an improved genetic algorithm (IGA). Specifically, we use a three-stage natural number coding to express the state of various types of cargoes placed in different types of containers, facilitating the algorithm to optimize both cargo loading and container types. Meanwhile, this paper designs the corresponding selection and crossover operators by combining the three-stage coding approach and proposes a new dynamic mutation operator to achieve dynamic mutation operation for different types of coding layers, enhancing the diversity of solutions. Besides, the tabu operator is added to the IGA to enhance the local search capability of the algorithm. Finally, the iteration is completed when the algorithm reaches the maximum number of iterations or the optimal solution unchanged for  $N$  successive generations, achieving the optimization of the solutions generated in the first two stages. The specific description of the proposed IGA is as follows.

### Setting of chromosome coding method

In this paper, we use the three-stage natural number coding to express the state of various types of cargoes placed in different types of containers. The corresponding encoded string is represented as:

$$\left( S_1, S_2, \dots, S_{N_c}, S_{N_c+1}, S_{N_c+2}, \dots, S_{N_c+N}, S_{N_c+N+1}, S_{N_c+N+2}, \dots, S_{N_c+2N} \right),$$

where  $S_1 \sim S_{N_c}$  represents the serial number of the container, and the serial number of container of the same type remains consistent;  $S_{N_c+1} \sim S_{N_c+N}$  represents the placement orientation of the corresponding number of cargoes in the container, and ranges from 1 to 6; and  $S_{N_c+N+1} \sim S_{N_c+2N}$  represents the sequence in which the cargoes are loaded into the container.

### Fitness function

Based on the space utilization rate and container usage cost, we establish the fitness function, and add the load balance constraint as a penalty function to limit the generation of infeasible schemes. The fitness function of the specific calculation formula is shown in Eqs. 24–25 (including  $\omega_1, \omega_2, \omega_3, \omega_4$  on behalf of the weight coefficient):

$$F = \omega_1 \times \sum_{k=1}^c \sum_{j=1}^m \frac{\sum_{i=1}^n v_i \alpha_i^{kj}}{V_{kj}} + \omega_2 \times \sum_{k=1}^c \sum_{j=1}^m \frac{\sum_{i=1}^n g_i \alpha_i^{kj}}{G_{kj}} + \omega_3 \times \left( \frac{1}{P_c} \right) + \omega_4 \times \frac{\sum_{u=1}^r S_u}{N_s}, \tag{24}$$

where  $P_c$  represents:

$$P_c = \frac{(\sum_{k=1}^c C_k \times N_k) \times \frac{1}{\sum_{k=1}^c N_k} - \min(C_1, C_2, C_3, \dots, C_c)}{\max(C_1, C_2, C_3, \dots, C_c) - \min(C_1, C_2, C_3, \dots, C_c)}. \tag{25}$$

### Population initialization

To ensure efficient algorithm iteration and promote population diversity, the first 60% of the population is generated following specified rules, while the remaining 40% of the population is randomly generated. The specific operation for population initialization is shown in Fig. 12.

### Selection, crossover, and mutation operators

The selection operator mainly adopts the tournament selection strategy. At the same time, to avoid the destruction of

excellent solutions in the iterative process, the elite individual retention strategy is added to the algorithm. Crossover operator is primarily generated using partial crossover mapping.

In this paper, based on the type of coding layer, a dynamic mutation operator is proposed. When selecting the sequential coding layer mutation, the operator mainly randomly selects one of the following three strategies: the two-point crossover strategy, the gene exchange strategy between two points, and the multi-point rearrangement strategy. When selecting the integer coding layer, random number mutation is used, that is,  $m$  positions are randomly selected in the interval  $[1, n]$  for mutation, and  $m$  random numbers are generated according to Eq. 26. Finally, the generated random number is corrected based on the range of genes in the coding layer, to generate new genes.

$$D_{\text{new}} = D + (D_{\text{max}} - D_{\text{min}}) \times \xi \times r, \tag{26}$$

where  $D_{\text{new}}$  represents the newly generated gene,  $D$  represents the original gene,  $D_{\text{max}}$  and  $D_{\text{min}}$  respectively the upper and lower bounds of gene  $D$ ,  $\xi$  represents the generation of random numbers from the interval  $[-1, 1]$ , and  $r$  represents the step size of the change, with the value being in  $[0, 1]$ .

### Tabu operator

To enhance the local search capability of the genetic algorithm, the tabu operator is integrated. The solution at the end of the genetic algorithm iteration is used as the initial solution of tabu search for local search. The main steps are as follows.

- Step 1** Input the neighborhood solutions to be searched;
- Step 2** Select the neighborhood structure and set the search step size based on the input neighborhood solution;
- Step 3** Search the neighborhood of the current solution based on the determined neighborhood structure and step size;
- Step 4** Select candidate solutions from the new neighborhood solutions;
- Step 5** Determine whether the candidate solution is consistent with the solution in the tabu list to avoid repeated searches for neighborhood solutions;
- Step 6** When the optimal solution has not changed for  $N$  consecutive generations or the maximum number of iterations has been reached, it terminates and outputs the results.

In summary, the specific steps of IGA are as follows.

- Step 1** Input the loading solutions  $V$  or  $V'$ , the parameters of the algorithm, and  $M$ ;
- Step 2** Determine the initial population  $pop$ ;

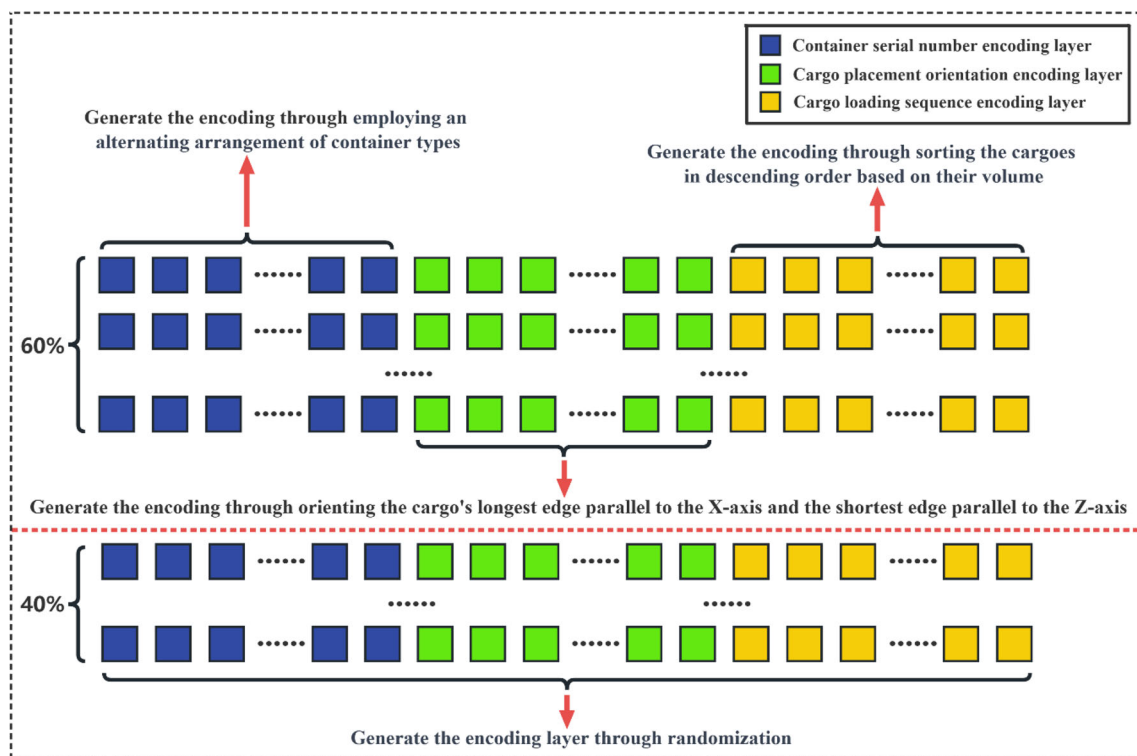


Fig. 12 The specific population initialization operation

**Step 3** Selection, crossover, and mutation operations. The population is updated based on the selection, crossover, and mutation operators designed in this paper. Calculate the fitness values of the new population.

**Step 4** Record the best individual fitness value *sofarbest* in each generation and the best fitness value *gbest* in the whole iteration process.

**Step 5** Determine whether the termination condition of IGA is reached. If this is achieved, output the solutions and proceed to Step 6. If the termination condition of the iteration is not met, return to Step 3 and continue the iteration.

**Step 6** Perform a tabu search. The input solutions are searched in the neighborhood and new neighborhood solutions are generated. Calculate the fitness value of the new neighborhood solutions.

**Step 7** Select the candidate solutions. The new neighborhood solutions are sorted in descending sequence based on their fitness values, and the top 50% are selected as candidate solutions.

**Step 8** Judge the tabu attribute of the candidate solutions. Check whether the candidate solutions appear in the tabu table, use the optimal solution *sofarbest* corresponding to the non-tabu object as the current solution, and replace the first object which is the first to enter the tabu table.

**Step 9** Update *sofarbest* and *gbest*. Determine whether the termination condition has been reached at this time. If not, go back to Step 6. If so, proceed to Step 10.

**Step 10** Output the solutions  $V''$ ,  $M$ , and determine if further adjustment and optimization of the container type are needed. If needed then backtrack to Stage 2. If not needed, directly output the solutions  $V''$  as the final solutions  $V_{best}$ .

The pseudocode of the specific procedure of the IGA is summarized as Algorithm 3.

**Algorithm 3** IGA (Stage 3)

---

**Input:** the loading solutions  $V$  or  $V'$ , the parameters of the algorithm, and  $M$

**Output:**  
the optimized solutions  $V''$  or  $V_{best}$ , and  $M$

- 1: determine the initial population  $pop$
- 2:  $fit_{best} \leftarrow$  calculate the fitness values of the initial population;  $fbestsofar \leftarrow -inf$
- 3: update the current optimal value  $fbestsofar$
- 4:  $Generation \leftarrow 0$ , and  $time \leftarrow 0$
- 5: **while**  $Generation \leq generationmax+1$  and  $time \leq timelimit$  **do**
- 6:    $Generation \leftarrow Generation + 1$
- 7:   perform selection operation, crossover operation and mutation operation
- 8:    $g_{best} \leftarrow$  calculate the fitness value of the new population
- 9:   update the current optimal value  $fbestsofar$
- 10: **endwhile**
- 11:  $GA\_pop \leftarrow$  the solutions at the end of the IGA iteration
- 12:  $iter \leftarrow 0$ , and  $time \leftarrow 0$
- 13: **while**  $iter \leq itermax + 1$  and  $time \leq timelimit$  **do**
- 14:    $iter \leftarrow iter + 1$
- 15:   **for**  $k=1$  to  $carnum$  **do**
- 16:     search the  $GA\_pop$  in the neighborhood and the new neighborhood solutions are generated
- 17:   **endfor**
- 18:   calculate the fitness value of the new neighborhood solutions
- 19:   select the candidate solutions and judge the tabu attribute of them
- 20:   update the current optimal value
- 21: **endwhile**
- 22:  $M \leftarrow 1$ , and determine if need to adjust the container types; if needed, backtrack to Algorithm 2
- 23: **return** the optimized solutions  $V''$  or  $V_{best}$ , and  $M$

---

## Computational experiments

This section conducts experimental analysis on the SCF\_DFA proposed in this paper. We conduct a performance comparison of SCF\_DFA with five state-of-the-art algorithms in “[Comparison with state-of-the-art algorithms](#)” and apply the SCF\_DFA to solve the real-world instances for the 3D-MBSBPP in “[3D-MBSBPP experimental analysis](#)”. To further evaluate and analyze the performance of our proposed algorithm, the ablation experiments are conducted in “[Ablation experimental analysis](#)”. The experiments are carried out in the MATLAB R2022a environment and the algorithm parameters are set as follows: population size  $pop$

$= 70$ , crossover probability  $pc = 0.8$ , mutation probability  $pm = 0.1$ , neighborhood search number  $carnum = 50$ , tabunum  $= 25$ , maximum iteration number  $generationmax = 500$ , maximum running time  $timelimit = 600$  s.

## Comparison with state-of-the-art algorithms

To demonstrate the performance of the SCF\_DFA, the standard instances proposed by Bischoff and Ratcliff are selected to verify the algorithm. The instances can be obtained from the OR-Library database. This dataset consists of 7 files (thpack1 to thpack7), and each file contains 100 sets of instances. The container is an international standard 20-foot container (with dimensions of  $587 \times 233 \times 220$  cm). In these

seven files, the number of different types of cargo gradually increases and the heterogeneity enhanced. Specifically, the number of different cargo types in the files is 3, 5, 8, 10, 12, 15, and 20, with each instance containing more than 100 pieces of cargo. We apply our proposed methods to solve these instances, conducting 10 random runs for each instance and calculating the average values. Besides, to validate the performance of the algorithm proposed in this paper, we compare the SCF\_DFA with the following state-of-the-art algorithms published in literature:

- **GRASP**: a greedy randomized adaptive search procedure approach proposed by Martínez et al. [9] (2015).
- **PRTS**: a progressively-refined tree search approach proposed by Wang et al. [24] (2019).
- **TSTDE**: a differential evolution algorithm combined with the ternary search tree model proposed by Huang et al. [22] (2022).
- **MDCLP**: Filella et al. [1] (2022) proposed two evaluation criteria to optimize the solutions: MDCLP-B and MDCLP-S.
- **IABC**: Bayraktar et al. [25] (2021) proposed three improved artificial bee colony algorithms to optimize the solutions.

The performance comparison of SCF\_DFA with other algorithms are shown in Table 3. For files thpack1 through thpack3, the PRTS approach by Wang et al. [24] yields superior results. The main reason is that these files contain fewer types and a smaller quantity of cargoes, and the container can generate a tidier residual space, which facilitates the PRTS approach to solve the cargo placement. As the instances scale increase, the SCF\_DFA proposed in this paper begins to demonstrate superiority. For example, in the files of thpack4, thpack5, and thpack7, the SCF\_DFA obtains optimal results. Especially for thpack4, the SCF\_DFA enhances container space utilization rate to 88.08%, which is 2.82% improvement over other methods. On the other hand, as the heterogeneity gradually increases, it may lead to irregularities in the residual space, thus affecting the algorithm's search for the optimal placement positions for cargo. However, as indicated in Table 3, except for our proposed algorithm and the IABC algorithm, the performance of remaining algorithms decreases with the rising cargo heterogeneity. In conclusion, it is evident from the last row of Table 3 that the SCF\_DFA achieves a container's average space utilization rate of 87.39%, which surpasses that of other state-of-the-art methods and demonstrates the superiority of the proposed SCF\_DFA.

Table 4 shows the details of the results obtained by the SCF\_DFA. Among them, the column labeled "Average" denotes the average value of the container space utilization rate obtained from all the instances in each file, and the

column labeled "Std" denotes the standard deviation of the container space utilization rate in the file. From Table 4, it is evident that the space utilization rates of the containers fall within the range of 78–92%, with the majority clustered around 88%. Besides, the standard deviation of each instance exhibits relatively low fluctuation, indicating that the container loading results are stable and demonstrating the effectiveness of the SCF\_DFA proposed in this paper.

Through further analysis of the above experimental results, there are two main reasons for the superiority of the proposed SCF\_DFA, as follows. Firstly, as the central component of the SCF\_HA, the assessment operators are designed to minimize gaps between cargoes and improve cargo load balance. Based on these objectives, they analyze and evaluate all placeable spatial corner. This enhances the exploration capability of the algorithm for identifying the optimal placement and makes the cargoes better utilize the remaining space within the container when they are loaded. Secondly, the proposed algorithm in the previous stages can obtain higher-quality initial solutions, enhancing the efficiency of the search when optimizing solutions quality in the improved methods stage. We also make some improvements to the genetic algorithm, such as adding the dynamic mutation operator and the tabu operator, which enhances the performance of the genetic algorithm. Therefore, the SCF\_DFA proposed in this paper shows significant advantage in solving the single-container type loading.

### 3D-MBSBPP experimental analysis

From "Comparison with state-of-the-art algorithms", the SCF\_DFA demonstrates superiority in solving the single-container type loading. When solving the 3D-MBSBPP, it needs to add the container type selection algorithm on the basis of solving the single-container type loading, realizing the corresponding adjustment and optimization of the container type. Since there is no standard instance for 3D-MBSBPP, we use the real-life instances mentioned in Han [56] for experimental analysis. The container information is shown in Table 5 and the cargo information is shown in Table 6. Among them, when a 40-foot container is fully loaded with cargo, the maximum lateral offset of the combined center of gravity is 350 mm, the maximum longitudinal offset is 100 mm and the maximum vertical offset is 100 mm. When a 20-foot container is fully loaded, the maximum lateral offset of the combined center of gravity is 350 mm, the maximum longitudinal offset is 150 mm, and the maximum vertical offset is 150 mm.

The container type selection algorithm constructed by Han is consistent with the most widely used container type selection algorithms that we mentioned in "Solving methods". To verify the superiority of the algorithm proposed in this paper, the container type selection algorithm constructed by Han is

**Table 3** Comparison of proposed SCF\_DFA with state-of-the-art algorithms

	GRASP	PRTS	MDCLP		TSTDE	IABC			SCF_DFA
			MDCLP-B	MDCLP-S		MIABC	GABC	JHABC	
thpack1	0.8748	<b>0.9046</b>	0.8200	0.8410	0.7662	0.7882	0.8219	0.8442	0.8596
thpack2	0.8498	<b>0.8959</b>	0.8190	0.8340	0.7487	0.7980	0.8139	0.8330	0.8712
thpack3	0.8244	<b>0.8849</b>	0.7960	0.8090	0.7094	0.8424	0.8545	0.8684	0.8795
thpack4	0.8103	0.8734	0.7870	0.8040	0.6866	0.8344	0.8398	0.8526	<b>0.8808</b>
thpack5	0.7978	0.8638	0.7820	0.7960	0.6659	0.8218	0.8420	0.8417	<b>0.8806</b>
thpack6	0.7834	0.8512	0.7730	0.7900	0.6469	0.8583	0.8784	<b>0.8933</b>	0.8766
thpack7	0.7642	0.8396	0.7630	0.7810	0.6236	0.8354	0.8570	0.8587	<b>0.8687</b>
Average	0.8150	0.8733	0.7914	0.8079	0.6925	0.8255	0.8439	0.8560	<b>0.8739</b>

Bold indicates the best results

**Table 4** Experimental results of SCF\_DFA

File cases	Cargo type	Instance number	Min	Max	Average	Std
thpack1	3	100	0.7791	0.9112	0.8596	2.9378
thpack2	5	100	0.8237	0.9195	0.8712	1.7884
thpack3	8	100	0.8419	0.9097	0.8795	1.4726
thpack4	10	100	0.8363	0.9153	0.8808	1.3131
thpack5	12	100	0.8483	0.9071	0.8806	1.1547
thpack6	15	100	0.8442	0.9004	0.8766	1.2132
thpack7	20	100	0.8384	0.8914	0.8687	1.1960
Average	10.43	100	0.8303	0.9078	0.8739	1.5823

**Table 5** Information of containers

Container type	Length (mm)	Width (mm)	Height (mm)	Maximum load (kg)	Cost (\$)
40-foot container	8500	2500	2500	2000	850
20-foot container	5870	2330	2200	2000	400

**Table 6** Information of cargoes to be loaded

Cargo type	Length (mm)	Width (mm)	Height (mm)	Quality (kg)	Quantity (pcs)
1	890	780	500	3	68
2	730	650	460	5	30
3	940	660	390	10	86
4	660	590	420	5	59
5	1060	930	700	9	42
6	800	710	620	6	65
7	600	360	260	2	90
8	720	310	310	4	70

used as a baseline method to compare with the SCF\_DFA. The algorithms are randomly run 10 times and the comparison of average performance is shown in Table 7.

In Table 7, the row labeled “Average space utilization rate” denotes the average container space utilization rate

over 10 experiments. The row labeled “Container load balance index” denotes the standard deviation of all container loading results in each experiment, which represents the balance of the container loading results. From the average performance comparison in Table 7, under the same usage cost of containers, the SCF\_DFA can improve the

**Table 7** Average performance comparison of SCF\_DFA with baseline method

	Baseline method	SCF_DFA
Average space utilization rate (%)	83.90	<b>85.38</b>
Total cost (\$)	2050	2050
Container load balance index	0.2368	<b>0.0253</b>

Bold indicates the best results

average space utilization rate by 1.48%. More importantly, the container load balance index of proposed SCF\_DFA is 0.2115 lower than that of baseline method, demonstrating that the obtained container loading results are more balanced. It ensures equal stress inside the container and improves the safety of cargo transportation. In conclusion, these aspects collectively demonstrate the superiority of the proposed SCF\_DFA in solving the 3D-MBSBPP.

To further demonstrate the performance of the SCF\_DFA, the results of the fifth experiment are selected for the specific analysis. The specific results of the fifth experiment are shown in Table 8. The specific loading results are visualized in Fig. 13.

For the cost of using containers, if all 510 pieces of cargo are loaded, a total of three containers are needed when only 40-foot container is used, and the total cost is \$ 2550. When only 20-foot containers are used, a total of five are needed, and the total cost is \$ 2000. From Table 8, the results obtained through the application of our proposed algorithm indicate the need for a total of four containers: one 40-foot container and three 20-foot containers. The total cost amounts to \$ 2050, demonstrating the cost-effectiveness of our algorithm in selecting container types. Also, from Table 8, it can be seen that the volume utilization rate of each container holds steady at more than 81%. The 179 pieces of cargo in the upper layer are all in a state of load balance and the actual center of gravity of each container falls within the safe deviation range.

For the 3D-MBSBPP, uneven container loading may occur. From the container load balance index in Table 7, it can be seen that the average container load balance index obtained from 10 experiments is 0.0253, indicating that the container loading results are stable. Especially from the specific space utilization rate of each container in Table 8, the SCF\_DFA proposed in this paper can improve the balance of container loading. As a comparison, the container load balance index obtained from the baseline method is 0.2368, which is 0.2115 higher than the results in this paper, indicating that the container loading results obtained from the baseline method are more fluctuated. Therefore, the SCF\_DFA proposed in this paper can effectively balance the

loading of each container and ensure equal stress inside the container.

Through the analysis of the above experimental results, it is evident that the SCF\_DFA proposed in this paper performs well in solving the 3D-MBSBPP due to the following two main reasons: firstly, the CTSA proposed in this paper can dynamically adjust and optimize the container types according to the feedback of the loading results, improving the flexibility of the algorithm. Especially during the trial repair phase of the CTSA, some containers with low-loading-rate will be replaced with new types and the cargoes in them will be reloaded. This makes the space utilization rate of each container more average and improves the balance of container loading. In contrast, the selection of the container type by the baseline method has been determined before the cargo is loaded, it may lead to irrationality in the set container type. For example, using a small type container is fully capable of carrying the remaining cargo, but a large type container is set up for loading, thus decreasing the performance of the method. Secondly, the objective function of the 3D-MBSBPP constructed in this paper is based on the container space utilization rate, the usage cost, and the cargo load balance. During the evolution process of the IGA, its fitness function also searches for the optimal solution based on these objectives. Therefore, the solutions obtained demonstrate superior performance across multiple dimensions.

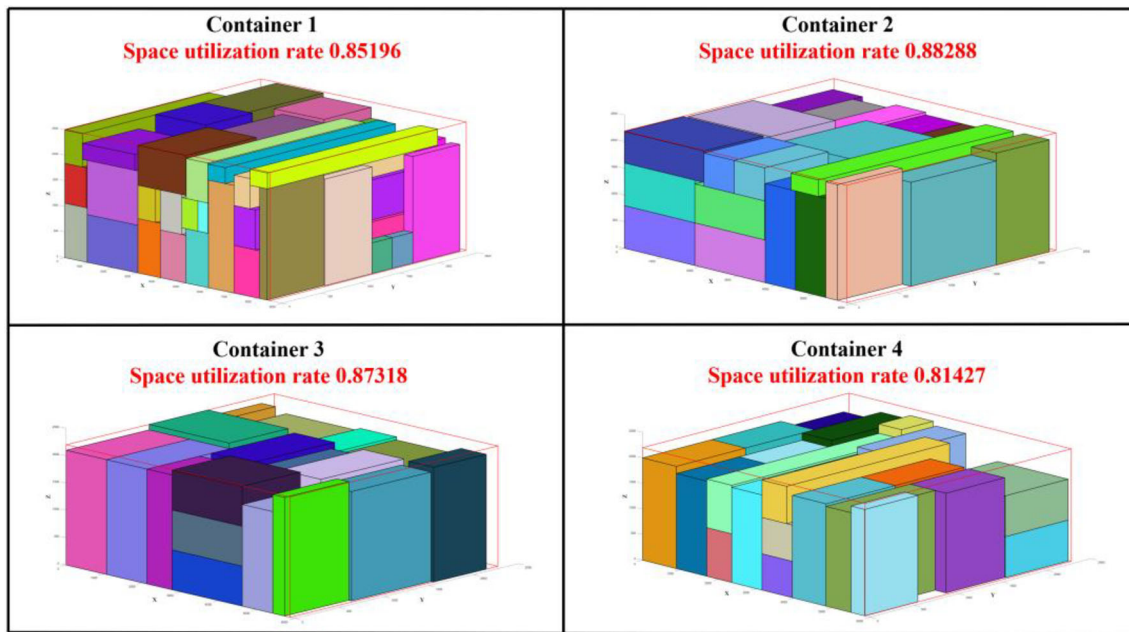
### Ablation experimental analysis

To further evaluate and prove the effectiveness of the SCF\_DFA proposed in this paper, we conduct ablation experiments using instances from the 3D-MBSBPP in “3D-MBSBPP experimental analysis”. As depicted in Table 9, we compare the SCF\_DFA with its constituent components, mainly the SCF\_HA, the combination of SCF\_HA with IGA, and the combination of SCF\_HA with CTSA. The column labeled “Methods” denotes the comparative methods in the ablation experiments (w/o denotes that the SCF\_DFA is without this component). We run each method 10 times randomly and calculate the average of the results from these 10 runs, as shown in the second column to the last column of the table.

The experimental results in Table 9 clearly demonstrate the impact of various components in the proposed algorithm on container space utilization rate and usage cost. Specifically, when only the SCF\_HA is employed, it is evident from the first row of Table 9 that the results are the worst. The primary reason is that this method is primarily designed to optimize the placement of cargoes without considering the selection of container types. When the CTSA proposed in this paper is integrated into the SCF\_HA, the second row shows a reduction of \$ 1650 in container usage costs. Also, with optimization of container types, the container space utilization rate is improved by 18.64%. This indicates that the

**Table 8** Results of SCF\_DFA in the fifth experiment

Index name	Container serial number			
	1	2	3	4
Container type	40-foot	20-foot	20-foot	20-foot
Cost (\$)	850	400	400	400
Actual center of gravity of the container	$X = 4297.0$	$X = 3259.9$	$X = 2792.6$	$X = 3166.0$
	$Y = 1346.8$	$Y = 1143.0$	$Y = 1310.6$	$Y = 1058.7$
	$Z = 1326.1$	$Z = 1111.9$	$Z = 1011.4$	$Z = 970.2$
Traversal offset of the center of gravity (mm)	47	324.9	142.4	231
Longitudinal offset of the center of gravity (mm)	96.8	22	145.6	106.3
Vertical offset of the center of gravity (mm)	76.1	11.9	88.6	129.8
Total quantity of cargoes to be loaded (pcs)	212	107	100	91
Quantity of upper cargo (pcs)	100	36	28	15
Quantity of upper cargoes that are in load balance (pcs)	100	36	28	15
Volume utilization rate (%)	85.20	88.29	87.32	81.43
Load utilization rate (%)	47.35	32.50	32.70	24.30

**Fig. 13** Loading diagram of each container**Table 9** Comparison of proposed SCF\_DFA without the components

Methods	Space utilization rate (%)	Total cost (\$)	Amount of container types	
			40-foot	20-foot
w/o (CTSA and IGA)	56.00	4150	3	4
w/o IGA	74.64	2500	2	2
w/o CTSA	67.13	3350	3	2
SCF_DFA	<b>85.38</b>	<b>2050</b>	<b>1</b>	<b>3</b>

Bold indicates the best results

CTSA proposed in this paper can effectively improve the container space utilization rate and reduce usage costs. When the IGA is integrated into the SCF\_HA, the third row shows an improvement of 11.13% in container space utilization rate and a reduction of \$ 800 in usage costs, demonstrating that the proposed IGA effectively improves the quality of solutions generated in the first stage. However, the lack of optimization specifically for container types has caused the improvement in container space utilization rate and usage costs is not desirable. Finally, when the CTSA and the IGA are integrated into the SCF\_HA, the results of the final row show a 29.38% improvement in container space utilization rate, and the usage costs is reduced by \$ 2100, demonstrating the effectiveness of the SCF\_DFA proposed in this paper. Importantly, it is also evident that all three components play crucial roles in improving the quality of solutions.

## Conclusion and future works

The main purposes of this study are to solve the difficulties existing in the current three-dimensional bin packing problem and improve the methods involved, thereby enhancing the efficiency and cost-effectiveness of cargo loading processes. Through an analysis and review of previous research on the 3D-MBSBPP, it is found that there are still difficulties and challenges in two specific aspects: the construction of multiple objective functions and the design of the algorithms. Therefore, we innovatively construct a multiple objective mixed integer programming mathematical model aiming at maximizing container space utilization rate and cargo load balance while minimizing container usage costs. Secondly, we introduce an innovative dynamic feedback algorithm based on spatial corner fitness (SCF\_DFA) to solve the 3D-MBSBPP. This algorithm improves the previous algorithm in terms of the rationality of the cargo placement position, the search of the remaining space, and the dynamic adjustment of the container type, assisting the company in improving container space utilization rate and reducing usage cost.

Specifically, during the construction methods stage, we propose a heuristic algorithm based on spatial corner fitness (SCF\_HA) to load the cargoes. Its purposes are to select the optimal positions to place the cargo and to reduce the generation of gaps in the remaining space. Subsequently, after loading the container, based on the feedback from the loading results, we use the container type selection algorithm (CTSA) proposed in this paper to adjust and optimize the container types dynamically. This stage aims to reduce usage costs and improve container space utilization rate. Finally, during the improvement methods stage, we apply an improved genetic algorithm (IGA) to improve the quality of solutions generated in the first two stages. Comparative experiments with other state-of-the-art algorithms reveal that the SCF\_DFA

proposed in this paper can improve the space utilization rate by 1.79%, demonstrating its superiority and effectiveness. On this basis, the SCF\_DFA is apply to solve the real-life instances of 3D-MBSBPP, and the final results show that the average space utilization rate of the containers remains at 85.38%, which is improved by 1.48% compare with the baseline method. The SCF\_DFA can also improve the balance of container loading and ensure equal stress inside the container. Thus, it indicates that the SCF\_DFA we proposed exhibits a certain superiority in solving the 3D-MBSBPP. Besides, to further evaluate and prove the effectiveness of the respective components of the algorithm, ablation experiments are conducted. The results show that each component in the algorithm plays a crucial role in improving the quality of solutions.

In future, we will conduct further research on 3D-MBSBPP based on the study in this paper, mainly including the following aspects: (i) Incorporating more practical constraints. For example, we will incorporate cargo fragility and extrusion constraints into the mathematical model. This aims to provide a more realistic simulation of the loading process in real-life scenarios. (ii) Exploring the identification of similar cargoes. We will try to use the clustering algorithm to identify and combine similar types of cargoes. (iii) Improving the algorithms proposed in this paper. In the construction methods stage, we will add more assessment operators for the spatial corner. In the container type selection algorithm, we will incorporate some perturbation solutions to better optimize the container types.

**Acknowledgements** This work was supported by the Key Project of the Natural Science Foundation of Fujian Province of China [Grant No. 2022J02053].

**Data availability** When using the data in this manuscript, readers can directly cite this manuscript or contact the authors for access.

## Declarations

**Conflict of 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.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## References

1. Filellaa GB, Trivella A, Corman F (2022) Modeling soft unloading constraints in the multi-drop container loading problem. *Eur J Oper Res* 308(01):336–352. <https://doi.org/10.1016/j.ejor.2022.10.033>
2. Gajda M, Trivella A, Mansini R, Pisinger D (2022) An optimization approach for a complex real-life container loading problem. *Omega* 107:1–16. <https://doi.org/10.1016/j.omega.2021.102559>
3. Silva E, Ramos AG, Oliveira JF (2018) Load balance recovery for multi-drop distribution problems: a mixed integer linear programming approach. *Transp Res Part B* 116:62–75. <https://doi.org/10.1016/j.trb.2018.08.001>
4. Wei LJ, Zhu WB, Lim A (2015) A goal-driven prototype column generation strategy for the multiple container loading cost minimization problem. *Eur J Oper Res* 241(01):39–49. <https://doi.org/10.1016/j.ejor.2014.08.015>
5. Martínez JC, Cuellar D, Álvarez-Martínez D (2018) Review of dynamic stability metrics and a mechanical model integrated with open source tools for the container loading problem. *Electron Notes Discrete Math* 69:325–332. <https://doi.org/10.1016/j.endm.2018.07.042>
6. Yang S, Song S, Chu SL, Song R, Cheng JY, Li YB, Zhang W (2023) Heuristics integrated deep reinforcement learning for online 3D bin packing. *IEEE Trans Autom Sci Eng*. <https://doi.org/10.1109/TASE.2023.3235742>
7. Que QQ, Yang F, Zhang DF (2023) Solving 3D packing problem using transformer network and reinforcement learning. *Expert Syst Appl* 214:1–10. <https://doi.org/10.1016/j.eswa.2022.119153>
8. Alonso MT, Alvarez-Valdes R, Iori M, Parreno F (2019) Mathematical models for multi container loading problems with practical constraints. *Comput Ind Eng* 127:722–733. <https://doi.org/10.1016/j.cie.2018.11.012>
9. Martínez DA, Alvarez-Valdes R, Parreño F (2015) A GRASP algorithm for the container l-oading problem with multi-drop constraints. *Pesquisa Operacional* 35(01):1–24. <https://doi.org/10.1590/0101-7438.2015.035.01.0001>
10. Lurkin V, Schyns M (2015) The airline container loading problem with pickup and deli-very. *Eur J Oper Res* 244(03):955–965. <https://doi.org/10.1016/j.ejor.2015.02.027>
11. Eley M (2002) Solving container loading problems by block arrangement. *Eur J Oper Res* 141(02):393–409. [https://doi.org/10.1016/S0377-2217\(02\)00133-9](https://doi.org/10.1016/S0377-2217(02)00133-9)
12. Araya I, Guerrero K, Nunez E (2017) VCS: a new heuristic function for selecting boxes i-n the single container loading problem. *Comput Oper Res* 82:27–35. <https://doi.org/10.1016/j.cor.2017.01.002>
13. Oliveira LD, de Lima VL, de Queiroz TA, Miyazawa FK (2021) The container loading problem with cargo stability: a study on support factors, mechanical equilibrium and grids. *Eng Optim* 53(07):1192–1211. <https://doi.org/10.1080/0305215X.2020.1779250>
14. Che CH, Huang WL, Lim A, Zhu WB (2011) The multiple container loading cost minimi-zation problem. *Eur J Oper Res* 214(03):501–511. <https://doi.org/10.1016/j.ejor.2011.04.017>
15. De Castro Silva JL, Soam NY, Maculan N (2003) A greedy search for the three-dimensional bin packing problem: the packing static stability case. *Int Trans Oper Res* 10(02):141–153. <https://doi.org/10.1111/1475-3995.00400>
16. Kacprzak L, Rudy J, Zelazny D (2015) Multi-criteria 3-dimision bin packing problem. *Int Trans Oper Res* 5(01):85–94
17. Kurpel DV, Scarpin CT, Junior JEP, Schenekemberg CM, Coelho LC (2020) The exact solutions of several types of container loading problems. *Eur J Oper Res* 284(01):87–107. <https://doi.org/10.1016/j.ejor.2019.12.012>
18. Harrath Y (2022) A three-stage layer-based heuristic to solve the 3D bin-packing problem under balancing constraint. *J King Saud Univ Comput Inf Sci* 34(08):6425–6431. <https://doi.org/10.1016/j.jksuci.2021.07.007>
19. Erbayrak S, Özkir V, Yildirim UM (2021) Multi-objective 3D bin packing problem with load balance and product family concerns. *Comput Ind Eng* 159:1–11. <https://doi.org/10.1016/j.cie.2021.10.7518>
20. Zhang DZ, Gu CH, Fang H, Ji CT, Zhang XG (2022) Multi-strategy hybrid heuristic algorithm for single container weakly heterogeneous loading problem. *Comput Ind Eng* 170:1–14. <https://doi.org/10.1016/j.cie.2022.108302>
21. Correcher JF, Alonso MT, Parreno F, Alvarez-Valdes R (2017) Solving a large multicontainer loading problem in the car manufacturing industry. *Comput Oper Res* 82:139–152. <https://doi.org/10.1016/j.cor.2017.01.012>
22. Huang Y, Lai L, Li W, Wang H (2022) A differential evolution algorithm with ternary search tree for solving the three-dimensional packing problem. *Inf Sci* 606:440–452. <https://doi.org/10.1016/j.ins.2022.05.063>
23. Goncalves JF, Resende MGC (2013) A biased random key genetic algorithm for 2D and 3-D bin packing problems. *Int J Prod Econ* 145(02):500–510. <https://doi.org/10.1016/j.ijpe.2013.04.019>
24. Wang Y, Li HL, Lei ZB, Ma DP, Fang Y (2019) Progressively-refined tree search for container loading problem. In: 2019 IEEE 21st International Conference on High Performance Computing and Communications :2520–2528. <https://doi.org/10.1109/HPCC/SmartCity/DSS.2019.00353>
25. Bayraktar T, Ersoz F, Kubat C (2021) Effects of memory and genetic operators on Artificial Bee Colony algorithm for Single Container Loading problem. *Appl Soft Comput* 108:1–17. <https://doi.org/10.1016/j.asoc.2021.107462>
26. Chen MZ, Huo JZ, Duan YR (2023) A hybrid biogeography-based optimization algorithm for three-dimensional bin size designing and packing problem. *Comput Ind Eng* 180:1–13. <https://doi.org/10.1016/j.cie.2023.109239>
27. Tresca G, Cavone G, Carli R, Cerviotti A, Dotoli M (2022) Automating bin packing: a layer building matheuristics for cost effective logistics. *IEEE Trans Autom Sci Eng* 19(03):1599–1613. <https://doi.org/10.1109/TASE.2022.3177422>
28. Gimenez-Palacios I, Alonso MT, Alvarez-Valdes R, Parreno F (2021) Logistic constraints in container loading problems: the impact of complete shipment conditions. *TOP* 29(01):177–203. <https://doi.org/10.1007/s11750-020-00577-8>
29. Elhedhli S, Gzara F, Yildiz B (2019) Three-dimensional bin packing and mixed-case palletization. *INFORMS J Optimiz*. <https://doi.org/10.1287/ijoo.2019.0013>
30. Bortfeldt A, Wascher G (2013) Constraints in container loading—a state-of-the-art review. *Eur J Oper Res* 229(01):1–20. <https://doi.org/10.1016/j.ejor.2012.12.006>
31. Colombi M, Corberan A, Mansini R, Plana I, Sanchis JM (2017) The directed profitable rural postman problem with incompatibility constraints. *Eur J Oper Res* 261(02):549–562. <https://doi.org/10.1016/j.ejor.2017.02.002>
32. Moon I, Nguyen TVL (2014) Container packing problem with balance constraints. *OR Spectrum* 36(04):837–878. <https://doi.org/10.1007/s00291-013-0356-1>
33. Baldi MM, Perboli G, Tadei R (2012) The three-dimensional knapsack problem with balancing constraints. *Appl Math Comput* 218(19):9802–9818. <https://doi.org/10.1016/j.amc.2012.03.052>
34. Amossen RR, Pisinger D (2010) Multi-dimensional bin packing problems with guillotine constraints. *Comput Oper Res* 37(11):1999–2006. <https://doi.org/10.1016/j.cor.2010.01.017>
35. Polyakovskiy S, M'Hallah R (2021) Just-in-time two-dimensional bin packing. *Omega (Westport)* 102:1–41. <https://doi.org/10.1016/j.omega.2020.102311>

36. Zhao XZ, Bennell JA, Bektas T, Dowsland K (2016) A comparative review of 3D container loading algorithms. *Int Trans Oper Res* 23(1–2):287–320. <https://doi.org/10.1111/itor.12094>
37. George JA, Robinson DF (1980) A heuristic for packing boxes into a container. *Comput Oper Res* 7(03):147–156. [https://doi.org/10.1016/0305-0548\(80\)90001-5](https://doi.org/10.1016/0305-0548(80)90001-5)
38. Pisinger D (2002) Heuristics for the container loading problem. *Eur J Oper Res* 141(02):382–392. [https://doi.org/10.1016/s0377-2217\(02\)00132-7](https://doi.org/10.1016/s0377-2217(02)00132-7)
39. da Silva EF, Leao AAS, Toledo FMB, Wauters T (2020) A metaheuristic framework for the Three-dimensional single large object placement problem with practical constraints. *Comput Oper Res* 124:1–33. <https://doi.org/10.1016/j.cor.2020.105058>
40. Lai KK, Xue J (1998) Container packing in a multi-customer delivering operation. *Comput Ind Eng* 35(1–2):323–326. [https://doi.org/10.1016/S0360-8352\(98\)00085-0](https://doi.org/10.1016/S0360-8352(98)00085-0)
41. Parreno F, Alvarez-Valdes R, Tamarit JM, Oliveira JF (2008) A maximal-space algorithm for the container loading problem. *INFORMS J Comput* 20(03):412–422. <https://doi.org/10.1287/ijoc.1070.0254>
42. Martello S, Pisinger D, Vigo D (2000) The three-dimensional bin packing problem. *Oper Res* 48(02):256–267. <https://doi.org/10.1287/opre.48.2.256.12386>
43. Crainic TG, Perboli G, Tadei R (2008) Extreme point-based heuristics for three-dimensional bin packing. *INFORMS J Comput* 20(03):368–384. <https://doi.org/10.1287/ijoc.1070.0250>
44. Trivella A, Pisinger D (2016) The load-balanced multi-dimensional bin-packing problem. *Comput Oper Res* 74:152–164. <https://doi.org/10.1016/j.cor.2016.04.020>
45. Crainic TG, Perboli G, Tadei R (2007) TS<sup>2</sup> PACK: a two-level tabu search for the three-dimensional bin packing problem. *Eur J Oper Res* 195(03):744–760. <https://doi.org/10.1016/j.ejor.2007.06.063>
46. Zachariadis EE, Tarantilis CD, Kiranoudis CT (2009) A guided tabu search for the vehicle routing problem with two-dimensional loading constraints. *Eur J Oper Res* 195(03):729–743. <https://doi.org/10.1016/j.ejor.2007.05.058>
47. Fuellerer G, Doerner KF, Hardl RF, Iori M (2009) Ant colony optimization for the two-dimensional loading vehicle routing problem. *Comput Oper Res* 36(03):655–673. <https://doi.org/10.1016/j.cor.2007.10.021>
48. Fuellerer G, Doerner KF, Hartl RF, Iori M (2010) Metaheuristics for vehicle routing problems with three-dimensional loading constraints. *Eur J Oper Res* 201(03):751–759. <https://doi.org/10.1016/j.ejor.2009.03.046>
49. Iori M, Martello S (2010) Routing problems with loading constraints. *TOP* 18(01):4–27. <https://doi.org/10.1007/s11750-010-0144-x>
50. Fekete SP, Schepers J, Van Der Veen JC (2007) An exact algorithm for higher-dimensional orthogonal packing. *Oper Res* 55(03):569–587. <https://doi.org/10.2307/25147100>
51. Ramos AG, Silva E, Oliveira JF (2018) A new load balance methodology for container loading problem in road transportation. *Eur J Oper Res* 266(03):1140–1152. <https://doi.org/10.1016/j.ejor.2017.10.050>
52. Moura A, Oliveira JF (2005) A GRASP approach to the container-loading problem. *IEEE Intell Syst* 20(04):50–57. <https://doi.org/10.1109/MIS.2005.57>
53. Li Y, Chen MZ, Huo JZ (2022) A hybrid adaptive large neighborhood search algorithm for the large-scale heterogeneous container loading problem. *Expert Syst Appl* 189:1–12. <https://doi.org/10.1016/j.eswa.2021.115909>
54. Li XP, Zhang KK (2015) A hybrid differential evolution algorithm for multiple container loading problem with heterogeneous containers. *Comput Ind Eng* 90(C):305–313. <https://doi.org/10.1016/j.cie.2015.10.007>
55. Piyachayawat T, Mungwattana A (2017) A hybrid algorithm application for the multi-size pallet loading problem case study: lamp and lighting factory. In: 2017 4th International Conference on Industrial Engineering and Applications :100–105. <https://doi.org/10.1109/iea.2017.7939187>
56. Han QW (2015) Optimization and visualization of multiple 3D container loading problem with non-identical items. Dalian University of Technology

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.