Multi-strategy hybrid heuristic algorithm for single container weakly heterogeneous loading problem

Multi-strategy hybrid heuristic algorithm for single container weakly heterogeneous loading problem

Dezhen Zhang[a], Chenhao Gu[a], Hui Fang[b], Chengtao Ji[c], Xiuguo Zhang[a]

[a]*College of information science and technology, Dalian Maritime University, Dalian, China*
[b]*Department of Computer Science, Loughborough University, England, United Kingdom LE11 3TU*
[c]*Department of Computing, School of Advanced Technology, Xi'an Jiaotong-Liverpool University, Suzhou, China.*

Acknowledgement

This work is funded by the National Natural Science Foundation of China (Grant Number 51579025), and the Natural Science Foundation of Liaoning Province of China (Grant Number 201602089).

*Corresponding author
Email addresses: dezhen@dlmu.edu.cn (Dezhen Zhang), h.fang@lboro.ac.uk (Hui Fang)

Abstract

Three-dimensional single container weakly heterogeneous loading problem is one of the most classical tasks which has various applications in manufacture and logistics industry. Solving this problem could improve transportation efficiency to bring great benefit to shipping customers. During the last two decades, many heuristic, meta-heuristic and hybrid algorithms have been proposed to maximize container volume utilization to reduce the waste of container space significantly. Despite their success in many real-world applications, it is still a challenging task to recommend satisfactory loading levels within a limited time frame when clients approach for options of different combinations of shipping items. In this paper, we propose a novel multi-strategy hybrid heuristic algorithm to achieve timely planning for clients in a required short time frame. In specific, a probabilistic model is used to combine the strength of two optimization strategies, i.e. an ant colony method and a constructive greedy method, to speed up the optimization process and ensure better convergence. In addition, a tree pruning strategy is designed to further improve the efficiency of the hybrid heuristic algorithm. Extensive experiments demonstrate the effectiveness of our method in terms of both volume utilization rate and algorithm processing speed compared to state-of-the-art methods. Based on the comparison results by using BR dataset, we achieved averagely 94.31% volume utilization rate and 50.16 seconds processing speed, which is the best performance by considering both algorithm effectiveness and efficiency. Further, our proposed method has been deployed in a real business case to provide plan solutions to individual customer ship-

ping requests and achieved high customer satisfaction rate.
Keywords: 3D-CLP, weakly heterogeneous single container loading problem, ant colony algorithm, constructive heuristic algorithm, multi-strategy optimization.

1. Introduction

Three-dimensional container loading problem (3D-CLP) attracts significant attentions because of its various applications in manufacture and logistics industry. 3D-CLP searches for solutions to arrange cargo layout in containers so that container space utilization is maximized, thereby minimizing the number of required trips across the global container transportation system (Zhoujing et al., 2008). Among different types of 3D-CLP, weakly heterogenous single container problem refers to loading many items which belong to a few item types to one type of containers (Gehring and ortfeldt, 1997). It has become one of the most classical tasks as it addresses key shipping concerns from logistics customers. Since the use of containers to load cargoes has been the most popular practice in sea and rail transportation, solving this weakly heterogenous problem under different constraints improves loading efficiency, thus serving mutual interests of wide range of stockholders, e.g., logistics companies, logistics customers, traders and environment agencies.

During these couple of decades, many optimization algorithms have been proposed to solve the 3D-CLP problem. These algorithms can be categorized into conventional heuristics, tree-search-based and meta-heuristics methods

(Fanslau and Bortfeldt, 2010). Conventional heuristics and tree-search-based methods rely on either human prior knowledge or local greedy search algorithms to maximize container volume utilization (Liu et al., 2014; Araya et al., 2017). In contrast, meta-heuristic algorithms introduce stochastic process in their search strategy for better convergence (Dereli and Das, 2011; Bayraktar et al., 2021). Recently, there are some hybrid algorithms introduced to further improve the loading performance (Saraiva et al., 2019; Ozcan and Evren, 2021). Despite high container volume utilization rate reported by many advanced algorithms, it is still a challenging task to achieve satisfactory loading levels within a limited time frame. In a real-world scenario, when a client approaches a logistics company for recommendations of different combinations of shipping items, it is expected to provide adaptive shipping options with high loading levels based on requests from customers within a limited processing time.

In this paper, to solve the above-mentioned problem, we propose a novel multi-strategy hybrid heuristic algorithm to improve single container weakly heterogeneous 3D-CLP optimization processing speed while ensuring better convergence compared to state-of-the-art 3D-CLP methods. In specific, we design a probabilistic model to control selections of a greedy stacking method (GSM) and an ant colony stacking method (ASM). At the early stage, the model generates high probability to select a GSM model to enhance pheromone accumulation of ASM. This strategy speeds up the processing of ASM significantly. While at the late stage, unlike the two-phase algorithms, our method still allows to reuse GSM to further fine-tune the solutions obtained from ASM by utilizing local greedy search. In addition,

the probabilistic model introduce randomness in the heuristic process which further enhance the algorithm reliability via an ensemble effect. This design makes the two heuristic strategies to complement with each other to achieve a synergy effect for optimization. Further, we build a tree-pruning matrix to narrow the searching space. In the pruning matrix, each state value corresponding to one loading chain. It reduces the feasible solution space of the problem, thereby effectively reducing the program operation efficiency. By using the proposed multi-strategy hybrid algorithm, both high container volume utilization rate and fast processing speed are achieved to improve the user experience of logistics customers.

We highlight the contributions of our work as: (i) we design a novel heuristic method to tackle one of the most classical single container weakly heterogeneous 3D-CLP. To our best knowledge, this is the first work to deploy probabilistic model to explore the complementary advantages of a heuristic and a meta-heuristic method during the entire process of the optimization; (ii) we introduce a pruning matrix in the optimization to reduce the searching space for fast process without compromising the level of container utilization; and (iii) we conduct comprehensive experiments, including a case study to optimize the logistics of a bicycle factory and quantitative comparison with state-of-the-art methods to demonstrate that the proposed method is superior in terms of both volume utilization rate and algorithm processing speed.

The remainder of the paper is structured as follows. Section 2 present the most related work of our paper. Section 3 provides the problem statement and preliminary defined in our work. Section 4 describes the proposed multi-strategy hybrid heuristic algorithm in detail. In Section 5, experimen-

tal results demonstrate the effectiveness of our proposed method. Finally, Section 6 draws the conclusion and discusses future work.

2. Related work

Both conventional heuristics and tree-search-based methods can be categorized as constructive heuristics to maximize container volume utilization. Pisinger et al. (Pisinger, 2002) proposed a wall-building based method to decompose container structure into a number of layers which are further split into a number of strips where the packing of a strip is formulated as a Knapsack Problem. Inspired by an old prover 'Gold corner, silver side and strawy void', Huang et al. (Huang and He, 2009) proposed a caving degree approach (CDA) to pack items into corner as its priority so that items are close to each other to reduce the space waste. He et al. (He and Huang, 2011) made improvements on the CDA and proposed a fit degree algorithm (FDA) to further narrow the searching space in CDA. Egeblad et al. (Egeblad and Pisinger, 2009) introduced an abstract representation of box placement, namely sequence triple, for the three-dimensional knapsack problem. In each iteration, the sequence triple is transformed to a packing solution to evaluate its objective value. Fanslau et al. (Fanslau and Bortfeldt, 2010) presented a tree search algorithm for the 3DCLP, where a special form of tree search to ensure a balance between its search range and diversity. Liu et al. (Liu et al., 2014) presented a binary tree search algorithm, where each tree node denotes a container loading plan. BSG (Araya and Riff, 2014; Araya et al., 2017) is another tree-search-based method based on beam search (Norvig, 1992).

Meta-heuristic algorithms, as stochastic search methods, are dominant in solving the 3D-CLP problem. Among them, evolutionary optimization algorithms, such as Tabu search (Lodi et al., 2002; Liu et al., 2011), genetic algorithm (Gehring and ortfeldt, 1997; Bortfeldt and Gehring, 2001; Kang et al., 2012; Jamrus and Chien, 2016),Bee Colony algorithm (Dereli and Das, 2011; Bayraktar et al., 2021) and differential evolution (Li and Zhang, 2015), are popular methods to tackle the loading task. Kang et al. (Kang et al., 2012) presented the Improved Deepest Bottom Left with Fill (I-DBLF) algorithm to utilize a hybrid genetic algorithm to solve the 3D bin packing problem (3D-BPP). Jamrus (Jamrus and Chien, 2016) developed an extended priority-based hybrid genetic algorithm (EP-HGA) for the present LCL problem to determine the loading patterns. Dereli (Dereli and Das, 2011) designed a bee(s) algorithm by hybridizing a heuristic filling procedure to work with discrete variables, and used different operators to reach neighborhood solutions. T. Bayraktar (Bayraktar et al., 2021) proposed a memory-integrated ABC algorithm to meticulously select useful search steps in local search, and a genetic operator-based ABC algorithm to intelligently generate the next search steps in global search inspired by efficient solutions. Moreover, a joint hybrid ABC algorithm with reinforcement approach was developed to provide effective solutions for single container loading problem sets with large number of constraints. However, compared to conventional heuristics, these meta-heuristic methods are more time consuming due to their nature of the searching process.

Many latest algorithms are designed by introducing stronger constraints, such as limited category number (do Nascimento et al., 2021), to further

improve the optimization performance. Ozcan et al. (Ozcan and Evren, 2021) developed a Filltype method, which enforced strong prior knowledge on ranking orders for loading and applied an INLP model to determine box sizes and box positions by using the layer-building approach. Oliviana et al. (do Nascimento et al., 2021) presented an exact approach that defined twelve practical constraints when linear programming is iteratively used to solve the packing in planes problem. Guillem et al. (Bonet Filella et al., 2021) proposed a mixed-integer linear programming for the MDCLP with soft unloading constraints. Although these methods are typically operated in a time-efficient manner, they converge to sub-optimal solutions due to the non-convex optimization searching space.

Multi-strategy methods are used for the optimal design in recent years. In (Peng et al., 2021b), Peng et al. proposed a multiple strategy serial framework to improve cuckoo search algorithm to avoid the local optimum due to its original unitary search strategy. In (Abbasi et al., 2021), Abbasi et al. introduced a chaotic initialization approach and several updated besiege strategies to improve Harris Hawk optimization (HHO) method for the optimal design of tapered roller bearings. Similarly, in (Li et al., 2021a), Li et al. presented to use logarithmic spiral and opposition-based learning and a local search technique to improve the global convergence of HHO. In (Peng et al., 2021a), Peng et al. considered a dynamic adaptive selection and different mutation and crossover strategies to present co-evolutionary differential evolution method for mixed-variable optimization. While in (Li et al., 2021b), Li et al. extend the differential evolution method with multi-population and ensemble of mutation strategies to improve its convergence. In (Lu et al.,

2021), Lu et al. presented the Niching method by deploying population-entropy, distribution-radius, and utility-fitness to achieve a better trade-off between exploitation and exploration for the optimization. In (Chen et al., 2021), Chen et al. designed pheromone fusion and pheromone recombination strategy to improve the convergence of ant colony algorithm.

We summarize the related work in a comprehensive appendix Table A1 shown as appendix.

3. Problem statement and Preliminary

Weakly heterogeneous single container loading problem is the most classical task in 3D-CLP. In our work, assume a list of loading cargoes which belong to a few cargo types is given by a shipping client, our proposed heuristic method aims to provide an optimal loading plan to maximize container space utilization in a short time period. It is worth to note that our work is built with the following assumptions:

- The shape of both containers and cargoes is cuboid, and the size of each cargo is smaller than the size of a container;
- The center of gravity of a cargo is its geometric center;
- The capacity of each cargo is not considered in our heuristic algorithm;
- The deformation of a cargo is not considered in our heuristic algorithm;

3.1. Notation Definition

We summarize all the variables and abbreviations which are used in our paper for clarification as follows:

N:	The total quantity of cargoes to be loaded;
n_k:	The loading quantity of class k cargoes in the range of $[n_{kmin}, n_{kmax}]$;
u_k:	The number of stacking layers of class k cargoes;
v_i:	The volume of each cargo i;
K:	Number of cargo types;
M:	Maximum number of iterations;
U_i:	Binary flag for cargo loading. If cargo i is loaded into the container, then $U_i = 1$, otherwise $U_i = 0$;
L, W, H:	The length, width and height of the container;
l_i, w_i, h_i:	The length, width and height of cargo i;
$P_i^T\left(x_i^T, y_i^T, z_i^T\right)$:	The right-front-upper corner of cargo i in the container and its coordinates;
$P_i^B\left(x_i^B, y_i^B, z_i^B\right)$:	The left-back-bottom corner point of cargo i in the container and its coordinates;
s:	Status value, which is a binary number generated by the type of cargoes loaded and the quantity of each type of cargoes;
$S_\Pi(i, p)$:	The projected area of the cargo i on the cargo p along the $-z$ axis on the XOY plane.

3.2. Practical constraints

As a loading task in a real-world application, we face several practical constraints which are defined as follows:

Quantity Constraint The actual loading quantity of each type of cargoes cannot exceed the given quantity range of this type of cargoes.

$$n_{kmin} \leq n_k \leq n_{kmax}, 1 \leq k \leq K \tag{1}$$

Direction Constraint Cargo placement direction is parallel or orthog-

onal to container surface on both sides of the container. The six placement directions are expressed with the follow equations:

$$l_i//H\&w_i//L\&h_i//W \Rightarrow z_i^T - z_i^B = l_i, x_i^T - x_i^B = w_i, y_i^T - y_i^B = h_i (2.1)$$
$$l_i//L\&w_i//H\&h_i//W \Rightarrow x_i^T - x_i^B = l_i, z_i^T - z_i^B = w_i, y_i^T - y_i^B = h_i (2.2)$$
$$l_i//W\&w_i//L\&h_i//H \Rightarrow y_i^T - y_i^B = l_i, x_i^T - x_i^B = w_i, z_i^T - z_i^B = h_i (2.3)$$
$$l_i//H\&w_i//W\&h_i//L \Rightarrow z_i^T - z_i^B = l_i, y_i^T - y_i^B = w_i, x_i^T - x_i^B = h_i (2.4)$$
$$l_i//L\&w_i//W\&h_i//H \Rightarrow x_i^T - x_i^B = l_i, y_i^T - y_i^B = w_i, z_i^T - z_i^B = h_i (2.5)$$
$$l_i//W\&w_i//H\&h_i//L \Rightarrow y_i^T - y_i^B = l_i, z_i^T - z_i^B = w_i, x_i^T - x_i^B = h_i (2.6)$$

Taking Eqn.2.5 as an example, the right-front-upper corner coordinate cargo i, the left-back-lower corner coordinate of cargo i, the positional relationship between it and the container coordinate are illustrated in Fig. 1:

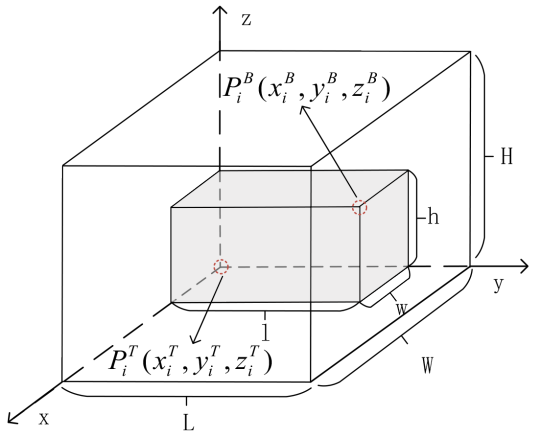


Fig. 1. Schematic diagram of cargo placement direction and marking points.

Place constraint The cargo must be placed on a solid surface, i.e. the bottom surface of the cargo on an upper layer must be in contact with the

top surface of the cargo on a lower layer. This relationship is expressed in the follow equations.

$$\sum S_{\Pi}\left(i,p\right)=\left(x_i^T-x_i^B\right)\times\left(y_i^T-y_i^B\right)$$

$$\begin{aligned}S_{\Pi}(i,p)=&(x_i^T-x_i^B+x_p^T-x_p^B-max(x_p^T,x_i^T)+min(x_p^B,x_i^B))\times\\&(y_i^T-y_i^B+y_p^T-y_p^B-max(y_p^T,y_i^T)+min(y_p^B,y_i^B))\end{aligned}$$

$$\begin{aligned}p\in\{p|U_p=U_i=l,z_p^T=z_i^B,x_p^B\leq x_i^B,x_p^T\geq x_i^T,y_p^B\leq y_i^B,y_p^T\geq y_i^T\}\cup\\\{p|U_p=U_i=1,z_p^T=z_i^B,max(x_p^T,x_i^T)-min(x_p^B,x_i^B)<x_i^T-x_i^B+x_p^T-x_p^B,\\max(y_p^T,y_i^T)-min(y_p^B,y_i^B)<y_i^T-y_i^B+y_p^T-y_p^B\}\end{aligned}$$

where $minx_i^B\geq 0$,$miny_i^B\geq 0$,$minz_i^B\geq 0$,$minx_i^T\geq 0$,$miny_i^T\geq 0$,$minz_i^T\geq 0$

Volume Constraint The total volume of the actual loaded cargo cannot be greater than the maximum loading volume of the container.

$$\sum_{i=1}^{N}l_i\times w_i\times l_i\leq L\times W\times H \tag{2}$$

Layer Constraint The actual number of longitudinal layers of the loaded cargo cannot exceed the maximum number of stacking layers of the lowest layer of cargo.

$$u_k\leq u_{kmax},l\leq k\leq n \tag{3}$$

Our proposed method generates an optimal loading plan by satisfying all the above mentioned constraints.

3.3. Objective function

The optimization goal of the single container weakly heterogeneous 3D-CLP is to minimize the idle space of containers when loading cargoes under certain constraints. In our work, we aim to maximize the objective function, i.e. volume utilization rate (VUR), as fast as possible when dealing with requests from logistics customers. The objective function is expressed as follows:

$$maxV = \frac{\sum_{i=1}^{N} l_i \times w_i \times h_i \times U_i}{L \times W \times H} \tag{4}$$

3.4. Spatial model

The cargo selection method is based on the relative size of the cargo volume, the current space to be loaded and the order of placement. After each selected cargo is placed in the space to be loaded, it makes structural changes on the space layout. The three-space division method (Bischoff, 2006) is used as the basis for space decomposition and comprehensive utilization. The spaces generated during loading are illustrated in the Fig. 2. They are defined as:

Definition 1. Unloaded space (VU) refers to the space where cargoes can be loaded at the current loading layer.

Definition 2. Remaining space (VR) refers to the space that can be used or combined for further loading at the current layer.

Definition 3. Idle space (VI) refers to the space where remaining cargoes can be loaded in future loading layers.

In the loading process, when filling the VU, the cargoes are placed in an order from bottom to top (z-axis direction), from left to right (y-axis

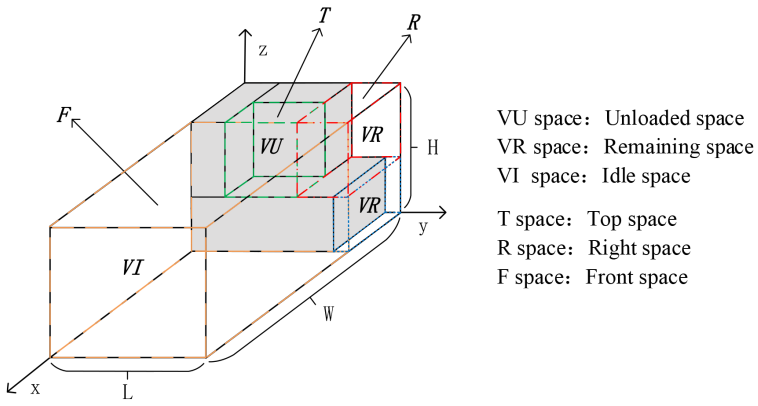


Fig. 2. Schematic diagram of space definition.

direction), and from back to front (x-axis direction). After filling each layer space on the YOZ plane, cargoes are placed on the next layer in VI until the container is full.

3.5. Remaining space merging algorithm

With continuous loading process of the container, more and more irregular "scattered" spaces are produced in different positions. Space merging refer to merging the small remaining spaces, and process them into usable rectangles, which is convenient to put more cargoes in the subsequent loading process, so as to improve the utilization rate of space. An example of remaining spaces $VR1$,$VR2$,$VR3$ before merging is illustrated in Fig. 3.

In our work, we utilize the following space merging rules to convert scattered small spaces into reusable spaces:

(1) Merge left and right remaining spaces: It is necessary to ensure that the right side of the remaining space $VR1$ on the left includes the left side of the remaining space $VR2$ on the right, under the premise of $\forall i \in N, i <$

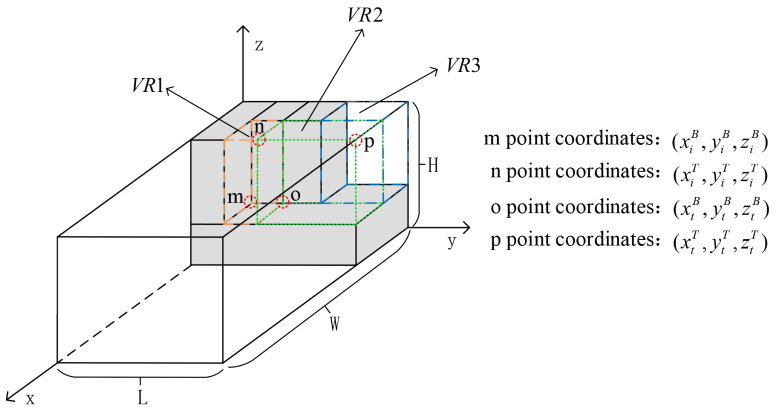


Fig. 3. Schematic diagram of remaining space.

T,$x_t^B + x_t^T - x_i^B - x_i^T < x_t^T - x_t^B + x_i^T - x_i^B$ and $z_t^B + z_t^T - z_i^B - z_i^T < z_t^T - z_t^B + z_i^T - z_i^B$ and $y_t^B = y_i^T$

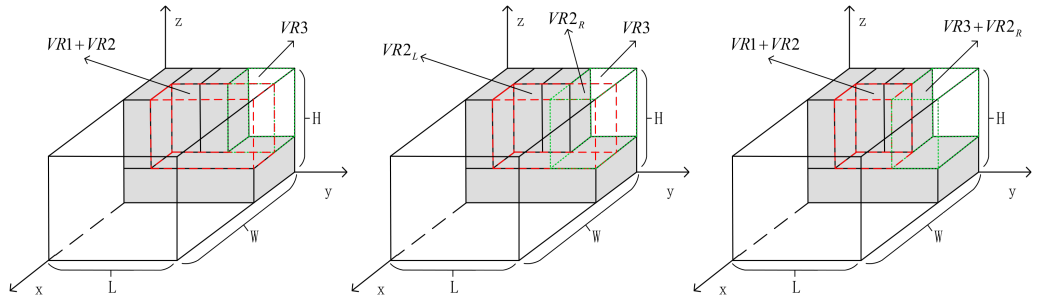
$$Y_{min} = min\left(y_t^B, y_i^B\right) \tag{5}$$

where the value of y_i^B is located in the remaining space $VR1$, the value of y_t^B is located in the remaining space $VR2$, Y_{min} stands for the minimum value of the remaining space in y coordinate after each space merging.

According to the selected Y_{min}, expand the space to the left and update the information in the remaining space Table. Fig. 4(a) illustrates the merging operation of a left remaining space and a right remaining space.

(2) Merge front and rear remaining spaces:Decompose the remaining space $VR2$ into two sub-spaces $VR2_L, VR2_R$ along the right side of box, as shown in Fig. 4(b).

Ensure that the back side of the remaining space $VR2_R$ on the front includes the front side of the remaining space $VR3$ on the rear, under the premise of $\forall i \in N, i < T$, $y_t^B + y_t^T - y_i^B - y_i^T < y_t^T - y_t^B + y_i^T - y_i^B$ and



(a) Merging left and right (b) Merging front and rear (c) The final merging result

Fig. 4. Illustration of the remaining spaces merging process.

$$z_t^B + z_t^T - z_i^B - z_i^T < z_t^T - z_t^B + z_i^T - z_i^B \text{ and } x_t^B = x_i^T$$

$$X_{min} = min\left(x_t^B, x_i^B\right) \tag{6}$$

where the value of x_i^B is located in the remaining space $VR3$, the value of x_t^B is located in the remaining space $VR2_R$, X_{min} stands for the minimum value of the remaining space in x coordinate after each space merging.

According to the selected X_{min}, expand the space to the rear and update the information in the remaining space table. After merging the front and rear spaces, the final effect is illustrated in Fig. 4(c).

4. Multi-strategy hybrid heuristic algorithm

Our multi-strategy hybrid heuristic algorithm is built on an ant colony stacking method (ASM). Classical ant colony algorithm shows great performance when dealing with extremely large and non-convex searching space. However, it has two well-known limitations: (i) at the early optimization

stage, ant colony algorithm has to take time to accumulate pheromone information to guide its convergence which slows down the optimization process significantly; and (ii) it misses the refinement to achieve the optimal solution although its design avoids the convergence at local minima. Therefore, in our method, we propose to utilize a probabilistic model to integrate a greedy stacking method (GSM) with the ant colony method to improve the convergence efficiency as well as the loading capacity via local refinement. With the help of the GSM, the pheromone information is accumulated much faster than a standard ASM method at the early heuristic searching stage. Furthermore, the GSM uses a local greedy strategy to refine the solutions from the ASM to achieve better convergence. Further, the probabilistic model introduces randomness during the heuristic process that behaves an ensemble model effect in the optimization. This innovative combination of ASM and GSM via a probabilistic model outperforms other two-phase methods and hybrid methods as the advantages of these two algorithms are fully explored during the entire heuristic searching process.

In this section, we explain the detail of our proposed method in five subsections: in Subsection 4.1, we define the searching space of the 3D-CLP with the constraints we face in a real-world application. In Subsection 4.2, we introduce our ASM heuristic strategy and its designed pheromone state. Following it, we present the GSM heuristic strategy in Subsection 4.3. Further, the proposed selection strategy which is the key to integrate ASM and GSM is explained in Subsection 4.4. Finally, we also propose a pruning matrix strategy to further speed up the heuristic process in Subsection 4.5.

4.1. Loading plan solution (searching) space definition

Before we design a heuristic method to optimize the loading plan, a searching space is defined to clarify how heuristic strategies are proposed to find an optimal loading solution from this non-convex space. For each loading solution, it is composed of numbers and types of loading cargoes as well as their loading sequence.

When a cargo item is loaded in a loading sequence, there are 6 options to add this item under the condition of satisfying the above-defined **direction constraint** and **place constraint**. These 6 options can be formalized in the following equation:

$$[x_i'^T, y_i'^T, z_i'^T] = [x_i^T, y_i^T, z_i^T] \times \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\alpha & -sin\alpha \\ 0 & sin\alpha & cos\alpha \end{bmatrix} \times \begin{bmatrix} cos\beta & 0 & sin\beta \\ 0 & 1 & 0 \\ -sin\beta & 0 & cos\beta \end{bmatrix} \times \begin{bmatrix} cos\gamma & -sin\gamma & 0 \\ sin\gamma & cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

$$\text{And} \begin{cases} \alpha = \{0°, 90°\}, \beta = 0°, \gamma = 0° \\ \alpha = \{0°, 90°\}, \beta = 90°, \gamma = 90° \\ \alpha = \{0°, 90°\}, \beta = 0°, \gamma = 90° \end{cases}.$$

where (x_i^T, y_i^T, z_i^T) refers the initial right front upper corner of the cargo item i and $(x_i'^T, y_i'^T, z_i'^T)$ is the corresponding corner after changing the posture of the item. Since the axis of the cargo body is parallel to the axis of the container, $\alpha, \beta, \gamma \in \{0°, 90°\}$.

In our work, during the placement process, it is realized that the generated remaining spaces have more regular shape when the width in the placement layer is considered as the highest priority. Simply put, the generated spaces

are easily reused to add more items. Therefore, under this circumstance with the priority for the local loading, we could convert this 3D placement problem into a 2D form. This assumption reduces the dimension of the searching space significantly without sacrificing the loading performance. As illustrated in Fig. 5, the cargo and the space to be loaded are consequently projected on the YOZ plane.

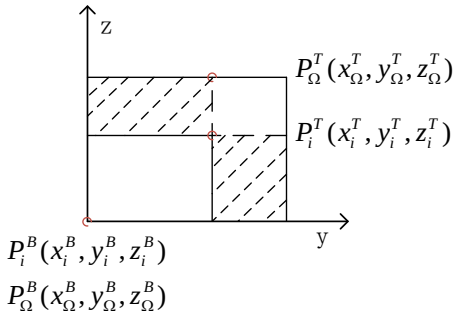


Fig. 5. Projection diagram of cargo and space to be loaded.

When we add new item into the remaining space, we design a local loading strategy which minimizes the sum of the remaining scale areas in the height direction and width direction (presented as the shading area in Fig. 5). Consequently, the local space utilization rate is maximized.

$$\begin{aligned} min\{&[((P_\Omega^T(y_\Omega^T) - P_\Omega^B(y_\Omega^B))\%(P_i^T(y_i^T) - P_i^B(y_i^B)))] * (P_i^T(z_i^T) - P_i^B(z_i^B)) \\ &+[((P_\Omega^T(z_\Omega^T) - P_\Omega^B(z_\Omega^B))\%(P_i^T(z_i^T) - P_i^B(z_i^B)))] * (P_i^T(y_i^T) - P_i^B(y_i^B))\} \end{aligned} \tag{8}$$

where, $P_\Omega^T(x_\Omega^T, y_\Omega^T, z_\Omega^T)$ and $P_\Omega^B(x_\Omega^B, y_\Omega^B, z_\Omega^B)$ are the upper right and lower left corner coordinates of the current space to be loaded. $P_i^T(x_i^T, y_i^T, z_i^T)$ and

$P_i^B(x_i^B, y_i^B, z_i^B)$ is the upper right and lower left corner coordinates of the loading cargo i respectively.

4.2. ASM strategy and its pheromone state

In each searching iteration, ant colony algorithm identifies numbers and types of cargoes, and their loading orders via using the pheromone information. After comparing different solution options at each iteration, the pheromone is updated for future usage. Therefore, the algorithm complexity heavily relies on the design of the pheromone state. For the key code state design, we propose a new state word representation. The formal representation of the state word is expressed as,

$$S_a^k(T, Q) = [t_1 t_2..t_k..t_K | q_1^Q(t_1) q_2^Q(t_2)..q_k^Q(t_k)..q_K^Q(t_K)] \tag{9}$$

where k is the number of cargoes to be loaded and a is the a ant.

$t_k = \{0, 1\}$ indicates whether the k-th cargo is loaded, 1 indicates loaded, and 0 indicates not loaded.

$q_k^Q(t_k) = [0..0]$ is a Q-digit consisting of 0 or 1 binary code, which indicates the loading quantity of the k-th kind of goods, and the maximum loading quantity is $2^Q - 1$ cargoes.

The key code state is illustrated in Fig. 6 and the structure of the status word is shown as:

In the ASM, we use a key-value tree structure for the pheromone storage. In specific, the $K * (1 + Q)$-bit binary code is used as the *key* value in the tree structure, and the pheromone states are saved as *value*. Noting that the relationship between keys and values is 1-to-n. The *value* of pheromone is

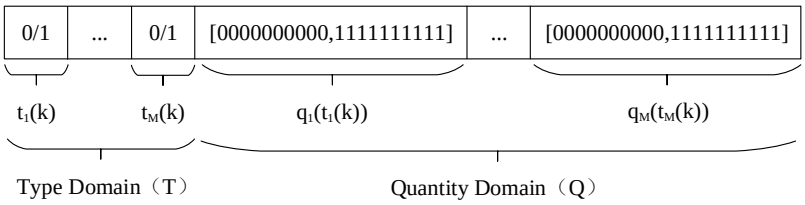


Fig. 6. State word representation.

formalized as follows:

$$value_i(S_a^k(T,Q)) = [k, d, x_i^B, y_i^B, z_i^B, x_i^T, y_i^T, z_i^T, VR_i] \tag{10}$$

where i represents the i_{th} current loading cargo, k represents the current loading cargo type, d represents the loading direction of the cargo, x_i^B, y_i^B, z_i^B are the left bottom corner coordinate of the cargo, x_i^T, y_i^T, z_i^T are the right top corner coordinate of the cargo after the posture adjustment, and VR_i is the remaining space of current layer after the loading. To ensure that, in the worst case, the time complexity of query, insert or delete is $O(logn)$, we utilize the red black tree (Bose et al., 2012) to access the loading information stored in the tree. Because of the exponential growth of the key-value pairs, the feasible solution space is untraceable. Despite its slow process, the ASM promises the convergence of the algorithm.

4.3. GSM strategy

GSM is a greedy heuristics method which has the advantage to converge to local minima in an extremely fast manner (He et al., 2012). Although it cannot guarantee global optimal solution convergence, it is a key element to speed up the ASM convergence as well as refine the ASM solutions in our

proposed method. Different to classical GSM used in (Zhu and Lim, 2012), we also update the pheromone information for the proposed ASM strategy by using the GSM strategy. The following procedure is the heuristic process in our GSM strategy:

Step 1: Initialize the pheromone of each node in the ASM.

Step 2: Judge whether the current space is a new layer. If yes, select a cargo with the largest volume and place it by using the loading heuristic algorithm which is explained in Section 4.1; if not, go directly to Step 3.

Step 3: Determine the space to be loaded based on the order of top-right-front of the previous placement of a cargo, and determine the space to be loaded based on the remaining space consolidation algorithm which is explained in the Section 3.5.

Step 4: Select a cargo with the largest volume that can be loaded in the space, place it by using the loading heuristic algorithm which is explained in Section 4.1, and update the pheromone of the current node of ASM;

Step 5: Repeat step 2 to step 4 until no remaining space is left in this layer.

We further provide an example to illustrate the procedure: considering the front cargo i-1 of the cargo i currently to be loaded. If $v_i + \triangle \leq v_{i-1}$, $\triangle \in [0, C]$ is a constant, then the placement rule is,

① If $min\{l_i, w_i, h_i\} \leq H - max\left(z_j^T\right)$, placed in the +Z direction;

② If $min\{l_i, w_i, h_i\} \leq H - max\left(y_j^T\right)$,place in the +Y direction;

③ If $min\{l_i, w_i, h_i\} \leq H - max\left(x_j^T\right)$, place it in the +X direction.

This GSM strategy ensures that the current space is loaded is as full as possible, and the loaded cargoes are as regular as possible.

4.4. Selection strategy

We design a selection probabilistic rule to switch between the GSM and ASM for the heuristic search. The probability values are generated based on the difference between a uniform random distribution function and a normal distribution function. For example, at current iteration τ, a random number between 0 and RAND_MAX is generated where RAND_MAX is the maximum value among all random numbers. For the convenience of comparison, it is normalized by taking the modulus form, which is expressed as the following equation:

$$R(\tau) = rand()\%RAND\_MAX \tag{11}$$

in which $R(\tau) \in [0, 1]$. With respect to the normal distribution function $P(\tau)$, we take the partial distribution on the right side of the symmetry axis that is greater than zero, and normalize it, which is expressed as the following equation:

$$P(\tau) = \frac{\int_0^M \frac{1}{\sqrt{2\Pi}\sigma} e^{-\frac{(k-\mu)^2}{2\sigma}} d(\tau)}{(k \times \frac{1}{\sqrt{2\Pi}\sigma})} \tag{12}$$

where τ is the current number of iterations, M is the maximum number of iterations, then $P(\tau) \in [0, 1]$. In particular, let σ be 1, $\tau = \mu$, and be a standard normal distribution. When $R(\tau) - P(\tau) \leq 0$, the GSM is used to select cargoes. While when $R(\tau) - P(\tau) > 0$, the ASM is used for the heuristic search.

The generated probabilistic values conform to the exponential distribution law perfectly which is illustrated in Fig. 7. It can be seen that, on the one hand, at the early stage of the algorithm, the probability of using

GSM to select cargoes is high (the range below the horizontal axis) while there is still a certain probability to use ASM (the range above the horizontal axis). On the other hand, with the increase of the number of iterations, the probability of using ASM algorithm increases gradually while there is still the probability of using GSM to select cargoes. This ensures that the two strategies complement with each other to improve the heuristic efficiency and better convergence.

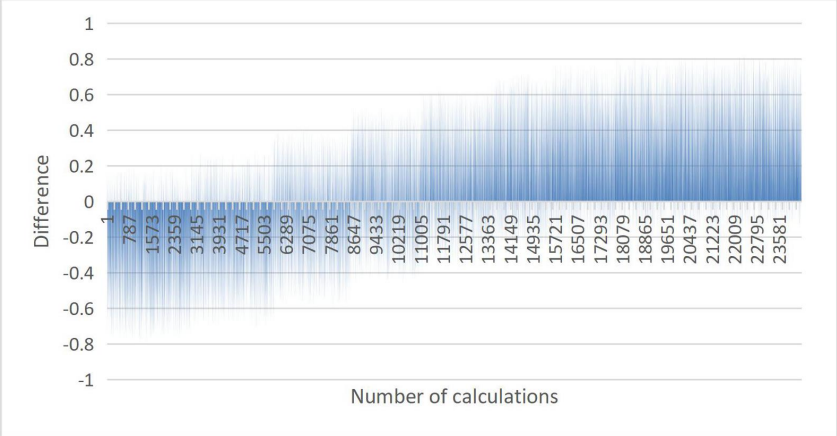


Fig. 7. Schematic diagram of dynamic hybrid of two selection methods.

The procedure of our selection strategy is shown in the following steps:

Step 1: Determine whether the current node pheromone concentration reaches the effective range, if so, use the current node pheromone; if not, initialize the current node pheromone concentration;

Step 2: Judge whether the current space is a newly opened layer, if yes, select a cargo with the largest volume that can be loaded, place it according to the heuristic algorithm for determining the direction of the cargo in 4.1, and proceed to the next step; if not, go to step 3;

Step 3: Determine the space to be loaded according to the order of top-right-front of the previous placement of the cargoes, and determine the space to be loaded according to the 3.5 remaining space consolidation algorithm;

Step 4: Determine the cargoes to be loaded according to the node pheromone concentration and the sequence of operators, and place them according to the heuristic algorithm for determining the direction of the cargoes in 4.1;

Step 5: Update the pheromone concentration and repeat steps 2 to 5 until the current layer is remaining space that is too small to load any a box;

Step 6: Open a new layer to determine whether the current layer can load the cargoes. If it can, go to step 2; if not, add 1 to the ant number to update the pheromone concentration;

Step 7: Judge whether the number of ants reaches the maximum number of ants, if so, end the loading and output the optimal loading chain; otherwise, execute step 1.

4.5. Pruning matrix strategy (PMS)

Considering that some solutions produce a large amount of very small remaining space at the beginning of operation, it is obviously unable to achieve the approximate optimal solution required by the algorithm and can be discarded. Therefore, we propose an early stop criterion to determine whether the current search needs to be continued based on the current solution that have been obtained from the previous layer. The red-black tree established by querying the status value and the size of the idle space performs pruning operations.

Assume K is the number of types of cargoes, T_i^* is the corresponding optimal type domain after placing the i-th cargo, Q_i^* is the corresponding

optimal quantity domain, and VR^* is the corresponding remaining space.

If there is an ant a that places the i-th cargo in space, the corresponding type domain is $\bigcup_{i=1}^{K} T_{a_i},\ i \in [1, K]$ and $T_i^* = \bigcup_{i=1}^{K} T_{a_i}$, the corresponding quantity domain is $\sum_{i=1}^{K} Q_{a_i}$ and $Q_i^* = \sum_{i=1}^{K} Q_{a_i}$, and its remaining space is $VR(\langle T_{a_1}||T_{a_2}||..||T_{a_K}, \sum_{i=1}^{K} Q_{a_i}\rangle)$, abbreviated as $VR\left\{\langle \bigcup_{i=1}^{K} T_{a_i}, \sum_{i=1}^{K} Q_{a_i}\rangle\right\}$, and $VR^* = VR\left\{\langle \bigcup_{i=1}^{K} T_{a_i}, \sum_{i=1}^{K} Q_{a_i}\rangle\right\}$.

For the current ant b, the corresponding type domain is $\bigcup_{i=1}^{K} T_{b_i}$ and $i \in [1, K]$ after the placement at a certain time, the corresponding cargo number domain is $\sum_{i=1}^{K} Q_{b_i}$, and its remaining space is $VR\left\{\langle \bigcup_{i=1}^{K} T_{b_i}, \sum_{i=1}^{K} Q_{b_i}\rangle\right\}$.

When $T_{a_i} = T_{b_i}$ and $Q_{a_i} = Q_{b_i}$,

① If $VR\left\{\langle \bigcup_{i=1}^{K} T_{a_i}, \sum_{i=1}^{K} Q_{a_i}\rangle\right\} > \left\{\langle \bigcup_{i=1}^{K} T_{b_i}, \sum_{i=1}^{K} Q_{b_i}\rangle\right\}$ give up ant a, record ant b to load the chain,and set $T_i^* = \bigcup_{i=1}^{K} T_{b_i},\ Q_i^* = \sum_{i=1}^{K} Q_{b_i}$, $VR^* = VR\left\{\langle \bigcup_{i=1}^{K} T_{b_i}, \sum_{i=1}^{K} Q_{b_i}\rangle\right\}$.

② If $VR\left\{\langle \bigcup_{i=1}^{K} T_{a_i}, \sum_{i=1}^{K} Q_{a_i}\rangle\right\} \leq \left\{\langle \bigcup_{i=1}^{K} T_{b_i}, \sum_{i=1}^{K} Q_{b_i}\rangle\right\}$keep ant a.

③ In particular, for the first ant c, $T_i^* = \bigcup_{i=1}^{K} T_{c_i},\ Q_i^* = \sum_{i=1}^{K} Q_{c_i}$, $VR^* = VR\left\{\langle \bigcup_{i=1}^{K} T_{c_i}, \sum_{i=1}^{K} Q_{c_i}\rangle\right\}$.

When the state value is the same, the load chain is preferably stored and the pruning matrix is updated. In the pruning matrix, each state value corresponds to a current optimal load chain. The status value is only related to the type and quantity of the loaded goods. Each state value only saves one piece of optimal loading chain information, which effectively reduces the solution space of the problem.

To further clarify the processing flow and termination criteria of our proposed method, we illustrate its flow diagram in Fig. 8.

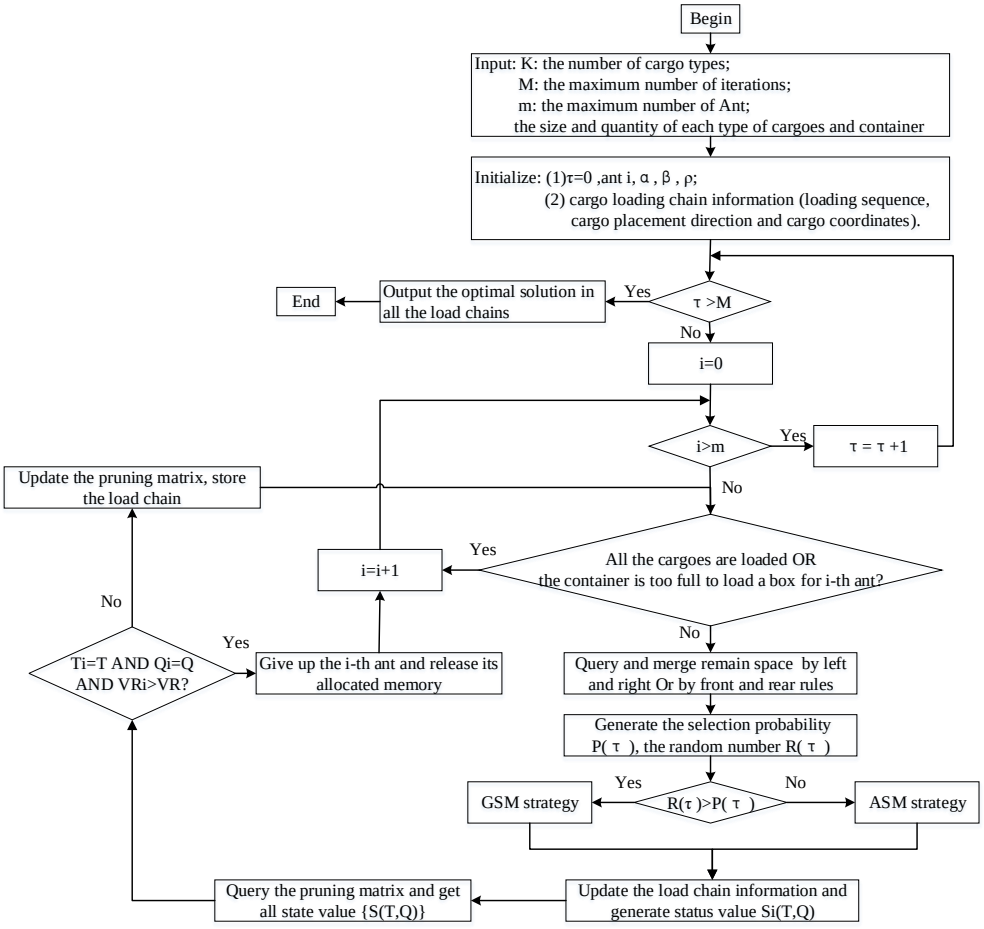


Fig. 8. Flow diagram of multi-strategy hybrid heuristic algorithm.

5. Computational Experiments

5.1. Experimental configurations and platform.

Our system is composed of data storage module, planning optimization module and simulation module. The experimental program runs on the Intel (R) core (TM) i5-6500 CPU @ 3.20GHz processor with 8G of memory. By

using Visual Studio C++ as the algorithm compiling environment, the loading results are output to the database, and the information in the database is read by using unity3D 5.5.0 to simulate the container loading process. The system framework architecture is shown in Fig. 9. The detail of these three modules are explained as follows.

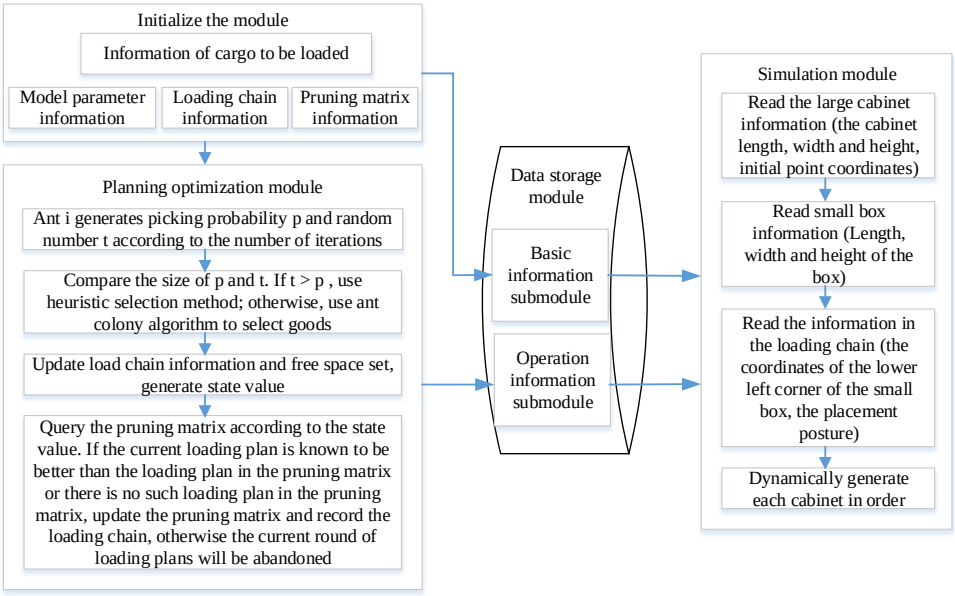


Fig. 9. Overview of system architecture.

(1) Data storage module: This module is divided into basic information submodule and operational output submodule. The basic information submodule is used to store the relevant information of containers and the cargo items to be loaded; the operational output submodule is used to store the output plans from the main optimization module, e.g., the information of the loading chain and the left rear lower angle coordinate information of

each cargo after loading.

(2) Planning optimization module: This module is mainly used for the key process of the proposed method. We provide the algorithm configuration parameters in the following Table 2. Among them, ρ, α and β are from the literature (Chen et al., 2021).

Table 2

Algorithm parameter values.

Parameter description	Parameter	Value
Volatile factor of pheromone	ρ	0.7
Heuristic factor of pheromone	α	1
Heuristic factor of the greedy rule	β	3
Number of ants	m	30
The maximum number of iterations	M	100

(3) Simulation module: the simulation module generates corresponding animations for visualization. After the loading plan, including container information, cargoes information and loading chain information are read, Unity3D script is written to simulate the dynamic loading. An example is illustrated in Fig. 12.

5.2. Comparison with state-of-the-art methods

A standard test data set (Bischoff and Ratcliff, 1995) is used in our work for performance comparison. This data set is composed of seven groups of 700 weakly heterogeneous examples. We set up eight algorithms as benchmarking methods, which include the layered greedy algorithm (H_BR) proposed by Bischoff E E et al. in (Bischoff and Ratcliff, 1995), the parallel genetic

algorithm (GA_GB) proposed by Gehring H et al. in (Gehring and ortfeldt, 1997), the Tabu Search Algorithm (TS_BG) proposed by Bortfeldt A et al. in (Bortfeldt and Gehring, 1998), the Hybrid Simulated Annealing Algorithm (HSA) proposed by Parreno et al. in (Parreño et al., 2008), the Greedy Random Adaptive Search Algorithm (GRASP) proposed by Moura A et al. in (Moura and Oliveira, 2005), the bee colony optimization algorithm (SOA) proposed by Zhou Qi-Feng et al. in (Zhou and Xiang, 2017), the multi-constraint heuristic algorithm (TS_CLP) proposed by Liu Sheng et al. in (Sheng et al., 2017), the tree search algorithm (PRTS) proposed by Wang Yan et al. in (Wang et al., 2019), the MDCLP-S strategy proposed by Guillem et al. in (Bonet Filella et al., 2021), JHABC algorithm proposed by T. Bayraktar et al. in (Bayraktar et al., 2021), and Gravitational Search Algorithm (GSA) proposed by Rashedi et al. in (Rashedi et al., 2009).

Table 3

Comparison of filling rate and time of various algorithms for weakly heterogeneous loading problem.

Test case	Fill rate(%)								Time(s)
	BR1	BR2	BR3	BR4	BR5	BR6	BR7	Avg+/-STD	Avg+/-STD
Box type	3	5	8	10	12	15	20	–	–
H_BR	83.79	84.44	83.94	83.71	83.80	82.44	82.01	83.50+/-3.33	180.00+/-0.00
GA_GB	85.80	87.26	88.10	88.04	87.86	87.85	87.68	87.50+/-2.42	180.00+/-0.00
TS_BG	87.81	89.40	90.48	90.63	90.73	92.72	90.65	90.10+/-2.25	121.00+/-41.98
HSA	93.81	93.94	93.86	93.57	93.22	92.72	91.99	93.23+/-1.98	65.87+/-27.83
GRASP	89.07	90.43	90.86	90.42	89.67	89.71	88.05	89.74+/-1.72	**33.71+/-14.85**
TS_CLP	90.62	91.51	92.43	92.35	92.45	92.37	92.13	91.98+/-1.67	125.30+/-18.74
SOA	92.67	93.19	93.44	93.21	92.94	92.70	92.31	92.92+/-1.76	121.87+/-17.36
PRTS	92.96	93.62	94.06	93.86	93.62	93.33	92.69	93.44+/-1.78	61.26+/-19.89
MDCLP-S	84.10	83.40	80.90	80.40	79.60	79.00	78.10	80.80+/-2.70	160.00+/-0.00
JHABC	84.42	83.30	86.84	85.26	84.17	89.33	85.87	85.60+/-1.59	147.88+/-22.58
GSA	94.63	93.16	91.41	90.28	88.61	87.52	85.24	90.12+/-2.39	67.20+/-16.12
Proposed	**95.33**	**94.87**	**94.76**	**94.43**	**93.94**	**93.52**	**93.36**	**94.31+/-1.75**	50.16+/-19.56

As shown in Table 3, it is found that: (1) when the types of cargoes increase, the container filling rate consistently drops for all the algorithms. It is understandable as the diversity of item shapes inevitably lead to an increase in the amount of remaining space, (2) compared to all other benchmarking methods, our proposed method achieves the best average filling rate which is 94.31% which is a significant improvement when the ceiling effect appears when using latest advanced optimization algorithms. and (3) w.r.t the efficiency, i.e. average loading time, the proposed method also performs well. When considering the filling rate above 90%, our proposed work in this paper is the most efficient method (running time: 50.16s) which outperforms the second HSA algorithm (65.87s), the third TS_CLP algorithm (125.3s). Although the average running time of the GRASP algorithm proposed by Moura A et al. is 33.71 seconds, the filling rate is about 4.57% lower than the result obtained by the algorithm in this paper, and the GRASP algorithm only considers a single constraint condition. Here the performance of MDCLP-S is the worst as it added a multi-drop unloading constraints. In addition, all the variations of fill rate and running time also confirm that the proposed method achieves comparable results with the methods which have the best performance.

To further analyze the variations of the proposed method due to the stochastic characteristic like all other evolutionary algorithms, we show the minimal and maximal loading capacity and running time in Table 4. In the table, the minimum, maximum and average values of the running time and filling rate of seven groups of test cases are given respectively. It is found that even with the worst cases with 20 box types, the algorithm still achieves

Table 4

Some test details of Multi Strategy dynamic hybrid optimization algorithm for weakly heterogeneous container loading.

Test case	Box type	Running time(s)				Fill rate(%)			
		Min	Max	Avg	STD	Min	Max	Avg	STD
BR1	3	1.54	44.02	6.80	12.44	89.76	97.76	95.33	1.96
BR2	5	2.19	46.08	19.86	10.54	91.30	97.36	94.87	1.55
BR3	8	2.92	67.21	47.76	16.07	90.87	96.12	94.76	1.48
BR4	10	3.86	74.36	48.95	16.86	90.17	97.61	94.43	1.74
BR5	12	6.20	101.33	66.31	22.20	89.56	96.03	93.94	1.77
BR6	15	8.81	131.52	76.10	27.63	90.32	96.58	93.52	1.74
BR7	20	12.09	150.60	85.37	31.21	88.64	95.71	93.36	1.99

very high filling rate with an acceptable running time frame.

In the Table 4, the standard variations of fill rate on all test cases range from 1.48 to 1.99 and these indicate a great convergence of the proposed method. In addition, the standard deviations of the running time increase from 12.44 in the simplest test case to 31.21 in the most complicated case. This trend reflects an objective observation that the time of convergence varies in more complicated cases.Overall, our proposed algorithm can guarantee both the effectiveness and efficiency.

5.3. Ablation study

To further evaluate and prove the effectiveness of the proposed algorithm, we conduct two ablation studies in our experiment. As depicted in Fig. 10, we compare the convergence of the GSM, ASM and our proposed model.

In the experiment, we run each algorithm with 10 times and plot the curves with average and standard deviation across iterations. Where, the upper and lower boundaries represent the 'minimal' and 'maximal' filling rates while the centre point represents the average filling rates of the 10 independent running times on each method. It is observed that the GSM method converges to local optimal solutions with small variations. While the ASM converges much slower with larger variations. Even after 100 iterations, it still has not reached the converged performance. In contrast, our proposed method combines the advantages of these two methods so that the method has strong exploration capability and ensures better convergence in limited iterations. These convergence curves prove both the effectiveness and efficiency of our method. As another element in our work, we design a pruning matrix to improve the convergence speed. To prove the importance of the pruning matrix strategy, we have made another ablation study by using the proposed method with and without the pruning matrix. The results are shown in Table 5. It can be observed that the pruning matrix plays a crucial role on improving the efficiency of our method.

Table 5

Comparison of filling rate and time of proposed algorithms with and without the pruning matrix.

Proposed	Fill rate(%)								Time(s)
	BR1	BR2	BR3	BR4	BR5	BR6	BR7	Avg	Avg
with PMS	95.33	94.87	94.76	94.43	93.94	93.52	93.36	94.31	50.16
w/o PMS	95.29	94.85	94.75	94.43	93.90	93.53	93.31	94.29	121.63

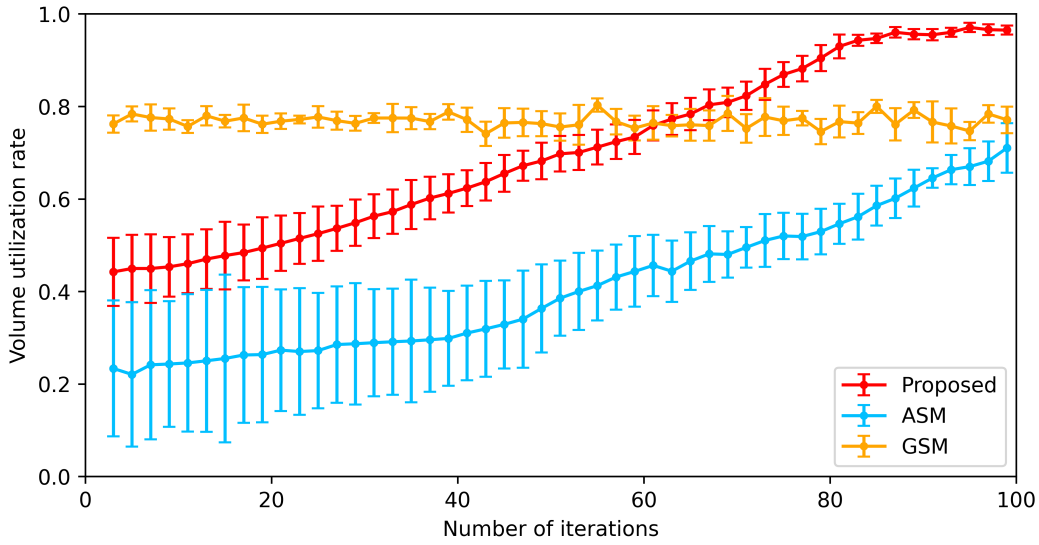


Fig. 10. Convergence curves of the three methods.

5.4. Case study

We further deploy our proposed method in a real business case for the logistic department of a bicycle factory in Tianjin, China. In the department, worldwide order requests are received to order different types of bicycles and ask for corresponding logistic plans. There are two main user requirements from the client: (1) replying customer requests in a timely response and (2) providing shipping animation for container loaders. In this case study, the container size is 40 feet (the size is $12.024m \times 2.35m \times 2.69m$) and 11 different item types are selected for shipping. The basic information of these types is shown in Table 6.

To satisfy the first user requirement, a website is built to reply customer requests by using the optimization engine driven by our proposed method. A screenshot of the website in production is illustrated in Fig. 11. When a

Table 6

Basic information of shipping items.

Number	Types	Length(m)	Width(m)	Height(m)	Volume(m^3)
1	HB02	1.2	0.26	0.48	0.1497
2	HB03	1.2	0.47	0.56	0.3158
3	HB04	0.69	0.31	0.58	0.1240
4	HB07	0.75	0.185	0.87	0.1207
5	HB09	0.86	0.19	0.77	0.1258
6	HE01	0.66	0.185	0.87	0.1062
7	HE03	0.66	0.3	0.6	0.1188
8	HE04	1.1	0.215	0.65	0.1537
9	HE07	0.87	0.36	0.73	0.2286
10	HE08	1.31	0.205	0.72	0.1933
11	HE10	1.4	0.2	0.74	0.2072

customer sends requests with their shopping combinations, our engine provides the optimized shipping options so that the customer has their flexibility to make better order combinations and understand their shipping cost. To satisfy the second user requirement, we uses Unity-3D to build an animation to guide shipping loaders when loading items. This is illustrated in Fig. 12. Here, we use four programs, which include three, five, six and seven item types respectively, to show the loading sequences.

In the evaluation stage, we compare the deployment performance of our system with the current onsite loading practices. The algorithm is used to calculate the above mentioned four loading programs, with three, five, six,

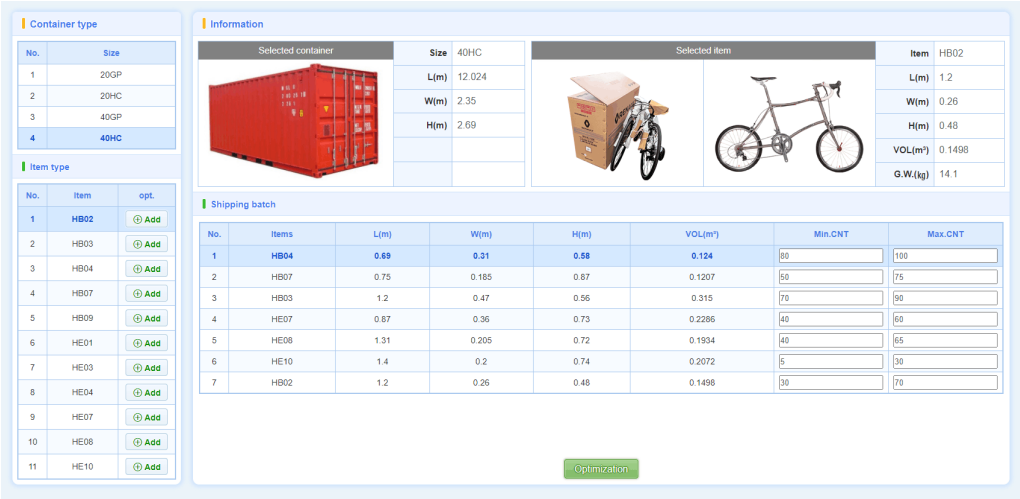


Fig. 11. The proposed algorithm has been deployed as the engine for arranging customer shipping plans.

seven shipping items respectively. From the comparison data in Table 7, it can be seen that the proposed multi-strategy heuristic algorithm improves the filling rate greatly compared to the current operation in our client. The improvement of the loading rate increases 5-10% based on various number of shipping item types. In addition, all of these are achieved in a timely manner, i.e. 2.92 seconds for three items, 3.75 seconds for five items, 9.86 for six items and 11.04 for seven items.

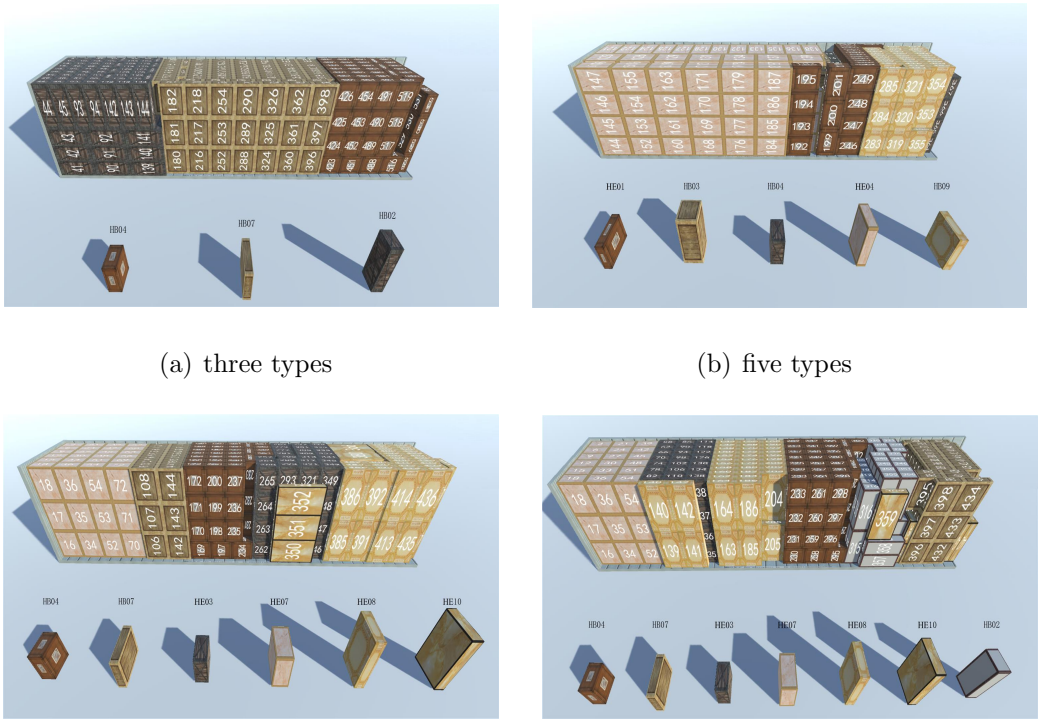
6. Concluding remarks

In this paper, a multi strategy dynamic hybrid heuristic algorithm is proposed to solve the single container weakly heterogeneous problem. Three space greedy cascade method and three space ant colony cascade method are integrated seemly based on a probabilistic selection function. In addition, a

Table 7

Loading scheme information.

No./ Type	Program							
	1		2		3		4	
Loading	OnSite	Ours	OnSite	Ours	OnSite	Ours	OnSite	Ours
1/HB02	150	144	-	-	-	-	46	55
2/HB03	-	-	110	120	-	-	-	-
3/HB04	100	133	8	12	99	109	91	98
4/HB07	250	254	-	-	66	72	66	72
5/HB09	-	-	91	100	-	-	-	-
6/HE01	-	-	61	65	-	-	-	-
7/HE03	-	-	-	-	85	96	78	84
8/HE04	-	-	61	70	-	-	-	-
9/HE07	-	-	-	-	68	72	48	54
10/HE08	-	-	-	-	54	62	61	57
11/HE10	-	-	-	-	26	30	15	27
Count	500	531	331	367	398	441	405	442
LR (%)	86.58	91.48	83.91	92.69	82.15	90.88	81.57	91.19
Time(s)	-	2.92	-	3.75	-	9.86	-	11.04



(a) three types

(b) five types

(c) six types

(d) seven types

Fig. 12. Several types of cargoes loading renderings in 40HC.

tree pruning algorithm is proposed to further improve the efficiency of solving the problem. During the loading process, each state value is preferentially stored and the pruning matrix is updated. Compared with the running results of general data set and public data set, the algorithm in this paper has better competitiveness under the premise of ensuring the packing rate. A case study from a client is used to further support the outstanding performance of the proposed algorithm. In future work, we will further extend the algorithm for strong heterogeneous problem and improve the efficiency when dealing with large number of item types, e.g., more than 30 types in real

business logistic planning.

References

Abbasi, A., Firouzi, B., Sendur, P., Heidari, A.A., Chen, H., Tiwari, R., 2021. Multi-strategy gaussian harris hawks optimization for fatigue life of tapered roller bearings. Engineering with Computers , 1–27doi:https://doi.org/10.1007/s00366-021-01442-3.

Araya, I., Guerrero, K., Nuñez, E., 2017. Vcs: A new heuristic function for selecting boxes in the single container loading problem. Computers & Operations Research 82, 27–35. doi:https://doi.org/10.1016/j.cor.2017.01.002.

Araya, I., Riff, M.C., 2014. A beam search approach to the container loading problem. Computers & Operations Research 43, 100–107. doi:https://doi.org/10.1016/j.cor.2013.09.003.

Bayraktar, T., Ersöz, F., Kubat, C., 2021. Effects of memory and genetic operators on artificial bee colony algorithm for single container loading problem. Applied Soft Computing 108, 107462. doi:https://doi.org/10.1016/j.asoc.2021.107462.

Bischoff, E.E., 2006. Three-dimensional packing of items with limited load bearing strength. European Journal of Operational Research 168, 952–966. doi:https://doi.org/10.1016/j.ejor.2004.04.037.

Bischoff, E.E., Ratcliff, M.S.W., 1995. Issues in the development of ap-

proaches to container loading. Omega 23, 377–390. doi:https://doi.org/10.1016/0305-0483(95)00015-G.

Bonet Filella, G., Trivella, A., Corman, F., 2021. Modeling soft unloading constraints in the multi-drop container loading problem. Social Science Electronic Publishing doi:dx.doi.org/10.2139/ssrn.3929994.

Bortfeldt, A., Gehring, H., 1998. A tabu search algorithm for weakly heterogeneous container loading problems. OR Spectrum .

Bortfeldt, A., Gehring, H., 2001. A hybrid genetic algorithm for the container loading problem. European Journal of Operational Research 131, 143–161. doi:https://doi.org/10.1016/S0377-2217(00)00055-2.

Bose, P., Douïeb, K., Morin, P., 2012. Skip lift: A probabilistic alternative to red–black trees. Journal of Discrete Algorithms 14, 13–20. doi:https://doi.org/10.1016/j.jda.2011.12.017.

Chen, D., You, X., Liu, S., 2021. Ant colony algorithm with stackelberg game and multi-strategy fusion. Applied Intelligence , 1573–7497doi:https://doi.org/10.1007/s10489-021-02774-9.

Dereli, T., Das, G.S., 2011. A hybrid 'bee(s) algorithm' for solving container loading problems. Applied Soft Computing 11, 2854–2862. doi:https://doi.org/10.1016/j.asoc.2010.11.017.

Egeblad, J., Pisinger, D., 2009. Heuristic approaches for the two- and three-dimensional knapsack packing problem. Computers & Operations Research 36, 1026–1049. doi:https://doi.org/10.1016/j.cor.2007.12.004.

Fanslau, T., Bortfeldt, A., 2010. A tree search algorithm for solving the container loading problem. INFORMS Journal on Computing 22, 222–235. doi:10.1287/ijoc.1090.0338.

Gehring, H., ortfeldt, A., 1997. A genetic algorithm for solving the container loading problem. International Transactions in Operational Research 4, 401–418. doi:https://doi.org/10.1016/S0969-6016(97)00033-6.

He, K., Huang, W., 2011. An efficient placement heuristic for three-dimensional rectangular packing. Computers & Operations Research 38, 227–233. doi:https://doi.org/10.1016/j.cor.2010.04.015.

He, Y., Wu, Y., de Souza, R., 2012. A global search framework for practical three-dimensional packing with variable carton orientations. Computers & Operations Research 39, 2395–2414. doi:https://doi.org/10.1016/j.cor.2011.12.007.

Huang, W., He, K., 2009. A caving degree approach for the single container loading problem. European Journal of Operational Research 196, 93–101. doi:https://doi.org/10.1016/j.ejor.2008.02.024.

Jamrus, T., Chien, C.F., 2016. Extended priority-based hybrid genetic algorithm for the less-than-container loading problem. Computers & Industrial Engineering 96, 227–236. doi:https://doi.org/10.1016/j.cie.2016.03.030.

Kang, K., Moon, I., Wang, H., 2012. A hybrid genetic algorithm with a new packing strategy for the three-dimensional bin packing problem. Applied

Mathematics and Computation 219, 1287–1299. doi:https://doi.org/10.1016/j.amc.2012.07.036.

Li, J., Chen, H., Jin, M., Ren, H., 2021a. Enhanced harris hawks optimization with multi-strategy for global optimization tasks. Expert Systems with Applications 185, 115499. doi:https://doi.org/10.1016/j.eswa.2021.115499.

Li, X., Wang, L., Jiang, Q., Li, N., 2021b. Differential evolution algorithm with multi-population cooperation and multi-strategy integration. Neurocomputing 421, 285–302. doi:https://doi.org/10.1016/j.neucom.2020.09.007.

Li, X., Zhang, K., 2015. A hybrid differential evolution algorithm for multiple container loading problem with heterogeneous containers. Computers & Industrial Engineering 90, 305–313. doi:https://doi.org/10.1016/j.cie.2015.10.007.

Liu, J., Yue, Y., Dong, Z., Maple, C., Keech, M., 2011. A novel hybrid tabu search approach to container loading. Computers & Operations Research 38, 797–807. doi:https://doi.org/10.1016/j.cor.2010.09.002.

Liu, S., Tan, W., Xu, Z., Liu, X., 2014. A tree search algorithm for the container loading problem. Computers & Industrial Engineering 75, 20–30. doi:https://doi.org/10.1016/j.cie.2014.05.024.

Lodi, A., Martello, S., Vigo, D., 2002. Heuristic algorithms for the three-dimensional bin packing problem. European Journal of Operational Re-

search 141, 410–420. doi:https://doi.org/10.1016/S0377-2217(02)00134-0.

Lu, H., Sun, S., Cheng, S., Shi, Y., 2021. An adaptive niching method based on multi-strategy fusion for multimodal optimization. Memetic Computing 13, 341–357. doi:https://doi.org/10.1007/s12293-021-00338-5.

Moura, A., Oliveira, J.F., 2005. A grasp approach to the container-loading problem. IEEE Intelligent Systems 20, 50–57. doi:10.1109/MIS.2005.57.

do Nascimento, O.X., Alves de Queiroz, T., Junqueira, L., 2021. Practical constraints in the container loading problem: Comprehensive formulations and exact algorithm. Computers & Operations Research 128, 105186. doi:https://doi.org/10.1016/j.cor.2020.105186.

Norvig, P., 1992. Paradigms of artificial intelligence programming: Case studies in common lisp .

Ozcan, K., Evren, M., 2021. An efficient method for the three-dimensional container loading problem by forming box sizes. Engineering Optimization 0, 1–16. doi:10.1080/0305215X.2021.1913734.

Parreño, F., Alvarez-Valdes, R., Tamarit, J.M., Oliveira, J.F., 2008. A maximal-space algorithm for the container loading problem. INFORMS Journal on Computing 20, 412–422. doi:10.1287/ijoc.1070.0254.

Peng, H., Han, Y., Deng, C., Wang, J., Wu, Z., 2021a. Multi-strategy co-evolutionary differential evolution for mixed-variable optimization. Knowledge-Based Systems 229, 107366. doi:https://doi.org/10.1016/j.knosys.2021.107366.

Peng, H., Zeng, Z., Deng, C., Wu, Z., 2021b. Multi-strategy serial cuckoo search algorithm for global optimization. Knowledge-Based Systems 214, 106729. doi:https://doi.org/10.1016/j.knosys.2020.106729.

Pisinger, D., 2002. Heuristics for the container loading problem. European Journal of Operational Research 141, 382–392. doi:https://doi.org/10.1016/S0377-2217(02)00132-7.

Rashedi, E., Nezamabadi-pour, H., Saryazdi, S., 2009. Gsa: A gravitational search algorithm. Information Sciences 179, 2232–2248. doi:https://doi.org/10.1016/j.ins.2009.03.004.

Saraiva, R., Nepomuceno, N., Pinheiro, P., 2019. A two-phase approach for single container loading with weakly heterogeneous boxes. Algorithms 12, 67. doi:10.3390/a12040067.

Sheng, L., Xiuqin, S., Changjian, C., Hongxia, Z., Dayong, S., Feiyue, W., 2017. Heuristic algorithm for the container loading problem with multiple constraints. Computers & Industrial Engineering 108, 149–164. doi:https://doi.org/10.1016/j.cie.2017.04.021.

Wang, Y., Li, H., Lei, Z., Ma, D., Fang, Y., 2019. Progressively-refined tree search for container loading problem, in: 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS).

Zhou, Q., Xiang, L., 2017. A swarm optimization algorithm for practical

container loading problem, in: IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society.

Zhoujing, W., Kevin, W.L., Jason, K.L., 2008. A heuristic for the container loading problem: A tertiary-tree-based dynamic space decomposition approach. European Journal of Operational Research 191, 86–99. doi:https://doi.org/10.1016/j.ejor.2007.08.017.

Zhu, W., Lim, A., 2012. A new iterative-doubling greedy–lookahead algorithm for the single container loading problem. European Journal of Operational Research 222, 408–417.

Appendix A.

Table A1

List documents in tabular format.

Categorization	Year	Titles	Features
Constructive heuristics	2002	Heuristics for the container loading problem(Pisinger, 2002)	Wall-building based method
	2009	A caving degree approach for the single container loading problem(Huang and He, 2009)	a caving degree approach(CDA)
	2011	An efficient placement heuristic for three-dimensional rectangular packing(He and Huang, 2011)	a fit degree algorithm (FDA)
	2009	Heuristic approaches for the two- and three-dimensional knapsack packing problem (Egeblad and Pisinger, 2009)	Sequence triple
	2010	A tree search algorithm for solving the container loading problem (Fanslau and Bortfeldt, 2010)	a tree search algorithm
	2014	A tree search algorithm for the container loading problem(Liu et al., 2014)	a binary tree search algorithm
	2014 2017	A beam search approach to the container loading problem(Araya and Riff, 2014) A new heuristic function for selecting boxes in the single container loading problem(Araya et al., 2017)	Tree-search-based method based on beam search
	2021	An efficient method for the three-dimensional container loading problem by forming box sizes(Ozcan and Evren, 2021)	INLP model, using the layer-building approach
	2021	Practical constraints in the container loading problem: Comprehensive formulations and exact algorithm(do Nascimento et al.,2021)	Integer linear programming

Continued on next page

Table A1 – continued from previous page

Categorization	Year	Titles	Features
	2021	Modeling soft unloading constraints in the multi-drop container loading problem (Bonet Filella et al., 2021)	Mixed-integer linear programming formulation with soft unloading constraints
	2002	Heuristic algorithms for the three-dimensional bin packing problem(Lodi et al., 2002)	Tabu search
	2011	A novel hybrid tabu search approach to container loading(Liu et al., 2011)	
Metaheuristics methods	1997	A genetic algorithm for solving the container loading problem (Gehring and ortfeldt, 1997)	Genetic algorithm
	2001	A hybrid genetic algorithm for the container loading problem (Bortfeldt and Gehring, 2001)	
	2012	A hybrid genetic algorithm with a new packing strategy for the three-dimensional bin packing problem (Kang et al., 2012)	
	2016	Extended priority-based hybrid genetic algorithm for the less-than-container loading problem(Jamrus and Chien, 2016)	
	2011	A hybrid 'bee(s) algorithm' for solving container loading problems (Dereli and Das, 2011)	Bee Colony algorithm
	2021	Effects of memory and genetic operators on artificial bee colony algorithm for single container loading problem(Bayraktar et al., 2021)	
	2015	A hybrid differential evolution algorithm for multiple container loading problem with heterogeneous containers(Li and Zhang, 2015)	Differential evolution

Continued on next page

Table A1 – continued from previous page

Categorization	Year	Titles	Features
	2012	A hybrid genetic algorithm with a new packing strategy for the three-dimensional bin packing problem(Kang et al., 2012)	the Improved Deepest Bottom Left with Fill (I-DBLF) algorithm
	2016	Extended priority-based hybrid genetic algorithm for the less-than-container loading problem(Jamrus and Chien, 2016)	an extended priority-based hybrid genetic algorithm (EP-HGA)
	2011	A hybrid 'bee(s) algorithm' for solving container loading problems (Dereli and Das, 2011)	a bee(s) algorithm by hybridizing a heuristic filling procedure
	2021	Effects of memory and genetic operators on artificial bee colony algorithm for single container loading problem(Bayraktar et al., 2021)	a memory-integrated ABC algorithm
	2021	Multi-strategy serial cuckoo search algorithm for global optimization(Peng et al.,2021)	Cuckoo search algorithm
	2021	Multi-strategy gaussian harris hawks optimization for fatigue life of tapered roller bearings(Abbasi et al.,2021)	Harris Hawk Optimization
	2021	Enhanced harris hawks optimization with multi-strategy for global optimization tasks(Li et al.,2021)	Harris Hawk Optimization
	2021	Multi-strategy co-evolutionary differential evolution for mixed-variable optimization(Peng et al.,2021)	Co-evolutionary differential evolution

Continued on next page

Table A1 – continued from previous page

Categorization	Year	Titles	Features
	2021	Differential evolution algorithm with multi-population cooperation and multi-strategy integration(Li et al.,2021)	Differential evolution algorithm
	2021	An adaptive niching method based on multi-strategy fusion for multimodal optimization(Lu et al.,2021)	Niching method
	2021	Ant colony algorithm with stackelberg game and multi-strategy fusion(Chen et al.,2021)	Ant colony algorithm