Multiobjective Approaches for the Ship Stowage Planning Problem Considering Ship Stability and Container Rehandles

Zizhen Zhang and Chung-Yee Lee

***Abstract*—The ship stowage planning problem (SSPP) is a very complex and challenging problem in the logistics industries because it affects the benefits of both shipping lines and port terminals. In this paper, we investigate a multiobjective SSPP, which aims to optimize the ship stability and the number of rehandles simultaneously. We use metacentric height, list value, and trim value to measure the ship stability. Meanwhile, the number of rehandles is the sum of rehandles by yard cranes and quay cranes and all necessary rehandles at future ports. To solve this problem, a variant of the nondominated sorting genetic algorithm III (NSGA-III) combined with a local search component is proposed. The algorithm can produce a set of nondominated solutions. Decision makers can then choose the most promising solution for practical implementation based on their experience and preferences. Extensive experiments are carried out on two groups of instances. The computational results demonstrate the effectiveness of the proposed algorithm compared to the NSGA-II and random weighted genetic algorithms, especially when it is applied in solving the six-objective SSPP.**

***Index Terms*—Evolutionary algorithms, multiobjective, rehandle, ship stowage planning, stability.**

I. INTRODUCTION

CONTAINERIZATION is one of the catalysts of global trade. It has resulted in a tremendous growth of the maritime freight industry. Nowadays, over 80% of world trade is carried by the maritime freight industry, which operates the container transportation business. The containers come in a few standard dimensions. Typically, containers are 20 or 40 ft long, 8 ft wide, and 8.5 or 9.5 ft high. The most common containers are 20 ft long, defined as a twenty-foot equivalent unit (TEU). The introduction of standard containers has made efficient multimodal transportation possible and significantly reduced the operational costs. In accordance with the standard, ship capacity is normally measured by the maximum number of TEUs that can be placed on the ship. For efficiency reasons, the ship size has risen from a few thousand TEUs a few years ago to over 18 000 TEUs today.

There are many challenging operational problems related to the shipping of containers. A practical one is termed the ship stowage planning problem (SSPP). Stowage planning is the process of deciding which container is to be placed in what location of the ship. Typically, a container ship will load and/or discharge its containers when it calls a port. Containers have different weights and are destined for different discharging ports where the ship is due to visit. In a port terminal, cranes are used to load and unload containers. Each crane can move within a certain range and perform container retrieval operations. Containers are piled up vertically forming stacks. If container x is stacked above another container y, x will have to be retrieved before y can be retrieved. In this case, container x is called a blocking container (of container y).

Designing a good stowage plan is important as it can significantly reduce the operational cost and raise the operational efficiency. Thus, it is one of the key steps to enhancing a shipping line's competitive advantage. To obtain a good plan, most of the existing container stowage planning literature takes one or several of the following requirements into consideration (see [1], [2]).

1) Minimizing ship turnaround time and reducing container rehandles. Turnaround time is one of the key productivity indicators of the shipping line and the port. The turnaround time of a ship is determined by the time span from its berthing to departure. From the shipping line's perspective, a lower turnaround time at the terminal can save port fees and accelerate the voyage so that the goods can reach their destinations on time. From the port's point of view, a shorter turnaround time can help to reduce congestion and maximize service productivity. Ship turnaround time is primarily dominated by the time necessary to unload and load containers. In order to minimize the turnaround time, it is important for stowage planners to reduce the redundant moves of containers (known as rehandling or reshuffling). Moreover, since moving a container is expensive (usually one move costs tens to hundreds of dollars), it also motivates the planners to reduce the number of rehandles.
2) Minimizing the makespan of quay cranes. The makespan of quay cranes is the latest completion time among all the handling tasks by cranes. It directly affects the ship

---

Manuscript received April 16, 2015; revised June 22, 2015; accepted August 20, 2015. Date of publication December 17, 2015; date of current version September 14, 2016. This work was supported by the Research Grants Council of Hong Kong under Theme-Based Research Grant T32-620/11. This paper was recommended by Associate Editor L. Fang. *(Corresponding author: Chung-Yee Lee.)*

Z. Zhang is with the School of Mobile Information Engineering, Sun Yat-sen University, Guangzhou 510275, China.

C.-Y. Lee is with the Department of Industrial Engineering and Logistics Management, Hong Kong University of Science and Technology, Hong Kong (e-mail: cylee@ust.hk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMC.2015.2504104

2168-2216 © 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

turnaround time and port charges. Because many container ships are large, the port terminal will normally launch two or more cranes simultaneously to speed up the unloading and loading procedure. In practice, container ships are divided longitudinally into several areas (or holds). Only one crane can work with one hold at a time. In addition, two adjacent cranes cannot work too closely for safety reasons. This problem is the so-called crane scheduling problem, which has attracted much academic attention. Although in the stowage planning step we may not be too concerned with the details of crane scheduling, sometimes we need to consider the dispersion of containers for the effective utilization of cranes.

3) Ensuring different kinds of rules for the ship and optimizing ship stability. The container ship is a complex object. To ensure a safe voyage, various types of constraints must be satisfied. These constraints can be classified into physical constraints and stability constraints. Physical constraints affect the operations of the ship. For instance, the last-in-first-out manner is applied to stacks. Refrigerated, hazardous, or specially treated containers must be stowed in particular slots. Stability constraints affect the transverse stability, longitudinal stability, and hydrostatic properties of the ship [3]. Stability is of vital importance to ship sailing. It is the ability to maintain the ship in the upright position. Ships are becoming ever larger and sailing ever faster, potentially increasing the risk of capsizing [4]. Moreover, there is a growing economic pressure to sail in adverse weather conditions, thereby potentially jeopardizing stability and safety [5]. Several of the previous works treated ship stabilities as hard constraints (see [6]). In actual operation, however, some noncritical stability issues are tractable. For example, ballast tanks can be used to hold seawater to weigh down the ship [7]. A good stowage plan needs to optimize the ballast water usage so as to decrease the draft of the ship and thus improve the fuel efficiency.

Based on the above discussions, the following two categories of objectives are of primary importance in ship stowage planning.

1) Maximization of ship stability.
2) Minimization of container rehandles.

These objectives are equally important and should be considered together. They relate to the profit and efficiency of both shipping line and port terminals. However, they may be in conflict sometimes, e.g., slightly increasing the number of rehandles may actually improve ship stability. We formally call such problem a multiobjective SSPP (MO-SSPP). The MO-SSPP is a very realistic problem in the logistics environment and thus deserves our attention. To date, multiobjective optimization has been applied in many fields including engineering, economics, and logistics. The optimal decision is often based on the tradeoff between two or more conflicting objectives. In order to solve the multiobjective optimization problem, two types of approaches are commonly used: the prior approach and the posterior approach.

The prior approach uses the weighted method to combine different objectives into one. The idea is simple but it is often quite difficult to work out the weighted coefficient for each objective. Moreover, in the MO-SSPP, ship stability must be within a certain range for safety and/or economic reasons. However, not all shipowners can tell exactly the range. Instead, it is much simpler for them to determine how stable the ship is given its layout. Therefore, the weighted method is sometimes not appropriate for addressing the MO-SSPP.

The posterior approach, on the other hand, has demonstrated its applicability to the MO-SSPP. It aims at generating a representative subset of Pareto-optimal solutions (globally or locally). A solution is Pareto-optimal if at least one of its objectives is noninferior to that of other solutions in the solution space. To be precise, a solution S_1 is said to dominate another solution S_2 if S_1 is no worse than S_2 for all objective values and S_1 is better than S_2 for at least one objective value. A solution S is Pareto-optimal if S is not dominated by any other solutions in the solution space.

By using the posterior approach, a set of Pareto-optimal solutions can be obtained within a single run without prior information. Then the decision makers can make a tradeoff among different objectives and select the most desirable solution for practical implementation.

This research contributes to the study of the SSPP by proposing a multiobjective approach to achieving tradeoff goals from the perspective of both shipping lines and port terminals. The remainder of this paper is organized as follows. In Section II, we briefly describe the relevant literature, including the studies concerning the SSPP and multiobjective framework. We then provide a formal definition of the MO-SSPP in Section III. In Section IV, we introduce a multiobjective approach, which combines a genetic algorithm with a local improvement procedure to solve this problem. To evaluate our approach, Section V reports a series of experiments that we conducted based on the generated benchmark instances. Section VI gives some closing remarks and some suggestions for future research in this area.

II. LITERATURE REVIEW

According to [1] and [2], stowage planning is one part of the ship planning process, which also includes the berth planning and crane split process. The first study on the SSPP was conducted by Webster and Van Dyke [8]. The problem they introduced was an oversimplified one, and their methods were not extensively tested to show their performance. In 1981, American President Line adopted a computer-aided preplanning system using simulation techniques and human interaction to generate container vessel plans, as described by Shields [9]. Theoretically speaking, a general SSPP can be described as an NP-hard problem. Avriel *et al.* [10] showed the relation between the SSPP and the problem of coloring circle graphs, which has proven to be NP-hard.

The SSPP is also known as the master bay plan problem (MBPP) [11]. Several integer programming models were introduced to formulate different versions of the SSPP or MBPP.

For example, Avriel *et al.* [12] presented a 0-1 linear programming formulation for finding the optimal solution to small-scale stowage planning cases. Ambrosino *et al.* [11], [13] described a basic 0-1 linear programming model for MBPP. Ambrosino *et al.* [14] extended the basic model by considering different types of containers (20 and 40 ft containers). Li *et al.* [15] proposed a 0-1 linear programming model for the stowage problem, in which they maximized the space utilization and minimized the operation cost for a multiport journey. Delgado *et al.* [16] also proposed an integer programming model for stowing a set of containers in a single bay section.

Most of the integer programming approaches can only solve SSPPs of a small scale. For the problem of a practical scale, multistage approaches were adopted instead. Wilson and Roach [17] proposed a method that embodied a two-stage process to computerized stowage planning. The first stage uses the generalized placement strategy, where generalized containers are assigned to a blocked cargo-space. The second stage is a specialized placement procedure, where specific containers are assigned to specific slots. Kang and Kim [18] divided the stowage planning problem into two subproblems. The first subproblem is to assign container groups to the holds and the second subproblem is to determine a loading pattern of containers assigned to each hold. Ambrosino *et al.* [14] developed a three-phase heuristic for the master bay problem, where each phase deals with a particular subproblem. Pacino *et al.* [19] proposed a two-phase approach for the stowage of large container ships. The first phase involves the multiport master planning process and the second phase concerns the slot planning process. Gumus *et al.* [20] also developed a four-stage decomposition heuristic that accounts for many complex real-world SSPPs.

Many heuristic or metaheuristic approaches for the SSPP can also be found in the literature. Avriel *et al.* [12] developed a heuristic called the suspensory heuristic procedure to deal with the SSPP with the objective of minimizing container rehandles (overstowage). In their problem setting, some practical constraints, such as the stability and strength of the ship, are relaxed. Sciomachen and Tanfani [21] presented a heuristic method for solving the MBPP based on its relation with the 3-D bin packing problem. They considered the structural and operational constraints related to the containers and the ship in their model. Wilson and Roach [22] used tabu search (TS) for generating a solution to the SSPP. TS progressively refines the placement of containers. Dubrovsky *et al.* [23] adopted a genetic algorithm and highlighted the efficiency of the encoding through simulations. Ambrosino *et al.* [24] tested the performance of three approaches for MBBP, namely a TS, a simple constructive heuristic and an ant colony optimization (ACO) approach. The results showed that ACO leads to good results for large-size instances, while TS is more desirable for medium-size instances.

Although the aforementioned papers discussed several objectives of the SSPP, most of them treated the SSPP as a single objective optimization problem. There have been few studies on MO-SSPP. Imai *et al.* [7] considered two criteria in the MO-SSPP, i.e., ship stability and the number of rehandles. However, they used the weighted sum method and devised a genetic algorithm to deal with the single objective SSPP. Liu *et al.* [25] took into account five objectives, i.e., the number of rehandles, the completion time of the longest crane, the number of stacks that exceed the weight limit, the number of idle slots, and horizontal moment difference and cross moment difference, in stowage planning. A randomized algorithm incorporating TS was developed for finding a set of Pareto-optimal solutions.

In recent decades, evolutionary approaches have become very popular and demonstrated great success in solving multiobjective optimization problems. Surprisingly, to the best of our knowledge, no existing paper in the literature has addressed the MO-SSPP using multiobjective evolutionary (posterior) approaches at the time of writing.

III. PROBLEM DEFINITION AND ASSUMPTIONS

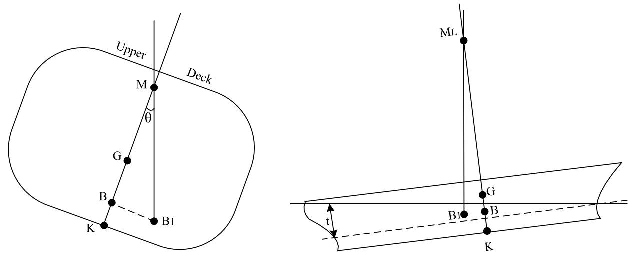
The MO-SSPP we study concerns obtaining a stowage plan that simultaneously optimizes two categories of objectives under various kinds of constraints. These constraints mainly affect the structure of the ship and yard, which are detailed in Section III-A. In Section III-B, the measurement of ship stability is discussed. In Section III-C, we elaborate on the stowage planning process, while we summarize different objectives of the SSPP in Section III-D.

A. Structure of the Ship and Yard

The cargo space of a container ship is split into several 20-ft-long areas. These areas are termed 20-ft bays. In addition, two contiguous 20-ft bays form a 40-ft bay which can be used to accommodate 40-ft containers. From the cross sectional view of a bay, the row denotes the position where the container is placed relative to the horizontal section of the corresponding bay. The planar positions of the container on the ship are uniquely defined by bay and row numbers. Each position is termed a cell. Within a cell, several containers can be piled up into a stack. A stack is made up of a number of tiers, which defines the vertical position of the containers. The container position in the ship is uniquely indexed by (bay, row, tier). Such three-tuple is called a slot.

Container ships are complex objects and have different layouts. To simplify the problem, the following assumptions related to ship structure are made in this research.

1) The container ship is box-shaped and of the lift-on/lift-off type. Containers are only accessible from the top, i.e., a container can only be retrieved if it is the topmost container in the stack.
2) We only consider the standard type of containers. Refrigerated, hazardous, or other nonstandard containers are treated specially.
3) 20- and 40-ft bays are commonly stacked separately. Our research only focuses on 20-ft containers.
4) We only consider the rectangular bays, and the number of rows in each bay is the same, denoted by r_S. The number of bays is assumed to be b_S in the ship.



(a) (b)

Fig. 1. Ship stability factors. (a) Transverse stability. (b) Longitudinal stability.

Thus, the number of stacks in the ship equals $b_S r_S$. We also assume that the number of tiers for each stack is bounded by h_S. We do not distinguish between the space inside the ship hold or that above deck.

The structure of the yard bays is similar to that of the ship bays. The number of yard bays, the number of rows in each yard bay, and height limits are denoted by b_Y, r_Y, and h_Y, respectively.

B. Stability Measurement

Ship stability depends on four factors, i.e., the segments of *KB*, *BM*, *KG*, and *GM* [3]. Fig. 1 shows the relative positions of the four segments. Among these factors, *GM* (also referred to as metacentric height) is the most important. *GM* is the distance between the center of gravity of a ship (with containers loaded) and its metacenter. *GM* mainly deals with transverse stability [see Fig. 1(a)], which relates to vertical equilibrium and cross equilibrium. *GM* can be calculated by the following formula:

$$GM = G_0M + \frac{\sum_i w_i(z_i - z_0)}{W + \sum_i w_i}. \tag{1}$$

In (1), G_0M is the initial metacentric height. The coordinate (x_0, y_0, z_0) denotes the initial center of gravity, and (x_i, y_i, z_i) is the location where the ith container is placed. w_i is the weight of the ith container. W is the weight of the ship without any containers loaded. In practice, a larger but not extremely large *GM* implies greater initial stability against overturning. To optimize *GM*, stowage planners will place lighter containers on top of heavier ones.

GM can also help to calculate the angle of list, which measures the degree to which a ship is leaning to one side. Let θ denote the angle of list. A small θ contributes to a ship's cross equilibrium, so θ should be minimized. Equation (2) shows a way to calculate θ

$$\tan\theta = \frac{\sum_i w_i(y_i - y_0)}{\left(W + \sum_i w_i\right) \cdot GM}. \tag{2}$$

The longitudinal stability [see Fig. 1(b)] is related to the trim value, which is defined as the difference between the draft at forward perpendicular and after perpendicular. Obviously, the absolute value of the trim value, denoted by $|t|$, should also be minimized. The calculation of t is given by (3), where GM_L is the distance between center of gravity G and longitudinal metacenter M_L. The approximated equation in (3) is obtained from $GM_L \simeq BM_L = W \cdot l^3/12(W + \sum_i w_i)$ when the box-shaped ship is considered (l denotes the length of the ship)

$$t = \frac{l\sum_i w_i(x_i - x_0)}{\left(W + \sum_i w_i\right) \cdot GM_L} \simeq \frac{12\sum_i w_i(x_i - x_0)}{W \cdot l^2}. \tag{3}$$

C. Ship Stowage Planning Process

Consider that a container ship calls some port terminal. The containers on the ship destined for the current port are to be discharged. Meanwhile, there are a number of containers destined for future ports stowed on the yard and to be loaded onto the ship. The port terminal operates cranes for loading and unloading containers. We assume that container unloading always take place before loading. This assumption is valid in many scenarios of terminal operations. In this case, the unloading process and loading process can be treated separately.

During the unloading process, the general goal is to reduce the import container discharging time, while the stability of the ship can be optimized through the loading process. Thus, the problem of container unloading can often be treated as a single objective one which minimizes the number of import container rehandles. We ignore this problem in this research and mainly focus on the stowage planning problem during the loading process. That is, given the ship layout after unloading, we aim to design a loading plan that indicates which export container to be placed in what location of the ship.

The rehandling of containers related to such stowage planning problem occurs in two scenarios. First, export containers (yard containers) arrive at the yard in random order several days before shipping. They are stowed in the bays forming stacks. A stowage plan should determine the loading sequence of yard containers. Following this sequence, yard containers are retrieved and transported by internal trucks from the yard to the quayside. The retrieval of yard containers by yard cranes according to the loading sequence may incur rehandling. It is worth noting that in reality, a premarshalling step (see [26]) can help to reduce the chances of rehandling during the container retrieval.

Second, container rehandling may also be needed when yard containers arrive at the quayside and have to be loaded onto the ship by quay cranes. This is because the stowage plan designates the location of each container. If a yard container is assigned to some location that is already occupied by another ship container, that container must first be shifted. In the literature, the rehandling of ship containers is also called voluntary shifting (refer to [12]). It can help to prevent costlier shifts at future ports. Through voluntary shifting, the number of blocking containers may be reduced significantly.

Example 1: Suppose that the ship calls port 1. There is a stack on the ship composed of only one container with its port of discharge (POD) equal to 2. Now there are four yard containers with POD = 3 to be loaded to this stack. If the ship container remains in the stack, all four yard containers will become blocking containers and four rehandles will be

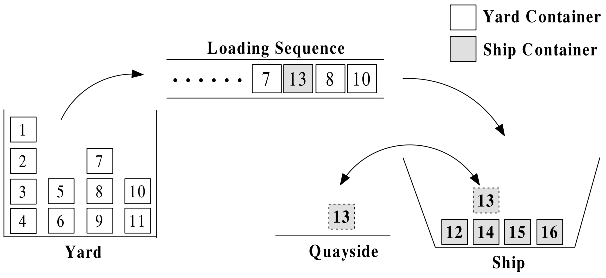


Fig. 2. Ship stowage planning process.

TABLE I
SIX OBJECTIVES OF THE MO-SSPP

Category	Objective	Range	Target
Stability	Metacentric height (f_1)	$[0, +\infty)$	max $f_1(S)$
	List value (f_2)	$(-\infty, +\infty)$	min $\|f_2(S)\|$
	Trim value (f_3)	$(-\infty, +\infty)$	min $\|f_3(S)\|$
re-handle	No. re-handles by yard cranes (f_4)	Z^+	min $f_4(S)$
	No. re-handles by quay cranes (f_5)	Z^+	min $f_5(S)$
	No. re-handles at future ports (f_6)	Z^+	min $f_6(S)$

required when the ship calls port 2. However, if the ship container is voluntarily shifted and is reloaded after the four yard containers are loaded, no rehandles will be needed at port 2.

In sum, the number of rehandles with respect to a given stowage plan consists of the following three parts.

1) Number of rehandles by yard cranes. Note that the problem of finding the minimum number of rehandles by yard cranes for a given loading sequence is known as the block relocation problem or container relocation problem [27], which is an NP-hard problem in the strong sense.
2) Number of rehandles by quay cranes (via voluntary shifting). For simplicity, we assume that if a ship container is rehandled, it is first temporarily shifted to the quayside and reloaded onto the ship at a later stage (known as external move). The movement of containers within the ship (known as internal move) is not considered.
3) Number of necessary rehandles at future ports. If a container is a blocking container, a rehandle will be needed when some container below it is to be retrieved. Therefore, we use the number of blocking containers to estimate the number of rehandles at future ports.

We illustrate the ship stowage planning process in Fig. 2.

D. Multiobjective Ship Stowage Planning Problem

As discussed above, there are six objectives in two categories to be optimized simultaneously. The first category is related to ship stability and the second one is related to container rehandles. Suppose that S is a feasible loading plan to the MO-SSPP. The six-objective function is defined as $f(S) = (f_1(S), \ldots, f_6(S))$, where the meanings of different objectives are summarized in Table I and the general form of the MO-SSPP is stated below

$$\min \quad \{-f_1(S), |f_2(S)|, |f_3(S)|, f_4(S), f_5(S), f_6(S)\} \tag{4}$$

$$\text{s.t.} \quad S \in \mathbb{S}. \tag{5}$$

Equation (4) is the objective function. Similar to [7], we can adopt the weighted method to integrate the six objectives into a single objective by introducing the weights $\alpha_1, \ldots, \alpha_6$. Another attractive approach is to adopt posterior approach to look for a set of Pareto-optimal solutions so that decisions makers can make a tradeoff among different objectives.

In (5), $\mathbb{S}$ is the feasible solution space. In this paper, we do not present the solution space in mixed integer linear programming (MILP) form, as it would introduce too many decision variables and constraints. Consider the block relocation problem, a sub-phase of the SSPP. The mathematical model of this problem can be found in [28]. The authors presented an MILP model to solve the problem. The model involves more than $r_Y^2 h_Y^2 NT$ binary variables, where N is the number of containers in the bay and T is an upper bound on the number of container moves. They also reported that the MILP solver can only solve very small-size instances, e.g., $r_Y h_Y \leq 4 \times 5$. Instead, we will discuss how to use a compact method to represent feasible solutions in the next section, which allows us to devise efficient heuristics for solving the problem.

IV. MULTIOBJECTIVE APPROACH

In this section, we devise a variant of the nondominated sorting genetic algorithm III (NSGA-III) [29] for solving the MO-SSPP. We first present an overview of our approach in Section IV-A. Then, the solution representation is discussed in Section IV-B. The evaluation of solution is given in Section IV-C. The method for generating an initial population is presented in Section IV-D. We then describe the recombination operator and mutation operator with respect to NSGA-III in Sections IV-E and IV-F, respectively. The local search process is introduced in Section IV-G. Section IV-H states the neighborhood generating operators.

A. Overview of the Approach

When we apply the posterior approach to the MO-SSPP, the solutions in $\mathbb{S}$ are compared based on Pareto dominance relations. In practice, it is always difficult to obtain all Pareto-optimal solutions (termed Pareto front) of the MO-SSPP due to its extremely large solution space. Therefore, we only seek a representative set of solutions that can approximate the Pareto front as closely as possible.

We adopt an NSGA-III to find the approximated Pareto front of the MO-SSPP. NSGA-III is an extension of NSGA-II [29], which is a very popular population-based multiobjective evolutionary algorithm. NSGA-III improves the selection operator in NSGA-II by maintaining the diversity of nondominated solutions with a set of well-distributed reference points. Compared to NSGA-II, NSGA-III is more adapted to solving many-objective optimization problems (many means greater than 3).

We also incorporate a local search procedure in NSGA-III, which can significantly improve individual solutions. The framework of NSGA-III is given in Algorithm 1.

Algorithm 1 Framework of NSGA-III for the MO-SSPP

```
1: $P_0 \leftarrow$ INITIAL_POPULATION( );
2: $Q_0 \leftarrow \emptyset$;
3: $t \leftarrow 1$;
4: while termination criteria not reached do
5:     $R_t \leftarrow P_{t-1} \cup Q_{t-1}$;
6:     $F \leftarrow$ FAST-NON-DOMINATED-SORT($R_t$);
7:     $P_t \leftarrow \emptyset$ and $i \leftarrow 1$;
8:     while $|P_t| + |F_i| \leq POP\_SIZE$ do
9:         $P_t \leftarrow P_t \cup F_i$;
10:        $i \leftarrow i + 1$;
11:    end while
12:    Members to be chosen from the last front $F_i$: $K = POP\_SIZE - |P_t|$;
13:    $P_t \leftarrow P_t \cup$ SELECTION($P_t$, $F_i$, $K$);
14:    $Q_t \leftarrow$ RECOMBINATION($P_t$);
15:    $Q_t \leftarrow$ MUTATION($Q_t$);
16:    $Q_t \leftarrow$ LOCAL-SEARCH($Q_t$);
17:    $t \leftarrow t + 1$;
18: end while
```

In this algorithm, P_t is the parent population in the tth generation, Q_t is the child population, and R_t is the generated population. The process fast-nondominated-sort (line 6) classifies R_t into several nondomination levels, i.e., $F_1, F_2, \ldots, F_i, \ldots$. The members from F_1 to F_{i-1} are already included to form a new population P_t, and the remaining K slots (line 12) are selected from the members of the last front F_i. Line 13 is the selection procedure, which involves normalization, association and niche-preservation operations. The procedures recombination and mutation exploit heuristic operators to evolve the population. Line 16 is the local-search procedure. For more details of NSGA-III, readers are referred to [30].

B. Solution Representation

A feasible solution S is associated with three decision sets, i.e., $S = (L, R_Y, R_S)$. Here, L is the loading plan for the containers (including all yard containers and some subset of ship containers). A loading plan should designate the container loading sequence and the position to be occupied by each container. R_Y is the rehandling plan for yard containers and R_S is the rehandling plans for the subset of ship containers.

Observe that once L is determined, the loading sequence is also realized. Given the loading sequence for yard containers, the optimal rehandling plan R_Y (in terms of the number of rehandles) can be obtained by solving the container relocation problem [27]. Given the loading sequence for the subset of ship containers, we know which ship containers should be rehandled, so the number of ship container rehandles can also be calculated. Moreover, once L is determined, the final container layout of the ship can be worked out and ship stability can be determined. In sum, it suffices to use the loading plan L to represent the solution.

Let C_Y be the set of yard containers, C_S be the set of ship containers, and $C_S(S) \subseteq C_S$ be some subset of ship containers to be rehandled according to solution S. Generally, L can be denoted by a sequence of triplets, i.e., $L = \langle (id_1, b_1, r_1), (id_2, b_2, r_2), \ldots, (id_n, b_n, r_n) \rangle$, where $n = |C_Y| + |C_S(S)|$. $id_i \in C_Y \cup C_S(S) (1 \leq i \leq n)$ is the identification of the container in the ith position of the loading sequence. b_i $(1 \leq b_i \leq b_S)$ and r_i $(1 \leq r_i \leq r_S)$ correspond to the bay number and row number of the ship for loading this container, i.e., (b_i, r_i) denotes a loading stack for container id_i.

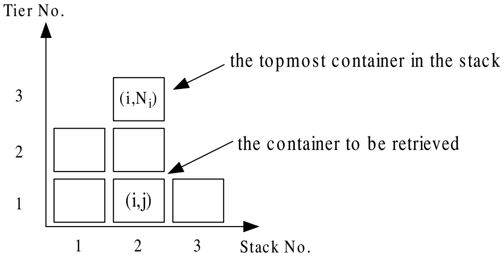

Tier No.
3
2
1
(i,N$_i$)
(i,j)
the topmost container in the stack
the container to be retrieved
1
2
3
Stack No.

Fig. 3. Example of a bay in the container relocation problem.

Example 2: For Fig. 2, suppose that a possible loading plan is $L = \langle (10, 1, 1), (8, 1, 2), (13, 1, 1), (7, 1, 3), \ldots \rangle$. The triplet (10, 1, 1) indicates that container 10 is first loaded to (bay 1, row 1) of the ship, then container 8 is loaded to (bay 1, row 2). Note that container 13 is a ship container that is temporarily shifted to the quayside. Then it is loaded to (bay 1, row 1) after container 8 is loaded. The loading sequence for yard containers is $\langle 10, 8, 7, \ldots \rangle$. From the information of L, the number of rehandles and the ship layout can be concluded.

Because the container relocation problem is NP-hard, given the loading plan L, we cannot obtain the optimal rehandling plan R_Y in polynomial time. Therefore, instead of seeking the optimal rehandling plan, we simply adopt a greedy heuristic method proposed by [31] in Section IV-C to find a near-optimal rehandling plan. The reasons are as follows.

1) We may need to validate and evaluate different loading plans in the solution space many times. A fast and effective heuristic is desirable.
2) Yard container rehandles cost much less than ship container rehandles. It is sufficient to first invoke a heuristic method to approximate the optimal R_Y for the loading plan L, and then optimize the approximated rehandling plan at a later stage.

C. Heuristic Method for the Container Relocation Problem

Let L_Y be the subsequence of L consisting of only yard containers. Given sequence L_Y and the yard layout, the container relocation problem aims to find the minimum number of rehandles for retrieving all yard containers according to L_Y. Because the relocation operation in a yard bay is independent of that in other yard bays, without loss of generality, we only present the method for a particular yard bay.

Fig. 3 illustrates a bay in the container relocation problem. Let us denote by (i, j) some container located in stack i and tier j of the yard bay. Denote by N_i $(1 \leq i \leq r_Y)$ the number of containers in stack i and by N the total number containers in the bay, i.e., $N = \sum_{i=1}^{r_Y} N_i$. Let $p_{i,j}$ be the priority value of container (i, j), i.e., $p_{i,j} = l$ means that the destination of container (i, j) is l. Let $p_i^{\max}$ and $p_i^{\min}$ be the maximum and minimum priorities of some container in stack i, i.e., $p_i^{\max} = \max_{1\le j\le N_i} p_{i,j}$ and $p_i^{\min} = \min_{1\le j\le N_i} p_{i,j}$, respectively.

The heuristic method introduced by Caserta *et al.* [31] makes use of simple greedy rules. Assume that container (i, j) is the target container to be retrieved according to L_Y. If $j = N_i$, container (i, j) can be directly retrieved without any rehandles. Otherwise, container (i, N_i), which is the topmost container in stack i, must be relocated to some other stack $k(k \neq i)$. Here, the stack to be chosen is determined by the following criteria.

1) If there exists a stack k such that $N_k < h_Y$ and container (i, N_i) does not block any containers in stack k (i.e., $p_{i,N_i} \le p_k^{\min}$), then stack k is chosen. If there is more than one stack satisfying these conditions, choose the one with the smallest $p_k^{\min}$.
2) If there does not exist a stack satisfying the above conditions, choose the stack k such that $N_k < h_Y$ and $p_k^{\max}$ is the largest one.

The heuristic method guarantees a feasible solution if indeed there is one. The heuristic executes in $O(Nh_Y r_Y) \le O(N^2)$ time. This is because there are N containers in total; each container may have at most $h_Y - 1$ blocking containers on top to be relocated to one of the remaining $r_Y - 1$ stacks.

D. Initial Population Generation

In NSGA-III, the initial population is made up of POP_SIZE initial solutions. Motivated by the manual planning procedure in practice, we employ a two-stage method to generate each initial solution.

In the first stage, we randomly apply one of the following mechanisms to obtain a loading sequence for the solution.

1) Each time, retrieve the topmost containers whose destination is the furthest among all the topmost containers in the yard stacks. If there are multiple choices, retrieve the heaviest one.
2) Each time, retrieve the container whose destination is the furthest among all yard containers. If there are multiple choices, retrieve the one with the fewest blocking containers on top of it. If there is still a tie, retrieve the heaviest one.

In the second stage, we use a heuristic to designate each container to a loading stack according to the loading sequence in the first stage. The selection of ship stack k to accommodate container x is decided in the similar manner described in Section IV-C as follows.

1) If there exists a stack k such that $N_k < h_S$ and container x does not block any containers in stack k, then stack k is chosen. If there is more than one stack satisfying these conditions, choose the one with the smallest $p_k^{\min}$.
2) If there does not exist a stack satisfying the above conditions, choose the stack k such that $N_k < h_S$ and $p_k^{\max}$ is the largest one.

After these two stages, a feasible initial solution can be obtained. Note that no ship container rehandling occurs in the initial solutions.

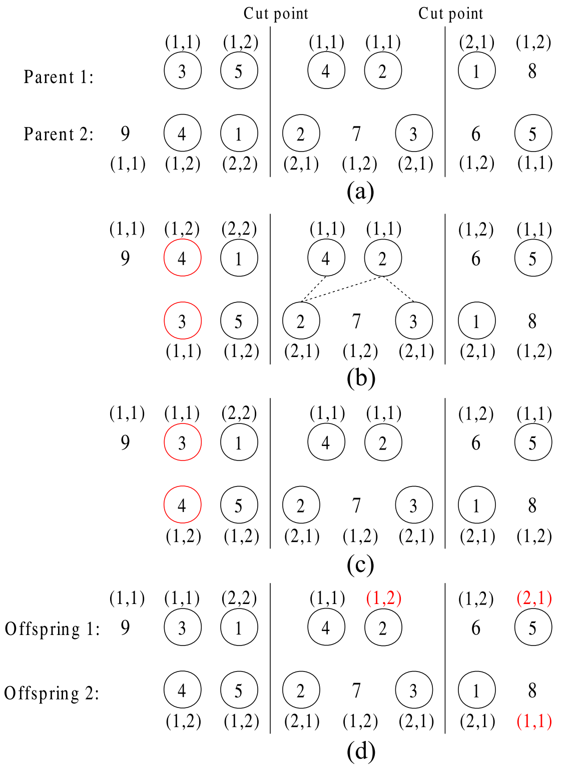


Fig. 4. Example illustrating the use of the recombination operator. (a) Original parent solutions. (b) Intermediate solutions after two-point crossover. (c) Intermediate solutions after PMX crossover. (d) Final offspring solutions.

E. Recombination

The recombination procedure is commonly used in evolutionary algorithms for producing offspring solutions from parent solutions. Generally, two parent solutions are selected to generate two offspring solutions after parent mating selection with the recombination probability P_r. In the MO-SSPP, designing the recombination operator is associated with two difficulties. First, the size of loading plans for different solutions may vary, so it is impossible to directly apply traditional recombination operators (e.g., two-point, partially mapped crossover (PMX), and so on). Second, the resultant loading plans may be infeasible if the recombination operator is not properly designed.

In what follows, we devise a novel and problem-specific recombination operator to deal with these difficulties. We also illustrate how the recombination operator works in Fig. 4. In this example, $b_S = 2$, $r_S = 2$, $h_S = 3$, containers 1–5 are yard containers, and containers 6–9 are ship containers. Note that parent 1 in Fig. 4(a) represents a solution with the loading plan $\langle(3, 1, 1), (5, 1, 2), (4, 1, 1), (2, 1, 1), (1, 2, 1), (8, 1, 2)\rangle$. The steps are as follows.

1) Identify the containers that appear in both parent solutions. These containers are marked with circles in the figure. Let m be the number of containers marked with circles in a parent solution ($m = 5$ in this example).
2) Randomly generate two integers x_0 and y_0 such that $0 \le x_0 \le y_0 \le m$. In our example, $x_0 = 2$ and $y_0 = 4$.
3) Let $\text{pos}[0] = 0$, $\text{pos}[m + 1] = n + 1$ and $\text{pos}[i]$ $(1 \le i \le m)$ be equal to the actual position of

the ith marked container in the corresponding solution. For example, pos1[2] = 2 and pos1[4] = 4 for parent 1 and pos2[2] = 3 and pos2[4] = 6 for parent 2.

4) Randomly generate two cut points $\langle x_1, y_1\rangle$ for parent 1 and $\langle x_2, y_2\rangle$ for parent 2. The cut points should satisfy $\text{pos1}[x_0] \le x_1 < \text{pos1}[x_0+1]$, $\text{pos1}[y_0] \le y_1 < \text{pos1}[y_0+1]$, $\text{pos2}[x_0] \le x_2 < \text{pos2}[x_0+1]$, and $\text{pos2}[y_0] \le y_2 < \text{pos2}[y_0+1]$. In our example, we have $2 \le x_1 < 3$, $4 \le y_1 < 5$, $3 \le x_2 < 4$, and $6 \le y_2 < 8$. Fig. 4(a) shows cut points $\langle 2, 4\rangle$ for parent 1 and cut points $\langle 3, 6\rangle$ for parent 2.
5) By cut points, the two-point crossover operator is performed. The first part and last part of parent solutions are swapped [see Fig. 4(b)].
6) Some marked containers can be duplicated in a solution after a two-point crossover. To overcome this problem, the well-known PMX procedure is invoked for position-wise exchanges. Fig. 4(c) illustrates the resulting intermediate solutions.
7) Some ship stacks may end up with excessive containers according to the loading plan in the previous step. We simply apply a greedy strategy to fix it. If loading a container causes a stack overflow, this container is loaded to the stack with the fewest containers. Fig. 4(d) shows the final offspring solutions.

F. Mutation

After performing the recombination operator, a mutation procedure is conventionally invoked to help evolutionary algorithms escape from local optima. We thus devise three types of operators to introduce a small amount of perturbation in the offspring population, which can help the process locate promising neighborhoods in a wider region of the search space. These operators are presented in Section IV-H. They are applied to the offspring solutions with the mutation probability P_m, i.e., they are performed $P_m \times n$ times for a solution.

G. Local Search

We incorporate a local search in the NSGA-III framework. The local search can significantly improve the individual solution guided by a search direction. It exploits the problem-knowledge as the population evolves and is more likely to achieve good results. This is in fact the concept behind memetic algorithms [32] in the literature.

Because the local search may take a lot of computation time, only a portion of the offspring solutions are selected for the local search. If an offspring solution is selected for local improvement, three types of operators, the same as those applied in mutation, are randomly chosen and applied to the offspring solution for ls_iter times. The solution is updated if and only if it is improved.

In the selection process, only $\lceil \text{POP\_SIZE} \times P_{\text{ls}} \rceil$ solutions from offspring pool Q_t are chosen for improvement, where P_{ls} is the local search probability. The selection of solutions is based on the local search direction.

Algorithm 2 Local Search Procedure

```
 1: function LOCAL_SEARCH($Q_t$)
 2:     $Q'_t \leftarrow \emptyset$;
 3:     for $i \leftarrow 1$ to $\lceil POP\_SIZE \times P_{ls} \rceil$ do
 4:         Randomly generate weights $\alpha_i$ $(1 \le i \le 6)$;
 5:         Perform tournament selection on $Q_t$ according to the aggregated objective, and solution $S$ is the winner;
 6:         $S' \leftarrow$ GUIDED-LOCAL-SEARCH($S, \alpha, ls\_iter$);
 7:         $Q'_t \leftarrow Q'_t \cup S'$;
 8:     end for
 9:     return $Q'_t$;
10: end function
```

Our local search aims to improve the solutions in Q_t by optimizing the aggregated single objective function: $-\alpha_1 f_1(S) + \alpha_2|f_2(S)| + \alpha_3 f_3(S) + \alpha_4 f_4(S) + \alpha_5 f_5(S) + \alpha_6 f_6(S)$ for $S \in Q_t$. Therefore, the search direction is actually guided by weights α_i $(1 \le i \le 6)$. Since there is no prior information about the importance of each objective, in order to reduce the scalar effect, α_i is randomly and uniformly drawn based on the following range setting:

$$\begin{cases} \alpha_i = 0, & \text{if } \max_{S\in Q_t} |f_i(S)| = 0 \\ 0 \le \alpha_i \le \left(\max_{S\in Q_t} |f_i(S)|\right)^{-1}, & \text{otherwise.} \end{cases} \tag{6}$$

Equation (6) can normalize the objectives. A large α_i indicates that the ith objective is important in the local search, and vice versa.

After all the weights α_i are determined, the aggregated objective for each offspring solution can be obtained. Tournament selection [33] with the tournament size τ is then performed on the offspring pool Q_t. The solution in the tournament with the best aggregated objective is selected for local search.

The local search procedure is summarized in Algorithm 2. The function guided-local-search (line 6) is to perform local improvement on solution S ls_iter times guided by weights α.

H. Neighborhood Operators

The neighborhood operators introduced in this section are invoked during the mutation and local search process. Note that all the neighborhood operators are applied to the loading plan L rather than the solution S. Therefore, every generated neighbor must be evaluated.

Recall that a loading plan consists of the loading sequence and the loading stacks of containers, i.e., $L = \langle (id_1, b_1, r_1), (id_2, b_2, r_2), \ldots, (id_n, b_n, r_n)\rangle$. Based on the scopes and effects of the neighborhood operators, we classify them into the following three categories.

1) Operators Applied to the Loading Sequence: This category of operators affects the loading sequence only, while the loading stacks of containers remain unchanged. It contains the following three operators.

1) *Forward Shift Operator:* Randomly generate two indices i and j $(1 \le i < j \le n)$, and then shift the triplet (id_j, b_j, r_j) to the position immediately before (id_i, b_i, r_i). The resultant loading plan is $\langle (id_1, b_1, r_1), \ldots, (id_j, b_j, r_j), (id_i, b_i, r_i), \ldots, (id_n, b_n, r_n)\rangle$.
2) *Backward Shift Operator:* Randomly generate two indices i and j $(1 \le i < j \le n)$, and then shift the

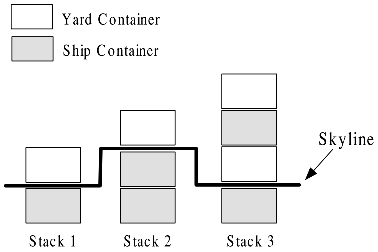


Fig. 5. Example of a skyline.

triplet (id_i, b_i, r_i) to the position immediately behind (id_j, b_j, r_j).

3) *Swap A Operator:* Randomly generate two indices i and j $(1 \leq i < j \leq n)$, and then swap triplets (id_i, b_i, r_i) and (id_j, b_j, r_j).

Note that the resultant loading plan is still feasible after applying this category of operators.

2) Operators Applied to Container Loading Stacks: This category of operators tries to alter the loading stacks of containers in the loading plan. Two operators are introduced as follows.

1) *Reassign Operator:* Randomly generate an index i $(1 \leq i \leq n)$ and a loading stack (b', r') such that $(b', r') \neq (b_i, r_i)$ and height $(b', r') < h_S$. Here, height (b', r') denotes the height of the stack (b', r') on the ship according to the loading plan. Subsequently, replace (id_i, b_i, r_i) in the loading plan with (id_i, b', r') to obtain a neighbor loading plan.
2) *Swap B Operator:* Randomly generate two indices i and j $(1 \leq i < j \leq n)$, and then swap the loading stacks of these two containers. The resultant loading plan becomes $\langle (id_1, b_1, r_1), \ldots, (id_i, b_j, r_j), \ldots, (id_j, b_i, r_i), \ldots, (id_n, b_n, r_n) \rangle$.

3) Operators Applied to Ship Containers: We have mentioned that a subset of ship containers, denoted by the set $C_S(S)$, are temporarily shifted to the quayside in order to reduce the number of blocking containers. This category of operators tries to produce a neighbor by adding/removing a ship container to/from $C_S(S)$.

To do this, we first introduce the concept of skyline. A skyline is a vertical partition that separates the final ship layout (determined by the loading plan L) into two parts. The upper part consists of the containers in $C_Y \cup C_S(S)$, i.e., all the yard containers and those ship containers to be rehandled. The lower part consists of only ship containers in $C_S \backslash C_S(S)$. Fig. 5 illustrates an example of a skyline.

With the assistance of the skyline, two operators are devised as follows.

1) *Add Ship Container Operator:* Randomly select a loading stack (b', r') that has more than one ship container below the skyline. Suppose that the topmost ship container under the skyline in the stack is container id. Randomly insert this container into the loading plan and randomly assign a loading stack value (b, r) to it. Then we can obtain a neighbor loading plan such as $\langle (id_1, b_1, r_1), \ldots, (id_{i-1}, b_{i-1}, r_{i-1}), (id, b, r), (id_i, b_i, r_i), \ldots, (id_n, b_n, r_n) \rangle$.

TABLE II
DATASET FOR THE MO-SSPP

Instance	Yard side					Ship side				
	b_Y	r_Y	h_Y	N	Source	b_S	r_S	h_S	N	Source
A1, B1	2	4	5	24	BF1_1 ∼ BF1_2	4	4	5	19	BF1_3 ∼ BF1_6
A2, B2	2	4	5	27	BF2_1 ∼ BF2_2	4	4	5	24	BF2_3 ∼ BF2_6
A3, B3	2	8	5	50	BF3_1 ∼ BF3_2	4	8	5	34	BF3_3 ∼ BF3_6
A4, B4	2	8	5	47	BF4_1 ∼ BF4_2	4	8	5	34	BF4_3 ∼ BF4_6
A5, B5	4	4	5	54	BF5_1 ∼ BF5_4	8	4	5	55	BF5_5 ∼ BF5_12
A6, B6	4	4	5	56	BF6_1 ∼ BF6_4	8	4	5	52	BF6_5 ∼ BF6_12
A7, B7	4	8	5	127	BF7_1 ∼ BF7_4	8	8	5	101	BF7_5 ∼ BF7_12
A8, B8	4	8	5	137	BF8_1 ∼ BF8_4	8	8	5	105	BF8_5 ∼ BF8_12
A9, B9	2	4	8	46	BF9_1 ∼ BF9_2	4	4	8	40	BF9_3 ∼ BF9_6
A10, B10	2	4	8	39	BF10_1 ∼ BF10_2	4	4	8	35	BF10_3 ∼ BF10_6
A11, B11	2	8	8	70	BF11_1 ∼ BF11_2	4	8	8	67	BF11_3 ∼ BF11_6
A12, B12	2	8	8	94	BF12_1 ∼ BF12_2	4	8	8	82	BF12_3 ∼ BF12_6
A13, B13	4	4	8	94	BF13_1 ∼ BF13_4	8	4	8	97	BF13_5 ∼ BF13_12
A14, B14	4	4	8	96	BF14_1 ∼ BF14_4	8	4	8	98	BF14_5 ∼ BF14_12
A15, B15	4	8	8	210	BF15_1 ∼ BF15_4	8	8	8	195	BF15_5 ∼ BF15_12
A16, B16	4	8	8	199	BF16_1 ∼ BF16_4	8	8	8	209	BF16_5 ∼ BF16_12

2) *Remove Ship Container Operator:* Randomly select a loading stack (b', r') such that there is more than one ship container above the skyline. Remove the bottommost ship container above the skyline from the loading plan.

V. EXPERIMENTAL ANALYSIS

In this section, we evaluate and analyze the performance of the multiobjective algorithm on a series of test instances. The algorithm was implemented in GNU C++, using the "−O2" option. All experiments were run on a Linux server with 8 GB of memory and Intel Xeon E5420 2.50 GHz processor. Computation times reported here are in CPU seconds on this machine.

A. Data Generation

The experiments were carried out on 32 instances with different ship and yard structures. These instances were generated based on the instance groups provided by Bortfeldt and Forster [34] (BF instances for short). In their paper, several groups of instances were proposed for solving the container premarshalling problem. Each group contains 20 container layouts for a rectangular bay. The bays within a group are of the same size. A rectangular bay is composed of a number of containers with different priority values.

Our generated MO-SSPP instances are classified into two types (types A and B) and each type contains 16 instances. For type A instances, all the containers have identical weight (1 unit). For type B instances, each container is randomly assigned a weight ranging from 10 to 20 units. The information of these instances is summarized in Table II. An instance may involve several yard bays and ship bays. The layouts of these bays were constructed from BF instances, where the bay labels are given in the column "source" in the table. For example, instance# A1 consists of two yard bays and four ship bays. The first yard bay is constructed from the instance "BF1_1," the number of stacks in the bay (r_Y) is 4, the height limit (h_Y) is 5, and the number of containers in this bay (N) is 24. The second ship bay is constructed from the instance "BF1_4." The data

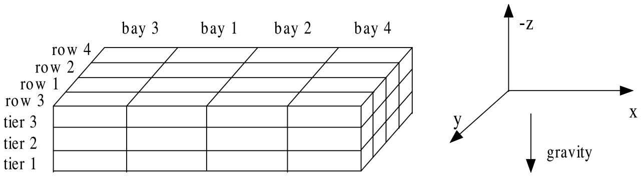


Fig. 6. Example of the ship structure.

TABLE III
COORDINATE OF A SLOT IN THE SHIP

Index	1	2	3	4	5	6	7	8
Bay coordinate (x_i)	-1	1	-3	3	-5	5	-7	7
Row coordinate (y_i)	-1	1	-3	3	-5	5	-7	7
Tier coordinate (z_i)	-1	-2	-3	-4	-5	-6	-7	-8

ensure that there is enough space in the ship to accommodate all the yard containers.

The coordinates of ship slots are needed to calculate the vertical, traverse, and longitudinal stability of the ship. By our assumptions, only box-shaped ship is considered. For such a ship, we number the ship bays and rows from the center to the two sides, while the tiers are numbered from bottom to top. Fig. 6 illustrates the structure of the ship with four bays, four rows, and three tiers. The coordinates of ship slots are set to the values given in Table III. For example, the coordinates (x_i, y_i, z_i) of the container i indexed by (bay 2, row 3, tier 1) are $(1, -3, -1)$. The initial gravity center (x_0, y_0, z_0) is set to $(0, 0, -h_S/2)$, which is the center of the box-shaped ship.

By the above setting, the optimization of vertical stability equation (1), traverse stability equation (2) and longitudinal stability equation (3) is equivalent to maximizing $\sum_{i\in C_Y\cup C_S(S)} w_i z_i$, minimizing $|\sum_{i\in C_Y\cup C_S(S)} w_i y_i|$ and minimizing $|\sum_{i\in C_Y\cup C_S(S)} w_i x_i|$, respectively. Therefore, we simply assume that $f_1(S) = \sum_{i\in C_Y\cup C_S(S)} w_i z_i$, $f_2(S) = \sum_{i\in C_Y\cup C_S(S)} w_i y_i$ and $f_3(S) = \sum_{i\in C_Y\cup C_S(S)} w_i x_i$ for our experiments.

B. Performance Measurement

The goals in designing a promising multiobjective algorithm lie in two aspects. First, the algorithm can find a set of solutions that are as close to the Pareto front as possible. Second, the algorithm can find a set of solutions that are as diverse as possible.

In order to evaluate the performance of the proposed algorithm and other algorithms, three indicators are introduced. The first one, the distance from reference set (or inverted generational distance) indicator [35], measures how "far" an approximated Pareto front is from the true Pareto front. The second one, the set coverage indicator [36], judges the dominance relations between two sets of nondominated solutions. And the third one, the maximum spread (MS) indicator [37], assesses the spread of a nondominated solution set.

1) Distance From Reference Set Indicator: The distance from reference set indicator (I_D) is used to measure the performance of both convergence and spread. Given a nondominated solution set A and a reference set P^*, I_D is defined by the following equation:

$$I_D\left(A, P^*\right) = \frac{1}{|P^*|}\sum_{y\in P^*} min_{x\in A}\ d(x, y) \tag{7}$$

where $d(x, y)$ indicates the distance between the solutions x and y, which is calculated using the relative Euclidean distance as follows:

$$\begin{aligned} d(x, y) &= \sqrt{\sum_{i}^{M}\left(\frac{f_i(x) - f_i^{\min}}{f_i^{\max} - f_i^{\min}} - \frac{f_i(y) - f_i^{\min}}{f_i^{\max} - f_i^{\min}}\right)^2} \\ &= \sqrt{\sum_{i}^{M}\left(\frac{f_i(x) - f_i(y)}{f_i^{\max} - f_i^{\min}}\right)^2}. \end{aligned} \tag{8}$$

In (8), $f_i^{\max}$ and $f_i^{\min}$ are the maximum and minimum values of the ith objective (M objectives in total) among the solutions in P^*, respectively. Note that we use the absolute values of f_2 and f_3 rather than the original values in this equation.

Ideally, the reference set P^* should be the true Pareto front. However, it is difficult to obtain the true front. In our experiment, P^* is set to the Pareto front of all the solutions obtained from the four compared algorithms. A small $I_D(A, P^*)$ indicates that front A is a good approximation of the reference set P^*. Thus, the algorithm that gives a small $I_D(A, P^*)$ value is preferred.

2) Set Coverage Indicator: Suppose that A and B are two sets of nondominated solutions, the set coverage indicator $C(A, B)$ is defined as follows:

$$C(A, B) = \frac{|\{x \in B \mid \exists y \in A : y \succeq x\}|}{|B|} \tag{9}$$

where $y \succeq x$ means that solution y dominates solution x. If $C(A, B) = 1$, every solution in B is dominated by some solution in A. If $C(A, B) = 0$, no solution in B is dominated by those in A. Note that $C(A, B)$ does not necessarily equal $1 - C(A, B)$. As suggested by Coello *et al.* [38], both $C(A, B)$ and $C(B, A)$ should be taken into consideration when comparing two nondominated sets. If $C(A, B) > C(B, A)$, A is considered better than B in terms of their dominance relations.

3) Maximum Spread Indicator: The MS indicator measures the distance between the boundary solutions in a nondominated solution set. For the nondominated solution set A, $MS(A)$ is calculated as follows:

$$MS(A) = \sqrt{\sum_{i=1}^{M}\left(\max_{x\in A}\frac{f_i(x) - f_i^{\min}}{f_i^{\max} - f_i^{\min}} - \min_{x\in A}\frac{f_i(x) - f_i^{\min}}{f_i^{\max} - f_i^{\min}}\right)^2}. \tag{10}$$

Generally, a large $MS(A)$ indicates that a wide range of objective values are covered by the nondominated solutions in A, so A may have a good extent.

C. Experimental Setup and Compared Algorithms

After some preliminary experiments, the final parameter settings for the proposed algorithm are shown in Table IV. The population size is set to 100 by a rule of thumb, which is a suitable size for decision makers to tradeoff solutions

TABLE IV
PARAMETER SETTINGS FOR THE PROPOSED ALGORITHM

Parameter	Value
Population size (POP_SIZE)	100
Number of reference points (H)	56
Recombination probability (P_r)	1.00
Mutation probability (P_m)	0.05
Local search probability (P_{ls})	0.20
Local search iteration (ls_iter)	100
Tournament size (τ)	$0.10 \times POP\_SIZE$
Execution time limit (T)	1 min. for $A1$ – $A4$, $B1$ – $B4$ 2 min. for $A5$ – $A12$, $B5$ – $B12$ 3 min. for $A13$ – $A16$, $B13$ – $B16$

TABLE V
PARAMETER TUNING FOR P_r AND P_m

P_r \ P_m	0	0.05	0.20
0	0.0572	0.0631	0.1155
0.5	0.0516	0.0492	0.0743
1	0.0505	**0.0457**	0.0701

TABLE VI
EFFECT OF THE NEIGHBORHOOD OPERATORS

N1+N2-N3-	N1-N2+N3-	N1-N2-N3+	N1+N2+N3-
0.0923	0.0842	0.1157	0.0762
N1-N2+N3+	N1+N2-N3+	N1+N2+N3+	
0.0822	0.0959	**0.0760**	

in the population. The number of reference points H is set according to Das and Dennis's systematic approach [39], i.e., $H = C_{M+p-1}^{M-1}$, where M is the number of objectives and p is some divisor. In our six-objective problem, we set $p = 3$ and thus $H = 56$.

It is worth noting that the recombination operator and/or mutation operators may have little effect during the evolution process. In our preliminary experiments, we tried to tune P_r and P_m by different combinations on the instances *#A1* and *#B1*, while keeping other parameters unchanged. For each instance and each combination, 30 independent runs were conducted. The reference set P^* for each instance is the set of nondominated solutions over all runs. The average I_D values are reported in Table V, where the smallest value is marked in bold. From this table, we can see that the recombination probability is important for achieving small I_D values. In addition, the effects of mutation probability show that a small P_m is likely to improve I_D values when P_r is large. Therefore, we set $P_r = 1$ and $P_m = 0.05$ as the final parameter settings.

We also studied the effect of different neighborhood operators; the results are presented in Table VI. In this table, $N1$–$N3$ correspond to the three categories of neighborhood operators presented in Section IV-H. The "+" sign (resp. "−" sign) means that the corresponding category of operators is included in (resp. excluded from) the mutation and local search process. The real numbers in the table are average I_D values which were obtained in a similar manner as described for Table V. As we can see, the impact of $N1$ is less significant than that of $N2$ but more significant than that of $N3$. Moreover, when all neighborhood operators were included, the corresponding I_D value is the smallest. This demonstrates the positive impact of the proposed neighborhood operators in solving the MO-SSPP.

TABLE VII
COMPARED ALGORITHMS

Label	Algorithm	Approach
NSGA-III-L NSGA-III-NL	NSGA-III with local search NSGA-III without local search	posterior
NSGA-II-L NSGA-II-NL	NSGA-II with local search NSGA-II without local search	posterior
RWGA-L RWGA-NL	RWGA with local search RWGA without local search	prior

In order to test the effectiveness of our proposed algorithm, the true Pareto front is needed. However, as the true Pareto front cannot be obtained in a reasonable time, it is impossible for us to directly measure the performance of the algorithm. Therefore, we implemented another five evolutionary algorithms for comparison. All these algorithms are listed in Table VII. In this table, NSGA-III-L is the proposed algorithm. NSGA-III-NL is the traditional NSGA-III, which can be obtained by disabling the local search component of NSGA-III-L. NSGA-II-L and NSGA-II-NL correspond to NSGA-II with and without local search, respectively. All four algorithms are posterior approaches. RWGA-L and RWGA-NL are random weighted genetic algorithms and are prior approaches. A random weighted genetic algorithm first randomly assigns relative weights to each individual. Then the selection process in each iteration preserves the most promising offspring solution for each individual. Readers are referred to [40] for more details about the random weighted genetic algorithm. Note that the parameters of these compared algorithms were set to the same values as those shown in Table IV.

The parameters for NSGA-II and RWGA are set to the same as those for NSGA-III shown in Table IV. Note that we do not need to introduce the number of reference points (H) for NSGA-II and RWGA. For each algorithm, 30 independent runs were carried out on each instance and the results were collected for the analysis.

D. Computational Results and Analysis

We conducted experiments by applying the four algorithms to 32 generated instances according to the settings given in Section V-C. For each algorithm and each instance, the results of 30 independent runs were collected. We then applied the above three indicators to test the performance of these four algorithms. Their average values are reported in Tables VIII–X.

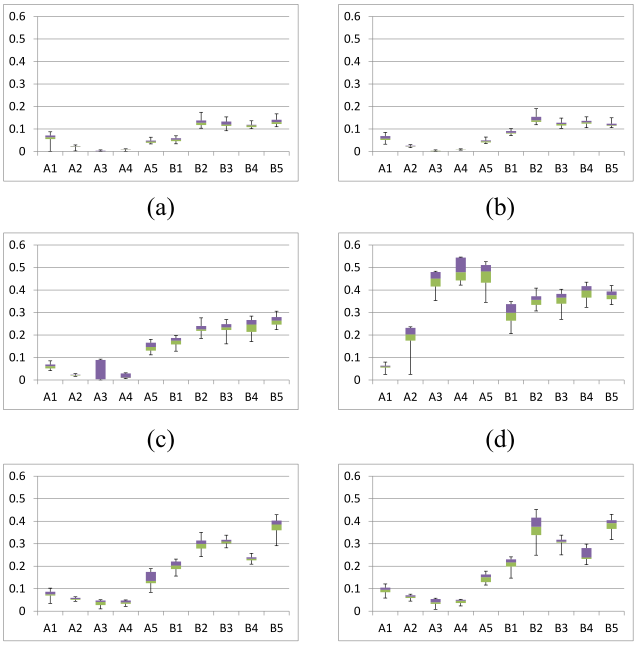
In Table VIII, the instance names are listed in the column instance. Other columns present the average values of I_D for the corresponding algorithms. We mark the value of I_D in bold if it is the smallest. In terms of the indicator I_D, the proposed algorithm (NSGA-III-L) is better than the others for 24 out of 32 instances. NSGA-III-NL performs slightly worse than NSGA-III-L, but it still outperforms the other four algorithms

TABLE VIII
COMPUTATIONAL RESULTS WITH RESPECT TO THE DISTANCE FROM REFERENCE SET INDICATOR FOR THE MO-SSPP

Instance	NSGA-III-L	NSGA-III-NL	NSGA-II-L	NSGA-II-NL	RWGA-L	RWGA-NL
A1	0.0620	**0.0588**	0.0610	0.0596	0.0782	0.0945
A2	0.0224	0.0237	**0.0219**	0.2176	0.0550	0.0659
A3	**0.0018**	0.0020	0.0485	0.4494	0.0418	0.0451
A4	**0.0087**	0.0088	0.0267	0.4915	0.0418	0.0460
A5	**0.0447**	0.0465	0.1514	0.4748	0.1486	0.1509
A6	**0.0877**	0.0973	0.2015	0.4908	0.2585	0.2562
A7	**0.1272**	0.1432	0.2564	0.3629	0.4128	0.4272
A8	**0.1324**	0.1788	0.2675	0.3267	0.5172	0.5112
A9	**0.0701**	0.0724	0.1516	0.4605	0.2751	0.2677
A10	**0.0422**	0.0605	0.1346	0.3949	0.2698	0.2542
A11	**0.0547**	0.0559	0.1521	0.4601	0.1461	0.1444
A12	0.1643	**0.1557**	0.3373	0.4612	0.3929	0.3872
A13	0.1218	**0.1200**	0.2436	0.4302	0.3599	0.3598
A14	**0.1354**	0.1466	0.2826	0.4042	0.4634	0.4684
A15	**0.0884**	0.1123	0.1804	0.2318	0.4440	0.4899
A16	**0.0911**	0.1147	0.2010	0.2548	0.3432	0.3652
Avg.	**0.0784**	0.0873	0.1699	0.3732	0.2655	0.2709
B1	**0.0525**	0.0849	0.1724	0.3007	0.2031	0.2137
B2	**0.1288**	0.1444	0.2270	0.3544	0.3006	0.3870
B3	**0.1219**	0.1227	0.2351	0.3573	0.3081	0.3091
B4	**0.1153**	0.1306	0.2399	0.3911	0.2347	0.2605
B5	0.1338	**0.1199**	0.2635	0.3823	0.3811	0.3882
B6	0.1883	**0.1800**	0.2749	0.3799	0.4664	0.4648
B7	**0.1231**	0.1232	0.2519	0.3024	0.5374	0.5722
B8	**0.1145**	0.1238	0.2443	0.2707	0.4744	0.5000
B9	**0.1408**	0.1493	0.2488	0.3545	0.4544	0.4369
B10	**0.1172**	0.1249	0.2564	0.3656	0.4519	0.4390
B11	**0.0948**	0.1013	0.2470	0.3535	0.3580	0.3488
B12	0.1224	**0.1206**	0.3044	0.3393	0.3849	0.3897
B13	0.1352	**0.1349**	0.2618	0.3233	0.4356	0.4393
B14	**0.1388**	0.1452	0.2827	0.3224	0.5094	0.5080
B15	**0.0953**	0.1018	0.1890	0.2202	0.4022	0.4603
B16	**0.0889**	0.0960	0.1932	0.2320	0.3768	0.4209
Avg.	**0.1195**	0.1252	0.2433	0.3281	0.3924	0.4087

TABLE IX
COMPUTATIONAL RESULTS WITH RESPECT TO THE SET COVERAGE INDICATOR FOR THE MO-SSPP

Instance	NSGA-III-L / NSGA-II-L		NSGA-III-L / RWGA-L		NSGA-II-L / RWGA-L	
	C(A,B)	C(B,A)	C(A,B)	C(B,A)	C(A,B)	C(B,A)
A1	**0.0056**	0.0000	**0.1689**	0.0000	**0.1677**	0.0022
A2	0.1181	**0.1207**	**0.2310**	0.0358	**0.2211**	0.0362
A3	**0.4416**	0.0384	**0.6797**	0.0044	**0.4198**	0.3403
A4	**0.4462**	0.0586	**0.3237**	0.0453	0.1990	**0.3144**
A5	**0.9814**	0.0006	**0.4136**	0.0282	0.0045	**0.6325**
A6	**0.9523**	0.0007	**0.3733**	0.0569	0.0013	**0.5763**
A7	**0.9644**	0.0034	0.0317	**0.2056**	0.0000	**0.5906**
A8	**0.9422**	0.0013	0.0284	**0.2287**	0.0000	**0.6824**
A9	**0.9336**	0.0009	**0.4874**	0.0202	0.0019	**0.4873**
A10	**0.9576**	0.0047	**0.6398**	0.0151	0.0675	**0.2893**
A11	**0.9846**	0.0006	**0.8365**	0.0112	0.0197	**0.7721**
A12	**0.9912**	0.0000	**0.3611**	0.2002	0.0000	**0.8645**
A13	**0.9633**	0.0033	**0.4854**	0.0763	0.0038	**0.7073**
A14	**0.8980**	0.0003	**0.4710**	0.0871	0.0000	**0.6610**
A15	**0.8506**	0.0046	0.1203	**0.2200**	0.0010	**0.4362**
A16	**0.9135**	0.0010	**0.2180**	0.1124	0.0000	**0.4688**
Avg.	**0.7715**	0.0149	**0.3669**	0.0842	0.0692	**0.4913**
B1	**0.9938**	0.0002	**0.8727**	0.0180	0.0000	**0.7464**
B2	**0.9289**	0.0000	**0.5653**	0.0717	0.0000	**0.7464**
B3	**0.9879**	0.0000	**0.4042**	0.0971	0.0000	**0.9311**
B4	**0.9714**	0.0001	**0.2895**	0.0843	0.0000	**0.9057**
B5	**0.9109**	0.0000	**0.2757**	0.0903	0.0000	**0.5316**
B6	**0.8674**	0.0000	0.0487	**0.1801**	0.0000	**0.6195**
B7	**0.7964**	0.0009	0.0014	**0.2128**	0.0000	**0.4921**
B8	**0.7798**	0.0001	0.0182	**0.1438**	0.0000	**0.5011**
B9	**0.8946**	0.0001	**0.1087**	0.0935	0.0000	**0.6121**
B10	**0.9180**	0.0000	**0.5789**	0.0389	0.0000	**0.6598**
B11	**0.9545**	0.0000	0.1265	**0.1823**	0.0000	**0.6878**
B12	**0.9493**	0.0000	0.0298	**0.1937**	0.0000	**0.5557**
B13	**0.8623**	0.0000	0.0134	**0.1883**	0.0000	**0.4131**
B14	**0.8846**	0.0000	0.0056	**0.2043**	0.0000	**0.6199**
B15	**0.7817**	0.0006	0.0793	**0.1572**	0.0001	**0.3191**
B16	**0.7436**	0.0012	0.0570	**0.1388**	0.0000	**0.3419**
Avg.	**0.8891**	0.0002	**0.2172**	0.1309	0.0000	**0.6052**



(a) (b)

(c) (d)

(e) (f)

Fig. 7. Box plots based on the distance from reference set indicator. (a) NSGA-III-L. (b) NSGA-III-NL.(c) NSGA-II-L. (d) NSGA-II-NL. (e) RWGA-L. (f) RWGA-NL.

for almost all instances. The observations also suggest that the local search component plays an important role in improving the solution quality.

To better represent the results with respect to the distance from reference set indicator, we use box plots to visualize the distribution of I_D values for instances $A1$–$A5$ and $B1$–$B5$ obtained with different algorithms (see Fig. 7). Each box summarizes 50% of data collected from 30 independent runs. The bottom and top of the box are the first and third quartiles. The band inside the box is the median. The figure clearly reflects that NSGA-III approaches can produce very promising nondominated solutions.

Table IX summarizes the set coverage indicator values of different algorithms. Since the algorithm with local search components is generally better than the one without, we only compare those algorithms with local search. From Table IX, we can see that the solutions produced by NSGA-III-L dominate most of the solutions produced by NSGA-II-L or RWGA-L, as $C(A, B) > C(B, A)$ holds for these instances. However, an interesting phenomenon is that for many instances, the solutions found with NSGA-II-L do not dominate any solutions found with RWGA-L. After having carefully analyzed the characteristics of these instances, we found that their solution spaces are extremely large. Because the prior approaches focus on exploring solutions for particular weight patterns, it is reasonable that RWGA-L performs better than NSGA-II-L according to the set coverage indicator. This fact also demonstrates the limitation of NSGA-II in solving many-objective optimization problems.

TABLE X
COMPUTATIONAL RESULTS WITH RESPECT TO THE MS INDICATOR FOR THE MO-SSPP

Instance	NSGA-III-L	NSGA-III-NL	NSGA-II-L	NSGA-II-NL	RWGA-L	RWGA-NL
A1	0.7803	0.7783	0.7803	0.7831	0.7775	0.7757
A2	0.8379	0.8329	0.8430	1.7489	0.8446	0.8267
A3	0.3211	0.3143	0.6045	1.9509	0.3161	0.2966
A4	0.6011	0.6056	0.7855	2.0098	0.6071	0.6052
A5	0.6775	0.6743	1.2627	1.9095	0.5784	0.5621
A6	0.8467	0.8222	1.4400	2.0133	0.4192	0.4210
A7	1.1619	1.2394	1.5267	1.8517	0.2181	0.1668
A8	1.2193	1.4215	1.5746	1.8301	0.2970	0.3684
A9	0.7469	0.8003	1.2879	1.8855	0.2962	0.2891
A10	0.7094	0.7466	1.2735	1.8776	0.1804	0.2834
A11	0.4889	0.5245	0.9920	1.7645	0.2758	0.2854
A12	0.8272	0.9667	1.4617	1.7630	0.1962	0.1633
A13	0.8844	0.9942	1.2583	1.7363	0.2870	0.2723
A14	1.0256	1.1513	1.3743	1.8033	0.2340	0.1892
A15	1.4232	1.4527	1.7305	1.9363	0.5842	0.4432
A16	1.3500	1.3693	1.6639	1.8871	0.4509	0.3518
Avg.	0.8688	0.9184	1.2412	1.7969	0.4102	0.3938
B1	0.6289	0.7881	1.4331	1.9467	0.3361	0.3574
B2	1.1478	1.2075	1.6928	2.0673	0.7860	0.7032
B3	0.7720	0.9383	1.5021	1.9882	0.3522	0.3316
B4	1.0042	1.1379	1.6028	2.0485	0.6339	0.4821
B5	0.9471	1.0184	1.6083	1.9414	0.2369	0.2226
B6	1.0572	1.1489	1.6757	2.0028	0.2202	0.2274
B7	1.2248	1.3448	1.5554	1.8346	0.1991	0.1714
B8	1.2315	1.3665	1.5943	1.7875	0.4166	0.3250
B9	1.0649	1.1518	1.5820	1.9343	0.2405	0.2872
B10	1.0952	1.1655	1.6322	1.9979	0.1547	0.2186
B11	1.0038	1.0337	1.4201	1.8609	0.1483	0.1887
B12	1.1307	1.1733	1.5475	1.8038	0.1429	0.1267
B13	1.0014	1.0904	1.5647	1.8590	0.1456	0.1415
B14	1.0920	1.2250	1.4760	1.7796	0.1517	0.1289
B15	1.3412	1.3845	1.8439	2.0029	0.4905	0.3192
B16	1.3107	1.3699	1.7311	1.9295	0.4572	0.3047
Avg.	1.0658	1.1590	1.5914	1.9241	0.3195	0.2835

TABLE XI
COMPUTATIONAL RESULTS WITH RESPECT TO THE DISTANCE FROM REFERENCE SET INDICATOR FOR THE BO-SSPP

Instance	NSGA-III-L	NSGA-III-NL	NSGA-II-L	NSGA-II-NL	RWGA-L	RWGA-NL
A1	**0.0000**	0.0111	0.0017	0.0074	0.0077	0.0231
A2	0.0189	0.0267	**0.0144**	0.0267	0.0222	0.0311
A3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
A4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
A5	**0.0231**	0.0432	0.0285	0.0434	0.0505	0.0552
A6	0.0381	0.0844	**0.0335**	0.0741	0.0433	0.0593
A7	0.0522	0.1629	**0.0514**	0.2092	0.0798	0.1248
A8	0.0442	0.1125	**0.0396**	0.1302	0.0576	0.0872
A9	0.0217	0.0849	0.0279	0.0812	**0.0171**	0.0643
A10	**0.0373**	0.0690	0.0441	0.0709	0.0406	0.0651
A11	0.0286	0.0443	**0.0232**	0.0504	0.0353	0.0326
A12	0.0285	0.0480	**0.0283**	0.0519	0.0446	0.0575
A13	0.0422	0.0705	**0.0413**	0.0697	0.0798	0.0991
A14	0.1010	0.1547	**0.0856**	0.1704	0.0967	0.1419
A15	**0.0243**	0.0577	0.0247	0.0659	0.0541	0.0840
A16	0.0306	0.0567	**0.0301**	0.0615	0.0394	0.0619
Avg.	0.0307	0.0642	**0.0296**	0.0695	0.0418	0.0617
B1	0.0478	0.0566	**0.0411**	0.0463	0.1618	0.1745
B2	0.0712	0.0910	**0.0649**	0.0729	0.3040	0.3072
B3	0.0690	0.0952	**0.0615**	0.0895	0.2675	0.2834
B4	0.0856	0.1029	**0.0649**	0.0868	0.2371	0.2423
B5	0.0827	0.0828	0.0673	**0.0605**	0.3281	0.3318
B6	0.0846	0.0953	**0.0683**	0.0868	0.3175	0.3227
B7	0.0549	0.1507	**0.0519**	0.1521	0.1513	0.2005
B8	0.0686	0.1219	**0.0644**	0.1198	0.1813	0.1989
B9	0.0574	0.0709	**0.0559**	0.0605	0.2202	0.2175
B10	0.0554	0.0586	**0.0503**	0.0630	0.2138	0.1994
B11	0.0702	0.0853	**0.0656**	0.0801	0.2739	0.2896
B12	0.0556	0.0735	**0.0492**	0.0725	0.1897	0.2062
B13	0.0539	0.0743	**0.0445**	0.0617	0.2078	0.2206
B14	0.0754	0.1020	**0.0681**	0.0957	0.1648	0.1786
B15	0.0316	0.0663	**0.0276**	0.0646	0.0785	0.1158
B16	0.0328	0.0638	**0.0314**	0.0616	0.0700	0.0972
Avg.	0.0623	0.0869	**0.0548**	0.0796	0.2105	0.2241

We then report the results for the MS indicator in Table X. As we can see, NSGA-II-NL achieves the largest *MS* values on average, and NSGA-II-L the second largest. Comparatively, the *MS* values obtained by RWGA-L and RWGA-NL are much smaller. NSGA-III approaches perform better than RWGA approaches but worse than NSGA-II approaches. These results show that NSGA-II-L and NSGA-II-NL spread over a wider extent. Decision makers may have more flexibility in trading off objectives among different solutions.

In reality, when the dimension of the objective space is high (e.g., greater than three), it is usually difficult for the decision makers to get an intuitive impression of the Pareto-optimal solutions. In what follows, we will reduce the dimension of the objective space and discuss a bi-objective optimization SSPP (BO-SSPP). For this problem, we can easily visualize the Pareto-optimal solutions in a planar graph.

Since there are six objectives that can be classified into two categories, we decided to manually set the relative weights for each category. A typical weight setting results in the following BO-SSPP:

$$\begin{aligned} \min \quad & \{g_1(S) = -f_1(S) + |f_2(S)| + |f_3(S)|, \\ & \quad g_2(S) = f_4(S) + 3f_5(S) + 2f_6(S)\} \qquad (11) \\ \text{s.t.} \quad & S \in \mathbb{S}. \qquad (12) \end{aligned}$$

In (11), $g_1(S)$ is the integrated stability function and $g_2(S)$ is the integrated rehandle function. For the function $g_1(S)$, the weights of vertical stability, traverse stability, and longitudinal stability are set to the same values. For the function $g_2(S)$, the weight of yard container rehandles is the smallest, which is one third of the weight of yard container rehandles and is one half of the weight of future port rehandles.

We have modified all of the above algorithms to make them capable of solving the BO-SSPP. In fact, only the dominance strategy part and the evaluation function part need to be rewritten. The experiments were also carried out using the same parameter settings given in Table IV except that H is set to $C_{99+2-1}^{2-1} = 100$ (100 reference points).

Analogous to Tables VIII–X, the computational results to the three indicators are reported in Tables XI–XIII. Table XI shows that NSGA-II-L achieves the best I_D values for 24 out of 32 instances compared to the other algorithms, so this algorithm has a greater advantage in approximating the true Pareto front when solving these instances. NSGA-III-L performs slightly worse than NSGA-II-L. This demonstrates that the Pareto-dominance relation method can provide stronger selection pressure than the reference point method for the problem with a small dimensional objective space. Note that for the instances $A3$ and $A4$, all six algorithms achieved the same nondominated solution (only one nondominated solution was found), so their I_D values are all equal to 0.

Table XII shows that NSGA-III-L beats NSGA-II-L for 13 instances but NSGA-II-L beats NSGA-III-L for 17 instances according to the set coverage indicator. Except for instances $A3$, $A4$, $A9$, $A15$, $A16$, $B15$, and $B16$, the nondominated solutions produced by NSGA-III-L and NSGA-II-L are more likely

TABLE XII
COMPUTATIONAL RESULTS WITH RESPECT TO THE SET COVERAGE INDICATOR FOR THE BO-SSPP

Instance	NSGA-III-L / NSGA-II-L		NSGA-III-L / RWGA-L		NSGA-II-L / RWGA-L	
	C(A,B)	C(B,A)	C(A,B)	C(B,A)	C(A,B)	C(B,A)
A1	**0.0222**	0.0000	**0.1000**	0.0000	**0.0933**	0.0156
A2	0.0626	**0.1070**	**0.0963**	0.0630	**0.1259**	0.0481
A3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
A4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
A5	**0.3103**	0.2009	**0.5249**	0.0268	**0.5182**	0.0318
A6	0.0491	**0.1576**	**0.1482**	0.0540	**0.1920**	0.0049
A7	0.3448	**0.5062**	**0.5024**	0.1718	**0.4979**	0.1795
A8	0.3101	**0.5775**	**0.3762**	0.3122	**0.4839**	0.2288
A9	**0.1302**	0.0761	0.0902	**0.0909**	0.0757	**0.1383**
A10	**0.3470**	0.2252	**0.3928**	0.2139	**0.3367**	0.2939
A11	0.2603	**0.3233**	**0.3207**	0.2314	**0.3551**	0.1934
A12	**0.4081**	0.3960	**0.4642**	0.2704	**0.4448**	0.2966
A13	0.3648	**0.3899**	**0.3854**	0.1910	**0.3964**	0.2007
A14	0.2832	**0.6044**	**0.3700**	0.2962	**0.5110**	0.2037
A15	**0.4863**	0.4404	0.3430	**0.3633**	0.3084	**0.3881**
A16	0.4465	**0.4733**	0.2434	**0.4625**	0.2564	**0.4146**
Avg.	0.2391	**0.2799**	**0.2724**	0.1717	**0.2872**	0.1649
B1	**0.4249**	0.4156	**0.6771**	0.0814	**0.6734**	0.0683
B2	0.4043	**0.5010**	**0.6737**	0.0581	**0.6923**	0.0508
B3	**0.4551**	0.4377	**0.2647**	0.1347	**0.1821**	0.1444
B4	0.4107	**0.4734**	**0.4541**	0.1297	**0.3815**	0.1172
B5	**0.4882**	0.4161	**0.5602**	0.0668	**0.5193**	0.0653
B6	0.3968	**0.5053**	**0.3477**	0.1185	**0.3457**	0.1062
B7	0.3888	**0.5501**	**0.2737**	0.1624	**0.2292**	0.1753
B8	0.3880	**0.5362**	**0.1993**	0.1812	**0.1975**	0.1852
B9	**0.4879**	0.4315	**0.4290**	0.1136	**0.3732**	0.1098
B10	0.4119	**0.5129**	**0.5386**	0.0640	**0.5128**	0.0598
B11	0.4267	**0.5002**	**0.2194**	0.1258	**0.2440**	0.1048
B12	**0.4773**	0.4644	**0.3407**	0.1915	**0.2875**	0.2021
B13	**0.4711**	0.4571	**0.2613**	0.1419	**0.2564**	0.1527
B14	0.4641	**0.4664**	**0.2797**	0.1812	**0.2850**	0.1626
B15	**0.5187**	0.4191	0.3064	**0.3535**	0.2565	**0.3674**
B16	0.4596	**0.4670**	0.1928	**0.3872**	0.1623	**0.3770**
Avg.	0.4421	**0.4721**	**0.3762**	0.1557	**0.3499**	0.1531

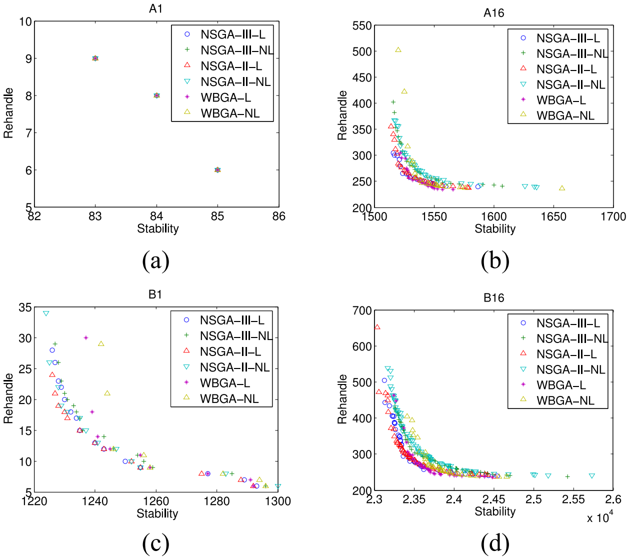
to dominate those produced by RWGA-L. From Table XIII, we can see that the NSGA-III and NSGA-II approaches produced better *MS* values than the RWGA approaches for most of the instances.

In Fig. 8, we illustrate the nondominated solutions produced by different algorithms for four instances (*A*1, *A*16, *B*1, and *B*16). For each algorithm and each instance, the nondominated solutions were collected based on the results from 30 independent runs ($30 \times 100 = 3000$ solutions in total). They are plotted using different markers. Note that the dominated solutions were discarded and not plotted. As we can see, the graphical results agree with the results in Tables VIII–X. The NSGA-III and NSGA-II approaches can cover the objective space more completely and coherently than the RWGA approaches.

To sum up, considering both convergence and diversity, no one algorithm is strictly better than the others. However, the above results confirmed that the proposed algorithm can produce a set of nondominated solutions with excellent quality and satisfactory diversity, especially for SSPP with high-dimensional objective space. Overall, we recommend using NSGA-III-L in practice.

TABLE XIII
COMPUTATIONAL RESULTS WITH RESPECT TO THE MS INDICATOR FOR THE BO-SSPP

Instance	NSGA-III-L	NSGA-III-NL	NSGA-II-L	NSGA-II-NL	RWGA-L	RWGA-NL
A1	0.2442	0.2762	0.2492	0.2620	0.2663	0.3109
A2	0.3673	0.3899	0.3544	0.3899	0.3770	0.4029
A3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
A4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
A5	0.4212	0.4674	0.4257	0.4595	0.5015	0.5173
A6	0.5621	0.4701	0.5436	0.4869	0.5835	0.5287
A7	0.2130	0.5515	0.2276	0.7097	0.1104	0.1338
A8	0.2099	0.3912	0.1702	0.4312	0.1424	0.1361
A9	0.3082	0.1413	0.2824	0.1565	0.3188	0.1783
A10	0.0641	0.1335	0.0671	0.1515	0.0896	0.1681
A11	0.3899	0.4244	0.3817	0.4408	0.3813	0.3762
A12	0.3510	0.4399	0.3551	0.4494	0.3040	0.3537
A13	0.6050	0.6072	0.5981	0.6110	0.5563	0.5376
A14	0.5267	0.4638	0.5530	0.4011	0.4964	0.4316
A15	0.6578	0.7583	0.6632	0.8369	0.5143	0.5975
A16	0.6363	0.7669	0.5375	0.8240	0.3254	0.4427
Avg.	0.3473	0.3926	0.3380	0.4132	0.3104	0.3197
B1	0.5457	0.6806	0.6009	0.6834	0.3179	0.3332
B2	0.7804	0.8541	0.7972	0.9181	0.3319	0.3718
B3	0.5027	0.7470	0.5635	0.8345	0.0617	0.0982
B4	0.3622	0.5586	0.4296	0.6841	0.0639	0.1218
B5	0.7296	0.8321	0.8094	0.9740	0.2959	0.3291
B6	0.6847	0.8258	0.7579	0.9344	0.2047	0.2778
B7	0.5420	0.8166	0.5763	0.8842	0.1454	0.2116
B8	0.5111	0.7325	0.5192	0.8162	0.2311	0.2435
B9	0.9353	0.8616	0.9262	0.9144	0.5973	0.5734
B10	0.6201	0.7407	0.6756	0.7703	0.3157	0.3963
B11	0.6695	0.8008	0.7611	0.9495	0.1641	0.2663
B12	0.7338	0.8014	0.8067	0.8696	0.3661	0.4136
B13	0.6883	0.7846	0.7467	0.9170	0.3171	0.3815
B14	0.6615	0.7105	0.7369	0.8263	0.5140	0.4900
B15	0.7375	0.7190	0.8335	0.8082	0.5888	0.5607
B16	0.7488	0.7642	0.7877	0.8766	0.4070	0.4171
Avg.	0.6533	0.7644	0.7080	0.8538	0.3077	0.3429



(a) (b)

(c) (d)

Fig. 8. Nondominated solutions produced by different algorithms on instances (a) A1, (b) A16, (c) B1, and (d) B16 for the BO-SSPP.

VI. CONCLUSION

In this paper, we study an MO-SSPP which simultaneously deals with ship stability and container rehandles. This problem is close to the realistic logistics environment. We consider six objectives in the problem: 1) metacentric height; 2) list; 3) trim; 4) yard container rehandles; 5) ship container rehandles; and 6) future port rehandles, where the first three objectives are related to ship stability and the last three to container rehandles. A tailored multiobjective evolutionary algorithm, which is a variant of the NSGA III, is proposed for solving this problem. The algorithm makes use of well-designed operators for evolving the population. In addition, it features a local search component to help improve the solution quality. To evaluate the performance of the algorithm, we conducted extensive experiments on two groups of instances generated by Bortfeldt and Forster [34]. Another five algorithms were also tested for comparison. The computational results verify the effectiveness of the proposed algorithm, especially when it is applied to solving a many-objective SSPP.

In practice, ship stowage planning is a very complex process. It relates to the benefits of both shipping lines and port terminals. Our approach can provide several tradeoff solutions to cater to the needs from different perspectives. Decision makers can select from among these solutions based on their experience.

Three extensions of the SSPP deserve further study. First, we can incorporate crane scheduling into the stowage planning process for a more realistic problem. Second, we can consider stowing a container ship during its multiport voyage. Given the information of export containers at each port, we can optimize the overall ship stability and the total number of container rehandles along the ship's whole journey. Third, the uncertainty issue in the ship stowage planning process can be better studied. In practice, the ship stability and the number of rehandles cannot be exactly measured beforehand. Robust optimization may be applied to solve this problem.

ACKNOWLEDGMENT

The authors would like to thank the anonymous referees for their helpful comments.

REFERENCES

[1] R. Stahlbock and S. Voß, "Operations research at container terminals: A literature update," *OR Spectr.*, vol. 30, no. 1, pp. 1–52, 2008.

[2] D. Steenken, S. Voß, and R. Stahlbock, "Container terminal operation and operations research—A classification and literature review," *OR Spectr.*, vol. 26, no. 1, pp. 3–49, 2004.

[3] J. F. Kemp and P. Young, *Ship Stability Notes and Examples*. London, U.K.: Stanford Maritime, 1971.

[4] V. L. Belenky, N. B. Sevastianov, R. Bhattacharyya, and M. E. McCormick, *Stability and Safety of Ships: Risk of Capsizing*, Jersey City, NJ, USA: Soc. Naval Archit. Marine Eng., 2007.

[5] Z. Jovanoski and G. Robinson, "Ship stability and parametric rolling," *Aust. J. Eng. Educ.*, vol. 15, no. 2, pp. 43–50, 2009.

[6] M. Y. H. Low *et al.*, "Improving safety and stability of large containerships in automated stowage planning," *IEEE Syst. J.*, vol. 5, no. 1, pp. 50–60, Mar. 2011.

[7] A. Imai, K. Sasaki, E. Nishimura, and S. Papadimitriou, "Multi-objective simultaneous stowage and load planning for a container ship with container rehandle in yard stacks," *Eur. J. Oper. Res.*, vol. 171, no. 2, pp. 373–389, 2006.

[8] W. C. Webster and P. Van Dyke, "Container loading: A container allocation model I and II: Introduction, background, strategy, conclusion," in *Proc. Comput.-Aided Ship Design Eng. Summer Conf.*, Detroit, MI, USA, 1970.

[9] J. J. Shields, "A computer aided preplanning system," *Marine Technol.*, vol. 21, no. 4, pp. 370–383, 1984.

[10] M. Avriel, M. Penn, and N. Shpirer, "Container ship stowage problem: Complexity and connection to the coloring of circle graphs," *Discrete Appl. Math.*, vol. 103, no. 1, pp. 271–279, 2000.

[11] D. Ambrosino, A. Sciomachen, and E. Tanfani, "Stowing a containership: The master bay plan problem," *Transp. Res. A Policy Pract.*, vol. 38, no. 2, pp. 81–99, 2004.

[12] M. Avriel, M. Penn, N. Shpirer, and S. Witteboon, "Stowage planning for container ships to reduce the number of shifts," *Ann. Oper. Res.*, vol. 76, pp. 55–71, Jan. 1998.

[13] D. Ambrosino, A. Sciomachen, and E. Tanfani, "A decomposition heuristics for the container ship stowage problem," *J. Heuristics*, vol. 12, no. 3, pp. 211–233, 2006.

[14] D. Ambrosino, D. Anghinolfi, M. Paolucci, and A. Sciomachen, "A new three-step heuristic for the master bay plan problem," *Maritime Econ. Log.*, vol. 11, no. 1, pp. 98–120, 2009.

[15] F. Li, C. Tian, R. Cao, and W. Ding, "An integer linear programming for container stowage problem," in *Computational Science—ICCS 2008*. Berlin, Germany: Springer, 2008, pp. 853–862.

[16] A. Delgado, R. M. Jensen, K. Janstrup, T. H. Rose, and K. H. Andersen, "A constraint programming model for fast optimal stowage of container vessel bays," *Eur. J. Oper. Res.*, vol. 220, pp. 251–261, Jul. 2012.

[17] I. D. Wilson and P. A. Roach, "Container stowage planning: A methodology for generating computerised solutions," *J. Oper. Res. Soc.*, vol. 51, no. 11, pp. 1248–1255, 2000.

[18] J.-G. Kang and Y.-D. Kim, "Stowage planning in maritime container transportation," *J. Oper. Res. Soc.*, vol. 53, no. 4, pp. 415–426, 2002.

[19] D. Pacino, A. Delgado, R. M. Jensen, and T. Bebbington, "Fast generation of near-optimal plans for eco-efficient stowage of large container vessels," in *Computational Logistics*. Berlin, Germany: Springer, 2011, pp. 286–301.

[20] M. Gumus, P. Kaminsky, E. Tiemroth, and M. Ayik, "A multi-stage decomposition heuristic for the container stowage problem," in *Proc. MSOM Conf.*, 2008.

[21] A. Sciomachen and E. Tanfani, "The master bay plan problem: A solution method based on its connection to the three-dimensional bin packing problem," *IMA J. Manage. Math.*, vol. 14, no. 3, pp. 251–269, 2003.

[22] I. D. Wilson and P. A. Roach, "Principles of combinatorial optimization applied to container-ship stowage planning," *J. Heuristics*, vol. 5, no. 4, pp. 403–418, 1999.

[23] O. Dubrovsky, G. Levitin, and M. Penn, "A genetic algorithm with a compact solution encoding for the container ship stowage problem," *J. Heuristics*, vol. 8, no. 6, pp. 585–599, 2002.

[24] D. Ambrosino, D. Anghinolfi, M. Paolucci, and A. Sciomachen, "An experimental comparison of different heuristics for the master bay plan problem," in *Experimental Algorithms*. Berlin, Germany: Springer, 2010, pp. 314–325.

[25] F. Liu *et al.*, "Randomized algorithm with tabu search for multi-objective optimization of large containership stowage plans," in *Computational Logistics*. Berlin, Germany: Springer, 2011, pp. 256–272.

[26] Y. Lee and N.-Y. Hsu, "An optimization model for the container pre-marshalling problem," *Comput. Oper. Res.*, vol. 34, no. 11, pp. 3295–3313, 2007.

[27] W. Zhu, H. Qin, A. Lim, and H. Zhang, "Iterative deepening a* algorithms for the container relocation problem," *IEEE Trans. Autom. Sci. Eng.*, vol. 9, no. 4, pp. 710–722, Oct. 2012.

[28] M. Caserta, S. Schwarze, and S. Voß, "A mathematical formulation and complexity considerations for the blocks relocation problem," *Eur. J. Oper. Res.*, vol. 219, no. 1, pp. 96–104, 2012.

[29] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[30] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, Part I: Solving problems with box constraints," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 577–601, Aug. 2014.

[31] M. Caserta, S. Voß, and M. Sniedovich, "Applying the corridor method to a blocks relocation problem," *OR Spectr.*, vol. 33, no. 4, pp. 915–929, 2011.

[32] P. Moscato and C. Cotta, "A gentle introduction to memetic algorithms," in *Handbook of Metaheuristics*. New York, NY, USA: Springer, 2003, pp. 105–144.

[33] B. L. Miller and D. E. Goldberg, "Genetic algorithms, tournament selection, and the effects of noise," *Complex Syst.*, vol. 9, no. 3, pp. 193–212, 1995.

[34] A. Bortfeldt and F. Forster, "A tree search procedure for the container pre-marshalling problem," *Eur. J. Oper. Res.*, vol. 217, no. 3, pp. 531–540, 2012.

[35] C. A. C. Coello and N. C. Cortés, "Solving multiobjective optimization problems using an artificial immune system," *Genet. Program. Evol. Mach.*, vol. 6, no. 2, pp. 163–190, 2005.

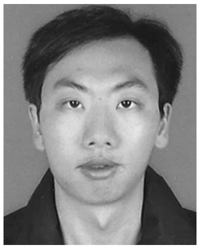
[36] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 117–132, Apr. 2003.

[37] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evol. Comput.*, vol. 8, no. 2, pp. 173–195, 2000.

[38] C. A. C. Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, vol. 242. New York, NY, USA: Springer, 2002.

[39] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems," *SIAM J. Optim.*, vol. 8, no. 3, pp. 631–657, 1998.

[40] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*, Chichester, U.K.: Wiley, 2001.



Zizhen Zhang received the B.S. and M.S. degrees with Sun Yat-sen University, Guangzhou, China, in 2007 and 2009, respectively, and the Ph.D. degree with the City University of Hong Kong, Hong Kong, in 2014.

He is currently an Assistant Professor with Sun Yat-sen University, and a Visiting Scholar with the Hong Kong University of Science and Technology, Hong Kong. His current research interests include computational intelligence and its applications in industrial engineering and operations research.

Chung-Yee Lee received the B.S. degree in electronic engineering and the M.S. degree in management sciences from National Chiao-Tung University, Hsinchu, Taiwan, in 1972 and 1976, respectively, the M.S. degree in industrial engineering from Northwestern University, Xi'an, China, in 1980, and the Ph.D. degree in operations research from Yale University, New Haven, CT, USA, in 1984.

He is currently a Chair Professor/Cheong Ying Chan Professor of Engineering with the Department of Industrial Engineering and Logistics Management, Hong Kong University of Science and Technology, Hong Kong, where he is also the Founding and Current Director of the Logistics and Supply Chain Management Institute and served as the Department Head from 2001 to 2008. He has published over 140 papers in refereed journals. According to an article in the *International Journal of Production Economics* in 2009, which looked at all papers published in the 20 core journals during the last 50 years in the field of production and operations management, he was ranked no. 6 among all researchers worldwide in h-index. His current research interests include logistics and supply chain management, scheduling, and inventory management.

Prof. Lee is a Fellow of the Institute of Industrial Engineers in U.S. and the Hong Kong Academy of Engineering Sciences.