Soft Computing (2021) 25:4465–4487
https://doi.org/10.1007/s00500-020-05456-9

METHODOLOGIES AND APPLICATION

Integrated optimization of feeder routing and stowage planning for containerships

Mingjun Ji[1] · **Lingrui Kong**[1] · **Yunxiao Guan**[1]

Published online: 7 January 2021

Abstract
The sailing safety constraints of containerships were ignored in the previous studies on feeder containership routing problems; however, they are especially critical for small-sized feeder containerships. In this study, we optimize feeder routes while incorporating stowage plans to address the sailing safety of containerships. Firstly, a mixed integer nonlinear programming model for integrated optimization is formulated. Next, a heuristic algorithm is designed, by which feeder routes can be updated through a variable neighborhood search, and stowage plans are obtained using a genetic algorithm. Finally, through the computational study, we confirm that the integrated optimization can meet the sailing safety requirements of containerships and effectively reduce the total cost of the feeder service. Moreover, through the sensitivity analysis, we discuss the performance robustness of the proposed algorithm and further demonstrate the significance of this study.

Keywords Sailing safety · Integrated optimization · Feeder routing · Stowage planning

1 Introduction

With the development of containership transportation, container shipping companies began to deploy large containerships to benefit from economies of scale. As large containerships can only be berthed in major hub ports with deep waters, a large containership typically serves a main route comprising some hub ports, while small ships are deployed to serve feeder routes with spoke ports (Zheng et al. 2015). This is called a hub-and-spoke (H&S) shipping network. Research on H&S shipping networks has mainly focused on liner transportation among hub ports (Hsu and Hsieh 2007; Gelareh and Pisinger 2011; Gelareh et al. 2013; Fontes and Goncalves 2017). Few studies have focused on the feeder service (spoke-level route). The feeder service is in charge of distributing the import volumes destined to all the spoke ports from the hub port and simultaneously collecting the export volumes from the spoke ports to be delivered to the hub port for further transportation. The main problem that needs to be addressed during a feeder service is the determination of feeder routes for containerships (i.e., the feeder routing problem).

Sambracos et al. (2004) thought of this problem as an extension of the vehicle routing problem (VRP) and formulated it as a capacitated VRP. Suban and Twrdy (2008) considered this problem as a VRP with both pickup and delivery and modeled it on a graph. Subsequently, Karlaftis et al. (2009) considered more realistic conditions while incorporating the time deadline constraints. Zhang et al. (2015) presented a feeder routing optimization method for containerships through an intelligent electronic chart and information system to minimize the sailing time of the containerships. Considering the size of the containership is also an important factor that may influence the cost of the feeder routing plans, Ji et al. (2015) established a routing optimization model for multi-type containerships.

Based on earlier studies, we know that the feeder routing problem is similar to network design problems, and in most related studies, this problem has been regarded as a VRP, with capacity and time window constraints. However, for the sailing of containerships in the real world, only meeting the capacity constraint of the containership is insufficient; some safety constraints, such as stability, strength, and trim, must be satisfied to ensure the sailing safety of the containership. These sailing safety constraints are especially critical for

Communicated by V. Loia.

✉ Mingjun Ji
jmj@dlmu.edu.cn

[1] Department of Transportation Engineering, Dalian Maritime University, Lingshui Road No.1th, Ganjingzi District, Dalian 116026, Liaoning Province, China

feeder containerships because they are usually small and have stricter safety requirements that must be satisfied to prevent overturning. Sailing safety indices are related to the loading locations of the containers on the containership, i.e., the stowage plan. If a feeder routing problem is addressed without considering the stowage plan, the resultant feeder routing plan may not satisfy the sailing safety requirements of the containership and may also result in a large number of container rehandles owing to an unfavorable arriving sequence along a route of the containership. Container rehandles occur when containers that are not destined to be discharged at a particular port need to be moved, to reach other containers that are required to be unloaded at that port. A large number of container rehandles will significantly increase the operational time and service cost of a containership at ports. Therefore, the stowage plan of the containership must be considered jointly with the optimization of the feeder containership routes.

In an H&S shipping network, many feeder ports are connected to a hub port; hence, the containership stowage planning problem mentioned here is actually a multi-port stowage planning problem (MP-SPP), which determines the containership stowage plan for each port along the route. Studies dealing with MP-SPPs were conducted by Wilson and Roach (2000), Ambrosino et al. (2015), Ambrosino et al. (2017), and Zhang et al. (2018). In a feeder service of the H&S network, no direct cargo flow exists between any two feeder ports (Alumur and Kara (2008)); therefore, it should be noted that the MP-SPP addressed here is different from that addressed in the aforementioned studies.

For the above analysis, we assert that it is necessary to optimize the feeder routes while incorporating the stowage plans to ensure the sailing safety of the containerships and reduce the total cost of the entire feeder service; this constitutes the scope of the current work. This study differs from other related research on the feeder containership routing problem in the following respects: (1) the feeder routing and stowage planning problems are optimized jointly, in which the sailing safety constraints are imposed for the containership, while in other feeder routing studies, only the capacity constraint of the containership was focused on, which is evidently insufficient for the navigation of the feeder containerships in reality; (2) the objective of this study is to minimize the transportation cost plus the service cost at each port, which includes the cost of container rehandles, while in other studies, the latter was ignored; (3) a novel formulation is developed for the integrated optimization problem, and an efficient heuristic algorithm is proposed for solving the problem.

The remainder of this paper is organized as follows. In Sect. 2, a mixed integer programming model for the integrated optimization of feeder routing and stowage planning of containerships is presented. In Sect. 3, a tailored heuristic algorithm is developed to solve the problem, in which the feeder routes are updated through a variable neighborhood search, and a special MP-SPP is described and solved by a genetic algorithm. The computational experiments and corresponding discussions are reported in Sect. 4. Finally, in Sect. 5, we summarize, draw conclusions, and provide suggestions for further research directions.

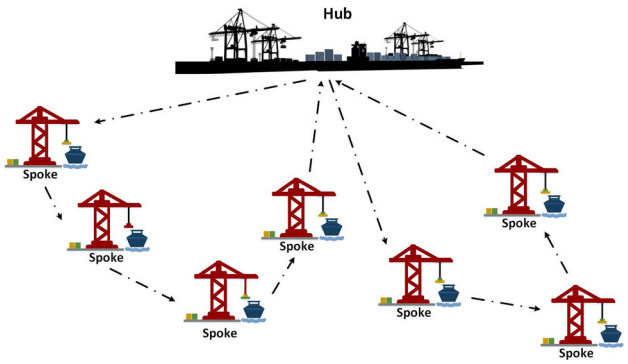

Fig. 1 Structure of a feeder shipping network

2 Model formulation

2.1 Problem description

As shown in Fig. 1, a feeder shipping network contains a hub port and several spoke ports. The feeder shipping company is responsible for distributing the containers from the hub port to the spoke ports and, at the same time, collecting the containers from each spoke port to the hub port for further transportation. The feeder routing problem involves assigning each spoke port that has transportation tasks with a specific containership and determining the arriving sequence of the containership along its route to minimize transportation costs, while satisfying the ship capacity constraints and time deadlines.

Sometimes, for a given feeder route of a containership, a feasible stowage plan cannot be obtained owing to violations of sailing safety constraints. In addition, an unfavorable arriving sequence of the containership may cause a large number of container rehandles, which have a significant impact on the service time and cost of the containership at the ports. Therefore, in this integrated optimization model, the feeder routes and related stowage plans of containerships are determined simultaneously, and the model aims to minimize the total cost of the feeder service, while ensuring the sailing safety requirements of the containerships.

The flows of containers for a feeder route are shown in Fig. 2 (ports 1 and 6 are hub ports). No container is directly transferred between any two feeder ports, and the containership must start from the hub port and finally return to the hub port.

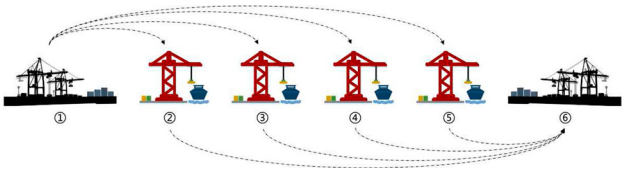

Fig. 2 Container flows for a feeder route

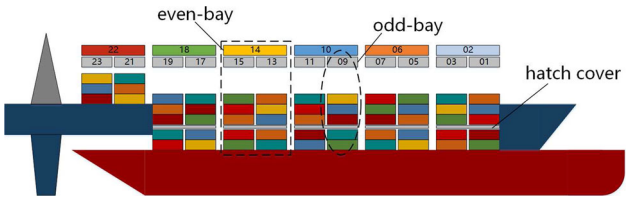

Fig. 3 Placement of a containership with a capacity of 200TEUs

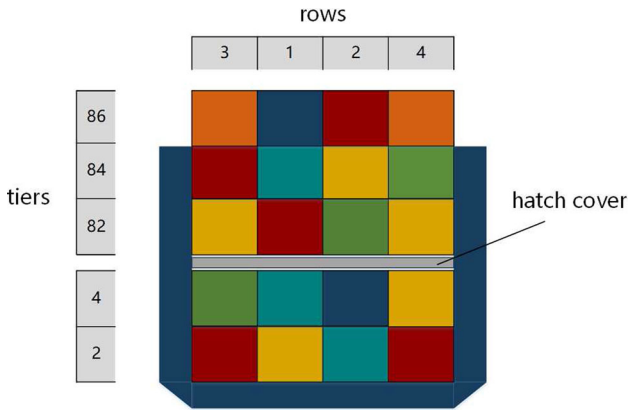

Fig. 4 Cross-sectional view of a bay

Figure 3 shows the placement of a feeder containership with a capacity of 200TEUs, i.e., it shows how containers are arranged in storage areas called bays, along the entire length of the vessel. Figure 4 shows the cross-sectional view of a bay. As shown in Fig. 4, a bay is composed of a number of slots. A slot in a bay is identified by a row number, which indicates its horizontal position within a bay, and a tier number, which indicates its vertical position within a bay. Therefore, a slot in a containership can be identified by its bay, row, and tier number.

In general, there is a distinction between the on-deck and below-deck areas of a bay. The below-deck (in holds) areas are closed by hatch covers, which are tight metallic structures that prevent water from entering. For the containership shown in Fig. 3, two adjacent odd-bays (or one even-bay) are (is) covered by a hatch cover. When unloading/loading a container from/into a slot in holds, the corresponding hatch cover must be opened, and containers loaded on-deck (above that hatch cover) must be removed (rehandled). Such container rehandles triggered by removing the hatch cover and regular container rehandles (i.e., moving containers onboard that are not destined to be discharged at a particular port to access others that are to be unloaded at that port) are both considered in the model. If a container is rehandled at a feeder port, it can be reloaded in a different slot of the containership.

Moreover, to add the adaptability to the model for practical applications, multi-type containerships with different capacities, speeds, and sailing costs are considered.

2.2 Model formulation

The notations used in the model are described as follows:

O: hub port (initial origination of all containerships);
O': virtual hub port (final destination of all containerships);
Ω: set of feeder ports;
K: set of containerships, where k is a containership belonging to K;
S_k: set of slots in containership k, s is a slot belonging to S_k;
G_k^0: set of slots on the deck (above hatch covers) of containership k;
G_k^1: set of slots in the holds (below hatch covers) of containership k;
$\theta(s)$: set of slots below the hatch cover that is related to slot s; more specifically, if slot s is above a hatch cover, then all the slots covered by that hatch cover are added to set $\theta(s)$, where $s \in G_k^0$;
Q_{ij}: set of containers required to be transported from port i to port j;
$\pi(s)$: set of slots that are directly below slot s;
e_i: earliest arrival time at feeder port i;
h_i: latest arrival time at feeder port i;
l_i: latest arrival time at the hub port for containers from feeder port i;
d_{ij}: distance between ports i and j (unit: mile);
v_k: speed of containership k (unit: knot);
C_k: loading capacity of containership k;
p_i: number of containers required to be unloaded at feeder port i;
q_i: number of containers required to be loaded at feeder port i;
w_c^{ij}: weight of container c originating from port i and with the destination of port j;
st_{ki}: time required to unload/load one container from/into containership k at port i (unit: h);
sc_{ki}: service cost for containership k at port i (unit: thousand USD/h);
f_{kij}: cost for containership k sailing from port i to port j (unit: thousand USD);
M: a sufficiently large positive number;

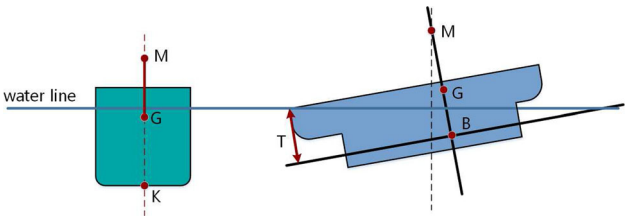

Fig. 5 GM and trim

GM: distance between the center of gravity (G) and the metacenter (M) of the ship, it is used for measuring the stability of the containership; as shown in Fig. 5, $GM = KM - KG$, where KM is the transverse metacenter above the base line, and KG is the vertical distance between the center of gravity and the base line;
GM_k^0: minimum GM value of containership k;
GM_k^1: maximum GM value of containership k;
Ms: hydrostatic bending moment at the middle of containership, it is an important index for measuring the longitudinal strength of the ship, and it is usually greater than 0;
Ms_k: maximum Ms value of containership k;
T: trim of the containership, which is defined as the difference between the draft at the stern and bow (T can also be illustrated as in Fig. 5, where B is the center of buoyancy); it needs to be as small as possible, such that it can be easily adjusted by the ballast;
T_k: maximum absolute value of T for containership k.

The decision variables are as follows:

x_{kij}: a binary, where $x_{kij} = 1$ if containership k travels directly from port i to j;
y_{ijc}^{bks}: a binary, where $y_{ijc}^{bks} = 1$ if container c originating from port i to port j is stowed at slot s of containership k at port b (after unloading and loading operations);
r_b^{ks}: a binary, where $r_b^{ks} = 1$ if slot s of containership k is occupied at port b (after unloading and loading operations);
e_b^{ks}: a binary, where $e_b^{ks} = 1$ if the container in slot s of containership k required to be rehandled at port b;
re_{kb}: number of container rehandles of containership k at port b, which can be calculated as $re_{kb} = \sum_{s \in S_k} e_b^{ks}$;
W_b^{ks}: deadweight of slot s in containership k at port b (after unloading and loading operations);
t_{ki}: arrival time of containership k at port i;
u_{ki}: load of containership k after leaving port i;
g_{ki}: service time of containership k at port i.

Using the above notations, the integrated optimization model can be presented as follows:

$$min \sum_{k\in K}\sum_{i\in\Omega\cup O}\sum_{j\in\Omega\cup O'\setminus i}(f_{kij}\times x_{kij}+(p_i+q_i)\times sc_{ki} \times st_{ki}\times x_{kij})+\sum_{k\in K}\sum_{b\in\Omega} re_{kb}\times sc_{kb}\times st_{kb}\times 2 \tag{1}$$

$$\sum_{j\in\Omega\cup O'} x_{kOj}=1 \quad k\in K \tag{2}$$

$$\sum_{i\in\Omega\cup O\setminus j} x_{kij}-\sum_{b\in\Omega\cup O'\setminus j} x_{kjb}=0 \quad k\in K, j\in\Omega \tag{3}$$

$$\sum_{k\in K}\sum_{i\in\Omega\cup O\setminus j} x_{kij}=1 \quad j\in\Omega \tag{4}$$

$$g_{kO}=\sum_{i\in\Omega\cup O}\sum_{j\in\Omega\setminus i} x_{kij}\times p_j\times st_{kO} \quad k\in K \tag{5}$$

$$g_{ki}=(p_i+q_i+re_{ki}\times 2)\times st_{ki} \quad k\in K, i\in\Omega \tag{6}$$

$$t_{ki}+g_{ki}+\frac{d_{ij}}{v_k}-t_{kj}\le(1-x_{kij})\times M \quad k\in K, i\in\Omega\cup O, j\in\Omega\cup O'\setminus i \tag{7}$$

$$e_i\le t_{ki}\le h_i \quad i\in\Omega, k\in K \tag{8}$$

$$t_{kO'}-l_j\le(1-\sum_{i\in\Omega\cup O\setminus j} x_{kij})\times M \quad k\in K, j\in\Omega \tag{9}$$

$$u_{kO}=\sum_{i\in\Omega\cup O}\sum_{j\in\Omega\setminus i} p_j\times x_{kij} \quad k\in K \tag{10}$$

$$u_{kj}\ge u_{ki}-p_j+q_j-(1-x_{kij})\times M \quad k\in K, i\in\Omega\cup O, j\in\Omega\cup O'\setminus i \tag{11}$$

$$u_{kj}\le u_{ki}-p_j+q_j+(1-x_{kij})\times M \quad k\in K, i\in\Omega\cup O, j\in\Omega\cup O'\setminus i \tag{12}$$

$$u_{ki}\le C_k \quad k\in K, i\in\Omega\cup O \tag{13}$$

Objective (1) of the model is to minimize the total cost of the feeder service, which includes the transportation cost and the service cost at the ports of the containerships. If a container is rehandled, it must be unloaded first and then reloaded; therefore, a container-rehandle is considered as two tasks. Constraints (2) along with constraints (3) guarantee that the route of each containership starts from and ends at the hub port, and a containership travels on one route at most ($x_{kOO'} = 1$ represents a virtual route of containership k). Constraints (4) ensure that each feeder port must be visited by a containership and be visited only once. Constraints (5) and (6) define the service time of the containerships at the hub and feeder ports, respectively. Constraints(7) describe successive arrival times at ports for containerships. Constraints (8) ensure the time window of each feeder port, while constraints (9) confine time deadlines at the hub port for containers from the feeder ports. Constraints (10) define the ship

loads at the hub port, while constraints (11) and (12) correspond to the ship loads at the feeder ports. Constraints (13) indicate that the capacity of any containership cannot be exceeded.

The constraints for containership stowage planning are as follows:

$$\sum_{i\in\Omega\cup O}\sum_{j\in\Omega\cup O'\backslash i}\sum_{c\in Q_{ij}} y_{ijc}^{bks} = r_b^{ks} \quad k\in K, s\in S_k, b\in\Omega\cup O \tag{14}$$

$$r_b^{ks} \le r_b^{k\pi(s)} \quad k\in K, s\in S_k, b\in\Omega\cup O \tag{15}$$

Constraints (14) define the relationship between decision variables y and r, and ensure that any slot stows at most one container at any port. Constraints (15) guarantee that there is no container hanging in the air for any containership at any port.

The sailing safety constraints are as follows:

$$W_b^{ks} = \sum_{i\in\Omega\cup O}\sum_{j\in\Omega\cup O'\backslash i}\sum_{c\in Q_{ij}} y_{ijc}^{bks}\times w_c^{ij} \quad k\in K, s\in S_k, b\in\Omega\cup O \tag{16}$$

$$GM_k^0 \le KM_k - \frac{\sum_{s\in S_k}(H_s\times W_b^{ks})}{D_k} \le GM_k^1 \quad k\in K, b\in\Omega\cup O \tag{17}$$

$$0 \le \frac{1}{2}(\gamma_k\times DL_k + \sum_{s\in S_k}(I_s\times W_b^{ks}) - D_k\times f_k \times L_k) \le Ms_k \quad k\in K, b\in\Omega\cup O \tag{18}$$

$$\left|\frac{(\sum_{s\in S_k}(I_s\times W_b^{ks}) - LCB_k\times D_k)}{(MCT_k\times 100)}\right| \le T_k \quad k\in K, b\in\Omega\cup O \tag{19}$$

Equations (16) define the deadweight of the slot in each containership at each port. Constraints (17)–(19) are restrictions for the stability (GM), longitudinal strength (Ms), and trim (T) of the containership, respectively. Notations in constraints (17)–(19) that are not introduced before, such as KM_k, H_s, and γ_k, are specific parameters of a containership. Detailed, formulations and corresponding explanations of GM, Ms, and T are presented in Appendix A.

The following constraints [constraints (20)–(28)] involve decision variables related to both the feeder routing and stowage planning and demonstrate the relationship between the two problems.

$$\sum_{s\in S_k}\sum_{c\in Q_{Oj}} y_{Ojc}^{Oks} = \sum_{i\in\Omega\cup O\backslash j} x_{kij}\times p_j \; k\in K, j\in\Omega \tag{20}$$

$$\sum_{s\in S_k}\sum_{c\in Q_{iO'}} y_{iO'c}^{iks} = \sum_{j\in\Omega\cup O\backslash i} x_{kji}\times q_i \quad k\in K, i\in\Omega \tag{21}$$

$$\sum_{s\in S_k}\sum_{i\in\Omega\cup O\backslash j}\sum_{c\in Q_{ij}} y_{ijc}^{jks} = 0 \; k\in K, j\in\Omega\cup O' \tag{22}$$

$$\sum_{s\in S_k}\sum_{c\in Q_{ij}} y_{ijc}^{bks} \le \sum_{s\in S_k}\sum_{c\in Q_{ij}} y_{ijc}^{mks} + (1-x_{kbm})M \quad k\in K, b, i\in\Omega\cup O, m\in\Omega, j\in\Omega\backslash m \tag{23}$$

$$\sum_{s\in S_k}\sum_{i\in\Omega\cup O}\sum_{j\in\Omega\cup O'\backslash i}\sum_{c\in Q_{ij}} y_{ijc}^{bks} = u_{kb} \; k\in K, b\in\Omega\cup O \tag{24}$$

Constraints (20) stipulate that all the containers destined to a feeder port along the route of a containership are loaded on it at the hub port. Constraints (21) indicate that if containership k arrives at feeder port i, then all the containers from port i must be loaded onto containership k at port i. Constraints (22) indicate that the container must be unloaded at its destination. Constraints (23) ensure that once a container is loaded on a containership, it must remain on that ship (after unloading and loading operations at the port(s)) until it reaches its destination. Constraints (24) confine the relationship between decision variables y and u, which in concert with constraints (20)–(23), can ensure that when a containership is leaving a port, only the containers from the port already visited by the containership are loaded on the containership at that time.

$$e_b^{ks} \ge \frac{1}{M}\times\sum_{s_1\in\pi(s)}\sum_{i\in\Omega\cup O}\sum_{c\in Q_{ib}} y_{ibc}^{mks_1}\times \sum_{i\in\Omega\cup O}\sum_{j\in\Omega\cup O'\backslash b}\sum_{c\in Q_{ij}} y_{ijc}^{mks} - (1-x_{kmb})\times M \quad k\in K, s\in S_k, m\in\Omega\cup O, b\in\Omega \tag{25}$$

$$e_b^{ks} \ge \frac{1}{M}\times\sum_{s_1\in\theta(s)}\sum_{i\in\Omega\cup O}\sum_{c\in Q_{ib}} y_{ibc}^{mks_1}\times \sum_{i\in\Omega\cup O}\sum_{j\in\Omega\cup O'\backslash b}\sum_{c\in Q_{ij}} y_{ijc}^{mks} - (1-x_{kmb})\times M \quad k\in K, s\in G_k^0, m\in\Omega\cup O, b\in\Omega \tag{26}$$

$$e_b^{ks} \ge \frac{1}{M}\times\sum_{i\in\Omega\cup O}\sum_{j\in\Omega\cup O'\backslash b}\sum_{c\in Q_{ij}} y_{ijc}^{mks}\times \sum_{s_1\in\theta(s)}\sum_{j\in\Omega\cup O'}\sum_{c\in Q_{bj}} y_{bjc}^{bks_1} - (1-x_{kmb})\times M \quad k\in K, s\in G_k^0, m\in\Omega\cup O, b\in\Omega \tag{27}$$

$$\left|y_{ijc}^{mks} - y_{ijc}^{bks}\right| \le e_b^{ks} + (1-x_{kmb})\times M \quad k\in K, s\in S_k, b\in\Omega, m\in\Omega\cup O, i\in\Omega\cup O\backslash b, j\in\Omega\cup O'\backslash b, c\in Q_{ij} \tag{28}$$

Constraints (25)–(28) are nonlinear inequalities used to define the container rehandles. Constraints (25) are used to calculate the regular container rehandles, while constraints (26) and (27) are used to calculate the container rehandles

triggered by removing the hatch cover during the unloading and loading operations, respectively. Constraints (28) ensure that if a container is not rehandled at the current port, the position of the container on the containership at the current port must be consistent with that at the former port along the route, and if the container is rehandled at a port, it can be reloaded into a different slot of the containership at that port.

The model mainly comprises three types of constraints: constraints related to feeder routing; constraints related to stowage planning; and constraints related to both feeder routing and stowage planning. Compared with the feeder routing problem discussed in the literature (e.g., Karlaftis et al. 2009; Zhang et al. 2015; Ji et al. 2015), we have also added the cost for container rehandles to the objective of the model. As shown in constraints (25)–(27), the containership route has a direct impact on the number of container rehandles at the ports. By including the cost of container rehandles in the objective, and optimizing the feeder routing and containership stowage planning problem jointly, we can obtain a better feeder routing plan and related containership stowage plans that minimize the total cost of the feeder service. Compared with the multi-port stowage planning problem discussed in the literature (Ambrosino et al. 2015; Ambrosino et al. 2017; Zhang et al. 2018), we have considered three types of container rehandles, namely the regular container rehandles, container rehandles triggered by removing the hatch cover for unloading the containers, and container rehandles triggered by removing the hatch cover for loading the containers. Furthermore, we have also discussed three main sailing safety requirements of the containership, which are especially important for feeder containerships. Moreover, the relationships between the two problems are well confined and the integrated optimization model can ensure the sailing safety requirements of the containership, which, otherwise, might be violated when discussing the two problems separately.

3 Solution method

The feeder routing problem is an extension of the VRP, which is NP-hard (Dethloff 2001); the stowage planning problem is also an NP-hard optimization problem (Avriel et al. 2000). Therefore, for real-life instances, it is difficult to obtain good solutions in a reasonable amount of time for the integrated optimization model. Hence, this section describes a heuristic algorithm for this problem. The framework of the heuristic algorithm is described in Sect. 3.1. Detailed descriptions are presented in Sects. 3.2, 3.3, and 3.4.

3.1 Overall description of the heuristic algorithm

Two main procedures are incorporated in the heuristic algorithm, namely a procedure for updating the feeder routing plans and a procedure for determining the containership stowage plans. These two procedures interact with each other iteratively to obtain (near) optimal solutions for the integrated optimization problem. For the description of the heuristic algorithm, the following notations are defined:

X is a solution of the integrated problem; X_1 is the feeder routing plan in solution X; X_2 is the stowage plans based on feeder routing plan X_1 in solution X; $C(X_1)$ is the cost of X_1, which includes the transportation cost and the service cost at the ports, but excludes the cost of container rehandles; $C(X_2)$ is the cost of X_2, which is the total cost of container rehandles for all the containerships in X_1; $C(X)$ is the total cost of solution X and $C(X) = C(X_1) + C(X_2)$; $Best\_X$ is the best integrated solution obtained so far; $C(Best\_X)$ is the cost of the best integrated solution obtained so far; ψ is a set used to record the feeder routing plans that have already been obtained; M is a sufficiently large number; gen is used to count the number of consecutive iterations without obtaining a good updated feeder routing plan; and $Maxgen$ is an index related to the stopping criteria of the heuristic.

The steps of the heuristic algorithm are listed as follows, and the flowchart of the heuristic algorithm is shown in Fig. 6.

Step 1: the indexes gen, ψ, and $C(Best\_X)$ are initialized, then go to Step 2;

Step 2: a greedy heuristic is used to generate an initial feeder routing plan X_1 (see details in Sect. 3.2), then go to Step 3;

Step 3: stowage plans for each containership based on feeder routing plan X_1 are constructed and recorded as X_2 (see details in Sect. 3.4), then go to Step 4;

Step 4: if feasible stowage plans can be obtained for all the containerships based on the feeder routes in X_1, then the cost of container rehandles for all the containerships are summed as $C(X_2)$, and by combing X_1 and X_2, an integrated solution X is generated, the related cost is calculated as $C(X) = C(X_1) + C(X_2)$, then go to Step 5; otherwise, go to Step 6;

Step 5: if the current integrated solution X is better than the best solution obtained so far, $Best\_X$ is replaced by X and $C(Best\_X) = C(X)$, then go to Step 6;

Step 6: update the current feeder routing plan X_1 as X_1' (see details in Sect. 3.3). If X_1' does not belong to ψ and $C(X_1') < C(Best\_X)$, go to Step 8; otherwise, go to Step 7;

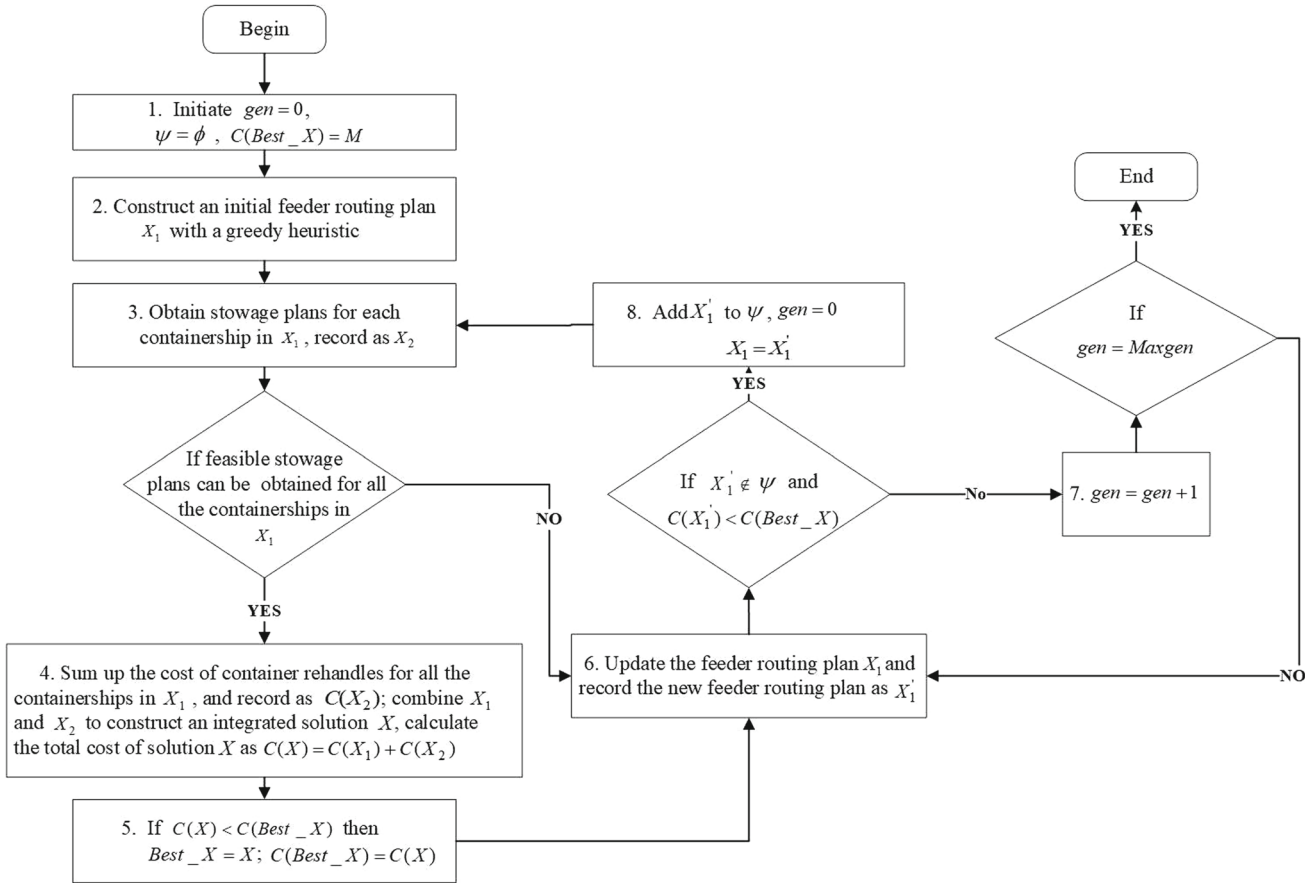

Fig. 6 Flowchart of the heuristic algorithm

Step 7: $gen = gen + 1$. If $gen = Maxgen$, the heuristic algorithm is stopped, and the best solution obtained so far ($Best\_X$) is output; otherwise, go to Step 6;

Step 8: add X_1' to ψ and reset index gen as $gen = 0$; X_1 is replaced by X_1', then go to Step 3.

As $C(X_1)$ and $C(X_2)$ are two nonnegative components, if $C(X_1') \geq C(Best\_X)$, the integrated solution related to X_1' cannot be better than $Best\_X$. Only when $C(X_1') < C(Best\_X)$ and $X_1' \notin \psi$ can the integrated solution based on X_1' be better than $Best\_X$. Therefore, only if the updated feeder routing plan satisfies $C(X_1') < C(Best\_X)$ and $X_1' \notin \psi$, can X_1' be accepted, and the stowage plans based on X_1' be constructed. As the number of iterations increases, $C(Best\_X)$ decreases and the number of elements in set ψ increases. Therefore, to obtain a good updated feeder routing plan X_1' that satisfies $C(X_1') < C(Best\_X)$, while not belonging to ψ, becomes increasingly difficult. If a good updated feeder routing plan cannot be obtained in $Maxgen$ iterations, the heuristic is stopped, and the best solution obtained so far ($Best\_X$) is output as the final solution of the heuristic algorithm.

3.2 Greedy heuristic for the initial feeder routing plan

We have developed a greedy heuristic to generate the initial feeder routing plan according to constraints (2)–(13) of the integrated optimization model.

Firstly, the largest containership is chosen and a route based on this containership is generated. This route begins at the hub port, and a feeder port that has not been visited before is inserted at the end of the current route at each iteration. No capacity violation is allowed during the insertion, and the time windows at the feeder ports and time deadlines at the hub port must be respected. If more than one feeder port meets the above requirements, the one with the least incremental cost is selected. In case no feeder port can be inserted to the current route, the hub port is added to the end of the route and the current route is output. If a smaller containership exists that can also make the current route feasible, then the largest containership is replaced by the smaller containership, and the cost of the current route is recalculated; otherwise, the cost based on the largest containership is recorded as the cost of the current route. Then, a new route is created to visit

the remaining feeder ports, and the abovementioned greedy heuristic is performed repeatedly until all the feeder ports are visited in the containership route.

Finally, the initial feeder routing plan is made up of all the routes generated through the greedy heuristic, and the cost of the initial feeder routing plan is the total cost of all the routes.

3.3 Heuristic for updating the feeder routing plan

We have designed a heuristic to update the feeder routing plan based on a variable neighborhood search. The variable neighborhood search (VNS) was first proposed by Hansen and Mladenovic (1997) and is a metaheuristic technique based on the concept of systematic change of neighborhoods during the search. The VNS heuristic was already successfully applied to variants of the VPR (e.g., Fleszar et al. 2009; Hansen et al. 2010; Kammoun et al. 2015). The feeder routing problem is also an extension of the VRP with both pickup and delivery, and with capacity and time window constraints. Therefore, a heuristic based on the VNS is developed to obtain the updated feeder routing plan.

The VNS considers a set of k_{max} neighborhood structures and alternately executes a shaking procedure and a local search procedure to escape from the local optima. At each iteration, a random solution is generated from the current neighborhood of the current solution, and then a local search procedure is applied to generate a new solution. If the new solution is better than the current solution, the current solution is replaced by the new solution and the above procedure is repeated by reinitializing the neighborhood; otherwise, the procedure is repeated by passing to the next neighborhood.

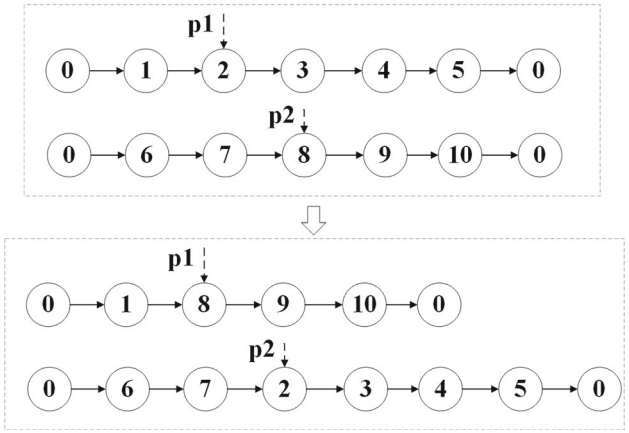

Fig. 7 Inter-route 2-opt

Table 1 Framework of the heuristic for updating the feeder routing plan

Design a set of neighborhood structures $N_k, k = 1, \ldots, k_{max}$
$s = X_1$, s represents the initial solution of the VNS
$gen = 0$
While true **do**
  $k = 1$
  While $k \leq k_{max}$ **do**
    $s' = PickAtRandom(N_k(s))$ – pick a random solution s' from the k-th neighborhood of s
    $X'_1 = LocalSearch(s')$ – get the best solution X'_1 among 30 neighbors of s' in $N_k(s')$
    If $C(X'_1) < C(s)$ **then**
      $s = X'_1$
      $k = 1$
    Else
      $k = k + 1$
    End if
    If $C(X'_1) < C(Best\_X)$ and $X'_1 \notin \psi$ **then**
      X'_1 is output as the updated feeder routing plan
      the VNS is **terminated**
    Else
      $gen = gen + 1$
    End if
    If $gen = Maxgen$ **then**
      the VNS is **terminated**
    End if
  End while
End while

The framework of the heuristic based on the VNS for updating the feeder routing plan is shown in Table 1. In Table 1, $N_k(k = 1, \ldots, k_{max})$ is a finite set of pre-determined neighborhood structures, and $N_k(s)$ is the set of solutions in the k-th neighborhood structure of solution s. X_1 is the current feeder routing plan that needs to be updated. The VNS starts with an initial solution $s = X_1$, and the first neighborhood structure $N_k(k = 1)$ is chosen in the first iteration. Then, the shaking procedure is implemented to obtain a random solution s' from $N_k(s)$, and the local search is executed to explore the neighbors of s' in $N_k(s')$. For the local search, a candidate list strategy is implemented. The candidate list is chosen randomly from the entire neighborhood and contains 30 non-repetitive neighbors. Then, the best one in the candidate list is accepted as the new solution(X_1'). If the new solution X_1' is better than the current solution s ($C(X_1') < C(s)$), then the current solution is updated as $s = X_1'$, and the first neighborhood structure ($k = 1$) is called in the next iteration; otherwise, the next neighborhood structure ($k = k + 1$) is explored in the next iteration. Each time, when X_1' is obtained, $C(X_1')$ is compared with $C(Best\_X)$. If $C(X_1') < C(Best\_X)$ and X_1' does not exist in set ψ, the VNS is terminated and X_1' is output. The VNS is also terminated when a good updated feeder routing plan cannot be obtained for $Maxgen$ iterations ($gen = Maxgen$).

According to the characteristics of the problem, seven neighborhood structures are designed to efficiently explore the solution space, and these are defined as: Inter-route 2-opt (N_1), Inter-route exchange (N_2), Inter-route insertion (N_3), Intra-route 2-opt (N_4), Intra-route exchange (N_5), Intra-route insertion (N_6), and Containership-size change (N_7). These structures are briefly explained below.

- Inter-route 2-opt (N_1). Two ports ($p1$ and $p2$) are chosen from two different routes, and connections between the two chosen ports and their previous ports are deleted. Then, the sub-route with $p1$ is connected to the preceding ports of $p2$, and the sub-route with $p2$ is connected to the preceding ports of $p1$. An example of the Inter-route 2-opt is presented in Fig. 7.
- Inter-route exchange (N_2). Two ports are chosen from two different routes, and the positions of the two ports are exchanged.
- Inter-route insertion (N_3). A port from one route is deleted and inserted into another existing route or a new route.
- Intra-route 2-opt (N_4). Two ports ($p1$ and $p2$) are chosen from a route, and the arriving sequence between ports $p1$ and $p2$ is reversed. An example of the Intra-route 2-opt is shown in Fig. 8.
- Intra-route exchange (N_5). Two ports are chosen from a route, and their positions are exchanged.

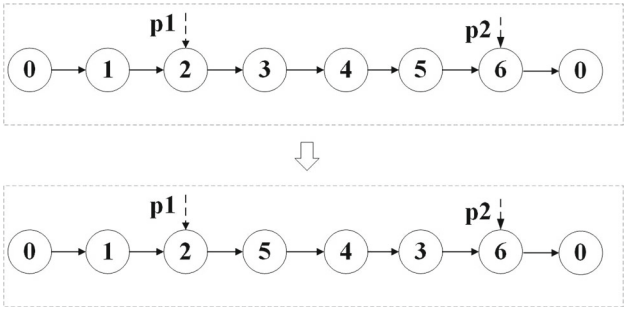

Fig. 8 Intra-route 2-opt

- Intra-route insertion (N_6). A port is removed from its route and reinserted again into that route.
- Containership-size change (N_7). The size of the containership for a route is changed.

All the above-mentioned operations are performed without violating the capacity constraints of the containerships or time-window constraints of the feeder ports and hub port.

3.4 Heuristic for containership stowage planning

For a route with n ports, let $T = [T_{ij}]$ be the $n \times n$ transportation matrix, where T_{ij} is the number of containers originating from port i and with port j as the destination . Based on features of a feeder route, there are only $2 \times (n-2)$ elements in matrix T that are more than 0, which are $[T_{12}, T_{13}, \ldots, T_{1(n-1)}]$ and $[T_{2n}, T_{3n}, \ldots, T_{(n-1)n}]$, while all the other elements in T are equal to 0.

We now assume that $T_{12}, T_{13}, \ldots, T_{1(n-1)}$ are all equal to 0, that is, there is no container to be loaded at the hub port. The other container groups ($T_{2n}, T_{3n}, \ldots, T_{(n-1)n}$) have the same destination, and therefore, they are loaded according to the arriving sequence of the containership and cannot be unloaded until the last port is reached. As a container cannot be hung in the air, when loading a container, it can only be loaded at the lowest empty tier of a row in a bay. If there are N slots in a tier of a containership, when loading a container, there exist N choices, and the complexity of the stowage planning (SP) problem described above can be calculated as $o(SP) = N^{T_{2n}} \times N^{T_{3n}} \cdots \times N^{(n-1)n} = N^{T_{2n}+T_{3n}\cdots+T_{(n-1)n}}$.

For a containership with a capacity of 200 slots, the number of slots in a tier can be considered to be 40. When the total number of containers that are required to be picked at the feeder ports along a route is 100, the complexity of the SP problem is at least $o(40^{100})$, and $o(SP)$ equals $o(40^{100})$ if and only if there is no container required to be delivered from the hub port ($T_{12} = T_{13} \cdots = T_{1(n-1)} = 0$), and no container is rehandled at any port along the route.

As can be observed, solving the SP problem is difficult and time-consuming. Therefore, we reduce its complexity by par-

Fig. 9 Serial numbers of container groups

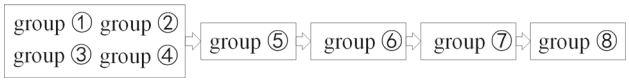

Fig. 10 Loading sequence of container groups (I)

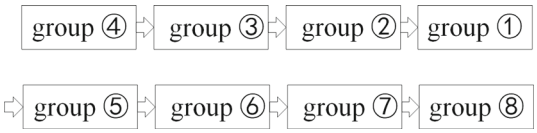

Fig. 11 Loading sequence of container groups (II)

titioning the containership into several blocks and splitting the containers into different groups to solve the SP problem more efficiently. The detailed procedures are given in Sects. 3.4.1, 3.4.2, 3.4.3 and 3.4.4.

3.4.1 Container division

First, the containers with the same origination and destination are placed in the same group. Figure 9 shows the route of a containership in a feeder routing plan. The number in the circle above or below the feeder port is the serial number of each container group. For example, ① represents the container group with containers from port 1 (hub port) to port 2; ② represents the container group with containers from port 1 to port 3; ⑤represents the container group with containers from port 2 to port 6 (hub port); ⑥represents the container group with containers from port 3 to port 6 (hub port). It can be noted that for a route with n ports, the number of container groups is $2 \times (n-2)$.

Based on the physical arriving sequence of the containership, there exists a certain loading sequence for all container groups; for example, containers in groups ①, ②, ③ and ④ are loaded at the first port; containers in group ⑤ are loaded at the second port; containers in group ⑥ are loaded at the third port; containers in group ⑦ are loaded at the fourth port; and finally, containers in group ⑧ are loaded. Thus, the loading sequence for the container groups is illustrated in Fig. 10.

For loading operations at the hub port, the container group that will be unloaded later should be loaded before the container group, which will be unloaded earlier, to avoid container rehandles. Therefore, the containers in group ④ should be loaded first at the hub port, and the containers in group ① should be loaded last. The final loading sequence of container groups is shown in Fig. 11.

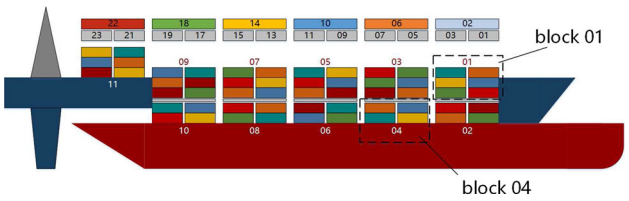

Fig. 12 Serial numbers of the containership blocks

Considering the safety of the containerships, containers with heavier weight should be loaded before (under) the containers with lesser weight. Here, we categorize the weight of containers into three classes, namely light-weight (less than 10 t), medium-weight (10–18 t), and heavy-weight (more than 18 t). Based on this principle, we continue to divide the container groups mentioned above into smaller groups based on their weights. Therefore, the number of container groups is now $3 \times 2 \times (n-2)$. The loading sequence of these smaller container groups is based on the sequence mentioned above (Fig. 11), and inside each group with the same origination and destination, the smaller group with a heavier weight will be loaded earlier.

Using the above rules, the containers are divided into groups, based on their origination, destination, and weight class, and the loading sequence of these container groups is defined.

3.4.2 Block division

In the integrated optimization model, the container rehandles triggered by removing the hatch cover are considered; therefore, for the convenience of calculations, the containership is divided into several blocks according to its hatch covers. A hatch cover of the containership in Fig. 12 crosses two odd bays and divides the two bays into an upper partition (on-deck) and a lower partition (in-hold). We define this upper partition/lower partition crossed by a hatch cover as a block. For the containership in Fig. 12, the number of blocks is 11 and they are numbered as shown in Fig. 12.

3.4.3 Obtaining a stowage plan

Given the containership blocks and container groups, we can obtain a stowage plan for the containership by distributing each container group sequentially with a specific containership block according to the loading sequence of the container groups at each port. Therefore, theoretically, for a 200TEU-containership with 11 blocks, if it travels on a route comprising six ports, the complexity of the problem can be seen as $11^{3\times2\times(6-2)} = 11^{24}$ (if no container is rehandled at ports along the route). Although the solution space is much smaller than the original SP problem, the complexity of the problem is still exponential. Therefore, a genetic algorithm

(GA), based on the above operations, is designed to solve the SP problem.

3.4.4 GA for the SP problem

The GA is selected for the following reasons:

(1) the complexity of the SP problem;
(2) the ability of GA to obtain a solution of good quality in an acceptable computational time;
(3) the wide applicability of GA for solving related stowage planning problems (Zhu and Ji 2014);
(4) a solution to this SP problem can be easily presented using the representation (coding) scheme of GA.

A GA for solving the SP problem is proposed as follows:

Table 2 Decoding procedure

1:	$i = 1, b = 1$
2:	**While** $i \leq n$ **do**
3:	**Unloading procedure**:
4:	The containers that need to be rehandled because of the unloading operations are removed from the containership and R_i^u is calculated; G_i^{max} and g_{ia} are updated
5:	Containers with the destination of port i are unloaded
6:	The number of container rehandles because of the loading operations is calculated as R_i^l and related containers are removed; G_i^{max} and g_{ia} are updated
7:	The number of container rehandles at port i is calculated as $R_i = R_i^u + R_i^l$
8:	The number of empty slots in each containership block is updated
9:	**Loading procedure**:
10:	a=1
11:	**While** $a < G_i^{max}$
12:	Loading containers in group g_{ia} to block b
13:	**If** $c(g_{ia}) < C(b)$ **then**
14:	Containers in group g_{ia} are loaded into block b from the lowest empty tier with one container occupies one slot
15:	$C(b) = C(b) - c(g_{ia})$
16:	$a = a + 1$
17:	$b = b + 1$
18:	**Else if** $0 < C(b) < c(g_{ia})$ **then**
19:	$C(b)$ Containers in group G_{ia} are loaded into block b from the lowest empty tier with one container occupies one slot
20:	$c(g_{ia}) = c(g_{ia}) - C(b)$
21:	$b = b + 1$
22:	**Else if** $C(b) = 0$ **then**
23:	$b = b + 1$
24:	**End**
25:	**End while**
26:	**If** the stowage plan at port i meeting the safety constraints of the containership **then**
27:	$i = i + 1$
28:	**Else**
29:	the decoding procedure is terminated and the current chromosome is infeasible for the SP problem
30:	**End**
31:	**End while**
32:	Output the stowage plan and calculate the total cost for all the container rehandles

a. Mode of encoding

A set of integer numbers is used to represent a chromosome. If a containership has N blocks and the number of container groups is M, then the length of the chromosome is set to $M \times N$, and each gene (integer) in the chromosome is generated randomly from 1 to N.

b. Mode of decoding

Each gene in a chromosome represents a containership block, and we assign each gene of the chromosome in sequence to container groups based on their loading order. Table 2 shows the detailed procedure of decoding, which is a dynamic block-assignment scheme that terminates with a stowage plan at each port or with the result that the chromosome is infeasible. The notations in Table 2 are defined as follows:

Index n is the number of ports along the route of the containership; G_i^{max} is the number of container groups that need to be loaded at port i; g_{ia} is the a-th container group that needs to be loaded at port i; $c(g_{ia})$ is the number of containers in container group g_{ia}; b is the b-th gene of the chromosome; $C(b)$ indicates the number of empty slots in block b; R_i^u is the number of container rehandles because of the unloading operations at port i; R_i^l is the number of container rehandles because of the loading operations at port i; and R_i is the total number of container rehandles at port i.

As seen in Table 2, when the containership arrives at a port, the unloading operations are executed first, followed by the loading operations. During the unloading operations, not only are the containers with the destination of the current port unloaded, but also the containers that need to be rehandled are unloaded (through Steps 4 and 6).

In Step 4, if a container needs to be rehandled during the unloading operations at port i, it should be removed first and added to a current container group or a new container group originating from port i and going to its original destination. Then, the data of container groups that should be loaded at port i are updated and the loading order of these container groups is regenerated according to the rules described in Sect. 3.4.1.

In Step 6, we calculate the number of container rehandles during the loading operations and remove these rehandled containers. First, we simulate the loading operation of the current container groups that need to be loaded at port i according to Steps 10–25, to check if there are any containers on-board that must be removed during the loading operation; if such containers exist, these containers are removed from the containership, and the data of the current container groups that need to be loaded at port i are updated. We continue to simulate the loading operation of the updated container groups and repeat the above procedures, until no container needs to be rehandled during the loading operation.

1	2	3	4	4	3	3	1	2	1	4	4	2	3	1	1	...

Fig. 13 Example of a chromosome

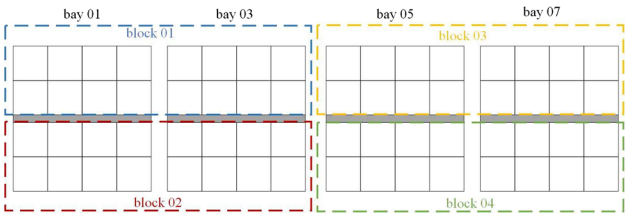

Fig. 14 Four odd bays of the containership

After executing Step 6, the number of container rehandles during the unloading and loading operations is calculated. The final data of container groups that need to be loaded at port i are generated, followed by Steps 10–25 to load these container groups at port i.

For a better explanation of the decoding procedure, a small example is designed and presented here. Figure 13 is an example of a chromosome. Assume there are four odd bays of an containership, with two odd bays sharing a hatch cover; thus, the containership is divided into four blocks by the two hatch covers (shown in Fig. 14). There are four ports along the route of the containership, therefore, the number of container groups is 12 at the beginning, and the number of containers in each container group is shown in Table 3. In Table 3, group ②-H contains heavy-weight containers from ports 1 to 3; group ②-M contains medium-weight containers from ports 1–3; group ②-L contains light-weight containers from ports 1–3, and so on.

At port 1, only the loading procedure is invoked. Container group ②-H is loaded into block 1; container group ②-M is loaded into block 2; container group ②-L is loaded into block 3; container group ①-H is loaded into block 4; and container group ①-M is assigned to block 4. When loading container group ①-M into block 4, only six empty slots are left in block 4; therefore, six containers from group ①-M are loaded into block 4 and the remaining containers are loaded into the next block (next gene of the chromosome—block 3). Likewise, all the container groups that need to be loaded at port 1 are loaded according to the chromosome shown in Fig. 13, and the stowage plan at port 1 is shown in Fig. 15a.

At port 2, the unloading procedure is called at first, containers destined to port 2 are marked as green in Fig. 15b. To unload these containers, six containers that are not destined to port 2 must be removed (marked as red in Fig. 15b). Then, the containers marked as green or red are removed from the containership (as shown in Fig. 15c).

The rehandled containers during the unloading operations are formed as a new container group that needs to be loaded at port 2 (②-L', which comprises six light-weight contain-

Table 3 Data of the container groups at the beginning

Loading order	Container group	Loading port	Unloading port	Weight class	Number of containers
1	②-H	1	3	Heavy	4
2	②-M	1	3	Medium	13
3	②-L	1	3	Light	6
4	①-H	1	2	Heavy	10
5	①-M	1	2	Medium	20
6	①-L	1	2	Light	7
7	③-H	2	4	Heavy	6
8	③-M	2	4	Medium	10
9	③-L	2	4	Light	11
10	④-H	3	4	Heavy	10
11	④-M	3	4	Medium	12
12	④-L	3	4	Light	6

ers originated from port 2 and destined to port 3). Now, the container groups that need to be loaded at port 2 are ③-H, ③-M, ③-L, and ②-L'.

Then, we simulate the loading operation according to Steps 10–25 in Table 2 for these four container groups, to calculate the number of containers that must be removed during the loading operations. When loading container group ③-H into block 2, the related hatch cover must be removed and the containers loaded in block 1 must be unloaded (marked as blue in Fig. 15c). Therefore, the containers marked as blue are removed from the containership (as shown in Fig. 15d) and are formed as a new container group that needs to be loaded at port 2 (②-H', which comprises four heavy-weight containers originating from port 2 and destined to port 3). Then, we continue to simulate the loading operation for these five container groups to see if any containers exist that must be removed during the loading operation. Apparently, no such containers exist. Therefore, the unloading procedure at port 2 is completed, and the final container groups that need to be loaded at port 2 are shown in Table 4. The total number of container rehandles at port 2 is ten. Then, the loading procedure is executed for the container groups presented in Table 4, and the stowage plan at port 2 is shown in Fig. 15e.

The unloading and loading procedures at port 3 can be executed in the same way as for port 2, resulting in a stowage plan; therefore, a detailed explanation for port 3 is not included here.

After a stowage plan is generated for the containership at a port, the feasibility of it is checked. If the stowage plan is feasible, the cost of container rehandles is recorded; otherwise, a new chromosome is generated and the decoding procedure is executed again. If we cannot obtain feasible initial stowage plans for all the ports along the route after a number of repetitions, we reorder the loading sequence of the container groups originating from the hub port, and repeat the above procedures. For this scenario, non-reasonable loading order at the hub port is applied at the expense of more container rehandles, to pursue feasible solutions. For a certain number of times, if initial feasible stowage plans for all the ports cannot be generated, we conclude that there is no feasible stowage plan for the containership based on the current feeder routing plan, and the GA used for solving the SP problem terminates.

c. As the encoding of the chromosome designed herein is universal, the selection/mutation/crossover operation is as simple as that in solving an ordinary traveling salesman problem using GA (Potvin 1996). Therefore, these procedures are not presented in this study.

d. Termination: the GA terminates when a better fitness value cannot be calculated after a number of generations or the initial feasible stowage plan cannot be generated for a certain number of times.

4 Computational study

A computational study was carried out to analysis the performance of the solution method proposed in this study. All the instances in the computational study were executed using a 1.8 GMz Intel Core E5 processor with 8GB of RAM and were implemented in MATLAB R2016a.

4.1 Data description

The proposed solution method was applied for routing a containership fleet from the hub port Nansha to a set of 29 spoke ports in the Pearl River Delta region of China. Table 5 shows the data for three different types of containerships used in this study.

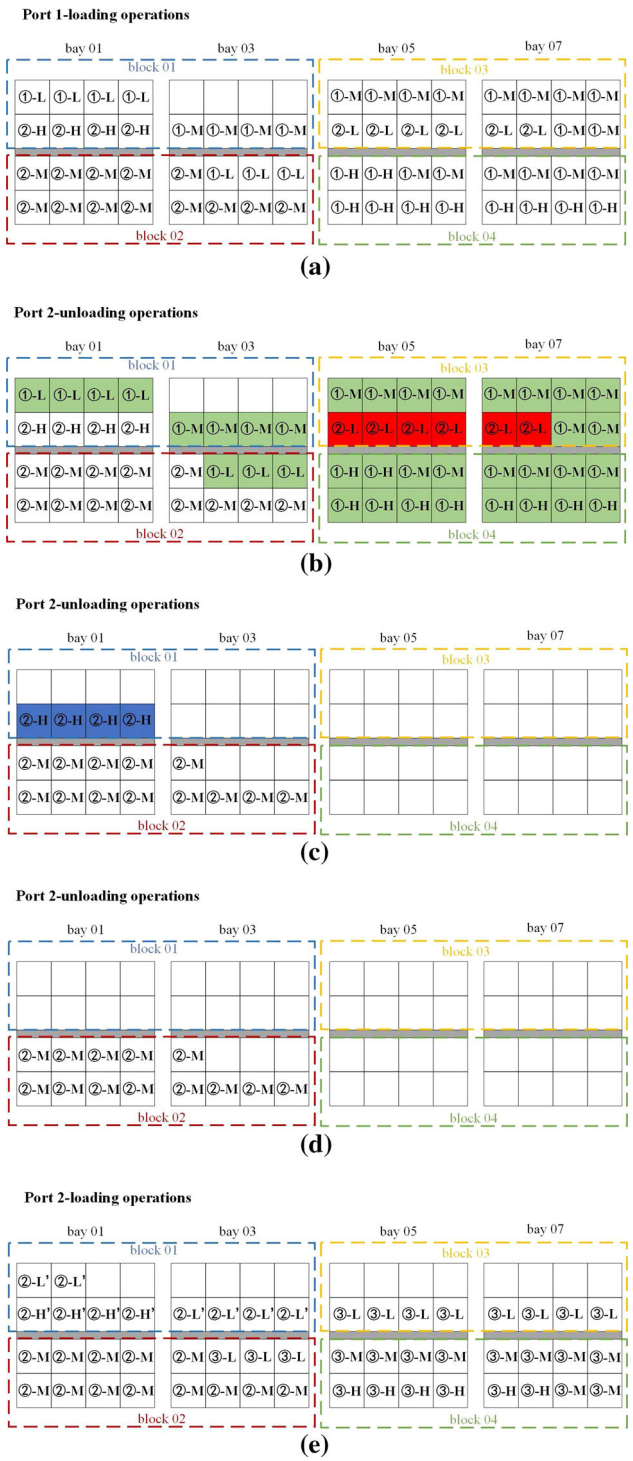

Fig. 15 Example of the decoding procedure

4.2 Hierarchical versus integrated optimization: value of integration

We compared the results of the hierarchical and integrated optimization to verify the value of the integrated optimization of the two problems. Here, two methods were designed to measure the results of the hierarchical optimization.

The first of these hierarchical methods, hierarchical optimization-I, is described as follows: an initial feeder routing plan is obtained by the greedy heuristic proposed in Sect. 3.2. Next, the initial feeder routing plan is improved by the VNS procedure until a better feeder routing plan cannot be obtained for $Maxgen$ iterations. Then, stowage plans are generated for the containerships based on the current feeder routes. The total cost of the feeder routing plan and related stowage plans is calculated as the result of hierarchical optimization-I. (The framework of hierarchical optimization-I is presented in Appendix B.)

The second hierarchical meta-heuristic, hierarchical optimization-II, is described as follows: an initial feeder routing plan X_1 is obtained by the greedy heuristic proposed in Sect. 3.2; the stowage plans X_2 based on feeder routing plan X_1 are generated, and the total cost of the initial solution is calculated as $C(X_1) + C(X_2)$. The feeder routing plan is updated through the VNS procedure. Once a better feeder routing plan X_1' with a lower cost is obtained $(C(X_1') < C(X_1))$, the current feeder routing plan X_1 is replaced by X_1', the stowage plans based on the current feeder routing plan X_1 are generated, and the total cost of the current solution is recorded. In hierarchical optimization-II, the two phases (feeder routing and stowage planning) are separately iterated through the whole process, and the hierarchical meta-heuristic is terminated when a better feeder routing plan cannot be generated through the VNS procedure for $Maxgen$ times. The feeder routing plan and the related stowage plans with the lowest total cost are output as the result from hierarchical optimization-II. (The framework of hierarchical optimization-II is presented in Appendix B.)

Different numbers of ports were selected, i.e., 15, 20, 25, and 30. The loading and unloading quantities at each port were randomly generated between 10 and 60. Based on the number of ports, five instances were randomly generated. Thus, 20 instances were generated in total.

We compared the results of the integrated optimization with the two hierarchical optimization schemes, individually. Each method was executed ten times for each instance, and the average results are presented in Tables 6 and 7. In these tables, index "C_1" represents the cost of the feeder routing plan; "C_2" represents the cost of container rehandles; "C" is the total cost of the entire solution, e.g., $C = C_1 + C_2$; "reduced cost" shows the difference of the average results between the hierarchical optimization and the integrated optimization. The unit of above indexes is thousand USD.

For each instance, the t-test is applied to analyze whether the difference between the results of the hierarchical optimization and the integrated optimization is significant. The index "t-value" in Table 6 shows the significant difference between the results of hierarchical optimization-I and the

Table 4 Data of container groups that need to be loaded at port 2

Loading order	Container group	Loading port	Unloading port	Weight class	Number of containers
1	③-H	2	4	Heavy	6
2	③-M	2	4	Medium	10
3	③-L	2	4	Light	11
4	②-H'	2	3	Heavy	4
5	②-L'	2	3	Light	6

Table 5 Data for different types of containerships

Type (TEU)	Economic speed (knots)	GM (m)	Safety range of T (m)	Ms (tf×m)
100	4.32	(0.8 – 1.54)	(-0.3 – 0.3)	8.2×10^3
150	5.18	(0.8 – 1.54)	(-0.45 – 0.45)	1.0×10^4
200	5.51	(0.8 – 1.54)	(-0.5 – 0.5)	1.2×10^4

Table 6 Computational results of hierarchical optimization-I and the integrated optimization

Instance	Number of ports	Hierarchical optimization-I			Integrated optimization			Reduced cost (1)–(2)	t-value	Significance (Y or N)
		C_1	C_2	C (1)	C_1	C_2	C (2)			
1	15	7.42	0.16	7.58	7.45	0.02	7.47	0.11	27.28	Y
2		7.36	0.13	7.49	7.40	0.01	7.41	0.08	16.13	Y
3		7.45	0.02	7.47	7.46	0.02	7.48	−0.01	−1.12	N
4		7.64	100#	107.64	7.79	0.05	7.84	99.80	> 10000	Y
5		7.40	90.03*	97.43	7.61	0.03	7.64	89.79	8.98	Y
6	20	12.24	0.00	12.24	12.24	0.00	12.24	0.00	1.05	N
7		11.92	0.22	12.14	11.99	0.03	12.02	0.12	38.72	Y
8		12.57	0.14	12.71	12.61	0.02	12.63	0.08	22.1	Y
9		12.84	0.19	13.03	12.86	0.01	12.87	0.16	61.19	Y
10		12.11	100#	112.11	12.79	0.06	12.85	99.26	> 10, 000	Y
11	25	17.81	100#	117.81	18.22	0.04	18.26	99.55	> 10, 000	Y
12		17.32	0.32	17.64	17.45	0.02	17.47	0.17	58.69	Y
13		18.26	0.27	18.53	18.38	0.03	18.41	0.12	48.08	Y
14		17.69	0.36	18.05	17.72	0.06	17.78	0.27	78.92	Y
15		17.43	80.03*	97.46	17.57	0.04	17.61	79.85	5.99	Y
16	30	26.92	0.39	27.31	27.01	0.11	27.12	0.19	105.28	Y
17		27.22	0.35	27.57	27.30	0.02	27.32	0.25	75.34	Y
18		27.49	90.04*	117.53	27.62	0.12	27.74	89.79	7.62	Y
19		26.31	100#	126.31	26.52	0.04	26.56	99.75	> 10, 000	Y
20		27.35	0.28	27.63	27.40	0.02	27.42	0.21	51.22	Y
Average				49.28			16.31	32.98		

integrated optimization, while the same in Table 7 shows the significant difference between the results of hierarchical optimization-II and the integrated optimization. The significance level (α) is set to 0.05, and the critical value of the t-test can be obtained as $t_{(\alpha)}(n-1) = t_{(0.05)}(9) = 2.26$. Therefore, for a specific instance, if the value of "t-value" is larger than 2.26 or smaller than −2.26, we can conclude that there is a significant difference between the two tested methods for that instance ("Significance"=Y); otherwise, there is no significant difference between the two tested methods for that instance ("Significance"=N).

For the hierarchical optimizations, situation may arise, wherein a feasible stowage plan cannot be obtained for the given feeder route; this will lead to an infeasible solution.

Table 7 Computational results of hierarchical optimization-II and the integrated optimization

Instance	Number of ports	Hierarchical optimization-II			Integrated optimization			Reduced cost (1)–(2)	t-value	Significance (Y or N)
		C_1	C_2	C (1)	C_1	C_2	C (2)			
1	15	7.44	0.09	7.53	7.45	0.02	7.47	0.06	5.12	Y
2		7.38	0.11	7.49	7.40	0.01	7.41	0.08	5.02	Y
3		7.45	0.02	7.47	7.46	0.02	7.48	−0.01	−1.32	N
4		7.89	0.06	7.95	7.79	0.05	7.84	0.11	18.60	Y
5		7.81	30.07*	37.88	7.61	0.03	7.64	30.24	2.49	Y
6	20	12.24	0.00	12.24	12.24	0.00	12.24	0.00	1.07	N
7		11.96	0.15	12.11	11.99	0.03	12.02	0.09	19.02	Y
8		12.58	0.07	12.65	12.61	0.02	12.63	0.02	1.89	N
9		12.85	0.17	13.02	12.86	0.01	12.87	0.15	29.61	Y
10		14.29	0.12	14.41	12.79	0.06	12.85	1.56	5.32	Y
11	25	19.25	0.23	19.48	18.22	0.04	18.26	1.22	7.09	Y
12		17.37	0.23	17.60	17.45	0.02	17.47	0.13	24.29	Y
13		18.34	0.08	18.42	18.38	0.03	18.41	0.01	1.63	N
14		17.71	0.29	18.00	17.72	0.06	17.78	0.22	65.34	Y
15		18.52	0.17	18.69	17.57	0.04	17.61	1.08	2.97	Y
16	30	26.99	0.29	27.28	27.01	0.11	27.12	0.16	35.57	Y
17		27.26	0.30	27.56	27.30	0.02	27.32	0.24	70.22	Y
18		29.57	20.04*	49.61	27.62	0.12	27.74	21.87	2.68	Y
19		27.06	0.19	27.25	26.52	0.04	26.56	0.69	5.08	Y
20		27.39	0.21	27.60	27.40	0.02	27.42	0.18	47.65	Y
Average				19.21			16.31	2.91		

If the solution is infeasible, the cost for stowage planning (C_2) is recorded as a sufficiently large number (in our cases, as 100); the result in Table 6 with the symbol "#" indicates that, for ten runs, no feasible solution was obtained for the corresponding instances. In addition, the results in Tables 6 and 7 with the symbol "*" indicate that, for ten runs, an infeasible solution was obtained at least once.

As can be seen in Table 6, for instances 5, 15, and 18, at least one solution was infeasible among the ten runs of hierarchical optimization-I. Furthermore, for instances 4, 10, 11, and 19, no feasible solution could be obtained for ten runs of hierarchical optimization-I. However, feasible solutions could be obtained for all the runs with the integrated optimization. Among the 20 instances, the average result of the integrated optimization was better than that of hierarchical optimization-I in 18 instances, and the difference was significant. For instance 3, the average result of hierarchical optimization-I is better than that of the integrated optimization; however, the difference is not significant. Moreover, we found that the best results among the ten runs of hierarchical optimization-I and the integrated optimization were the same for instance 3. This is because the (near) optimal feeder routing plan obtained at the first phase of hierarchical optimization-I for instance 3 resulted in stowage plans that were feasible and had only a few container rehandles; then, the (near) optimal feeder routing plan and the related stowage plans were formed as a solution with the lowest total cost. A similar explanation can be offered for instance 6.

Through the analysis of the results in Table 6, we can conclude that the integrated optimization is better than hierarchical optimization-I for providing feasible solutions and obtaining results with a lower total cost. In hierarchical optimization-I, the second phase (stowage planning) is conditioned by the feeder routing decisions fixed in the first phase; therefore, we further explored the effectiveness of our integrated optimization method by comparing it with hierarchical optimization-II, where the two phases (feeder routing and stowage planning) are separately iterated, and the results are summarized in Table 7.

As shown in Table 7, for instances 5 and 18, at least one solution was infeasible among the ten runs of hierarchical optimization-II. For instances 3, 6, 8, and 13, there was no significant difference between the results of hierarchical optimization-II and the integrated optimization. For the rest of the instances, the integrated optimization could provide results with better total cost, and the difference between the integrated optimization and hierarchical optimization-II was significant.

As can be seen in Tables 6 and 7, from the perspective of the feasibility and the average cost of the solution, hierarchical optimization-II is better than hierarchical optimization-I, and the integrated optimization is more efficient than either of the hierarchical methods. Thus, the validity of the integrated optimization for the two problems has been verified.

For a better explanation of the differences among the three optimization methods considered, we show the detailed solution results of the these methods for solving instance 19. The best solutions from the ten runs of hierarchical optimization-I, hierarchical optimization-II, and the integrated optimization are shown in Tables 8, 9, and 10, respectively.

In hierarchical optimization-I, the (near) optimal feeder routing plan is generated at the first phase; therefore, the cost for feeder routing in the solution of hierarchical optimization-I was the lowest among the three methods. However, as shown in Table 8, the feasible stowage plan could not be obtained for route (containership) 3 based on the (near) optimal feeder routing plan: for any of the scenarios, a feasible stowage plan could not be obtained for containership 3 at the fifth port along its route (Macau), either owing to violations of the safety constraint of stability (GM) or not satisfying the safety constraint for trim (T). Therefore, the solution of hierarchical optimization-I was infeasible.

In hierarchical optimization-II, more combinations of the feeder routing plan and related stowage plans are generated through the entire iteration. As can be seen, in the solution of hierarchical optimization-II, the cost for feeder routing was larger than the cost of the (near) optimal feeder routing plan. However, by adding a route (containership), feasible stowage plans could be obtained for all the containerships. As can be seen from Table 9, many container rehandles existed along routes 2 and 7 because of the suboptimal arriving sequences of the containerships. For route 2, for the obtained (near) optimal stowage plans, when containership 2 arrived at Haikou to unload the containers that were destined to Haikou, 17 containers were to be rehandled; when containership 2 arrived at Shuidong to unload containers that were destined to Shuidong, and load containers originating from Shuidong, 21 containers were to be rehandled.

In the solution of the integrated optimization, the cost of the feeder routing was equal to that of the solution of hierarchical optimization-I; however, the arriving sequence of route 2/7 was reversed, which resulted in stowage plans with fewer container rehandles.

In hierarchical optimization-II, the two phases were iterated separately. When the feeder routing plan in Table 9 was obtained, it continued to search for a better feeder routing plan with a cost lower than the current feeder routing plan. However, in the integrated optimization, when the feeder routing plan (as shown in Table 9) was obtained, it continued to search for a feeder routing plan with a cost that was lower

Table 8 Best solution from the ten runs of hierarchical optimization-I

Route		Type of ship (TEU)	Number of container rehandles	C_1	C_2	$C = C_1 + C_2$
1	Nansha-Huizhou-Yantian-Nansha	100	0	26.31	100	126.31
2	Nansha-Xinfeng-Old Huangpu-Zengcheng-New huangpu -Sanshan-Beijiao-Xinhui-Jiangmen-Taiping-Nansha	200	21			
3	Nansha-Nanwei-Shekou-Zhuhai-Macau-Gaolan-Doumen-Zhongshan-Nansha	200	Infeasible			
4	Nansha-Sanshui-Sanrong-Rongqi -Nansha	100	0			
5	Nansha-Haifang-Yangjiang-Nansha	100	0			
6	Nansha-Shuidong-Zhanjiang-Haikou-Yangpu-Beihai-Fangcheng-Nansha	200	24			

Table 9 Best solution from the ten runs of hierarchical optimization-II

Route		Type of ship (TEU)	Number of container rehandles	C_1	C_2	$C = C_1 + C_2$
1	Nansha-Fangcheng-Beihai-Haifang-Nansha	100	0	26.51	0.35	26.86
2	Nansha-Zhanjiang-Yangpu-Haiko-Shuidong-Yangjiang-Nansha	150	38			
3	Nansha-Beijiao-Xinhui-Jiangmen-Rongqi-Xinfeng-Nanwei-Nansha	150	0			
4	Nansha-Zhongshan-Doumen-Zhuhai-Nansha	100	0			
5	Nansha-Huizhou-Yantian-Shekou-Nansha	100	0			
6	Nansha-Sanshan-Zengcheng-New Huangpu-Old Huangpu-Taiping-Nansha	150	6			
7	Nansha-Macau-Gaolan-Sanshui-Sanrong-Nansha	100	22			

Table 10 Best solution from the ten runs of the integrated optimization

Route		Type of ship (TEU)	Number of container rehandles	C_1	C_2	$C = C_1 + C_2$
1	Nansha-Fangcheng-Beihai-Haifang-Nansha	100	0	26.51	0.04	26.55
2	Nansha-Yangjiang-Shuidong-Haiko-Yangpu-Zhanjiang-Nansha	150	2			
3	Nansha-Beijiao-Xinhui-Jiangmen-Rongqi-Xinfeng-Nanwei-Nansha	150	0			
4	Nansha-Zhongshan-Doumen-Zhuhai-Nansha	100	0			
5	Nansha-Huizhou-Yantian-Shekou-Nansha	100	0			
6	Nansha-Sanshan-Zengcheng-New Huangpu-Old Huangpu-Taiping-Nansha	150	6			
7	Nansha-Sanrong-Sanshui-Gaolan-Macau-Nansha	100	0			

than the total cost of the best integrated solution obtained so far. For the whole iteration of the integrated optimization, the two phases (feeder routing and stowage planning) were interactive, and the stopping criteria were related to both phases; therefore, more effective combinations of the feeder routing plan and the related stowage plans were explored and resulted with the best solution among the three methods.

Overall, by comparing the results of the integrated optimization with hierarchical optimization-I, we have demonstrated that when the (near) optimal feeder routing plan was generated in the first phase, the stowage plans based on this feeder routing plan might have been infeasible. With the combined optimization of the feeder routing plan and the stowage plan, we could obtain feasible integrated solutions and the total cost of the feeder service could also be reduced. Therefore, the significance of optimizing the two problems jointly has been demonstrated.

By comparing the results of the integrated optimization with hierarchical optimization-II (in which different combinations of the feeder routing plan and stowage plans are explored), we have demonstrated the advantages and effectiveness of the integrated optimization method developed in our study.

4.3 Sensitivity analysis

In this section, we present the sensitivity analysis for examining the effects of the three sailing safety indices, time deadlines, and types of containerships.

In the sensitivity analysis, the number of ports was set to 30, and the loading and unloading quantities were randomly generated between 10 and 60.

To test one safety index, the other two were relaxed, and five conditions were generated for 1.2, 1.1, 1.0, 0.9, and 0.8 times the original safety range of the tested indices. For each condition, 10 cases were randomly generated, and each case was solved by the integrated optimization method and hierarchical optimization-II. Hierarchical optimization-I was not tested because of its inferior performance.

The sensitivity analysis for the range of the three sailing safety indices is presented in Table 11, wherein "R1" represents the number of cases, in which a solution from the hierarchical optimization was not feasible; "R2" represents the number of cases, in which the result of the integrated optimization was better than that of the hierarchical optimization;"R3" represents the number of cases, in which the result of the integrated optimization was same as that of the hierarchical optimization. Therefore, the number of cases, in which the result of the hierarchical optimization is better than the integrated optimization, can be calculated as 10-R2-R3. "Average cost" is the average cost for the results of the integrated optimization method. "Time" is the average computational time of the integrated optimization method.

As can be observed for index "R1," with the decreasing safety range, the number of infeasible solutions obtained by the hierarchical optimization tended to increase for all the three indices. In other words, the higher the restrictions on the safety indices, the more likely that an infeasible solution would be obtained through the hierarchical optimization. As discussed previously, the feeder containership often imposes stricter requirements on sailing safety constraints; therefore, the integrated optimization is critically essential. As seen for indexes "R2" and "R3," although for all the conditions, there existed cases, in which the integrated optimization and the hierarchical optimization yielded similar results, for each condition, the integrated optimization was better than the hierarchical optimization for a majority of cases.

Table 11 also shows that with an increasing restriction on the sailing safety indices of the containership, the cost for the solution obtained by the integrated optimization increased. This is because a feasible stowage plan could become infeasible if the range of the sailing safety index decreased, and the algorithm would continue to search for a feasible solution at the expense of increased costs (such as adding new containerships/routes). Moreover, when the three indices were compared, it can be observed that GM was the most sensitive factor among the three indices and had the highest impact on the solution, while Ms was the least sensitive factor, and had the lowest impact on the solution.

Next, a sensitivity analysis was performed on the types of containerships (100TEUs, 150TEUs, 200TEUs, and multi-type) and time deadlines (L1, L2, L3, L4, and L5, which represent 0.8, 0.9, 1.0, 1.1, and 1.2 times the original time deadlines, respectively). The integrated optimization was executed ten times for each condition, and the average results are shown in Table 12.

As expected, the average cost (average C) for the multi-type containerships was the lowest. In addition, the value of C decreased, as the time deadline was increased. This is because when the time deadline was extended, it could allow the containership to visit more ports, and fewer containerships were required, which was beneficial in terms of the economies of scale.

The sensitivity analysis for the main parameters of the problem indicates the consistency of the results under the various changes and further demonstrates the significance of this study. Moreover, the reasonable execution time (4–14 min) of the integrated optimization can ensure multiple runs for refining routing results under real life operational conditions.

5 Conclusion

In this study, we optimize feeder routing and containership stowage plans jointly to ensure the sailing safety of con-

Table 11 Sensitivity analysis for three sailing safety indices

Relaxed indexes	Tested indexes	R1	R2	R3	Average cost	Time (min)
Ms, T	GM1(1.2 GM)	0	6	3	22.66	7.26
	GM2(1.1 GM)	1	6	3	22.60	7.31
	GM3(1.0 GM)	1	7	3	24.10	9.07
	GM4(0.9 GM)	2	8	2	27.39	12.73
	GM5(0.8 GM)	4	8	2	29.83	13.22
Gm, Ms	T1(1.2 T)	0	7	3	23.27	7.09
	T2(1.1 T)	0	8	2	23.96	7.92
	T3(1.0T)	0	6	4	24.85	8.41
	T4(0.9 T)	1	7	3	25.41	9.37
	T5(0.8 T)	3	7	3	28.92	12.36
T,Gm	Ms1(1.2 Ms)	0	7	2	23.54	6.83
	Ms2(1.1 Ms)	0	7	3	23.27	7.31
	Ms3(1.0 Ms)	0	7	3	23.74	7.86
	Ms4(0.9 Ms)	0	8	2	23.91	7.37
	Ms5(0.8 Ms)	1	7	3	25.54	9.02

Table 12 Sensitivity analysis for types of containerships and time deadlines

Types of ship	Time deadlines	Number of routes	C_1	C	Time (min)
100TEUs	L1(0.8L)	12.4	31.31	31.32	7.82
	L2(0.9L)	11.3	31.36	31.37	7.46
	L3(1.0L)	10.5	30.80	30.83	7.02
	L4(1.1L)	10.2	30.34	30.36	6.79
	L5(1.2L)	9.9	30.17	30.21	6.55
	Average C		30.82		
150TEUs	L1(0.8L)	7.9	33.52	33.54	6.02
	L2(0.9L)	7.5	32.26	32.27	5.94
	L3(1.0L)	7.1	31.15	32.22	5.79
	L4(1.1L)	6.4	30.26	30.34	5.20
	L5(1.2L)	6.1	28.27	28.38	5.12
	Average C		31.35		
200TEUs	L1(0.8L)	6.9	34.07	34.07	5.23
	L2(0.9L)	6.3	32.63	32.64	5.08
	L3(1.0L)	5.8	30.95	31.06	5.96
	L4(1.1L)	5.2	29.01	29.14	4.75
	L5(1.2L)	4.8	27.53	27.69	4.62
	Average C		30.92		
Multi-type	L1(0.8T)	8.7	29.41	29.42	12.98
	L2(0.9T)	8.1	27.73	27.76	12.54
	L3(1.0T)	7.4	26.54	26.57	10.02
	L4(1.1T)	6.9	25.35	25.44	9.46
	L5(1.2T)	6.2	24.97	25.08	7.27
	Average C		26.85		

tainerships. To this end, we develop a formulation and a tailored heuristic algorithm for solving this complex model. The computational results show that compared with the hierarchical optimization, our method can effectively reduce the total cost of the feeder service and obtain feeder routing plans that could ensure the sailing safety of containerships within an acceptable computational time. Through the sensitivity analysis, we further confirm the consistent performance of the developed algorithm and the significance of this study.

The contribution of this study is twofold: at the planning level, we have extended the previous research on feeder containership routing by incorporating stowage plans, and the results show that the integrated optimization can not only ensure the sailing safety of the containerships (an issue that was not addressed in previous studies), but also reduce the total cost of the feeder service. At the model and algorithm level, we have designed a novel integrated optimization model, in which relationships between the feeder routing and stowage planning problems are well defined. We also develop a heuristic procedure tailored for the integrated optimization problem, in which a variable neighborhood search heuristic is proposed to update the feeder routing plan, and a special multi-port stowage planning problem is addressed and solved by GA.

Any future work, based on the current study, may proceed in several directions. Firstly, different types of containers can be considered. Secondly, the speed of the containerships can be considered as a variable. Finally, better heuristic approaches that use completely different strategies can be developed.

Acknowledgements This work was funded by the National Nature Science Foundation of China [Grant Numbers 71971035 71572022].

Compliance with ethical standards

Conflicts of interest Lingrui Kong declares that she has no conflict of interest. Mingjun Ji declares that he has no conflict of interest. Yunxiao Guan declares that he has no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

Appendix A

Formulations for GM, T, and Ms

a. GM can be calculated as $GM = KM - KG$. KM is the distance between the keel and metacenter of the containership, it is related to the draft, and can be accurately determined using the hydrostatic curve provided by the containership; KG is the distance between the keel and center of gravity of the containership, which can be calculated as $KG = \frac{\sum_{s \in S_k} W_b^{ks} \times H_s}{D}$, where D is the loaded displacement of the containership and H_s is the height of the center of slot s.

b. Ms can be calculated as $Ms = \frac{1}{2} \times (\gamma \times DL + \sum_{s \in S_k} W_b^{ks} \times I_s - D \times f \times L)$. γ is the equivalent arm of the total moment of the deadweight of the first and the latter half to the center cross section of the containership; DL is the light displacement of the containership; I_s is the

Table 13 Framework of hierarchical optimization-I

Generate an initial feeder routing plan s
$gen = 0$
While true **do**
  $k = 1$
  While $k \leq k_{max}$ **do**
    $s' = PickAtRandom(N_k(s))$ – pick a random solution s' from the k-th neighborhood of s
    $X'_1 = LocalSearch(s')$ – get the best solution X'_1 among 30 neighbors of s' in $N_k(s')$
    If $C(X'_1) < C(s)$ **then**
      $s = X'_1$
      $k = 1$
    Else
      $k = k + 1$
      $gen = gen + 1$
    End if
    If $gen = Maxgen$ **then**
      the while(true) loop is **terminated**
    End if
  End while
End while
Stowage plans are generated for containerships based on feeder routing plan s
Feeder routing plan s and related stowage plans are output
The total cost of feeder routing plan s and related stowage plans is calculated as the result of hierarchical optimization-I

Table 14 The framework of hierarchical optimization-II

Generate an initial feeder routing plan X_1
The stowage plans X_2 based on feeder routing plan X_1 are generated
The total cost of the initial solution is recorded($C(X_1) + C(X_2)$)
$gen = 0$
While true **do**
  $k = 1$
  While $k \leq k_{max}$ **do**
    $s' = PickAtRandom(N_k(X_1))$—pick a random solution s' from the k-th neighborhood of X_1
    $X_1' = LocalSearch(s')$—get the best solution X_1' among 30 neighbors of s' in $N_k(s')$
    If $C(X_1') < C(X_1)$ **then**
      $k = 1$
      $gen = 0$
      $X_1 = X_1'$
      The stowage plans X_2 based on feeder routing plan X_1 are generated
      The total cost of the current solution is recorded
    Else
      $k = k + 1$
      $gen = gen + 1$
    End if
    If $gen = Maxgen$ **then**
      the while(true) loop is **terminated**
    End if
  End while
End while
The feeder routing plan and related stowage plans with the lowest total cost are output
The lowest total cost is regarded as the result of hierarchical optimization-II

horizontal distance between slot s and the vertical center of the containership; f is the equivalent arm of the buoyancy of the containership; and L is the length of the containership.

c. The trim can be calculated as $T = \frac{\sum_{s \in S_k} I_s \times W_b^{ks} - LCB \times D}{MCT \times 100}$. LCB is the distance between the center of buoyancy and the vertical center of the containership and can be obtained from the trim diagram of the containership; and MCT is the moment required to change the trim of the containership by 1 cm, which can also be obtained from the trim diagram.

Appendix B

The framework of Hierarchical optimization-I and hierarchical optimization-II is shown in Tables 13 and 14, respectively.

References

Alumur SA, Kara BY (2008) Network hub location problems: the state of the art. Eur J Oper Res 190(1):1–21

Ambrosino D, Paolucci M, Sciomachen A (2015) Experimental evaluation of mixed integer programming models for the multi-port master bay plan problem. Flex Serv Manuf J 27(2–3):263–284

Ambrosino D, Paolucci M, Sciomachen A (2017) Computational evaluation of a mip model for multi-port stowage planning problems. Soft Comput 21(7):1753–1763

Avriel M, Penn M, Shpirer N (2000) Container ship stowage problem: complexity and connection to the colouring of circle graphs. Discrete Appl Math 103:271–279

Dethloff J (2001) Vehicle routing and reverse logistics: the vehicle routing problem with simultaneous delivery and pick-up. Or Spektrum 23(1):79–96

Fleszar K, Osman I, Hindi KS (2009) A variable neighborhood search algorithm for the open vehicle routing problem. Eur J Oper Res 195:803–809

Fontes F, Goncalves G (2017) A new hub network design integrating deep sea and short sea services at liner shipping operations. Int J Ship Trans Log 9(5):580

Gelareh S, Maculan N, Mahey P, Monemi RN (2013) Hub-and-spoke network design and fleet deployment for string planning of liner shipping. Appl Math Model 37(5):3307–3321

Gelareh S, Pisinger D (2011) Fleet deployment, network design and hub location of liner shipping companies. Trans Res E-Log 47(6):947–964

Hansen P, Mladenovic N (1997) Variable neighborhood search for the p-median. Locat Sci 5:207–226

Hansen P, Mladenovic N, Perez JAM (2010) Variable neighborhood search: methods and applications. Ann Oper Res 175:367–407

Hsu CI, Hsieh YP (2007) Routing, ship size, and sailing frequency decision-making for a maritime hub-and-spoke container network. Math Comput Model 45(7–8):899–916

Ji MJ, Shen LX, Shi BS, Xue Y, Wang F (2015) Routing optimization for multi-type containerships in a hub-and-spoke network. J Traffic Transp Eng (Engl Ed) 2(5):362–372

Kammoun M, Derbel H, Ratli M, Jarboui B (2015) A variable neighborhood search for solving the multi-vehicle covering tour problem. Electron Notes Discrete Math 47:285–292

Karlaftis MG, Kepaptsoglou K, Sambracos E (2009) Containership routing with time deadlines and simultaneous deliveries and pick-ups. Transp Res E-Log 45(1):210–221

Potvin JY (1996) Genetic algorithms for the traveling salesman problem. Ann Oper Res 63(3):337–370

Sambracos E, Paravantis JA, Tarantilis CD, Kiranoudis CT (2004) Dispatching of small containers via coastal freight lines: the case of the Aegean Sea. Eur J Oper Res 152(2):365–381

Suban TD, Twrdy E (2008) Decision support for optimal repositioning of containers in a feeder system. Promet -Traffic Transp 20(2):71–77

Wilson ID, Roach PA (2000) Container stowage planning: a methodology for generating computerised solutions. J Oper Res Soc 51(11):1248–1255

Zhang E, Mei QH, Liu M, Zheng FF (2018) Stowage planning in multiple ports with shifting fee minimization. Sci Program Sci Program 2018:1–9. https://doi.org/10.1155/2018/3450726

Zhang XY, Ji MJ, Yao S, Chen X (2015) Optimising feeder routing for container ships through an electronic chart display and information system. J Navig 68(5):848–868

Zheng J, Meng Q, Sun Z (2015) Liner hub-and-spoke shipping network design. Transp Res E-Log 75:32–48

Zhu HL, Ji MJ (2014) Optimal model and improved genetic algorithm of containership stowage on full route. J Traffic Transp Eng 14(5):59–67